THE ROLE OF HUBNESS IN HIGH-DIMENSIONAL DATA ANALYSIS

Nenad Tomašev

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, September 2013

Evaluation Board:

Prof. Dr. Matjaž Gams, Chairman, Jožef Stefan Institute, Jamova 39, Ljubljana, Slovenia

Prof. Dr. Dunja Mladenić, Member, Jožef Stefan Institute, Jamova 39, Ljubljana, Slovenia

Asst. Prof. Miloš Radovanović, Member, Department of Mathematics and Informatics, Trg Dositeja Obradovića 4, Novi Sad, Serbia

Prof. Dr. Mirjana Ivanović, Member, Department of Mathematics and Informatics, Trg Dositeja Obradovića 4, Novi Sad, Serbia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Nenad Tomašev

THE ROLE OF HUBNESS IN HIGH-DIMENSIONAL DATA ANALYSIS

Doctoral Dissertation

VLOGA ZVEZDIŠČNOSTI PRI ANALIZI VISOKODIMENZIONALNIH PODATKOV

Doktorska disertacija

Supervisor: Prof. Dr. Dunja Mladenić

Ljubljana, Slovenia, September 2013

Index

Abstract

This thesis examines the role of hubness in many important data mining contexts.

Hubness is a recently described aspect of the well known *curse of dimensionality*. It is an intrinsic property of high-dimensional data, where hubs emerge as centers of influence in k-nearest neighbor (kNN) topologies. They often exhibit a detrimental influence on the learning process. As most data of practical concern is high-dimensional, this is an important issue.

This thesis presents a set of novel, hubness-aware nearest neighbor data analysis techniques. They are shown to be more robust and reliable when working with high-dimensional data under the assumption of hubness. This is confirmed by an extensive experimental evaluation on a wide range of datasets from various domains, including images, text and time series.

The role of hubs as potential prototypes in high-dimensional data clustering was examined and it was shown that node degree in such k-nearest neighbor graphs is an appropriate measure of local cluster centrality. Several proof-of-concept clustering methods have been proposed: global K-hubs (GKH), global hubness-proportional clustering (GHPC) and global hubness-proportional K-means (GHPKM). They have been shown to perform well on high-dimensional data, even under large quantities of noise and in presence of outliers.

The impact of hubs on k-nearest neighbor classification was evaluated and several hubness-aware classification methods have been proposed and shown to be quite robust and accurate in classifying high-dimensional data. The hubness-fuzzy k-nearest neighbor (h-FNN), hubness information k-nearest neighbor (HIKNN) and naive hubness-Bayesian k-nearest neighbor (NHBNN) are novel classification approaches based on building the neighbor k-occurrence model on the training data and learning from past occurrences.

The impact of the choice of the underlying feature representation was examined in object recognition. Different feature types were shown to induce different degrees of hubness.

A novel secondary hubness-aware shared-neighbor similarity, $simhub_s$ has been proposed and shown to significantly improve the structure of the kNN graph reduce the overall hubness of the data, as well as the percentage of label mismatches.

The role of hubness and hub points has also been examined in other contexts, including class imbalance, instance selection, cross-lingual document retrieval, anomaly detection and bug duplicate detection.

Povzetek

V predloženi doktorski disertaciji smo preučili vlogo zvezdišč v številnih kontekstih strojnega učenja. Zvezdiščnost je lastnost visokodimenzionalnih podatkov, ki je povezana s t.i. prekletstvom dimenzionalnosti. Zvezdišča se v visokodimenzionalnih podatkih pojavljajo kot centri vpliva v topologijah kNN (k najbližjih sosedov). Zvezdišča imajo pogosto škodljiv vpliv na proces učenja, kar je v praksi pomembno zaradi pogostosti visokodimenzionalnih učnih problemov.

V pričujoči disertaciji smo predstavili niz novih, zvezdiščno-prilagojenih kNN metod za analizo podatkov. Predlagane metode omogočajo robustnejšo in zanesljivejšo analizo visokodimenzionalnih podatkov, kar je podprto z obsežno eksperimentalno raziskavo na slikah, tekstu in senzorskih podatkih.

Pri nalogi visokodimenzionalnega gručenja podatkov smo raziskali vlogo zvezdišč kot prototipov. Izkazalo se je, da je stopnja vozlišča v ustreznih kNN grafih primerna za merjenje lokalne centralnosti. Predlagali smo tri nove zvezdiščno-prilagojene metode za gručenje podatkov: globalno K-zvezdiščno gručenje (GKH), globalno zvezdiščno-sorazmerno gručenje (GHPC) in globalno zvezdiščno-sorazmerno različico K-means algoritma (GHPKM). Eksperimentalno smo ugotovili, da predlagane metode vodijo do izboljšav, kar pa je najbolj izrazito v prisotnosti visokih nivojev šuma.

Preučili smo vpliv zvezdišč na kNN klasifikacijo in predlagali več novih zvezdiščno-prilagojenih kNN klasifikacijskih metod. Pri nalogi klasifikacije visokodimenzionalnih podatkov so se predlagane metode izkazale za zelo robustne in točne. Metode temeljijo na analizi pojavitev točk v kNN okolicah. Na tem principu so osnovane metode: Hubness-fuzzy k-nearest neighbor (h-FNN), Hubness Information k-nearest neighbor (HIKNN) ter Nad've Hubness-Bayesian k-nearest neighbor (NHBNN).

Vpliv izbire atributov na zvezdiščnost smo preučili na problemu zaznavanja objektov. Različni atributi inducirajo različne stopnje zvezdiščnosti. Ta opažanja so vodila do zasnove nove sekundarne zvezdiščno-prilagojene metrike, simhubs. Pokazali smo, da nova zasnova bistveno izboljša strukturo kNN grafa, kar omogoča lažje strojno učenje in boljšo klasifikacijo.

Vlogo zvezdišč smo preučili tudi v številnih drugih kontekstih, vključno z neravnovesjem razredov, izborom primerov, medjezičnim iskanjem informacij, zaznavanjem anomalij in iskanjem podvojenih poročil hroščev.

Abbreviations and Notation

AI	=	artificial intelligence
AKNN	=	adaptive k-nearest neighbor
BN_k	=	bad occurrence frequency
CCA	=	canonical correlation analysis
D_k	=	k-neighbor set
dwh-FNN	=	distance-weighted h-FNN
DM	=	data mining
FNN	=	fuzzy k-nearest neighbor
GN_k	=	good occurrence frequency
HIKNN	=	hubness information k-nearest neighbor
h-FNN	=	hubness-based fuzzy k-nearest neighbor
HoG	=	histograms of gradients
hw-kNN	=	hubness-weighted k-nearest neighbor
KCCA	=	kernel canonical correlation analysis
KM	=	K-means
KM++	=	K-means++
kNN	=	k-nearest neighbor
MCC	=	Matthews correlation coefficient
MDS	=	multidimensional scaling
ML	=	machine learning
NHBNN	=	naive hubness-Bayesian k-nearest neighbor
NWKNN	=	neighbor-weighted k-nearest neighbor
N_k	=	k-occurrence frequency, point hubness
$N_{k,c}$	=	class-specific k-occurrence frequency, class-hubness
$N_{k,c_1}^{c_2}$	=	class to class hubness
NN	=	nearest neighbor
PCA	=	principal component analysis
PNN	=	probabilistic k-nearest neighbor
RImb	=	relative imbalance coefficient
RNN	=	reverse nearest neighbor
SIFT	=	scale-invariant feature transform
$simcos_s$	=	shared neighbor cosine similarity
$simhub_s$	=	hubness-aware shared neighbor similarity
SN_k	=	skewness of the k-occurrence frequency distribution, data hubness
SVD	=	singular value decomposition
SVM	=	support vector machines
TFIDF	=	term frequency - inverse document frequency
VSM	=	vector space model

1 Introduction

1.1 General Introduction

We live in the age of information. Never before has the information been so easily available and abundant. This only became possible in the recent decades, thanks to the great advances in digital computing technology. Indeed, computers that were once big and expensive machines now permeate every aspect of our daily lives. We use them both for work and pleasure, business and entertainment alike.

Yet, computers are more than just simple machines. They require programming to run and the design of adequate software is just as important as the underlying hardware architecture. Information must be properly handled.

Intelligent data analysis requires flexible, adaptive methods, capable of processing large quantities of complex, noisy data in an effective and efficient way. Ever since the first general purpose computers were unveiled, people have been investigating the possibilities for creating some sort of intelligent machines. As the only sort of intelligence we are closely familiar with is our own, it seemed as if the long-term aim should be set at creating systems which would reason just as people do. This is usually referred to as the strong AI (artificial intelligence) hypothesis [Russell and Norvig, 2003].

Alan Turing (1912–1954), who is often rightfully so referred to as the father of computer science, was the first to propose such a concept by saying "A computer would deserve to be called intelligent if it could deceive a human into believing that it was human.". This test was later challenged by some philosophers, most notable Searle in his famous Chinese room argument. Regardless, the Turing test remains the most natural approach to testing general-purpose intelligence in artificial systems. Even though it seems as quite an ambitious goal, it has already been achieved in some limited context. Computer games are often used as testing environments for AI methods, as it is easy to vary and control the simulation parameters [Champandard, 2003]. In one such game, the Unreal Tournament 2004, a well-known FPS, a competition was organized with the goal of creating bots (AI players) which would appear to be human to other human players in the game. The results of the Botprize 2K competition (http://www.botprize.org) were presented in September 2012 at the IEEE Conference on Computational Intelligence and Games and have drawn quite a bit of attention. The winning bots achieved a humanness rating of 52 percent, whereas the actual human players received an average humanness rating of only 40 percent. In other words, the artificial computer programs seemed to behave more human than humans. This is quite astonishing, even though it is only a limited set of actions in a rather small, simulated environment. We have no reason to be skeptical about the limits of artificially intelligent systems and no one can say with certainty what the future holds.

The approaches to intelligent system design have evolved over time and the focus is set on learning and adaptability to changing conditions in dynamic, complex environments. Yet, this was not always the case.

Back in 1997, all eyes were set on what will later be regarded as "the most spectacular chess event in history", the clash between the reigning world champion and one of the greatest chess players of all time Garry Kasparov and the IBM chess playing supercomputer

Deep Blue. Deep Blue won the event by a slight margin (3.5:2.5) and landed a major blow to the mental supremacy of man over machine, especially as chess was always thought of as one of the ultimate mental battlegrounds. Yet, what was celebrated back then as a victory of artificial over natural intelligence was no more than an illusion. If Deep Blue was good at anything, it was "number crunching". Its triumph was a consequence of unparalleled computational power, not intelligence as we define it today. Its design was no different from that of the modern chess engines (Rybka, Hydra, Shredder, Fritz, etc.). A chess engine has at its disposal an extensive base of opening knowledge, encoded by the world's top authorities, as well as an endgame tablebase where the exact step-by-step procedures of finishing a game are provided for a vast set of endgame positions. During middlegame, the engine plays by searching through a variation three and choosing the most promising continuation based on the heuristic evaluation function which was, again, directly encoded by chess experts. As such engines are capable of checking and evaluating millions of variations, they are certainly powerful opponents and able to outplay even the world champions. However, such computer programs exhibit no actual understanding of either the position itself or how the various higher-order chess concepts define it. The algorithm is only implicitly aware of it through the predefined heuristic. Even more importantly, such computer programs are incapable of learning and incapable of adapting their strategy to specific opponents and their style of play. This was later sharply criticized by Garry Kasparov himself: "This is our last chess metaphor then, a metaphor for how we have discarded innovation and creativity in exchange for a steady supply of marketable products. The dreams of creating an artificial intelligence that would engage in an ancient game symbolic of human thought have been abandoned.","Like so much else in our technology-rich and innovation-poor modern world, chess computing has fallen prey to incrementalism and the demands of the market.","Such thinking should horrify anyone worthy of the name of scientist, but it seems, tragically, to be the norm."[Rasskin-Gutman, 2009]. Nevertheless, there is still ongoing research in this area and it was shown that it is indeed possible to automatically infer chess-related concepts from past observations [Možina et al., 2012][Guid et al., 2010], as well as utilize reinforcement learning for improving search and heuristics [Veness et al., 2009][Block et al., 2008].

As the chess arena was conquered, it was time to tackle more difficult problems. It was February 2011 when Watson, a machine programmed by the IBM team, won against strong human competitors in a live TV-broadcast quiz "Jeopardy!". The system was built on top of a knowledge base containing more than 200 million pages of dictionary, encyclopedia and news content - including the entire off-line copy of Wikipedia. It implemented the state-of-the-art techniques for natural language processing, information retrieval, knowledge representation and reasoning. The announced commercial applications include clinical decision support in treatment recommendation and disease diagnosis.

Even though both Watson and Deep Blue were incredible engineering achievements, the purpose of most AI software is neither to compete with people nor to try to replace them. Most AI research is actually directed at creating special-purpose intelligent programs which would automate data processing and help people in solving complex real-world problems.

So, what is intelligence and which properties do we expect to see in an artificially intelligent system? Obviously, an intelligent system needs knowledge representation in order to facilitate understanding and it needs to embed some problem-solving procedures. However, a genuinely intelligent system must first and foremost have learning capabilities. Without learning, a system would not be able to adapt to a sudden change in environment and its predefined reactions would fall short of their marks [Russell and Norvig, 2003].

1.2 Background

Machine learning is among the central fields of modern AI research. It is the study of algorithms which generate models, patterns and predictions based on observations. Data mining

and knowledge discovery from databases are closely related and the disciplines frequently overlap in methods and ideas. They are roughly distinguished by the fact that machine learning is mostly concerned with future predictions while data mining usually refers to the discovery of new patterns, previously unknown and potentially useful properties of the data [Witten and Frank, 2005b].

Data science is more than just research. The advances in data processing techniques are unlocking new effective ways of extracting business value from the available customer and market information. Large companies are struggling with information that comes in volumes and varieties never seen before. Analyzing such big data without automated intelligent assistance is impossible. Adaptive tools and systems have become a necessity. Even a small increase in the system performance might translate to millions in profit.

These advanced learning techniques are already in ubiquitous use in many domains, such as: market analysis, recommendation systems, bioinformatics, genetics, medicine, education, search, question answering [Marinčič et al.], social network analysis, power distribution, sensor networks. Data mining has also been successfully applied for analyzing the macroeconomic indicators of a development of state economy [Vidulin and Gams, 2011].

We can distinguish between several major machine learning approaches and methodologies, each tailored for a specific sort of data. Combinatorial techniques are often employed for frequent subset mining in market basket analysis. In general, discrete data can easily be modeled by rule-based systems or decision trees. Alternatively, probabilistic graphical models, like Bayesian networks, can be used to model the conditional dependencies between different representational features and infer the unobserved feature values [Han, 2005]. Linear models are also frequently used, especially when extended by the 'kernel trick', an implicit non-linear mapping to a (usually) higher dimensional feature space where the data clusters are more easily separable by hyper-planes. This involves using kernel functions which act as an implicit replacement for the scalar product in the target space. The most famous large-margin algorithm is the Support Vector Machines (SVM) classifier, which is considered to be a state-of-the-art in many domains [Schölkopf and Smola, 2001]. Gaussian mixture models are also frequently used in non-linear systems, as well as density-based methods. Artificial Neural Networks (ANN) offer great flexibility, though it is usually a-priori unclear which network topology to choose or how to infer it from the data. Additionally, the downside is the *black-box* nature of the neural network approaches, as such decision systems have low interpretability. Lastly, instance-based learning is often used in practice, i.e. nearest-neighbor methods.

In instance-based learning, we infer the desired information about a new observation by comparing it to the previously observed and categorized examples. The approach is based on an intuitive notion that similar instances usually belong to same data categories or clusters. This similarity needs to be captured by an appropriate metric function in the representational feature space. The problem of generating and choosing the optimal feature set and the similarity function is non-trivial and problem-specific. Different representations and different metrics are used in different contexts, depending on the task at hand. Nearest-neighbor methods are widely used in many machine learning applications.

The *curse of dimensionality* has been one of the main topics in machine learning and data mining research ever since its discovery [Bellman, 1961]. It is a term commonly used to refer to inherent difficulties involving high-dimensional data analysis. All high-dimensional data is sparse and an exponential number of examples is required to reach reliable density estimates. This has subtle and profound consequences, which often counter our intuition and pose a serious challenge for many traditional data mining approaches. This is why we are usually forced to design special, modified algorithms, capable of properly handling complex, high-dimensional data.

The emergence of *hubs* in k-nearest neighbor graphs is an important consequence of the dimensionality curse, affecting all nearest-neighbor methods in high-dimensional data. The

overall data *hubness* can be seen as the increasing skewness (third standard moment) of the neighbor occurrence distribution. Hubs are frequent nearest neighbors, points which are frequently retrieved by the system. If the relevance of such points were proportional to their occurrence frequency, this would not necessarily have negative consequences. However, careful practical examination has determined that this is frequently not the case [Radovanović et al., 2009][Radovanović et al., 2010a][Radovanović et al., 2010b]. Data hubness is highly dependent on the particular choice of feature representation, normalization and similarity. Intrinsically high-dimensional data is known to exhibit hubness, but its degree and distribution over the examples may vary. Therefore, hubs might even sometimes be interpreted as 'noise' when they are retrieved. Hubness is closely related to the distance concentration phenomenon [Aggarwal et al., 2001][François et al., 2007], another counter-intuitive property of high-dimensional data.

1.3 Nearest Neighbor Approaches in Machine Learning and Data Mining

The basic k-nearest neighbor (kNN) rule [Cover and Hart, 1967] is a well known statistical method used for class density estimation and, consequently, classification. As such, it is a prototypical example of what is usually referred to as a *lazy* classifier. This is because no model is learned from the data during the training phase of the basic kNN approach and all inference is done run-time while considering individual examples. This means that all the training examples need to be stored as they are accessed during subsequent system queries. Many k-nearest neighbor methods have been developed over the years and applied to solving different types of data mining and machine learning tasks.

The underlying idea of all k-nearest neighbor methods is that the inference in the point of interest is made based on its k nearest neighbors, i.e. k examples from the training data that are most similar to the point that is currently being inspected. Intuitively, this means that the similarity calculated in the chosen feature space might imply the similarity among the 'hidden' variables of the compared points.

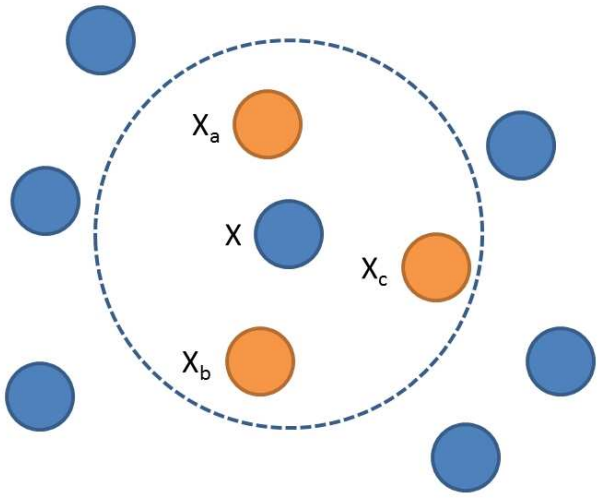


Figure 1: The 3 nearest neighbors to point X in this 2D data are points X_a, X_b and X_c.

The k-nearest neighbor problem is closely related to the concept of Voronoi diagrams. Each fixed set of points for a fixed neighborhood size k defines a unique Vornoi tesselation in the feature space. In case of $k = 1$, all points from the same cell have the same nearest neighbor. An example is shown in Figure 2. It is also possible to define higher-order Voronoi diagrams when $k > 1$, where all points in the same region have the same set of

k-nearest neighbors. As calculating the exact split can sometimes be time consuming, some approximate algorithms have also been used in practice [Arya, 2002]. The Voronoi split for a given dataset can sometimes be used for speeding up the search [Kolahdouzan and Shahabi, 2004][Liotta et al., 1996].

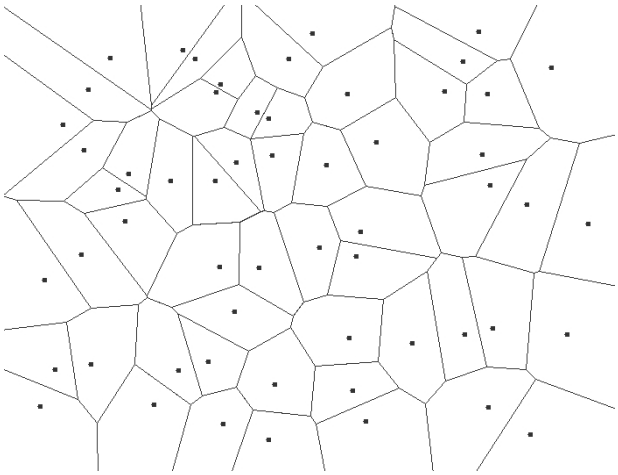


Figure 2: The Voronoi tessellation in the plane for $k = 1$ for a given set of points.

The kNN method is probably most widely known in the machine learning community for its use in classification. It is one of the simplest available classifiers, easy to implement and test. The label of a new instance is determined by a majority vote of its k-nearest neighbors (kNN) from the training set, as shown in Figure 3. This simple rule has some surprising properties which go in its favor. For instance, when there is no overlap between the classes, 1-nearest neighbor is asymptotically optimal [Cover and Hart, 1967][Devroye, 1981]. As for the kNN rule, it has been shown to be universally consistent under some strong assumptions, namely $k \to \infty$ and $k/n \to 0$ [Stone, 1977][L. Devroye and Lugosi, 1994]. Let $D = (x_1, y_1), (x_2, y_2), ..(x_n, y_n)$ be the data set, where each $x_i \in R^d$. In this case, the x_i are feature vectors which reside in d-dimensional Euclidean space, and $y_i \in c_1, c_2, ..c_C$ are the labels. It can be shown that in the hypothetical case of an infinite data sample, the probability of a nearest neighbor of x_i having label c is asymptotically equal to the posterior class probability in point x_i, namely $p(c|x_i) = \lim_{n\to\infty} p(c|\mathrm{NN}(x_i))$. These conditions are never exactly met in real data, which is often sparse if the feature space is high-dimensional. Additionally, k-nearest neighbor methods do not explicitly require the vector space model (VSM) of the data [Raghavan and Wong, 1986], they allow for more general types of inference. As long as it is possible to define a similarity measure between all pairs of instances or produce a ranking for each query, most basic types of k-nearest neighbor classification methods would be able to output a prediction.

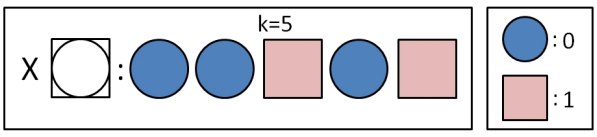


Figure 3: An example of k-nearest neighbor classification rule for k=5 in a binary classification case. The 5 nearest neighbors of x are shown and 3 of them share the label 0, while 2 share the label 1. According to the nearest neighbor rule, $p(y = 0|\mathrm{NN}_5(x)) = 0.6$ and $p(y = 1|\mathrm{NN}_5(x)) = 0.4$, so the point x would be assigned label 0. The ratio between the two label probabilities can also be viewed as an estimate of the ration between class probability density functions in point x.

Various extensions and variations of the original kNN classifier have been proposed over the years. The fuzzy k-nearest neighbor classifier is frequently used in the biomedical do-

main and represents a 'soft' alternative, as it is able to handle fuzzy class labels [Keller et al., 1985][Jensen and Cornelis, 2008][Shang et al., 2006]. One of the state-of-the-art contemporary classifiers is the large margin kNN classifier, based on learning the Mahalanobis distance matrix from the data via semidefinite programming [Weinberger et al., 2005][Min et al., 2009]. Feature weighting is also common [Han et al., 2001], as well as approaches based on adaptive distance measures and neighborhoods [Wang et al., 2007][Song et al., 2007][Hastie and Tibshirani, 1996][Ougiaroglou et al., 2007][Short and Fukunaga, 1981]. These sometimes include kernel-based approaches [Peng et al., 2004][Zhang et al., 2006]. Multi-label k-nearest neighbor classifiers have also recently been given some attention [Zhang and Zhou, 2007][Zhang and Zhou, 2005][Younes et al., 2008][Chiang et al., 2012]. The practical application of k-nearest neighbor classification methods are numerous [Shen and Chou, 2005][Chou and Shen, 2006][Liao and Vemuri, 2002][Rajagopalan and Lall, 1999].

Extensions of kNN are considered to be well suited for classification under the assumption of class imbalance, which is an important learning context. The basic k-nearest neighbor classifier does not build a model and thus performs essentially no generalization. This results in kNN having a high specificity bias, which stops it from over-generalizing. Over-generalization under class imbalance usually entails that the minority class examples end up being assigned to the majority class, especially in the borderline regions where classes overlap and the minority class has a comparably lower density and representation. This usually causes a significant misclassification rate [Holte et al., 1989][van den Bosch et al., 1997]. Several kNN classifiers that are appropriate for use in the imbalance class domain have been proposed [Garcia et al., 2008][Hand and Vinciotti, 2003][Liu and Chawla, 2011][Tan, 2005a][Wang et al., 2010b][Li and Zhang, 2011].

Clustering is one of the most important unsupervised tasks in data mining, as it helps in discovering the underlying structure of the data and is often one of the first steps in data analysis. The k-nearest neighbor approaches are frequently encountered in various clustering applications. The standard way to reach a clustering via kNN is to infer linkage properties from the k-nearest neighbor graph and perform some form of graph clustering [Maier et al., 2009][Franti et al., 2006][Maier et al., 2007][Lucinska and Wierzchon, 2012][Arefin et al., 2012]. Another way would be to use density-based clustering methods based on the kNN density estimate [Biçici and Yuret, 2007; Tran et al., 2003; Zhang et al., 2007]. Of course, there is always the issue of choosing the proper neighborhood size, since both small and large values of k can cause problems for such density-based approaches [Hader and Hamprecht, 2003]. Shared neighbor clustering methods have become especially popular for high-dimensional data clustering [Jarvis and Patrick, 1973][Ertz et al., 2001][Yin et al., 2005][Moëllic et al., 2008][Patidar et al., 2012][Zheng and Huang, 2012]., including the clustering of large image databases. Yet, other less typical kNN clustering approaches have been proposed as well [Hsieh et al., 2010][Hu and Bhatnagar, 2012][Sun and Wang, 2010].

The k-nearest neighbor methods turn up naturally in information retrieval, as well as collaborative filtering [Töscher et al., 2008] and recommendation systems [Liu et al., 2012][Pan et al., 2013][Gemmell et al., 2009]. It is easy to see why these types of approaches are well suited for collaborative filtering [Töscher et al., 2008][Gjoka and Soldo, 2008][Grčar et al., 2006][Xiaoyuan Su, 2008]. Namely, the behavior and the preferences of a user can be modeled by observing the most similar previously analyzed users. The same can be said for product-oriented analysis, where one would be, for instance, interested in finding the preferentially most similar movies or books to any given movie/book. Therefore, the k-nearest neighbors are often used to make inferences about future preferences, ratings or purchases.

Apart from the mentioned machine learning applications, k-nearest neighbor estimators have been used in various traditional statistical applications as well. Apart from density estimation [Mack and Rosenblatt, 1979][Fukunaga and Hostetler, 1975], it is also frequently used in entropy estimation [Beirlant et al., 1997][Sricharan et al., 2011][Singh et al., 2003], which is of high importance in many fields and domains [Hnizdo et al., 2008]. More specifi-

cally, if we have the entropy estimate derived from a set of n points by H_n and the distance from point x_i to its nearest neighbor as $\rho_{n,i} = d(x_i, \mathrm{NN}(x_i))$, the nearest neighbor entropy estimate is given as: $H_n = \frac{1}{n}\sum_{i=1}^{n} \ln(n \cdot \rho_{n,i}) + \ln 2 + C_E$, where C_E is the Euler's constant which yields $C_E = -\int_0^\infty e^{-t} \ln t dt$. The differential entropy estimates can be used to find independent subspaces [Póczos and Lörincz, 2005]. This is a common task in independent component analysis [Faivishevsky and Goldberger, 2008], where independent sources need to be detected and separated in a mixed signal.

The k-nearest neighbor regression is another widely used predictive method. Unlike in classification, the target variable is continuous. The basic idea is similar to classification, as the predicted value is taken either as a simple or a weighted average of the target variable values among the k-nearest neighbors. However, more sophisticated techniques are also possible [Hamed et al., 2012][Kramer, 2011][Kramer, 2012].

In all kNN-related machine learning and data mining methods, the required functionality is the ability to efficiently and effectively calculate the k-nearest neighbors for a query point. When the total number of queries is low, the brute force method of calculating all the distances and partially sorting in order to find the k smallest ones is usually sufficient. However, as the queries become more frequent and the size of the data increases, this can become computationally expensive and reduce the overall responsiveness of the system by significantly increasing the time required to process each individual query. This is why many efficient approaches for speeding-up the k-nearest neighbor search have been proposed over the years [Jagadish et al., 2005][Katayama and Satoh, 1997][Tao et al., 2010][Tao et al., 2009][Seidl and Kriegel, 1998][Iwerks et al., 2003][Yu et al., 2005][Song and Roussopoulos, 2001]. There is a difference between speeding up external searches over a fixed database and wanting to calculate the complete k-nearest neighbor graph for a given dataset. The latter is often required for estimating the kNN topology of the data. In our experiments, we have mostly relied on the approximate method based on recursive Lanczos bisections [Chen et al., 2009]. However, locality sensitive hashing (LSH) techniques are probably the most widely used in practice, as they have been shown to be quite reliable [Kulis and Grauman, 2011][Pauleve et al., 2010][Haghani et al., 2009][Rasheed et al., 2012][Chen et al., 2011][Gorisse et al., 2011]. By using the locality sensitive hashing, it is possible to apply the k-nearest neighbor algorithms for analyzing very large datasets. Different approaches to parallelization of approximate kNN graph construction are also available [Wang et al., 2012]. Most approximate approaches are closely tied to a predefined metric, which allows them to achieve the speed-up, though some generic methods exist as an alternative [Dong et al., 2011]. Apart from the up-scaling approaches, there has also been recent interest in privacy-preserving applications and privacy-preserving kNN search [Shaneck et al., 2009].

Data reduction is another common approach for speeding-up the k-nearest neighbor methods, as well as removing noisy points or outliers that might negatively affect the inference [Garcia et al., 2012][Liu, 2010][Liu and Motoda, 2002]. Many prototype selection methods are based on analyzing the k-nearest neighbor sets on the training data and this includes ENN [Wilson, 1972], CNN [PE, 1968], GCNN [Chou et al., 2006], RT3 [Wilson and Martinez, 1997], AL1 [Dai and Hsu, 2011] and INSIGHT [Buza et al., 2011].

Despite their widespread use, the k-nearest neighbor approaches run into some difficulties when applied to intrinsically high-dimensional data, one of which is the emergence of hubs that skew the underlying distribution of influence [Radovanović et al., 2009][Radovanović et al., 2010a]. The phenomenon of hubness will be discussed in more detail in the following Section.

1.4 The Hubness Phenomenon

Hubs emerge naturally as centers of influence in many different contexts. This thesis deals primarily with the consequences of the existence of hub points in k-nearest neighbor topolo-

gies of intrinsically high-dimensional data [Radovanović et al., 2009][Radovanović et al., 2010a], but there are many other situations in which similar uneven distributions of influence arise.

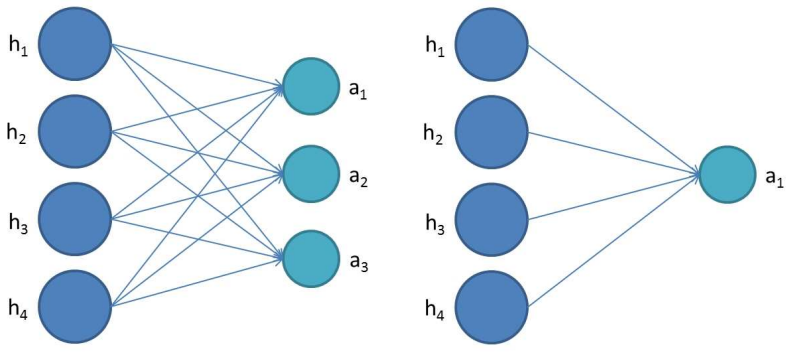
1.4.1 Hubs and Authorities in Directed Graphs

The k-nearest neighbor graph is a special case of a more general class of directed graphs and networks, where hubs have been known to emerge in certain contexts. One such network is the internet.

The distribution of popularity and relevance on the world wide web is known to be skewed [Adamic and Huberman, 2002]. The internet hosts a relatively small percentage of large, highly connected and highly relevant web pages, followed by a wide spectrum of relatively small non-influential sources. This is a common pattern in many well-known large-scale networks and will be discussed in more detail in Section 1.4.2.

The concepts of *hubs* and *authorities* are frequently encountered in network link analysis [David and Jon, 2010] and their meaning in that context will be outlined here, for purposes of comparison and disambiguation. Hubs are informally defined as the centers of influence, nodes with a high outgoing degree in the network graph. Authorities, on the other hand, are nodes that are linked to by many hubs. A page can be both a hub and an authority at the same time, though most pages are neither. Given a directed graph $G = (V, E)$, a *hub-score* $h(v)$ and an *authority score* $a(v)$ are usually assigned to each node in the graph, in order to measure the degree to which it is a hub or an authority. It is, of course, possible to define a threshold θ for a cut-off point and denote hubs as all $v \in G : h(v) > \theta$ or the authorities as all $v \in G : a(v) > \theta$.

The motivation for observing these particular types of points stems from the structure of the internet and the need for browsing and searching for relevant pages on different topics. Hubs correspond to pages holding carefully picked catalogues of links to relevant pages in a given domain. An example of a hub would be a hotel booking website that links to all hotels in a given area. Detecting the relevant hubs for a given query can help to improve the result sets in internet search [Chirita et al., 2003], as it helps in finding the authorities that are the ultimate goal. Authority pages are important because them being linked to by many hubs suggests that their content might be potentially highly relevant for the user. Simply counting the incoming and/or outgoing links is unfortunately not enough to satisfy the requirements for reliable search, as demonstrated by the example in Figure 4.



(a) Reliable authority pages linked to by several reliable hubs.

(b) An unreliable authority page that is linked to by pages of low hubness score.

Figure 4: An example showing how the nature of the linking hub-points influences the reliability of target authority pages.

The Hubs and Authorities (HITS) algorithm [Kleinberg, 1999b][Kleinberg, 1999a] is an iterative procedure that determines the hub and authority scores for web pages and was a

precursor to the well-known PageRank score [Brin and Page, 1998] that Google search is largely based upon. The HITS algorithm is a fairly simple one: the hub-score of a node is updated by assigning to it the sum of authority scores of the pages it links to. The authority update is analogous. Normalization of the scores is performed at each iteration. Each node is assigned a score of 1 initially. This is shown in Equation 1.

$$\begin{aligned} \forall v \in V : h(v) &\longleftarrow \sum_{v \to y} a(y) \\ \forall v \in V : h(v) &\longleftarrow \frac{h(v)}{\sqrt{\sum_{v \in V} h^2(v)}} \\ \forall v \in V : a(v) &\longleftarrow \sum_{y \to v} h(y) \\ \forall v \in V : a(v) &\longleftarrow \frac{a(v)}{\sqrt{\sum_{v \in V} a^2(v)}} \end{aligned} \tag{1}$$

Modifications to the original algorithm have also been considered [Benzi et al., 2012] [Miller et al., 2001]. It has also been used in querying multi-relation databases [Li et al., 2012], extracting features for detection of gene orthology [Towfic et al., 2010] and social network analysis [Ovalle-Perandones et al., 2009]. The semantics of the hub and authority scores may vary depending on the type of network and the domain it represents.

The definition of hubs and hubness that is used throughout this thesis is not a converging limit of an iterative process, but is derived directly from the node degree distribution. These concepts are defined precisely in Section 1.4.3.2 within the context of the thesis.

1.4.2 Hubs and Power Laws

Hubs often arise as centers of influence in scale-free networks. High-dimensional kNN topologies that are the focus of this thesis are not necessarily scale-free, but sometimes they tend to be [Radovanović, 2011]. Examining the role of hubs in such networks can help shed some light on the wider context of the problem.

Many complex systems share the same underlying structure, as a result of common organizing principles. Power laws are surprisingly common in large networks. Networks where the degree distribution follows a power law are known as *scale-free networks*. Let $P(k)$ be the fraction of nodes adjacent to exactly k edges in the graph. In scale-free networks, this is proportional to a decreasing power of k, like this: $P(k) \propto k^{-\gamma}$. In other words, we would expect to see a large number of low-degree nodes and a small number of highly connected centers of influence, *hubs*, in the long tail of the distribution [Barabási and Bonabeau, 2003][Wang and Chen, 2003].

The reason why the power law distributions are called scale-free is that scaling the function argument by a constant factor results in the proportional scaling of the original function. For instance, let $P(k) = \lambda \cdot k^{-\gamma}$ be a power law. Then, $P(\alpha \cdot k) = \lambda \cdot \alpha^{-\gamma} \cdot k^{-\gamma} = \alpha^{-\gamma} P(k) \propto P(k)$. It follows that all the power laws with the same scaling constant γ are equivalent up to constant factors, as they are in fact scaled versions of each other.

Studying the properties of such network structures is important, as the world that we live in is itself a hierarchy of interaction networks operating at different scales and levels of granularity. Societies are networks of people, connected by their friendships, professional ties and interests. People use their brains to govern their decisions and brains are networks of neurons connected via synapses by axons and dendrites. Cells themselves are networks of complex molecules that interact in many biochemical reactions [Albert, 2005]. Similar analogies can be seen while modeling ecosystems, markets [Kim et al., 2002], internet, and so on. Many of these networks exhibit scale-free properties [Barabási and Bonabeau, 2003]. Even the language that we use follows many power laws. The most famous is certainly

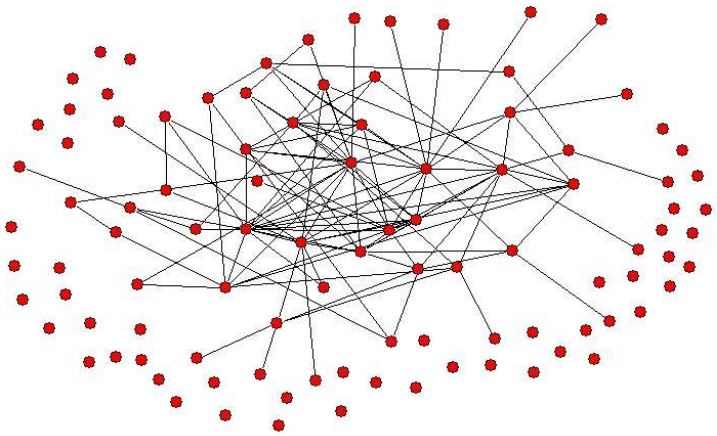


Figure 5: A randomly generated scale-free network of 200 nodes. The graph contains many disconnected nodes and leaves, that are dominated by a certain number of highly connected hub nodes.

Zipf's law, which describes the power law distribution of word frequencies that are inversely proportional to the word frequency ranks. Furthermore, it has been shown that some semantic word networks, like networks of synonyms, follow a power law degree distribution and that this distribution seems to be *language independent* [Makaruk and Owczarek, 2008], which is an interesting property that might shed some light on the universal laws behind the forming of languages in human societies. Zipf's law is captured by the zeta distribution, $p_s(x=k) = \frac{1}{k^s \cdot \zeta(s)}$. The exchange of parameters can transform this into a Pareto distribution, that is a power law distribution of a fitness (survival) function that was first used to model the overall distribution of wealth [Levy et al., 2005].

The emergence of scale-free networks can sometimes be interpreted based on the mechanisms of *preferential attachment*, or as it is less formally called, the 'rich get richer' process. This sort of a process was first studied by Yule in 1925 [Yule, 1925], in his paper on the genus size among the flowering plants. When a new species appears, it is placed in a new genus if it is sufficiently different from all the known species within the genus of its parent species, from which it diverged. The more species the genus already has, the higher the probability that the new species will stay within the existing genus. This is an obvious positive reinforcement loop that results in a power law distribution of genus sizes. Similar mechanisms can be observed in, for instance, collaboration graphs of co-authorship in scientific publishing. The more co-authors someone has, the higher the likelihood of their next paper also being co-authored [Farkas et al., 2002]. The preferential attachment model for growing random scale-free networks was developed in 1999 by Barabási and Albert [Barabási and Albert, 1999] and was applied to explaining the structure of the world wide web.

Scale free networks often exhibit the *small-world* property [Milgram, 1967][Wang and Chen, 2003], i.e. high connectivity. A famous example is the conjecture of *six degrees of separation* [Barabasi, 2003], where every living person on the planet would be connected to any other person by no more than six friendship links. The accuracy of this notion has been questioned on several occasions and it isn't entirely clear whether there are exceptions to the rule, but the fact remains that we are living in a highly connected world. The reason for this lies in hubs of the network, in this case people that know many other people, social hubs. Hubs increase the connectivity of networks and this can usually be seen as a good thing, though it can also be a problem, for instance when it comes to disease spread [Dezso and Barabási, 2001]. However, specifically targeting hubs while taking measures towards epidemic containment has proven to be an effective strategy [Dezso and Barabási, 2001]. In general, hubs make the scale-free networks more resistant to random failures, though much more susceptible to coordinated attacks against their hubs [Barabási and Bonabeau, 2003][Zhao and Xu, 2009].

Same power law degree distributions can exhibit slightly different topological properties and there is an ongoing debate on how to best define what is meant by the term of scale-free networks. The degree distribution landscapes can be characterized in different ways and studied in more depth [Axelsen et al., 2006]. It all depends on how hubs are connected, whether there is a strong or weak correlation between degrees of neighboring nodes. The structure can take on either a strictly hierarchical shape with a dominating hub in the center or act as a collection of several strongly or weakly connected local hierarchies. This can be quantified by a scale-free metric [Li et al., 2005], shown in Equation 2 for a graph $G = (V,E)$, where s_{max} is a normalization coefficient that denotes the maximum $s(H)$ taken over the set of all graphs H that follow the same degree distribution as G. A low value of $S(G)$ would characterize a *scale-rich* network and a high value a *scale-free* network.

$$\begin{aligned} s(G) &= \sum_{(u,v)\in E} deg(u)\cdot deg(v) \\ S(G) &= \frac{s(G)}{s_{max}} \end{aligned} \tag{2}$$

The emergence of hubs has a profound effect on many network properties and it is often important to focus the analysis on hub points and analyze them more thoroughly. For instance, the heterogeneity of the degree distribution makes the hub points follow a slightly different spectral dimension [Hwang et al., 2013]. The need for visualization in the process of analyzing scale-free networks has given rise to some novel graph visualization techniques [Jia et al., 2008].

The reason why scale-free networks are relevant as related work to the study of hubs in k-nearest neighbor graphs of high-dimensional data is that one of the main results of Radovanović [Radovanović, 2011] was to show that the neighbor degree distribution in randomly generated data of increasing dimensionality asymptotically approaches a power law, i.e. the kNN graph becomes scale-free. This has been demonstrated for a class of different probability distributions of data points.

1.4.3 Hubs as Very Frequent Nearest Neighbors

1.4.3.1 Curse of Dimensionality

The term "Curse of Dimensionality" was first coined by Bellman [Bellman, 1961] to denote difficulties that come up when working with high-dimensional data. It is an umbrella term for various problems that may arise, not only in machine learning and data mining, but also in numerical analysis, sampling, Bayesian statistics and combinatorics.

When the dimensionality of the data increases, so does the containing volume. This leads to sparsity of the data and sparse data is difficult to handle. It is difficult to obtain reliable density estimates. The amount of data required to derive statistically sound estimates rises exponentially with the number of dimensions. This means that in cases of high data dimensionality, there is virtually never enough data to overcome these difficulties and even if there were, processing all of it would not be feasible.

The curse of dimensionality poses new challenges for similarity search [Chávez and Navarro, 2001][Chavez and Navarro, 2003][Yianilos, 2000]. Classification tends to become more difficult and new approaches are required [Serpen and Pathical, 2009]. Kernel methods are also subject to the dimensionality curse, as their performance tends to be impaired [Bengio et al., 2005][Evangelista et al., 2006]. High dimensionality is known to cause problems for privacy-preserving data randomization [Aggarwal, 2007][Aggarwal, 2005]. Even the neural networks suffer from problems stemming from high data dimensionality [Verleysen et al., 2003].

While considering the negative effects of high dimensionality, it is important to make a clear distinction between the embedding dimensionality and intrinsic dimensionality, as the two are not always equal. The embedding dimensionality represents the number of features in the data representation, while the intrinsic dimensionality is defined as the minimal number of features needed to fully represent the data.

In many applications, data representations contain at least some attributes that are not mutually independent and some correlation and redundancy can be observed. The intrinsic dimensionality of such data is then lower than the number of features currently used. Many different approaches to estimating the intrinsic dimensionality of the data have been proposed [Pettis et al., 1979][Gupta and Huang, 2012][Farahmand and Szepesvári, 2007][Carter et al., 2010][Rozza et al., 2012][Camastra and Vinciarelli, 2002]. Some machine learning methods work under the assumption that data can be projected onto a lower-dimensional manifold and that learning might be easier after such dimensionality reduction [Talwalkar et al., 2008][Zhang et al., 2012b].

Apart from sparsity, one of the main problems in analyzing high-dimensional data stems from what is known as the concentration of distances. The distance concentration is also responsible for difficulties in nearest-neighbor search and partly related to the hubness phenomenon.

Distance concentration is a counterintuitive property of many intrinsically high-dimensional datasets [Pestov, 2000][François et al., 2007]. In high-dimensional data, the relative contrast between distances calculated on pairs of examples sampled from the same distribution decreases. This tendency makes it hard to distinguish between close and distant pairs of points, which is essential in many practical applications. The fact that the difference between nearest and farthest neighbors sometimes vanishes in high-dimensional spaces has lead some researchers to question the very notion of nearest neighbors in high-dimensional data [Beyer et al., 1999][Durrant and Kabán, 2009][Hinneburg et al., 2000]. However, it is usually still possible to distinguish between points originating from different, non-overlapping distributions.

If we are given a finite sample $S \subset \mathbb{R}^N$ denote by d_M the maximal observed distance from a fixed query point to points in the sample and by d_m the minimal observed distance, i.e. $d_M = \max_{x_i,x_j \in S} d(x_i,x_j)$ and $d_m = \min_{x_i,x_j \in S} d(x_i,x_j)$. Let $\rho_d^n = \frac{d_M - d_m}{d_m}$. This quantity is referred to as the relative contrast (RC). Alternatively, we can also observe the relative contrast on distances between all pairs of points. The distance concentration phenomenon means that as the dimensionality increases, the contrast goes to zero. In other words, $\lim_{n\to\infty} \rho_d^n = 0$. Of course, this is an asymptotic result and real world data is not expected to exhibit such severe concentration.

The problem arises when the expected value for the distance increases with increasing dimensionality, while the variance remains constant.

There is an ongoing research on determining the exact conditions for stability of distance functions in high dimensional data [Hsu and Chen, 2009][Kabán, 2012]. The problem is that many standard metrics suffer from severe distance concentration, as for example the widely used Euclidean distance. Same can be said for other members of the Minkowski distance family and it also holds for fractional distances, though to a somewhat lower extent [François et al., 2007]. Redesigning metrics for high-dimensional data analysis has become an important topic [Aggarwal, 2001]. It has been argued that unbounded distance measures for which the expectation does not exist are more stable and should be preferred in such cases [Jayaram and Klawonn, 2012]. Another approach is to start with a common primary distance measure and proceed by defining a secondary distance measure that takes the primary measure as its functional parameter. Shared neighbor distances represent a class of secondary distance measures that is often used in high-dimensional clustering applications [Houle et al., 2010][Jarvis and Patrick, 1973][Yin et al., 2005][Moëllic et al., 2008]. Some other secondary measures have recently been proposed, like mutual proximity [Schnitzer et al., 2011].

Distance concentration has been shown to impede certain sorts of distance-preserving dimensionality reduction methods, which has lead to the design of some new techniques [Lee and Verleysen, 2011]. In general, some sorts of data mining techniques are more and some are less susceptible to the distance concentration phenomenon [Kabán, 2011].

1.4.3.2 Emergence of Hubs in kNN Graphs

Before proceeding with an overview of the related work on k-nearest neighbor hubs, it is imperative that all the relevant terms and concepts are properly defined in the context that is being discussed. Some formal notation must also be introduced.

Let $D = (x_1, y_1), (x_2, y_2), ..(x_n, y_n)$ be the data set, where each $x_i \in R^d$. The x_i are feature vectors residing in some high-dimensional Euclidean space, and $y_i \in c_1, c_2, ..c_C$ are instance labels. Denote by $D_k(x_i)$ the k-neighborhood defined by the nearest neighbors of x_i. Also, let $N_k(x_i)$ be the number of k-occurrences (occurrences in k-neighbor sets) of x_i and by $N_{k,c}(x_i)$ the number of such occurrences in neighborhoods of elements from class c. The phenomenon of *hubness* is induced by the skewed distribution of the neighbor occurrence frequency N_k.

Def. *Absolute total hubness* of a data point x_i equals its total observed neighbor occurrence frequency $N_k(x_i)$. Normalizing either by the data size or the maximum observed absolute hubness yields the *relative total hubness* of the data point x_i, i.e. $\frac{N_k(x_i)}{n}$ or $\frac{N_k(x_i)}{\max_{i \leq n} N_k(x_i)}$. We will refer to the absolute total hubness of particular points simply as their *hubness*, for short, as the relative hubness scores are rarely used in practice.

Def. *Absolute good hubness* of a data point x_i equals the number of its neighbor occurrences where there is no label mismatch between x_i and its reverse nearest neighbor. In other words, $GN_k(x_i) = |x_j : x_i \in D_k(x_j) \wedge y_i = y_j|$.

Def. *Absolute bad hubness* of a data point x_i equals the number of its neighbor occurrences where there is a label mismatch between x_i and its reverse nearest neighbor. In other words, $BN_k(x_i) = |x_j : x_i \in D_k(x_j) \wedge y_i \neq y_j|$.

Therefore, the absolute total hubness of a neighbor point x_i is a sum of its good and bad hubness and it can also be further decomposed into the class-specific hubness scores.

$$N_k(x_i) = GN_k(x_i) + BN_k(x_i) = \sum_{c \in C} N_{k,c}(x_i) \tag{3}$$

The motivation behind the basic definitions is clear, as the point hubness corresponds to how much of a hub a point is. The more it occurs, the higher its degree in the k-nearest neighbor graph is going to be. Label mismatches in k-neighbor sets and bad hubness in general constitute semantic similarity breaches that usually reflect negatively on data modeling and analysis by using the k-nearest neighbor methods. Trivially, when performing k-nearest neighbor classification, we would like to have each point surrounded mostly by the neighbors of its own class, at least for smaller values of k. This is not the same as pure density estimation, as the primary goal is to achieve the highest possible classification accuracy by avoiding all possible misclassifications.

Apart from the point-wise hubness scores, the total hubness of the entire dataset will also be of interest as a quantity that allows for characterization of different types of datasets and comparisons between them.

Def. *Hubness* of a dataset D is defined as the third standard moment (skewness) of the neighbor occurrence degree distribution and will be denoted by $SN_k = \frac{\frac{1}{n}\sum_{i=1}^{n}(N_k(x_i)-k)^3}{(\frac{1}{n}\sum_{i=1}^{n}(N_k(x_i)-k)^2)^{3/2}}$.

Therefore, high hubness is equivalent to high neighbor k-occurrence skewness. An example of the rising hubness in Gaussian high-dimensional data can be seen in Figure 6. The low-dimensional data forms a familiar bell-shaped neighbor occurrence distribution curve, while the high-dimensional data form a *fat-tailed* distribution that slowly takes the form of a power law.

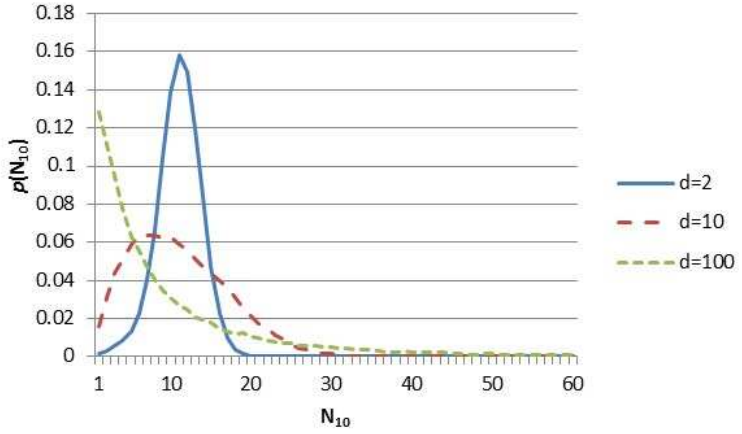


Figure 6: The change in the distribution shape of 10-occurrences (N_{10}) in i.i.d. Gaussian data with increasing dimensionality when using the Euclidean distance. The graph was obtained by averaging over 50 randomly generated data sets. Hub-points exist also with $N_{10} > 60$, so the graph displays only a restriction of the actual data occurrence distribution.

The term *hubness* is used in both of its meanings throughout the text. If it is a quantity tied to a specific point, it denotes the neighbor occurrence frequency. If it is a quantity describing a dataset, it denotes the skewness of the k-occurrence distribution. This is consistent with the notation in earlier work on the topic of k-nearest neighbor hubness [Radovanović, 2011][Radovanović et al., 2009][Radovanović et al., 2010a][Radovanović et al., 2010b][Radovanović et al., 2010][Nanopoulos et al., 2009]. For the sake of consistency, the same definitions will be used when it comes to defining hubs, as well.

Def. *A Hub* in a k-nearest neighbor graph is a point that occurs much more frequently than other points in that its observed occurrence frequency exceeds the mean by more than two standard deviations. More specifically, $\text{hubs}(D) = \{x_i : N_k(x_i) \geq k + 2 \cdot \text{stdev}(N_k)\}$, where $\text{stdev}(N_k) = \sqrt{\frac{\sum_{i=1}^{n}(N_k(x_i)-k)^2}{n}}$.

Points that occur very infrequently will be informally denoted as *anti-hubs*, though the exact definition of what constitutes an anti-hub might differ based on the context and task at hand. Similarly, let *orphans* be the points that never occur as neighbors.

The *hubness* phenomenon is closely related to other aspects of the dimensionality curse, including sparsity and distance concentration. One of the main results of Radovanović [Radovanović, 2011] was to show that the neighbor degree distribution in randomly generated data of increasing dimensionality asymptotically approaches a power law. Furthermore, it was suggested [Radovanović et al., 2009][Radovanović et al., 2010a][Radovanović et al., 2010b][Berenzweig, 2007] that hubs are expected to emerge in inherently high-dimensional data. In other words, they are not to be considered an artefact of a particular feature representation or a probability distribution.

The distribution of nearest neighbor occurrences has been a topic of study for a while among the mathematical probability community and several theoretical asymptotic tendencies have been proven for certain types of distributions [Tversky et al., 1983][Yao and G.Simons, 1996][Tversky and Hutchinson, 1986][Maloney, 1983][Newman and Rinott, 1985] [C. M. Newman and Tversky, 1983]. In the majority of studied settings (the Poisson process, d-dimensional torus), the distribution of N_1 was shown to converge to a Poisson distribution with $\lambda = 1$ when the Euclidean distance is used for measuring dissimilarity and when the number of points and the number of dimensions go to infinity. The Poisson distribution is a limit case of the binomial distribution with infinite granularity. In the more general case of the Poisson process, the distribution of N_k would also converge under the same conditions to the Poisson distribution with $\lambda = k$. The shape of the Poisson distribution would

thus suggest that the emergence of hubs is not to be expected under such conditions, which is opposite to what has actually been observed in practical analysis of high-dimensional datasets [Radovanović et al., 2010a]. However, a careful interpretation of the results is required. It was suggested [Tversky et al., 1983][Newman and Rinott, 1985] that an emergence of a small number of hub-points is to be expected when the intrinsic dimensionality of the feature space is high relative to the number of points. This is precisely the setup that is frequently encountered when analyzing real-world high-dimensional sparse datasets with limited sample sizes.

Nearest neighbor hubs have first been observed while examining music recommendation systems [Aucouturier and Pachet, 2004][Aucouturier, 2006]. Music recommendation is based on suggesting potentially interesting songs to users, according to what they had already been listening to. Similarity search for a particular query song is among the basic required system functions. During the analysis, it was determined that some songs end up being very frequently retrieved by the system, more than what could be explained by the semantics of the data. The calculated similarity measure between the songs did not capture the perceptual similarity well, which suggested that the problem might stem from an improper feature representation or an inappropriate similarity measure. Indeed, the symmetric Kullback-Leibler divergence [Kullback and Leibler, 1951] that was used in some earlier recommendation systems does suffer from high hubness. However, the phenomenon remains present to some degree even if the data representation or the metric are changed. Several recent papers have proposed and evaluated different techniques for reducing hubness in music recommendation systems and improving system performance [M. and A., 2012][Flexer A., 2012][Schnitzer et al., 2011][Gasser M., 2010].

Audio data in general has also been shown to fall prey to the hubness phenomenon. Like in music recommendation, speaker verification is also prone to experiencing problems due to the curse of dimensionality. Speaker verification systems compute the distance between the audio generated by the speakers and the generated statistical speaker models. The 'Doddington zoo' effect [Doddington et al., 1998] describes the pitfalls of automated speaker recognition by defining four types of speakers, based on the system performance. The average users for which the system exhibits normal, acceptable behavior are known as *sheep*. Users that are especially difficult to recognize are known as *goats*. Speakers that are able to easily impersonate other users are known as *wolves*. Speakers that are particularly easy to imitate are called *lambs*. It soon becomes apparent that the terms of wolves and goats correspond well to the concepts of hubs and anti-hubs (orphans). This has been discussed and evaluated in a recent paper [Dominik Schnitzer, 2012].

Another form of data that exhibits substantial hubness in the kNN topology is the PPI (protein-protein interaction) data [Patil et al., 2010]. The significance of hub proteins has been emphasized on many occasions, as it was shown that the hub proteins from the PPI networks of different biological entities are essential proteins that play significant roles in the complex biochemical processes that happen in the body or the cells [He and Zhang, 2006][Ekman et al., 2006][Batada et al., 2006]. This means that any advances in understanding the mechanisms behind the hubness phenomenon might eventually be beneficial to the bioinformatical research as well.

In the k-nearest neighbor topology, different k-neighborhoods have different diameters, depending on the local data density. If the local density is higher in x, the radius of the minimal hyper-sphere containing $D_k(x)$ centered at x would be smaller. A slightly different topological examination might involve considering fixed diameter neighborhoods instead of the k-nearest neighbor sets. Geometric graphs obtained in such a way are usually referred to as the ε-neighborhood graphs [Penrose, 2003]. Unlike the k-nearest neighbor graphs, the neighbor occurrence frequency in ε-neighborhood graphs does not exhibit high skewness and the phenomenon of hubness is not present [Radovanović, 2011]. However, these graphs are not easy to use in data analysis, as it often proves to be difficult to set a global ε threshold.

If there are great differences in densities over the data space, it might even prove impossible to assure that each neighborhood is non-empty while not breaching the locality assumption around data points in higher density regions. This is why the k-nearest neighbor methods are much more frequently used in practice, even though the phenomenon of hubness remains a constant concern in high-dimensional data.

1.5 Aims and Hypotheses

Several studies have previously shown hubness to be highly detrimental in various practical applications. Nevertheless, that research was mostly theoretical, focusing mainly on the mathematical mechanisms behind the emergence of hubs. Useful as that may be, little was actually done on improving the system performance under the assumption of hubness in high-dimensional data.

The primary goal of the research presented in this thesis was to design a set of hubness-aware algorithms, in order to show that the knowledge and understanding of the underlying data hubness can be exploited for increasing the effectiveness of ML/DM methods.

During the course of our research, we have made several hypotheses about the role that hub-points play in high-dimensional data analysis. The main hypotheses are listed below. In order to clarify the context, a comment on the prior state of the art is given in each case, outlining the information that was already known prior to the research described in this thesis.

Hypothesis 1: *Hubs can be effectively used as cluster prototypes when clustering intrinsically high-dimensional data.*

Prior state of the art: It was already known that there exists a strong positive correlation between *anti-hubs* and outliers, points that are isolated and located far away from cluster means and regions of high density in the feature space [Radovanović et al., 2010a]. Additionally, it was known that hub points cluster badly on average. This had been attributed to their low average inter-cluster distances, that cause them to act as 'links' between different high-dimensional data clusters. It was conjectured that the existence of hubs hampers clustering, as they reduce the average cluster separation.

Aim: We have taken a novel perspective on the role of hubs in high-dimensional data clustering. Our aim was to show that the hubness information can be exploited for improving the effectiveness of clustering algorithms in intrinsically high-dimensional data by using hubs as prototypes that represent local data clusters. In order to test our hypothesis, we have proposed several novel clustering methods based on the stated assumptions.

Hypothesis 2: *Class-conditional neighbor occurrence models learned on the training data can improve the effectiveness of k-nearest neighbor classification in intrinsically high-dimensional data.*

Prior state of the art: Reducing the voting weights of bad hubs has been shown to be beneficial for kNN classification in hw-kNN [Radovanović et al., 2009]. However, the existing algorithm did not utilize the class-specific neighbor occurrence information and was based only on bad occurrence counts (label mismatches).

Aim: We conjectured that almost all neighbor occurrences carry some class-discriminative information, whether they are considered good or bad in terms of label math/mismatch. In order to evaluate whether this is true, we have proposed to use the class-conditional neighbor occurrence models in order

to decompose bad hubness into class-conditional neighbor occurrence frequencies. Our goal was to use the neighbor occurrence models as a basis for designing hubness-aware kNN classification methods.

Hypothesis 3: *In class imbalanced intrinsically high-dimensional data, minority hubs often induce label mismatches and can cause a severe misclassification of points from the majority class.*

Prior state of the art: Learning under class imbalance is a well known and important topic, as many real-world problems are known to be highly imbalanced. The standard working hypothesis is as follows: *Due to an average relative difference in densities in class overlap regions, the majority class often causes a severe misclassification of points from the minority class* [He and Garcia, 2009]. This is a reasonable assumption that takes into account the generalization bias of many classification models and it certainly holds in low or medium-dimensional data. However, this is exactly the opposite of the hypothesis that we have made for the intrinsically high-dimensional case.

Aim: Our goal was to examine and better understand the role of the minority hubs in learning under class imbalance. We have selected a series of real-world and synthetic datasets that have been shown to exhibit significant hubness and have performed a thorough experimental evaluation. Furthermore, our goal was also to compare how different types of k-nearest neighbor classifiers handle the negative effects of minority hubs and to see if the proposed class-conditional neighbor occurrence models can be used to improve the classification performance.

Hypothesis 4: *Class-conditional neighbor occurrence models can be used to capture the instance selection bias during data reduction and exploit this information for improving classification performance.*

Prior state of the art: Most data mining and machine learning applications are working with large quantities of data, which needs to be processed efficiently and effectively. However, many state-of-the-art algorithms in terms of the overall performance do not scale well with data size, as they perform computationally intensive analysis. This is why data reduction / instance selection is common in practice, so that the model is inferred from a sub-sample of the original data set [Liu, 2010]. In complex data, the information loss usually incurs a downgrade in accuracy. This is why the sampling is usually not done randomly, there exist many complex instance selection methods. Each of these methods encapsulates a certain bias, a selection criterion.

Aim: An implicit assumption contained in our hypothesis is that the hubness properties of the selected prototypes on the test data can be different when the whole training set is used and when only the prototypes are used as potential neighbor points. We have tested several different instance selection strategies over a wide range of intrinsically high-dimensional datasets from different domains, in order to test whether this is indeed true. After the initial tests, we have proposed to use the hubness-aware kNN classification algorithms based on the class-conditional neighbor occurrence models in order to model the changes in hubness induced by the instance selection bias and increase the overall classification performance.

Hypothesis 5: *The information contained in the class-conditional neighbor occurrence models can be used for metric learning in order to improve the semantic consistency in kNN sets, as well as the overall kNN classification performance.*

Prior state of the art: Secondary metrics have often been used in the past in order to circumvent the distance concentration phenomenon. Shared-neighbor distances are a widely used family of secondary distances that is often used in high-dimensional data analysis [Houle et al., 2010]. In the standard shared-neighbor approach, a similarity between two points is determined by the cardinality of the intersection of their kNN sets in the original (primary) metric space. Usually, a large value of k is used.

Aim: Our aim was to demonstrate the potential usefulness of the information contained in the class-conditional neighbor occurrence models by extending and improving the basic shared-neighbor distance framework. Our main idea was to assign different weights to different points while counting the size of the kNN set intersections and to base these weights on the properties of the neighbor occurrence profiles of the shared neighbor points. Hubs are shared by more pairs of points, so their discriminative potential is somewhat lower and we have assigned them a lower weight. Additionally, bad hubs contribute to increasing the average inter-class similarity, so they were assigned a low weight, in order to improve the separation between different classes in the data.

1.6 Scientific Contributions

The major scientific contributions of this thesis can be grouped based on their relation to the 5 major hypotheses that we have examined, as previously outlined in Section 1.5. Topic-wise, these correspond to the problems of data clustering, classification, class imbalance, instance selection and metric learning, respectively.

SC 1. Clustering: Our work on exploring the role of hubs in clustering high-dimensional data [Tomašev et al., 2011d][Tomašev et al., 2013c] was the first to try and explicitly exploit hubs for high-dimensional **data clustering**.

SC 1.1: We have shown that the neighbor occurrence frequency is a good measure of **local cluster centrality** in high-dimensional Gaussian distributions.

SC 1.2: We have proposed three new clustering algorithms that use hubs as cluster prototypes and hubness as a measure of centrality: K-hubs, global hubness-proportional clustering (GHPC) and global hubness-proportional K-means (GHPKM).

SC 1.3: It was determined that the proposed methods achieve their improvements primarily by **improving the clustering quality of hub points**. This is an important novelty, as previous work had suggested that hubs often cluster badly, almost as bad as outliers [Radovanović, 2011].

SC 2. Classification: We have proposed several novel kNN **classification algorithms** designed specifically for high-dimensional data analysis.

SC 2.1: A set of hubness-based fuzzy measures was proposed, which were used to infer a novel hubness-based fuzzy k-nearest neighbor classification method (h-FNN) [Tomašev et al., 2011b][Tomašev et al., 2013b]. This method was the first one to use the *class-conditional* hubness estimates derived from the neighbor occurrence model.

SC 2.2: We have also proposed the hubness information k-nearest neighbor algorithm (HIKNN) as an extension of the h-FNN framework [Tomašev and Mladenić, 2012]. The novelty in HIKNN was its ability to increase the influence of locally relevant points on the classification outcome.

SC 2.3: A Bayesian approach to interpreting neighbor occurrences was examined and a novel hubness-aware classifier was proposed, the naive hubness-Bayesian k-nearest neighbor (NHBNN) [Tomašev et al., 2011c].

SC 3. Class imbalance: This thesis also presents the first results aimed at exploring the link between **class imbalance** in high-dimensional data and underlying data hubness.

SC 3.1: The performed experiments suggest that **hubs of the minority classes** often induce severe misclassification of the points that belong to the majority class. This phenomenon will be referred to as *the curse of minority hubs*.

SC 3.2: The proposed hubness-aware k-nearest neighbor classification approaches have been shown to decrease the negative influence of the bad minority hubs and improve the classification performance in class imbalanced high-dimensional data.

SC 4. Instance selection: The role of hubs in **instance selection** has been given comparatively little attention in the past and this thesis presents one of the first in-depth discussions on the topic.

SC 4.1: A comparison between a series of standard instance selection strategies has revealed that some of them are more and some are less biased towards selecting hubs as prototypes.

SC 4.2: Some selection strategies have been shown to consistently overestimate and some underestimate the bad hubness of the selected prototypes, which results in biased and less reliable kNN classification models.

SC 4.3: We have proposed a novel hubness-aware instance selection classification framework based on using an **unbiased hubness estimator** in conjunction with the proposed hubness-aware kNN classification algorithms based on the class-conditional neighbor occurrence models.

SC 5. Metric learning: We have extended the shared-neighbor similarity framework in order to include the relevant hubness information.

SC 5.1: We have proposed a novel **secondary shared-neighbor similarity measure**, $simcos_s$ [N. and D., 2012][Tomašev and Mladenić, 2013], by incorporating a hubness-aware instance weighting scheme into the standard shared-neighbor similarity measure, $simcos_s$ [Houle et al., 2010].

SC R. Other/Remaining: The remaining scientific contributions are related to some specific practical applications of the developed hubness-aware methodology.

SC R.1: A correlation between different **feature representations** and the arising data hubness was examined for image data in context of object recognition [Tomašev et al., 2011a].

SC R.2: It was demonstrated that the bad hubness information could be potentially useful for detecting anomalous behaviour in oceanographic sensor measurements.

SC R.3: The role of hubs in **cross-lingual document retrieval** was also examined [Tomašev et al., 2013d] and it was shown that there exists a high correlation between hubness of different language representations of aligned document corpora. This information was exploited for designing a new instance weighting scheme for Canonical Correlation Analysis, which improved the quality of document retrieval.

SC R.4: Learning from past occurrences was employed in order to design a novel hubness-aware self-adaptive **re-ranking** procedure that was successfully used in improving the system performance in bug duplicate detection. The notion of *temporal hubness* was introduced, as hubness of bug reports changes over time with changing user interest and report distribution.

The majority of listed contributions represent first steps towards successfully exploiting the hubness information in high-dimensional data analysis. The experimentally observed improvements suggest that taking data hubness into account might be beneficial for many different types of machine learning applications.

1.7 Thesis Structure

The thesis starts with a brief outlining of the general background and motivation, proceeding quickly to describing the context of the problems related to hubness in high-dimensional data. The emergence of hubs is high-dimensional k-nearest neighbor topologies is discussed and an overview of previous work done in this field is provided.

Chapter 2 deals with our newly proposed hubness-aware data analysis algorithms. It is composed of three separate thematic sections. Section 2.1 starts by examining an interplay between hubness and clustering and covers the scientific contributions listed in (SC 1.1-1.3). The results presented in the paper *The Role of Hubness in Clustering High-dimensional Data* [Tomašev et al., 2011d][Tomašev et al., 2013c] are shown. The original paper is inserted in its entirety and the section follows its structure. Section 2.2 deals with hubness-aware classification and the scientific contributions (SC 2.1-2.3). The proposed hubness-aware classifiers, hubness-based fuzzy k-nearest neighbor (h-FNN) [Tomašev et al., 2011b][Tomašev et al., 2013b], hubness-information k-nearest neighbor (HIKNN) [Tomašev and Mladenić, 2012] and naive hubness-Bayesian k-nearest neighbor (NHBNN) [Tomašev et al., 2011c] are discussed in great detail. Two journal papers are presented here, *Hubness-Based Fuzzy Measures for High-Dimensional k-Nearest Neighbor Classification*[Tomašev et al., 2013b] that analyzes the h-FNN classifier and *Nearest Neighbor Voting in High-dimensional Data: Learning from Past Occurrences* [Tomašev and Mladenić, 2012] that analyzes HIKNN. This is followed by subsections 2.2.3 and 2.2.4 dealing with instance selection (SC 4.1-4.3) and class imbalance (SC 3.1-3.2), respectively. Subsection 2.3 discussed the role of hubs in metric learning (SC 5.1) and presents the results of the journal paper titled *Hubness-aware Shared-neighbor Distances for High-dimensional k-nearest Neighbor Classification* [Tomašev and Mladenić, 2013].

Chapter 3 deals with the applications of hubness-aware methodologies in practical data analysis systems. Section 3.1 shows the potential uses of hubness-aware approaches in object recognition systems from image data (SC R.1). Image Hub Explorer visualization application is presented and its use cases are discussed. Oceanographic survey sensor data are analyzed in Section 3.2 where hub points are used to mark potentially anomalous sensor signals (SC R.2). The beneficial use of hubness-aware instance weighting is demonstrated in cross-lingual document retrieval in Section 3.3, where a new way of forming the common semantic representation is proposed (SC R.3). Finally, Section 3.4 demonstrates how temporal hubness information can be successfully exploited for re-raking in a bug duplicate detection system (SC R.4).

Chapter 4 concludes the thesis by looking back at the main scientific contributions and clearly stating objectives and directions for future work and research on the topic of hubness in high-dimensional data.

2 Hubness-aware Algorithms

The almost ubiquitous presence of hubs and hubness in inherently high-dimensional data [Radovanović, 2011] impedes the performance of many classical similarity-based data mining and machine learning methods and therefore outlines a need for novel algorithms, capable of achieving good results under the burden of the dimensionality curse. These kNN-based methods will be referred to as being *hubness-aware*, since their design directly embodies the robustness with respect to handling difficult hub neighbor points. In this chapter, we will show several such methods that we have recently developed and applied to different domains, including clustering, classification and metric learning.

The methods described in this chapter represent the core of this thesis. The experimental evaluation has shown them to be quite robust and reliable when analyzing high-dimensional data under the assumption of hubness. They are not domain-specific and can be utilized by experts in various fields in order to avoid the negative aspects of the dimensionality curse in k-nearest neighbor methods.

2.1 Hubness-aware Clustering

This Section presents the paper titled *The Role of Hubness in Clustering High-dimensional Data* by Nenad Tomašev, Miloš Radovanović, Dunja Mladenić and Mirjana Ivanović. The paper was first presented at the PAKDD (Pacific Asian Knowledge Discovery and Data Mining) conference in Shenzhen, China, in April 2011 [Tomašev et al., 2011e] and was awarded the **best research paper runner-up award** for its scientific contributions to the field of clustering and innovative algorithm design. The paper was later extended by including new experiments and analysis of the observed improvements and was published in the IEEE Transactions on Knowledge and Data Engineering journal in 2013 [Tomašev et al., 2013c].

The paper was based on an interpretation of the theoretical results behind the emergence of hubness [Radovanović et al., 2010a][Radovanović, 2011] where it was hypothesized that the points closer to the centers of hyper-spheres that high-dimensional data approximately lies upon have a greater tendency towards becoming hubs in the k-nearest neighbor topology of the data. The idea behind the approaches taken in the paper is that hubs can be used to model the local cluster centers and exploited within a center-based clustering framework.

In order to show the potential of such an approach, hubs were compared to cluster centroids and medoids and the comparisons were performed mostly from within the widely used K-means++ clustering framework [Arthur and Vassilvitskii, 2007].

One of the main experimental results of the paper is that hubness was evaluated as a much better measure of local centrality than density, when it comes to evaluating centrality in high-dimensional datasets. The correlation between hubness and density decreases with increasing dimensionality, as the k-nearest neighbor density estimates become more and more affected by the curse of dimensionality.

The paper proposes several hubness-based clustering approaches and evaluates their performance over a wide spectrum of high-dimensional clustering tasks. The proposed algorithms are: K-hubs, global hubness-proportional clustering (GHPC) and global hubness-

proportional K-means (GHPKM). Local and global cluster hubness estimates were compared and the global estimates were shown to be better in the experimental evaluation.

The evaluation shows that the proposed stochastic hubness-based algorithms clearly outperform the K-Means++ baseline in almost all testing setups. The improvements in the clustering quality index were thoroughly analyzed and it was determined that they stem either from avoiding to prematurely converge to local optima or from increasing the clustering quality index of hub-points. This is very important, as earlier work has shown that hubs cluster badly on average, almost as bad as outliers and anti-hubs [Radovanović, 2011].

The algorithms proposed in the paper have an interesting property that their effectiveness increases with the inherent dimensionality of the data. They are not appropriate for clustering in the low-dimensional feature spaces. For them, high dimensionality is not a curse, but a blessing.

The current versions of the algorithms are not able to properly handle clusters of irregular shape and will probably need to be extended by including the kernel trick or some other form of non-linearity. Shared-neighbor clustering approaches are an alternative [Yin et al., 2005].

The Role of Hubness in Clustering High-Dimensional Data

Nenad Tomašev, Miloš Radovanović, Dunja Mladenić, and Mirjana Ivanović

Abstract—High-dimensional data arise naturally in many domains, and have regularly presented a great challenge for traditional data-mining techniques, both in terms of effectiveness and efficiency. Clustering becomes difficult due to the increasing sparsity of such data, as well as the increasing difficulty in distinguishing distances between data points. In this paper we take a novel perspective on the problem of clustering high-dimensional data. Instead of attempting to avoid the curse of dimensionality by observing a lower-dimensional feature subspace, we embrace dimensionality by taking advantage of inherently high-dimensional phenomena. More specifically, we show that hubness, i.e., the tendency of high-dimensional data to contain points (hubs) that frequently occur in k-nearest neighbor lists of other points, can be successfully exploited in clustering. We validate our hypothesis by demonstrating that hubness is a good measure of point centrality within a high-dimensional data cluster, and by proposing several hubness-based clustering algorithms, showing that major hubs can be used effectively as cluster prototypes or as guides during the search for centroid-based cluster configurations. Experimental results demonstrate good performance of our algorithms in multiple settings, particularly in the presence of large quantities of noise. The proposed methods are tailored mostly for detecting approximately hyperspherical clusters and need to be extended in order to properly handle clusters of arbitrary shapes.

Index Terms—Clustering, curse of dimensionality, nearest neighbors, hubs.

1

1 Introduction

Clustering in general is an unsupervised process of grouping elements together, so that elements assigned to the same cluster are more similar to each other than to the remaining data points [1]. This goal is often difficult to achieve in practice. Over the years, various clustering algorithms have been proposed, which can be roughly divided into four groups: *partitional*, *hierarchical*, *density-based*, and *subspace* algorithms. Algorithms from the fourth group search for clusters in some lower-dimensional projection of the original data, and have been generally preferred when dealing with data that is high-dimensional [2], [3], [4], [5]. The motivation for this preference lies in the observation that having more dimensions usually leads to the so-called *curse of dimensionality*, where the performance of many standard machine-learning algorithms becomes impaired. This is mostly due to two pervasive effects: the empty space phenomenon and concentration of distances. The former refers to the fact that all high-dimensional data sets tend to be sparse, because the number of points required to represent any distribution grows exponentially with the number of dimensions. This leads to bad density estimates for high-dimensional data, causing difficulties for density-based approaches. The latter is a somewhat counterintuitive property of high-dimensional data representations, where all distances between data points tend to become harder to distinguish as dimensionality increases, which can cause problems with distance-based algorithms [6], [7], [8], [9].

The difficulties in dealing with high-dimensional data are omnipresent and abundant. However, not all phenomena which arise are necessarily detrimental to clustering techniques. We will show in this paper that *hubness*, which is the tendency of some data points in high-dimensional data sets to occur much more frequently in k-nearest neighbor lists of other points than the rest of the points from the set, can in fact be used for clustering. To our knowledge, this has not been previously attempted. In a limited sense, hubs in graphs have been used to represent typical word meanings in [10], which was not used for data clustering. A similar line of research has identified essential proteins as hubs in the reverse nearest neighbor topology of protein interaction networks [11]. We

- *N. Tomašev and D. Mladenić are with the Jožef Stefan Institute, Artificial Intelligence Laboratory and Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: {nenad.tomasev, dunja.mladenic}@ijs.si*
- *M. Radovanović and M. Ivanović are with the University of Novi Sad, Department of Mathematics and Informatics, Trg D. Obradovića 4, 21000 Novi Sad, Serbia. E-mail: {radacha, mira}@dmi.uns.ac.rs*

1. DOI:http://doi.ieeecomputersociety.org/10.1109/TKDE.2013.25

have focused on exploring the potential value of using hub points in clustering by designing hubness-aware clustering algorithms and testing them in a high-dimensional context. The hubness phenomenon and its relation to clustering will be further addressed in Section 3.

There are two main contributions of this paper. First, in experiments on synthetic data we show that hubness is a good measure of point centrality within a high-dimensional data cluster and that major hubs can be used effectively as cluster prototypes. In addition, we propose three new clustering algorithms and evaluate their performance in various high-dimensional clustering tasks. We compared the algorithms with a baseline state-of-the-art prototype-based method (K-means++ [12]), as well as kernel-based and density-based approaches. The evaluation shows that our algorithms frequently offer improvements in cluster quality and homogeneity. The comparison with kernel K-means [13] reveals that kernel-based extensions of the initial approaches should also be considered in the future. Our current focus was mostly on properly selecting cluster prototypes, with the proposed methods tailored for detecting approximately hyperspherical clusters.

The rest of the paper is structured as follows. In the next section we present the related work, Section 3 discusses in general the phenomenon of hubness, while Section 4 describes the proposed algorithms that are exploiting hubness for data clustering. Section 5 presents the experiments we performed on both synthetic and real-world data. We expect our observations and approach to open numerous directions for further research, many of which are outlined by our final remarks in Section 6.

2 Related Work

Even though hubness has not been given much attention in data clustering, hubness information is drawn from k-nearest-neighbor lists, which have been used in the past to perform clustering in various ways. These lists may be used for computing density estimates, by observing the volume of space determined by the k nearest neighbors. Density-based clustering methods often rely on this kind of density estimation [14], [15], [16]. The implicit assumption made by density-based algorithms is that clusters exist as high-density regions separated from each other by low-density regions. In high-dimensional spaces this is often difficult to estimate, due to data being very sparse. There is also the issue of choosing the proper neighborhood size, since both small and large values of k can cause problems for density-based approaches [17].

Enforcing k-nearest-neighbor consistency in algorithms such as K-means was also explored [18]. The most typical usage of k-nearest-neighbor lists, however, is to construct a k-NN graph [19] and reduce the problem to that of graph clustering.

Consequences and applications of hubness have been more thoroughly investigated in other related fields: classification [20], [21], [22], [23], [24], image feature representation [25], data reduction [23], [26], collaborative filtering [27], text retrieval [28], and music retrieval [29], [30], [31]. In many of these studies it was shown that hubs can offer valuable information that can be used to improve existing methods and devise new algorithms for the given task.

Finally, the interplay between clustering and hubness was briefly examined in [23], where it was observed that hubs may not cluster well using conventional prototype-based clustering algorithms, since they not only tend to be close to points belonging to the same cluster (i.e., have low intra-cluster distance) but also tend to be close to points assigned to other clusters (low inter-cluster distance). Hubs can therefore be viewed as (opposing) analogues of outliers, which have high inter- and intra-cluster distance, suggesting that hubs should also receive special attention [23]. In this paper we have adopted the approach of using hubs as cluster prototypes and/or guiding points during prototype search.

3 The Hubness Phenomenon

Hubness is an aspect of the curse of dimensionality pertaining to nearest neighbors which has only recently come to attention, unlike the much discussed distance concentration phenomenon. Let $D \subset \mathbb{R}^d$ be a set of data points and let $N_k(x)$ denote the number of *k-occurrences* of point $x \in D$, i.e., the number of times x occurs in k-nearest-neighbor lists of other points from D. As the dimensionality of data increases, the distribution of k-occurrences becomes considerably skewed [23]. As a consequence, some data points, which we will refer to as *hubs*, are included in many more k-nearest-neighbor lists than other points. In the rest of the text we will refer to the number of k-occurrences of point $x \in D$ as its *hubness score*. It has been shown that hubness, as a phenomenon, appears in high-dimensional data as an inherent property of high dimensionality, and is not an artefact of finite samples nor a peculiarity of some specific data sets [23]. Naturally, the exact degree of hubness may still vary and is not uniquely determined by dimensionality.

3.1 The Emergence of Hubs

The concentration of distances enables one to view unimodal high-dimensional data as lying approximately on a hypersphere centered at the data distribution mean [23]. However, the variance of distances to the mean remains non-negligible for any finite number of dimensions [7], [32], which implies that some of the points still end up being closer to the data mean than other points. It is well known that points closer to the mean tend to be closer (on average) to

all other points, for any observed dimensionality. In high-dimensional data, this tendency is amplified [23]. Such points will have a higher probability of being included in k-nearest neighbor lists of other points in the data set, which increases their influence, and they emerge as neighbor-hubs.

It was established that hubs also exist in clustered (multimodal) data, tending to be situated in the proximity of cluster centers [23]. In addition, the degree of hubness does not depend on the embedding dimensionality, but rather on the *intrinsic* data dimensionality, which is viewed as the minimal number of variables needed to account for all pairwise distances in the data [23].

Generally, the hubness phenomenon is relevant to (intrinsically) high-dimensional data regardless of the distance or similarity measure employed. Its existence was verified for Euclidean (l_2) and Manhattan (l_1) distances, l_p distances with $p > 2$, fractional distances (l_p with rational $p \in (0, 1)$), Bray-Curtis, normalized Euclidean, and Canberra distances, cosine similarity, and the dynamic time warping distance for time series [22], [23], [28]. In this paper, unless otherwise stated, we will assume the Euclidean distance. The methods we propose in Section 4, however, depend mostly on neighborhood relations that are derived from the distance matrix, and are therefore independent of the particular choice of distance measure.

Before continuing, we should clearly define what constitutes a hub. Similarly to [23], we will say that hubs are points x having $N_k(x)$ more than two standard deviations higher than the expected value k (in other words, significantly above average). However, in most experiments that follow, we will only concern ourselves with one major hub in each cluster, i.e. the point with the highest hubness score.

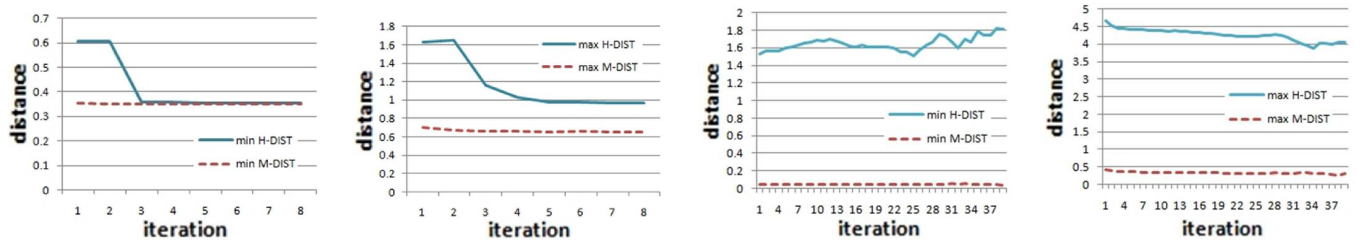
3.2 Relation of Hubs to Data Clusters

There has been previous work on how well high-hubness elements cluster, as well as the general impact of hubness on clustering algorithms [23]. A correlation between low-hubness elements (i.e., *antihubs* or *orphans*) and outliers was also observed. A low hubness score indicates that a point is on average far from the rest of the points and hence probably an outlier. In high-dimensional spaces, however, low-hubness elements are expected to occur by the very nature of these spaces and data distributions. These data points will lead to an average increase in intra-cluster distance. It was also shown for several clustering algorithms that hubs do not cluster well compared to the rest of the points. This is due to the fact that some hubs are actually close to points in different clusters. Hence, they lead to a decrease in inter-cluster distance. This has been observed on real data sets clustered using state-of-the art prototype-based methods, and was identified as a possible area for performance improvement [23]. We will revisit this point in Section 5.4.

It was already mentioned that points closer to cluster means tend to have higher hubness scores than other points. A natural question which arises is: *Are hubs medoids?* When observing the problem from the perspective of partitioning clustering approaches, of which K-means is the most commonly used representative, a similar question might also be posed: *Are hubs the closest points to data centroids in clustering iterations?* To answer this question, we ran K-means++ [12] multiple times on several randomly generated 10000-point Gaussian mixtures for various fixed numbers of dimensions (2, 5, 10, 20, 30, 50, 100), observing the high-dimensional case. We measured in each iteration the distance from current cluster centroid to the medoid and to the strongest hub, and scaled by the average intra-cluster distance. This was measured for every cluster in all the iterations, and for each iteration the minimal and maximal distance from any of the centroids to the corresponding hub and medoid were computed.

Figure 1 gives example plots of how these ratios evolve through iterations for the case of 10-cluster data, using neighborhood size 10, with 30 dimensions for the high-dimensional case, and 2 dimensions to illustrate low-dimensional behavior. The Gaussian mixtures were generated randomly by drawing the centers from a $[l_{bound}, u_{bound}]^d$ uniform distribution (as well as covariance matrices, with somewhat tighter bounds). In the low-dimensional case, hubs in the clusters are far away from the centroids, even farther than average points. There is no correlation between cluster means and frequent neighbors in the low-dimensional context. This changes with the increase in dimensionality, as we observe that the minimal distance from centroid to hub converges to minimal distance from centroid to medoid. This implies that some medoids are in fact cluster hubs. Maximal distances to hubs and medoids, however, do not match. There exist hubs which are not medoids, and vice versa. Also, we observe that maximal distance to hubs drops with iterations, suggesting that as the iterations progress, centroids are becoming closer and closer to data hubs. This already hints at a possibility of developing an iterative approximation procedure.

To complement the above observations and explore the interaction between hubs, medoids and the classic notion of density, and illustrate the different relationships they exhibit in low- and high-dimensional settings, we performed additional simulations. For a given number of dimensions (5 or 100), we generated a random Gaussian distribution centered around zero and started drawing random points from the distribution one by one, adding them sequentially to a synthetic data set. As the points were being added, hubness, densities, distance contrast and all the other examined quantities and correlations between them



(a) Minimal distances, $d = 30$ (b) Maximal distances, $d = 30$ (c) Minimal distances, $d = 2$ (d) Maximal distances, $d = 2$

Fig. 1. Evolution of minimal and maximal distances from cluster centroids to hubs and medoids on synthetic data for neighborhood size 10, and 10 clusters.

(most of which are shown in Fig. 2 and Fig. 3) were calculated on the fly for all the neighborhood sizes within the specified range $\{1, 2, \ldots, 20\}$. The data sets started with 25 points initially and were grown to a size of 5000. The entire process was repeated 20 times, thus in the end we considered 20 synthetic 5-dimensional Gaussian distributions and 20 synthetic 100-dimensional Gaussian distributions. Figures 2 and 3 display averages taken over all the runs.[2] We report results with Euclidean distance, observing similar trends with Manhattan and $l_{0.5}$ distances.

Figure 2 illustrates the interaction between norm, hubness and density (as the measurement, not the absolute term) in the simulated setting. From the definition of the setting, the norm of a point can be viewed as an "oracle" that expresses exactly the position of the point with respect to the cluster center.[3] As can be seen in Fig. 2(a), strong Pearson correlation between the density measurement and norm indicates that in low dimensions density pinpoints the location of the cluster center with great accuracy. In high dimensions, however, density loses it connection with centrality (Fig. 2(b)), and is no longer a good indicator of the main part of the cluster.

Hubness, on the other hand, has some correlation with the norm in low dimensions (Fig. 2(c)), albeit weak. It is in the high-dimensional setting of Fig. 2(d) that hubness begins to show its true potential, as the correlation becomes much stronger, meaning that the hubness score of a point represents a very good indicator of its proximity to the cluster center. In both charts, a trend of slight weakening of correlation can be observed as the number of points increases. Meanwhile, strengthening of correlation can be seen for an increasing number of neighbors k, indicating that larger values of k can be used to adjust to larger data-set sizes. Quite expectedly, density and hubness are well correlated in low dimensions, but not in the high-dimensional setting (Fig. 2(e, f)).

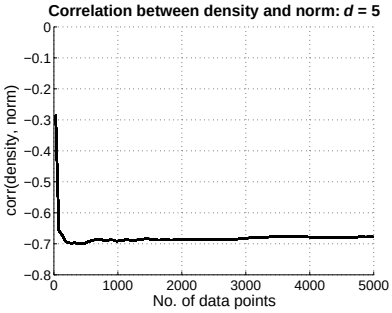
Figure 3 shows the interaction between hubs, medoids and other points in the simulated setting, expressed through distances. Based on the ratio between the average distance to the strongest hub and average distance to the medoid, from Fig. 3(a, b) it can be seen that in high dimensions the hub is equally informative about the location of the cluster center as the medoid, while in low dimensions the hub and medoid are unrelated. At the same time, generally the hub and the medoid are in neither case the same point, as depicted in Fig. 3(c, d) with the distances from hub to medoid which are always far from 0. This is also indicated in Fig. 3(e, f) that shows the ratio between hub to medoid distance and average pairwise distance. In addition, Fig. 3(f) suggests that in high dimensions the hub and medoid become relatively closer to each other.

This brings us to the idea that will be explained in detail in the following section: *Why not use hubs as cluster prototypes?* After all, it is expected of points with high hubness scores to be closer to centers of clustered subregions of high-dimensional space than other data points, making them viable candidates for representative cluster elements. We are not limited to observing only points with the highest hubness scores, we can also take advantage of hubness information for any given point. More generally, in case of irregularly shaped clusters, hubs are expected to be found near the centers of compact sub-clusters, which is also beneficial. In addition, hubness of points is straightforward to compute exactly, while the computation cluster centroids and medoids must involve some iterative inexact procedure intrinsically tied to the process of cluster construction. The remaining question of how to assign individual hubs to particular clusters will be addressed in the following section.

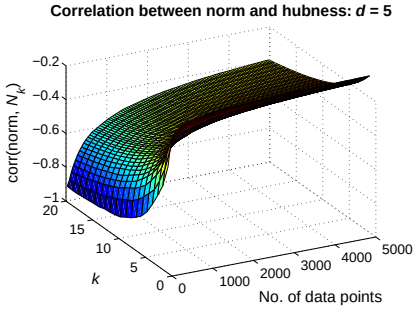
4 Hub-Based Clustering

If hubness is viewed as a kind of local centrality measure, it may be possible to use hubness for clustering in various ways. In order to test this hypothesis, we opted for an approach that allows observations about the quality of resulting clustering configurations to be related directly to the property of hubness, instead of being a consequence of some other attribute of the clustering algorithm. Since it is expected of hubs to be located near the centers of compact sub-clusters in high-dimensional data, a natural way to test the

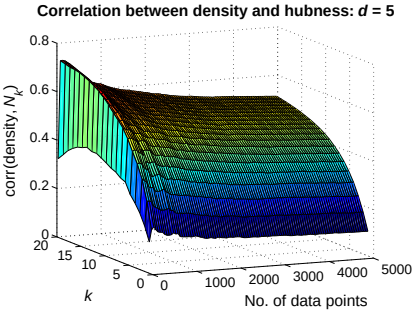
2. This is the reason why some of the graphs are not smooth.
3. In realistic scenarios, such indicators are not available.



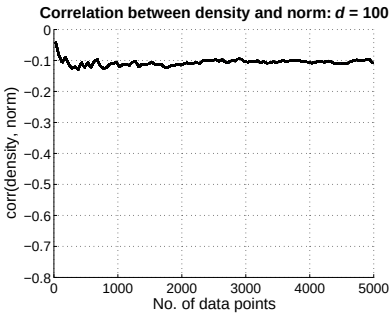
(a) Correlation between density and norm for $d = 5$



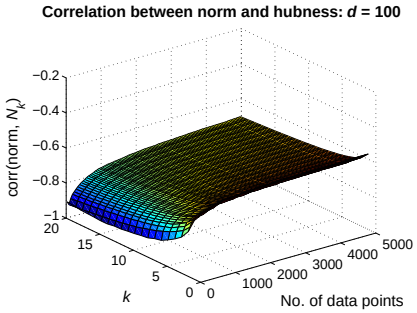
(c) Correlation between norm and hubness for $d = 5$



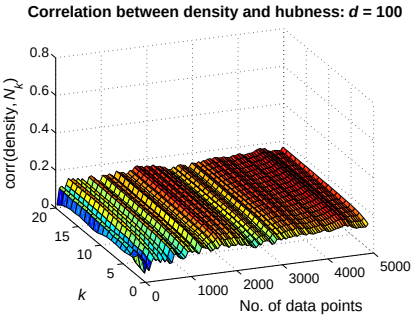
(e) Correlation between density and hubness for $d = 5$



(b) Correlation between density and norm for $d = 100$

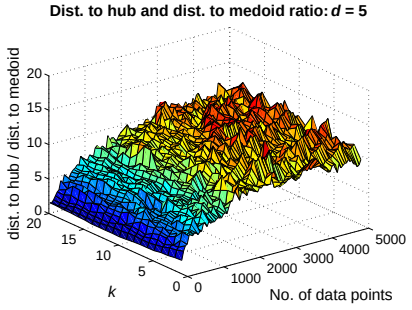


(d) Correlation between norm and hubness for $d = 100$

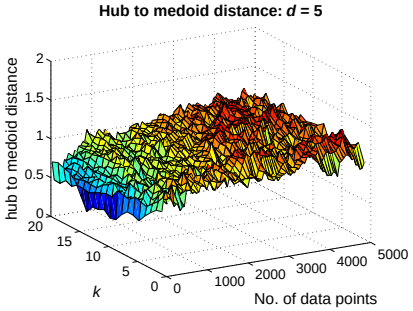


(f) Correlation between density and hubness for $d = 100$

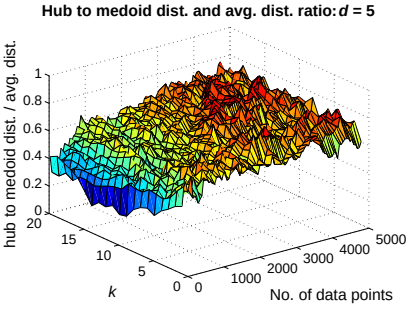
Fig. 2. Interaction between norm, hubness and density in the simulated setting, in low- and high-dimensional scenarios.



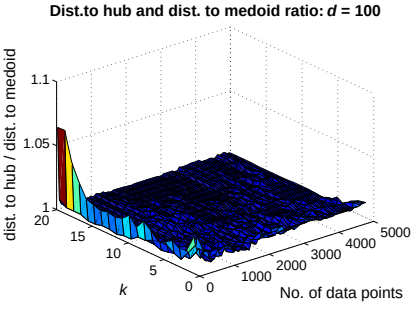
(a) Ratio between distance to hub and distance to medoid for $d = 5$



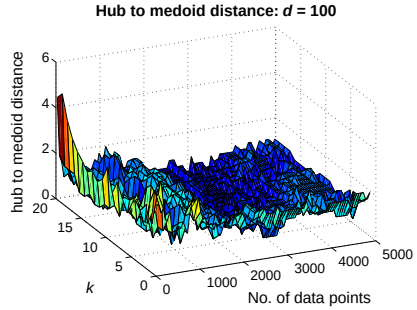
(c) Hub to medoid distance for $d = 5$



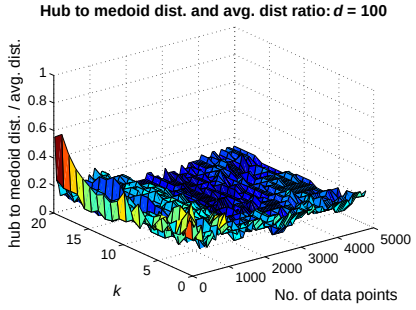
(e) Ratio between hub to medoid distance and average pairwise distance for $d = 5$



(b) Ratio between distance to hub and distance to medoid for $d = 100$



(d) Hub to medoid distance for $d = 100$



(f) Ratio between hub to medoid distance and average pairwise distance for $d = 100$

Fig. 3. Interaction between hubs, medoids and other points in the simulated setting, expressed through distances, in low- and high-dimensional scenarios.

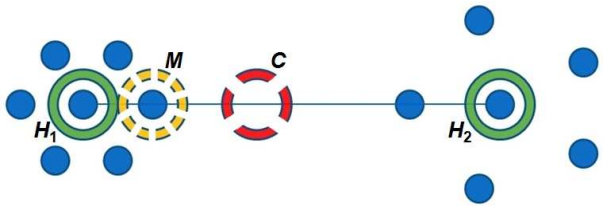


Fig. 4. Illustrative example: The red dashed circle marks the centroid (C), yellow dotted circle the medoid (M), and green circles denote two elements of highest hubness (H_1, H_2), for neighborhood size 3.

feasibility of using them to approximate these centers is to compare the hub-based approach with some centroid-based technique. For this reason, the considered algorithms are made to resemble K-means, by being iterative approaches for defining clusters around separated high-hubness data elements.

Centroids and medoids in K-means iterations tend to converge to locations close to high-hubness points, which implies that using hubs instead of either of these could actually speed up the convergence of the algorithms, leading straight to the promising regions in the data space. To illustrate this point, consider the simple example shown in Fig. 4, which mimics in two dimensions what normally happens in multi-dimensional data, and suggests that not only might taking hubs as centers in following iterations provide quicker convergence, but that it also might prove helpful in finding the best end configuration. Centroids depend on all current cluster elements, while hubs depend mostly on their neighboring elements and therefore carry localized centrality information. We will consider two types of hubness below, namely *global* hubness and *local* hubness. We define local hubness as a restriction of global hubness on any given cluster, considered in the context of the current algorithm iteration. Hence, the local hubness score represents the number of k-occurrences of a point in k-NN lists of elements within the same cluster.[4]

The fact that hubs emerge close to centers of dense subregions might suggest some sort of a relationship between hubness and the density estimate at the observed data point. There are, however, some important differences. First of all, hubness does not depend on scale. Let D_1 and D_2 be two separate sets of points. If the local distance matrices defined on each of them separately are proportional, we might think of D_1 and D_2 as two copies of the same abstract data model appearing at different scales. Even though the density estimate might be significantly different, depending on the defining volumes which are affected by scale, there will be a perfect match in hubness scores of the corresponding points. However, there is a more subtle difference. Let $D_k(x)$ be the set of points where x is among the k nearest neighbors. Hence, the hubness score of x is given by $N_k(x) = |D_k(x)|$. For each $x_i \in D_k(x)$, whether point x is among the k nearest neighbors of x_i depends on two things: $distance(x, x_i)$, and the density estimate at point x_i, not the density estimate at point x. Consequently, a hub might be a k-neighbor for points where density is high, as well as for points where density is low. Therefore, there is no direct correspondence between the magnitude of hubness and point density. Naturally, since hubs tend to be close to many points, it would be expected that density estimates at hub points are not low, but they may not correspond to the points of highest density in the data. Also, in order to compute the exact volume of the neighborhood around a given point, one needs to have a suitable data representation. For hubness, one only needs the distance matrix.

Computational complexity of hubness-based algorithms is mostly determined by the cost of computing hubness scores. Several fast approximate approaches are available. It was demonstrated [33] that it is possible to construct an approximate k-NN graph (from which hubness scores can be read) in $\Theta(ndt)$ time, where the user-defined value $t > 1$ expresses the desired quality of graph construction. It was reported that good graph quality may be achieved with small values of t, which we were able to confirm in our initial experiments. Alternatively, locality-sensitive hashing could also be used [34], as such methods have become quite popular recently. In other words, we expect our algorithms to be applicable in big data scenarios as well.

4.1 Deterministic Approach

A simple way to employ hubs for clustering is to use them as one would normally use centroids. In addition, this allows us to make a direct comparison with the K-means method. The algorithm, referred to as *K-hubs*, is given in Algorithm 1.

Algorithm 1 K-hubs

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
repeat
    for all Cluster c ∈ clusters do
        DataPoint h = findClusterHub(c);
        setClusterCenter(c, h);
    end for
    clusters = formClusters();
until noReassignments
return clusters
```

After initial evaluation on synthetic data, it became clear that even though the algorithm manages to find good and even best configurations often, it is quite sensitive to initialization. To increase the probability of finding the global optimum, we resorted to the stochastic approach described in the following section.

4. Henceforth, we will use uppercase K to represent the desired number of clusters and lowercase k for neighborhood size.

However, even though K-hubs exhibited low stability, it converges to cluster configurations very quickly, in no more than four iterations on all the data sets used for testing, most of which contained around 10000 data instances.

4.2 Probabilistic Approach

Even though points with highest hubness scores are without doubt the prime candidates for cluster centers, there is no need to disregard the information about hubness scores of other points in the data. In the algorithm described below, we implemented a squared hubness-proportional stochastic scheme based on the widely used simulated annealing approach to optimization [35]. The temperature factor was introduced to the algorithm, so that it may start as being entirely probabilistic and eventually end by executing deterministic K-hubs iterations. We will refer to this algorithm, specified by Algorithm 2, as *hubness-proportional clustering* (HPC).

Algorithm 2 HPC

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
float t = t0; {initialize temperature}
repeat
  float θ = getProbFromSchedule(t);
  for all Cluster c ∈ clusters do
    if randomFloat(0,1) < θ then
      DataPoint h = findClusterHub(c);
      setClusterCenter(c, h);
    else
      for all DataPoint x ∈ c do
        setChoosingProbability(x, N_k^2(x));
      end for
      normalizeProbabilities();
      DataPoint h = chooseHubProbabilistically(c);
      setClusterCenter(c, h);
    end if
  end for
  clusters = formClusters();
  t = updateTemperature(t);
until noReassignments
return clusters
```

The reason why hubness-proportional clustering is feasible in the context of high dimensionality lies in the skewness of the distribution of k-occurrences. Namely, there exist many data points having low hubness scores, making them bad candidates for cluster centers. Such points will have a low probability of being selected. To further emphasize this, we use the square of the actual hubness score instead of making the probabilities directly proportional to $N_k(x)$.

We have chosen to use a rather trivial temperature schedule in the getProbFromSchedule(t) function. The number of probabilistic iterations N_{Prob} is passed as an argument to the algorithm and the probability $\theta = \min(1, t/N_{Prob})$. Different probabilistic schemes are possible and might even lead to better results.

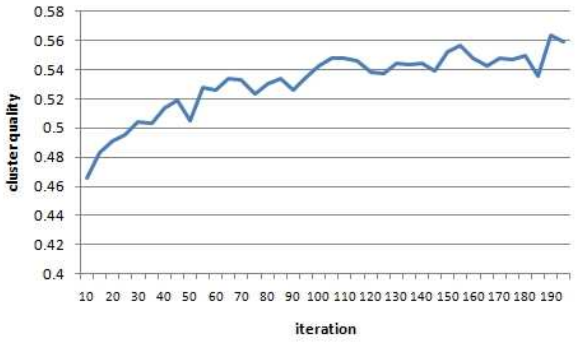


Fig. 5. Estimated quality of clustering for various durations of probabilistic search in HPC.

The HPC algorithm defines a search through the data space based on hubness as a kind of a local centrality estimate. To justify the use of the proposed stochastic scheme, we executed a series of initial tests on a synthetic mixture of Gaussians, for dimensionality $d = 50$, $n = 10000$ instances, and $K = 25$ clusters in the data. Neighborhood size was set to $k = 10$ and for each preset number of probabilistic iterations in the annealing schedule, the clustering was run 50 times, each time re-initializing the seeds. The results are displayed in Fig. 5. The silhouette index [36] was used to estimate the clustering quality. Due to the significant skewness of the squared hubness scores, adding more probabilistic iterations helps in achieving better clustering, up to a certain plateau that is eventually reached. The same shape of the curve also appears in the case of not taking the last, but the error-minimizing configuration.

4.3 A Hybrid Approach

The algorithms outlined in Sections 4.1 and 4.2 share a property that they do not require knowledge of data/object representation (they work the with distance matrix only), so all that is required is a distance/similarity measure defined for each pair of data objects. However, if the representation is also available such that it is possible to meaningfully calculate centroids, there also exists a third alternative: use point hubness scores to guide the search, but choose a centroid-based cluster configuration in the end. We will refer to this algorithm as *hubness-proportional K-means* (HPKM). It is nearly identical to HPC, the only difference being in the deterministic phase of the iteration, as the configuration cools down during the annealing procedure: instead of reverting to K-hubs, the deterministic phase executes K-means updates.

There are, indeed, cases when HPKM might be preferable to the pure hubness-based approach of K-hubs and HPC. Even though our initial experiments (Fig. 3) suggest that the major hubs lie close to local cluster means in high-dimensional data, there is no guarantee that this would hold for every cluster in every possible data set. It is reasonable to expect

Algorithm 3 HPKM

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
float t = t0; {initialize temperature}
repeat
  float θ = getProbFromSchedule(t);
  for all Cluster c ∈ clusters do
    if randomFloat(0,1) < θ then
      DataPoint h = findClusterCentroid(c);
      setClusterCenter(c, h);
    else
      for all DataPoint x ∈ c do
        setChoosingProbability(x, N_k^2(x));
      end for
      normalizeProbabilities();
      DataPoint h = chooseHubProbabilistically(c);
      setClusterCenter(c, h);
    end if
  end for
  clusters = formClusters();
  t = updateTemperature(t);
until noReassignments
return clusters
```

there to be distributions which lead to such local data structure where the major hub is not among the most central points. Also, an *ideal* cluster configuration (with minimal error) on a given real-world data set is sometimes impossible to achieve by using points as centers, since centers may need to be located in the empty space between the points.

In fact, we opted for the hybrid approach only after observing that, despite the encouraging initial results on synthetic data discussed in Section 5.1, hubness-based algorithms were not consistently better on real-world data sets. This is why we tried to take "the best of both worlds," by combining the centroid-based cluster representation with the hubness-guided search. This way, we are hoping to avoid premature convergence to a local optimum. We must keep in mind, however, that it is not as widely applicable as K-hubs and HPC, since it only makes sense with data where centroids can actually be defined.

5 Experiments and Evaluation

We tested our approach on various high-dimensional synthetic and real-world data sets. We will use the following abbreviations in the forthcoming discussion: KM (K-Means), ker-KM (kernel K-means), GKH (Global K-Hubs), LKH (Local K-Hubs), GHPC (Global Hubness-Proportional Clustering) and LHPC (Local Hubness-Proportional Clustering), HPKM (Hubness-Proportional K-Means), *local* and *global* referring to the type of hubness score that was used (see Section 4). For all centroid-based algorithms, including KM, we used the D^2 (K-means++) initialization procedure [12].[5] The neighborhood size

5. Hubness could also be used for cluster initialization, an option which we have not fully explored yet.

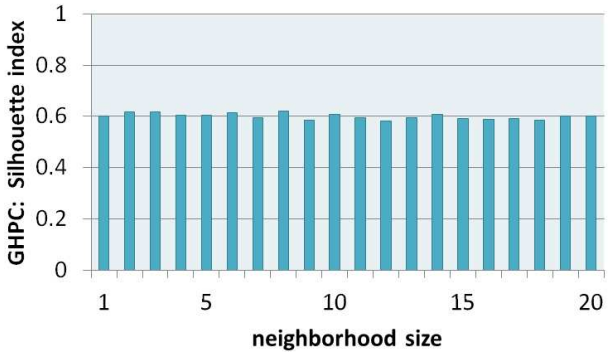


Fig. 6. Sensitivity of the quality of GHPC clustering on neighborhood size (k), measured by silhouette index.

of $k = 10$ was used by default in our experiments involving synthetic data and we have experimented with different neighborhood size in different real-world tests.

There is no known way of selecting the best k for finding neighbor sets, the problem being domain-specific. To check how the choice of k reflects on hubness-based clustering, we ran a series of tests on a fixed 50-dimensional 10-distribution Gaussian mixture for a range of k values, $k \in \{1, 2, \ldots, 20\}$. The results are summarized in Fig. 6. It is clear that, at least in such simple data, the hubness-based GHPC algorithm is not overly sensitive on the choice of k.

In the following sections, K-means++ will be used as the main baseline for comparisons, since it is suitable for determining the feasibility of using hubness to estimate local centrality of points. Additionally, we will also compare the proposed algorithms to kernel K-means [13] and one standard density-based method, GDBScan [37]. Kernel K-means was used with the non-parametric histogram intersection kernel, as it is believed to be good for image clustering and most of our real-world data tests were done on various sorts of image data.

Kernel methods are naturally much more powerful, since they can handle non-hyperspherical clusters. Yet, the hubness-based methods could just as easily be "kernelized," pretty much the same way it was done for K-means. This idea requires further tests and is beyond the scope of this paper.

For evaluation, we used repeated random sub-sampling, training the models on 70% of the data and testing them on the remaining 30%. This was done to reduce the potential impact of overfitting, even though it is not a major issue in clustering, as clustering is mostly used for pattern detection and not prediction. On the other hand, we would like to be able to use the clustering methods not only for detecting groups in a given sample, but rather for detecting the underlying structure of the data distribution in general.

5.1 Synthetic Data: Gaussian Mixtures

In the first batch of experiments, we wanted to compare the value of *global* vs. *local* hubness scores. These initial tests were run on synthetic data and do not include HPKM, as the hybrid approach was introduced later for tackling problems on real-world data.

For comparing the resulting clustering quality, we used mainly the silhouette index as an unsupervised measure of configuration validity, and average cluster entropy as a supervised measure of clustering homogeneity. Since most of the generated data sets are "solvable," i.e., consist of non-overlapping Gaussian distributions, we also report the normalized frequency with which the algorithms were able to find these perfect configurations. We ran two lines of experiments, one using 5 Gaussian generators, the other using 10. For each of these, we generated data of ten different high dimensionalities: $10, 20, \ldots, 100$. In each case, 10 different Gaussian mixtures were generated, resulting in 200 different generic sets, 100 of them containing 5 data clusters, the others containing 10. On each of the data sets, KM++ and all of the hub-based algorithms were executed 30 times and the averages of performance measures were computed.

The generated Gaussian distributions were hyperspherical (diagonal covariance matrices, independent attributes). Distribution means were drawn randomly from $[l^m_{bound}, u^m_{bound}]^d$, $l^m_{bound} = -20, u^m_{bound} = 20$ and the standard deviations were also uniformly taken from $[l^\sigma_{bound}, u^\sigma_{bound}]^d$, $l^\sigma_{bound} = 2, u^\sigma_{bound} = 5$.

Table 1 shows the final summary of all these runs. (Henceforth, we use boldface to denote measurements that are significantly better than others, in the sense of having no overlap of surrounding one-standard deviation intervals.) Global hubness is definitely to be preferred, especially in the presence of more clusters, which further restrict neighbor sets in the case of local hubness scores. Probabilistic approaches significantly outperform the deterministic ones, even though GKH and LKH also sometimes converge to the best configurations, but much less frequently. More importantly, the best overall algorithm in these tests was GHPC, which outperformed KM++ on all basis, having lower average entropy, a higher silhouette index, and a much higher frequency of finding the perfect configuration. This suggests that GHPC is a good option for clustering high-dimensional Gaussian mixtures. Regarding the number of dimensions when the actual improvements begin to show, in our lower-dimensional test runs, GHPC was better already on 6-dimensional mixtures. Since we concluded that using global hubness leads to better results, we only consider GKH and GHPC in the rest of the experiments.

5.2 Clustering and High Noise Levels

Real-world data often contains noisy or erroneous values due to the nature of the data-collecting process. It can be assumed that hub-based algorithms will be more robust with respect to noise, since hubness-proportional search is driven mostly by the highest-hubness elements, not the outliers. In the case of KM++, all instances from the current cluster directly determine the location of the centroid in the next iteration. When the noise level is low, some sort of outlier removal technique may be applied. In setups involving high levels of noise this may not be the case.

To test this hypothesis, we generated two data sets of 10000 instances as a mixture of 20 clearly separated Gaussians, farther away from each other than in the previously described experiments. The first data set was 10-dimensional and the second 50-dimensional. In both cases, individual distribution centers were drawn independently from the uniform $[l^m_{bound}, u^m_{bound}]^d$ distribution, $l^m_{bound} = -150, u^m_{bound} = 150$. The covariance matrix was also random-generated, independently for each distribution. It was diagonal, the individual feature standard deviations drawn uniformly from $[l^\sigma_{bound}, u^\sigma_{bound}]^d$, $l^\sigma_{bound} = 10, u^\sigma_{bound} = 60$. Cluster sizes were imbalanced. Without noise, both of these data sets represented quite easy clustering problems, all of the algorithms being able to solve them very effectively. This is, regardless, a more challenging task than we had previously addressed [38], by virtue of having a larger number of clusters.

To this data we incrementally added noise, 250 instances at a time, drawn from a uniform distribution on hypercube $[l^n_{bound}, u^n_{bound}]^d$, $l^n_{bound} = -200, u^n_{bound} = 200$, containing all the data points. The hypercube was much larger than the space containing the rest of the points. In other words, clusters were immersed in uniform noise. The highest level of noise for which we tested was the case when there was an equal number of actual data instances in original clusters and noisy instances. At every noise level, KM++, GKH, GHPC and GHPKM were run 50 times each. We used two different k-values, namely 20 and 50. We have used somewhat larger neighborhoods in order to try to smooth out the influence of noisy data on hubness scores. The silhouette index and average entropy were computed only on the non-noisy restriction of the data, i.e., the original Gaussian clusters. This is an important point, as such measures quantify how well each algorithm captures *the underlying structure* of the data. Indeed, if there is noise in the data, we are not overly interested in how well the noisy points cluster. Including them into the cluster quality indices might be misleading.

A brief summary of total averages is given in Table 2, with the best Silhouette index value and the best entropy score in each row given in boldface. The probabilistic hub-based algorithms show substantial improvements with higher noise levels, which is a very useful property. GHPKM is consistently better than KM++ for all noise levels, especially in terms of

TABLE 1
Averaged results of algorithm runs on high-dimensional mixtures of Gaussians

		LKH	GKH	LHPC	GHPC	KM++
$K = 5$	Silhouette	0.46 ± 0.03	0.51 ± 0.02	$\mathbf{0.61} \pm 0.02$	$\mathbf{0.61} \pm 0.02$	0.56 ± 0.02
	Entropy	0.32 ± 0.04	0.17 ± 0.01	0.09 ± 0.02	$\mathbf{0.06} \pm 0.01$	0.10 ± 0.01
	Perfect	0.32 ± 0.05	0.39 ± 0.05	$\mathbf{0.75} \pm 0.07$	$\mathbf{0.76} \pm 0.06$	0.54 ± 0.04
$K = 10$	Silhouette	0.38 ± 0.02	0.46 ± 0.01	0.52 ± 0.02	$\mathbf{0.57} \pm 0.01$	0.52 ± 0.01
	Entropy	0.52 ± 0.07	0.22 ± 0.01	0.22 ± 0.03	$\mathbf{0.08} \pm 0.01$	0.13 ± 0.01
	Perfect	0.05 ± 0.01	0.06 ± 0.02	0.30 ± 0.05	$\mathbf{0.39} \pm 0.06$	0.11 ± 0.02

TABLE 2
Estimated cluster quality at various noise levels for synthetic data composed of 20 different clusters

(a) d=10, k=20

	GKH		GHPC		KM++		GHPKM	
	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.
Noise<10%	0.29	0.41	0.37	0.18	0.34	0.22	**0.38**	**0.10**
Noise 10-20%	0.31	0.44	0.38	0.27	0.36	0.28	**0.39**	**0.20**
Noise 20-30%	0.31	0.50	0.35	0.33	0.36	0.38	**0.39**	**0.27**
Noise 30-40%	0.29	0.52	0.36	**0.32**	0.35	0.44	**0.37**	0.36
Noise 40-50%	0.29	0.55	0.35	**0.38**	0.33	0.53	**0.36**	0.45
AVG	0.30	0.50	0.36	**0.31**	0.34	0.41	**0.37**	**0.31**

(b) d=10, k=50

	GKH		GHPC		KM++		GHPKM	
	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.
Noise<10%	0.29	0.44	0.35	0.18	0.33	0.23	**0.39**	**0.10**
Noise 10-20%	0.29	0.50	0.36	0.25	0.36	0.27	**0.39**	**0.15**
Noise 20-30%	0.30	0.53	0.35	0.32	0.36	0.35	**0.38**	**0.24**
Noise 30-40%	0.29	0.59	0.35	0.35	0.35	0.44	**0.38**	**0.32**
Noise 40-50%	0.29	0.60	0.33	**0.39**	0.33	0.50	**0.36**	0.43
AVG	0.29	0.55	0.35	0.33	0.34	0.39	**0.38**	**0.29**

(c) d=50, k=20

	GKH		GHPC		KM++		GHPKM	
	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.
Noise<10%	0.37	0.45	0.49	0.12	0.48	0.16	**0.55**	**0.03**
Noise 10-20%	0.38	0.54	0.50	0.20	0.46	0.30	**0.55**	**0.02**
Noise 20-30%	0.37	0.54	0.47	0.23	0.42	0.44	**0.55**	**0.04**
Noise 30-40%	0.36	0.58	0.46	0.28	0.40	0.54	**0.53**	**0.09**
Noise 40-50%	0.34	0.64	0.43	0.40	0.38	0.59	**0.51**	**0.17**
AVG	0.36	0.57	0.46	0.27	0.42	0.46	**0.53**	**0.09**

(d) d=50, k=50

	GKH		GHPC		KM++		GHPKM	
	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.	Sil.	Ent.
Noise<10%	0.37	0.36	0.51	0.05	0.48	0.18	**0.55**	**0.02**
Noise 10-20%	0.40	0.39	0.51	0.09	0.46	0.33	**0.56**	**0.02**
Noise 20-30%	0.36	0.44	0.49	0.14	0.43	0.42	**0.55**	**0.03**
Noise 30-40%	0.36	0.45	0.47	0.20	0.42	0.52	**0.54**	**0.10**
Noise 40-50%	0.33	0.46	0.45	0.25	0.41	0.57	**0.52**	**0.17**
AVG	0.36	0.43	0.48	0.17	0.43	0.44	**0.54**	**0.09**

cluster homogeneity. The difference in average cluster entropy is quite obvious in all cases and is more pronounced in the 50-dimensional case, where there is more hubness in the data.

Figure 7 shows the rate of change in algorithm performance under various noise levels. We see that the achieved improvement is indeed stable and consistent, especially in the high-dimensional case. The difference increases with increasing noise, which means that HPC and HPKM are not only less affected by the curse of dimensionality, but also more robust to the presence of noise in the data.

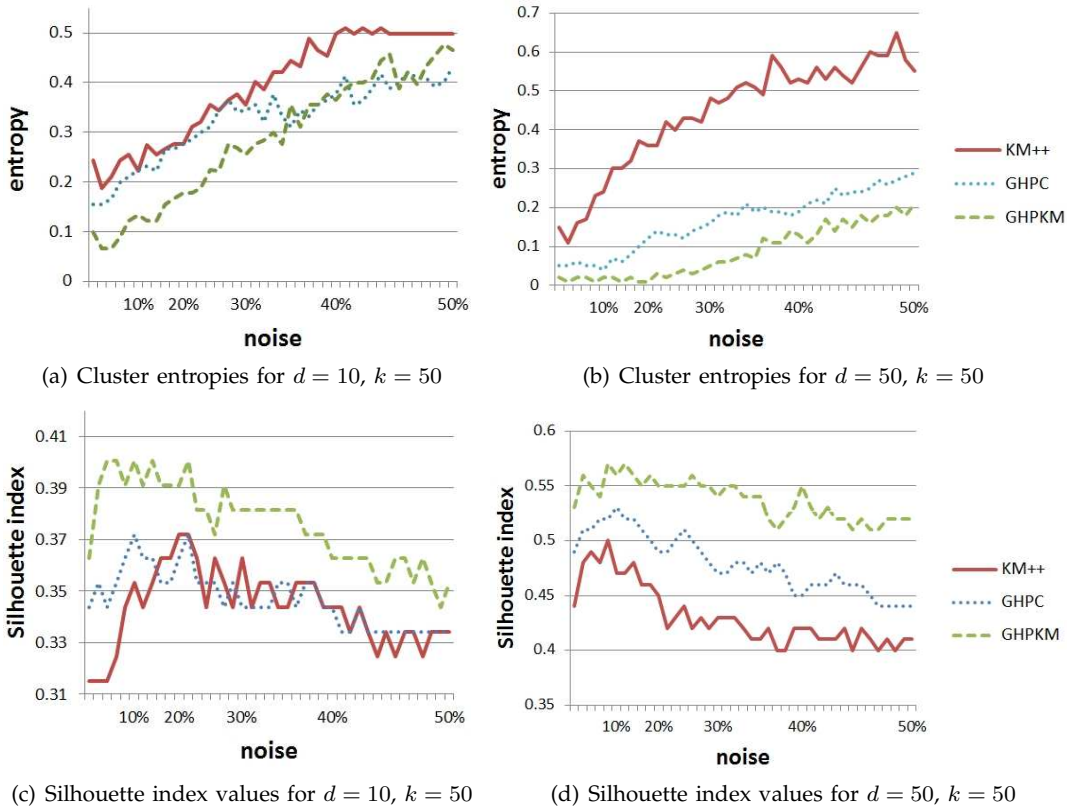
5.3 Experiments on Real-World Data

Real-world data is usually much more complex and difficult to cluster, therefore such tests are of a higher practical significance. As not all data exhibits hubness, we tested the algorithms both on intrinsically high-dimensional, high-hubness data and intrinsically low-to-medium dimensional, low-hubness data. There were two different experimental setups. In the first setup, a single data set was clustered for many different K-s (number of clusters), to see if there is any difference when the number of clusters is varied. In the second setup, 20 different data sets were all clustered by the number of classes in the data (the number of different labels).

The clustering quality in these experiments was measured by two quality indices, the silhouette index and the isolation index [39], which measures a percentage of k-neighbor points that are clustered together.

In the first experimental setup, the two-part MissAmerica data set (cs.joensuu.fi/sipu/datasets/) was used for evaluation. Each part consists of 6480 instances having 16 dimensions. Results were compared for various predefined numbers of clusters in algorithm calls. Each algorithm was tested 50 times for each number of clusters. Neighborhood size was 5.

The results for both parts of the data set are given in Table 3. GHPC clearly outperformed KM and other hubness-based methods. This shows that hubs can serve as good cluster center prototypes. On the other hand, hyperspherical methods have their limits and kernel K-means achieved the best overall cluster quality on this data set. Only one quality estimate is



(a) Cluster entropies for $d = 10$, $k = 50$

(b) Cluster entropies for $d = 50$, $k = 50$

(c) Silhouette index values for $d = 10$, $k = 50$

(d) Silhouette index values for $d = 50$, $k = 50$

Fig. 7. Gradual change in cluster quality measures with rising noise levels. The difference between the algorithm performances is much more pronounced in the high-dimensional case.

given for GDBScan, as it automatically determines the number of clusters on its own.

As mostly low-to-medium hubness data (with the exception of spambase), we have taken several UCI data sets (archive.ics.uci.edu/ml/datasets.html). Values of all the individual features in the data sets were normalized prior to testing. The data sets were mostly simple, composed only of a few clusters. The value of k was set to 20. The results are shown in the first parts of Tables 4(a) and 4(b).[6] In the absence of hubness,[7] purely hubness-based methods do not perform well. Note, however, that they score comparably to KM++ on several data sets, and that GHPC did as well as KM++ on the Iris data set, which is only 4-dimensional. On the other hand, hubness-guiding the K-means in HPKM neither helps nor hurts the K-means base in such cases.

As intrinsically high-dimensional, high-hubness data, we have taken several subsets of the ImageNet public repository (www.image-net.org). These data sets are described in detail in [20], [25]. We examine two separate cases: Haar wavelet representation and

6. Some entries for GDBScan are marked as "-" and in those cases the standard parametrization of the algorithm produced a single connected cluster as a result. Due to space considerations, we show only averages for the isolation index in Table 4(b).

7. We quantify hubness using the skewness measure, i.e., the standardized third moment of the distribution of N_k, signified as S_{N_k}. If $S_{N_k} = 0$ there is no skewness, positive (negative) values signify skewness to the right (left).

TABLE 3
Clustering quality on the Miss-America data set

(a) Silhouette index

	K	2	4	6	8	10	12	14	16
Part I	GKH	0.28	0.14	0.12	0.08	0.07	0.05	0.06	0.05
	GHPC	**0.38**	0.29	0.25	0.21	0.15	0.10	0.10	0.09
	KM++	0.14	0.12	0.09	0.08	0.07	0.07	0.07	0.07
	GHPKM	0.28	0.18	0.17	0.14	0.13	0.11	0.10	0.08
	ker-KM++	0.33	**0.36**	**0.36**	**0.34**	**0.35**	**0.22**	**0.28**	**0.14**
	GDBScan	-0.27							
Part II	GKH	0.33	0.21	0.13	0.08	0.08	0.07	0.06	0.06
	GHPC	0.33	0.27	0.22	0.26	0.18	0.19	0.12	0.11
	KM++	0.18	0.12	0.10	0.08	0.07	0.08	0.07	0.07
	GHPKM	0.33	0.22	0.18	0.14	0.12	0.11	0.10	0.08
	ker-KM++	**0.46**	**0.30**	**0.41**	**0.46**	**0.29**	**0.28**	**0.24**	**0.23**
	GDBScan	-0.25							

(b) Isolation index

	K	2	4	6	8	10	12	14	16
Part I	GKH	0.83	0.58	0.53	0.38	0.27	0.22	0.21	0.15
	GHPC	**0.91**	0.89	0.71	0.53	0.42	0.33	0.30	0.26
	KM++	0.62	0.46	0.34	0.23	0.19	0.16	0.13	0.12
	GHPKM	0.85	0.54	0.45	0.38	0.29	0.26	0.24	0.23
	ker-KM++	0.77	**0.92**	**0.93**	**0.92**	**0.95**	**0.91**	**0.91**	**0.80**
	GDBScan	0.12							
Part II	GKH	0.82	0.56	0.35	0.26	0.21	0.17	0.15	0.14
	GHPC	0.80	0.64	0.45	0.48	0.37	0.35	0.26	0.23
	KM++	0.62	0.35	0.28	0.20	0.16	0.14	0.11	0.09
	GHPKM	0.77	0.50	0.36	0.29	0.26	0.24	0.22	0.19
	ker-KM++	**0.88**	**0.78**	**0.90**	**0.94**	**0.91**	**0.89**	**0.90**	**0.91**
	GDBScan	0.12							

TABLE 4
Clustering quality on low to medium-hubness data sets from the UCI repository and subsets of high-hubness ImageNet data

(a) Silhouette index

data set	size	d	K	S_{N_1}	GKH	GHPC	KM++	GHPKM	ker-KM	GDB-Scan
wpbc	198	33	2	0.64	0.16	0.16	0.16	0.16	0.17	-
spamb.	4601	57	2	21.46	0.29	**0.38**	0.31	**0.50**	0.13	0.01
arcene	100	1000	2	1.08	0.21	0.22	0.20	**0.23**	0.21	-
ovarian	253	15154	2	1.20	0.17	0.17	0.18	0.19	0.13	-
iris	158	4	3	0.46	0.48	0.47	0.49	0.49	0.38	**0.62**
parkins.	195	22	2	0.39	0.25	0.30	0.37	0.37	**0.45**	-
sonar	208	60	2	1.35	0.11	0.11	**0.19**	0.15	0.13	-
wine	178	13	3	0.76	0.27	0.33	0.34	0.35	0.12	-
abalone	4177	8	29	0.92	0.22	0.20	0.26	0.27	0.26	0.05
spectr.	531	100	10	1.20	0.16	0.16	0.23	**0.25**	0.15	0.12
AVG-UCI					0.23	0.25	0.27	**0.30**	0.21	0.20
ds3haar	2731	100	3	2.27	0.62	0.67	**0.70**	**0.70**	0.61	0.63
ds4haar	6054	100	4	2.44	0.53	0.59	0.62	**0.64**	0.52	0.56
ds5haar	6555	100	5	2.43	0.56	0.58	0.65	**0.69**	0.50	0.51
ds6haar	6010	100	6	2.13	0.49	0.55	0.56	**0.58**	0.48	0.50
ds7haar	10544	100	7	4.60	0.33	0.65	0.63	**0.68**	0.50	0.58
AVG-Haar					0.51	0.61	0.63	**0.66**	0.52	0.55
ds3sift	2731	416	3	15.85	0.08	**0.12**	0.05	0.05	0.05	**0.12**
ds4sift	6054	416	4	8.88	0.06	0.06	0.02	0.03	0.02	**0.18**
ds5sift	6555	416	5	26.08	0.05	0.06	0.01	0.02	0.09	**0.11**
ds6sift	6010	416	6	13.19	0.01	0.02	0.01	0.02	**0.11**	0.09
ds7sift	10544	416	7	9.15	0.04	0.05	0.01	0.03	**0.19**	0.16
AVG-Sift					0.05	0.06	0.02	0.03	0.09	**0.13**
AVG-Img					0.28	0.34	0.33	**0.35**	0.31	0.34
AVG-Total					0.26	0.30	0.30	**0.33**	0.26	0.27

(b) Isolation index

data sets	GKH	GHPC	KM++	GHPKM	ker-KM	DB-Scan
AVG-UCI	0.48	0.47	0.44	0.47	**0.64**	0.55
AVG-Haar	0.64	0.69	0.71	**0.73**	0.70	0.72
AVG-Sift	0.35	0.38	0.37	0.41	**0.79**	0.32
AVG-Img	0.50	0.54	0.54	0.57	**0.76**	0.52
AVG-Total	0.49	0.51	0.49	0.52	**0.70**	0.54

SIFT codebook + color histogram representation [40], [41]. This totals to 10 different clustering problems. We set k to 5. The results are given in the second parts of Tables 4(a) and 4(b).

We see that the Haar wavelet representation clusters well, while the SIFT + color histogram one does not. This is not a general conclusion, but rather a particular feature of the observed data. GHPKM is clearly the best amongst the evaluated algorithms in clustering the Haar representation of the images. This is encouraging, as it suggests that hubness-guided clustering may indeed be useful in some real-world high-dimensional scenarios.

The fact that kernel K-means achieves best isolation in most data sets suggests that accurate center localization is not in itself enough for ensuring good clustering quality and the possibilities for extending the basic HPKM and HPC framework to allow for non-hyperspherical and arbitrarily shaped clusters need to be considered. There are many ways to use hubs and hubness in high-dimensional data clustering. We have only considered the simplest approach here and many more remain to be explored.

5.4 Interpreting Improvements in Silhouette Index

This section will discuss the reason why hubness-based clustering can offer better performance when compared to K-means in terms of intra- and inter-cluster distance expressed by the silhouette index.

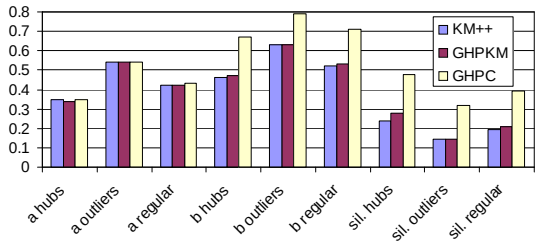
Let us view the a (intra) and b (inter) components of the silhouette index separately, and compute a, b and the silhouette index on a given data set for hubs, outliers and "regular" points.[8] Let n_h be the number of hubs selected. Next, we select as outliers the n_h points with the lowest k-occurrences. Finally, we select all remaining points as "regular" points.

Figure 8 illustrates the described break-up of the silhouette index on the Miss-America data set (we have detected similar trends with all other data sets where hubness-based methods offer improvement), for $k = 5$ and $K = 2$. It can be seen that all clustering methods perform approximately equally with respect to the a (intra) part, but that the hubness-based algorithms increase the b (inter) part, which is the main reason for improving the silhouette index. The increase of b is visible in all three groups of points, but is most prominent for hubs. Earlier research [23] had revealed that hubs often have low b-values, which causes them to cluster badly and have a negative impact on the clustering process. It was suggested that they should be treated almost as outliers. This why it is encouraging to see that the proposed clustering methods lead to clustering configurations where hubs have higher b-values than in the case of K-means.

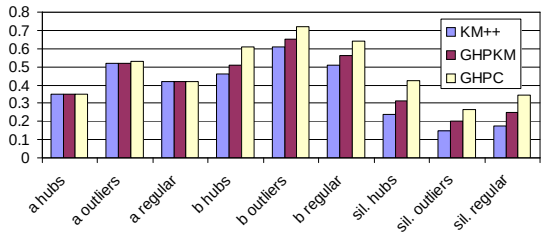
5.5 Visualizing the Hubness-Guided Search

In order to gain further insight, we have visualized the hubness-guided search on several low-to-medium-dimensional data sets. We performed clustering by the HPC algorithm and recorded the history of all iteration states (visited hub-points). After the clustering was completed, the data was projected onto a plane by a standard multi-dimensional scaling (MDS) procedure. Each point was drawn as a circle of radius proportional to its relative hubness. Some of the resulting images generated for the well-known Iris data set are shown in Fig. 9.

8. For the ith point, a_i is the average distance to all points in its cluster (intra-cluster distance), and b_i the minimum average distance to points from other clusters (inter-cluster distance). The silhouette index of the ith point is then $(b_i - a_i)/\max(a_i, b_i)$, ranging between -1 and 1 (higher values are better). The silhouette index of a set of points is obtained by averaging the silhouette indices of the individual points.



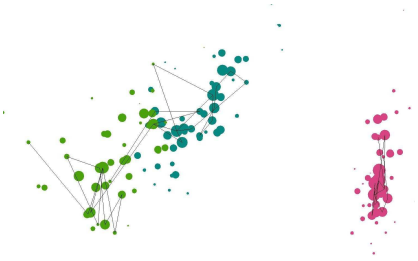
(a) Miss-America, Part I



(b) Miss-America, Part II

Fig. 8. Break-up od the silhouette index into its constituent parts, viewed separately for hubs, outliers and regular points on the Miss-America data set.

(a) k=1



(b) k=10

Fig. 9. Hubness-guided search for the best cluster hub-configuration in HPC on Iris data.

It can be seen that HPC searches through many different hub-configurations before settling on the final one. Also, what seems to be the case, at least in the majority of generated images, is that the search is somewhat *wider* for lower k-values. This observation is reasonable due to the fact that with an increase in neighborhood size, more points have hubness greater than a certain threshold and it is easier to distinguish between genuine outliers and slightly less central regular points. Currently, we do not have a universal robust solution to the problem of choosing a k-value. This is, on the other hand, an issue with nearly all kNN-based methods, with no simple, efficient, and general work-around.

6 Conclusions and Future Work

Using hubness for data clustering has not previously been attempted. We have shown that using hubs to approximate local data centers is not only a feasible option, but also frequently leads to improvement over the centroid-based approach. The proposed GHPKM method (Global Hubness-Proportional K-Means) had proven to be more robust than the K-Means++ baseline on both synthetic and real-world data, as well as in the presence of high levels of artificially introduced noise. This initial evaluation suggests that using hubs both as cluster prototypes and points guiding the centroid-based search is a promising new idea in clustering high-dimensional and noisy data. Also, global hubness estimates are generally to be preferred with respect to the local ones.

Hub-based algorithms are designed specifically for high-dimensional data. This is an unusual property, since the performance of most standard clustering algorithms deteriorates with an increase of dimensionality. Hubness, on the other hand, is a property of intrinsically high-dimensional data, and this is precisely where GHPKM and GHPC excel, and are expected to offer improvement by providing higher inter-cluster distance, i.e., better cluster separation.

The proposed algorithms represent only one possible approach to using hubness for improving high-dimensional data clustering. We also intend to explore other closely related research directions, including kernel mappings and shared-neighbor clustering. This would allow us to overcome the major drawback of the proposed methods – detecting only hyperspherical clusters, just as K-Means. Additionally, we would like to explore methods for using hubs to automatically determine the number of clusters in the data.

Acknowledgments. This work was supported by the Slovenian Research Agency, the IST Programme of the EC under PASCAL2 (IST-NoE-216886), and the Serbian Ministry of Education, Science and Technological Development project no. OI174023.

References

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. Morgan Kaufmann Publishers, 2006.

[2] C. C. Aggarwal and P. S. Yu, "Finding generalized projected clusters in high dimensional spaces," in *Proc. 26th ACM SIGMOD Int. Conf. on Management of Data*, 2000, pp. 70–81.

[3] K. Kailing, H.-P. Kriegel, P. Kröger, and S. Wanka, "Ranking interesting subspaces for clustering high dimensional data," in *Proc. 7th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2003, pp. 241–252.

[4] K. Kailing, H.-P. Kriegel, and P. Kröger, "Density-connected subspace clustering for high-dimensional data," in *Proc. 4th SIAM Int. Conf. on Data Mining (SDM)*, 2004, pp. 246–257.

[5] E. Müller, S. Günnemann, I. Assent, and T. Seidl, "Evaluating clustering in subspace projections of high dimensional data," *Proceedings of the VLDB Endowment*, vol. 2, pp. 1270–1281, 2009.
[6] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional spaces," in *Proc. 8th Int. Conf. on Database Theory (ICDT)*, 2001, pp. 420–434.
[7] D. François, V. Wertz, and M. Verleysen, "The concentration of fractional distances," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 7, pp. 873–886, 2007.
[8] R. J. Durrant and A. Kabán, "When is 'nearest neighbour' meaningful: A converse theorem and implications," *Journal of Complexity*, vol. 25, no. 4, pp. 385–397, 2009.
[9] A. Kabán, "Non-parametric detection of meaningless distances in high dimensional data," *Statistics and Computing*, vol. 22, no. 2, pp. 375–385, 2012.
[10] E. Agirre, D. Martínez, O. L. de Lacalle, and A. Soroa, "Two graph-based algorithms for state-of-the-art WSD," in *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2006, pp. 585–593.
[11] K. Ning, H. Ng, S. Srihari, H. Leong, and A. Nesvizhskii, "Examination of the relationship between essential genes in PPI network and hub proteins in reverse nearest neighbor topology," *BMC Bioinformatics*, vol. 11, pp. 1–14, 2010.
[12] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007, pp. 1027–1035.
[13] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proc. 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2004, pp. 551–556.
[14] T. N. Tran, R. Wehrens, and L. M. C. Buydens, "Knn density-based clustering for high dimensional multispectral images," in *Proc. 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas*, 2003, pp. 147–151.
[15] E. Biçici and D. Yuret, "Locally scaled density based clustering," in *Proc. 8th Int. Conf. on Adaptive and Natural Computing Algorithms (ICANNGA), Part I*, 2007, pp. 739–748.
[16] C. Zhang, X. Zhang, M. Q. Zhang, and Y. Li, "Neighbor number, valley seeking and clustering," *Pattern Recognition Letters*, vol. 28, no. 2, pp. 173–180, 2007.
[17] S. Hader and F. A. Hamprecht, "Efficient density clustering using basin spanning trees," in *Proc. 26th Annual Conf. of the Gesellschaft für Klassifikation*, 2003, pp. 39–48.
[18] C. Ding and X. He, "K-nearest-neighbor consistency in data clustering: Incorporating local information into global optimization," in *Proc. ACM Symposium on Applied Computing (SAC)*, 2004, pp. 584–589.
[19] C.-T. Chang, J. Z. C. Lai, and M. D. Jeng, "Fast agglomerative clustering using information of k-nearest neighbors," *Pattern Recognition*, vol. 43, no. 12, pp. 3958–3968, 2010.
[20] N. Tomašev, M. Radovanović, D. Mladenić, and M. Ivanović, "Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification," in *Proc. 7th Int. Conf. on Machine Learning and Data Mining (MLDM)*, 2011, pp. 16–30.
[21] ——, "A probabilistic approach to nearest-neighbor classification: Naive hubness bayesian kNN," in *Proc. 20th ACM Int. Conf. on Information and Knowledge Management (CIKM)*, 2011, pp. 2173–2176.
[22] M. Radovanović, A. Nanopoulos, and M. Ivanović, "Time-series classification in many intrinsic dimensions," in *Proc. 10th SIAM Int. Conf. on Data Mining (SDM)*, 2010, pp. 677–688.
[23] ——, "Hubs in space: Popular nearest neighbors in high-dimensional data," *Journal of Machine Learning Research*, vol. 11, pp. 2487–2531, 2010.
[24] N. Tomašev and D. Mladenić, "Nearest neighbor voting in high dimensional data: Learning from past occurrences," *Computer Science and Information Systems*, vol. 9, no. 2, pp. 691–712, 2012.
[25] N. Tomašev, R. Brehar, D. Mladenić, and S. Nedevschi, "The influence of hubness on nearest-neighbor methods in object recognition," in *Proc. 7th IEEE Int. Conf. on Intelligent Computer Communication and Processing (ICCP)*, 2011, pp. 367–374.
[26] K. Buza, A. Nanopoulos, and L. Schmidt-Thieme, "INSIGHT: Efficient and effective instance selection for time-series classification," in *Proc. 15th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD), Part II*, 2011, pp. 149–160.
[27] A. Nanopoulos, M. Radovanović, and M. Ivanović, "How does high dimensionality affect collaborative filtering?" in *Proc. 3rd ACM Conf. on Recommender Systems (RecSys)*, 2009, pp. 293–296.
[28] M. Radovanović, A. Nanopoulos, and M. Ivanović, "On the existence of obstinate results in vector space models," in *Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2010, pp. 186–193.
[29] J. J. Aucouturier and F. Pachet, "Improving timbre similarity: How high is the sky?" *Journal of Negative Results in Speech and Audio Sciences*, vol. 1, 2004.
[30] J. J. Aucouturier, "Ten experiments on the modelling of polyphonic timbre," Ph.D. dissertation, University of Paris 6, Paris, France, 2006.
[31] D. Schnitzer, A. Flexer, M. Schedl, and G. Widmer, "Local and global scaling reduce hubs in space," *Journal of Machine Learning Research*, vol. 13, pp. 2871–2902, 2012.
[32] S. France and D. Carroll, "Is the distance compression effect overstated? Some theory and experimentation," in *Proc. 6th Int. Conf. on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, 2009, pp. 280–294.
[33] J. Chen, H. Fang, and Y. Saad, "Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection," *Journal of Machine Learning Research*, vol. 10, pp. 1989–2012, 2009.
[34] V. Satuluri and S. Parthasarathy, "Bayesian locality sensitive hashing for fast similarity search," *Proceedings of the VLDB Endowment*, vol. 5, no. 5, pp. 430–441, 2012.
[35] D. Corne, M. Dorigo, and F. Glover, *New Ideas in Optimization*. McGraw-Hill, 1999.
[36] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Addison Wesley, 2005.
[37] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: The algorithm gdbscan and its applications," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169–194, 1998.
[38] N. Tomašev, M. Radovanović, D. Mladenić, and M. Ivanović, "The role of hubness in clustering high-dimensional data," in *Proc. 15th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD), Part I*, 2011, pp. 183–195.
[39] G. Frederix and E. J. Pauwels, "Shape-invariant cluster validity indices," in *Proc. 4th Industrial Conf. on Data Mining (ICDM)*, 2004, pp. 96–105.
[40] D. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. on Computer Vision (ICCV)*, vol. 2, 1999, pp. 1150–1157.
[41] Z. Zhang and R. Zhang, *Multimedia Data Mining*. Chapman and Hall, 2009.

Nenad Tomašev is a PhD student at the Artificial Intelligence Laboratory at Jožef Stefan Intitute in Ljubljana. He graduated in 2008 from the Department of Mathematics and Informatics at the University of Novi Sad. His research focus is in the area of Machine Learning and Data Mining, as well as Stochastic Optimization and Artificial Life. He has actively participated as a teaching assistant in Petnica Science Center and Višnjan Summer School.

Miloš Radovanović is Assistant Professor at the Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Serbia, where he received his BSc, MSc and PhD degrees. He was/is a member of several international projects supported by DAAD, TEMPUS, and bilateral programs. From 2009 he is Managing Editor of the Computer Science and Information Systems journal. He (co)authored one programming textbook, a research monograph, and over 40 papers in Data Mining and related fields.

Dunja Mladenić is an expert on study and development of Machine Learning, Data and Text Mining, Semantic Technology techniques and their application on real-world problems. She is associated with the J. Stefan Institute since 1987 and she is currently leading the Artificial Intelligence Laboratory at the Institute. She received her BSc, MSc and PhD at University of Ljubljana all in Computer Science. She was a visiting researcher at School of Computer Science, Carnegie Mellon University, USA (1996–1997 and 2000–2001). She has published papers in refereed conferences and journals, served in the program committee of international conferences, organized international events and co-edited several books.

Mirjana Ivanović holds the position of Full Professor at Faculty of Sciences, University of Novi Sad. She is the Head of Chair of

Computer Science and a member of University Council for Informatics. She is author or co-author of 13 textbooks and of more than 230 research papers on multi-agent systems, e-learning, intelligent techniques (CBR, data and web mining), most of which are published in international journals and conferences. She is/was a member of Program Committees of more than 100 international conferences and is the Editor-in-Chief of Computer Science and Information Systems journal.

2.2 Hubness-aware Classification

Classification is among the main applications of k-nearest neighbor methods in practice. As most real-world data of interest for automated analysis and modeling is complex and high-dimensional, developing robust, hubness-aware classification techniques is of prime interest. The only existing explicitly hubness-aware approach that was in use prior to the developments proposed in the papers comprising this thesis was the instance weighting method [Radovanović et al., 2009] that reduced the voting weights of bad hubs. This simple idea was shown to be helpful in classification under high hubness of the data. Nevertheless, its simplicity suggested that more complex and more effective approaches might be possible. This has lead to the development of several new hubness-aware classification algorithms that we have proposed and will describe in more detail in this chapter.

We have proposed three novel approaches to hubness-aware k-nearest neighbor classification: h-FNN [Tomašev et al., 2011b][Tomašev et al., 2013b], NHBNN [Tomašev et al., 2011c], HIKNN [Tomašev and Mladenić, 2011c][Tomašev and Mladenić, 2012]. The two of these approaches, h-FNN and HIKNN are based on fuzzy class hubness kNN occurrence models and will be presented and analyzed in more detail in Section 2.2.1 and Section 2.2.2, respectively. Here, we will outline the details of the third approach, the Naive Hubness-Bayesian k-nearest neighbor.

All three proposed hubness-aware approaches share a common theme: learning from past occurrences. Some form of an occurrence model is built from the occurrences on the training set and is used in predicting the labels in future queries. The goal is to predict $p(y = c|D_k(x))$ for every $c \in C$ and to assign a label based on $argmax_{c \in C} p(y = c|D_k(x))$. The observed k-neighbor set $D_k(x)$ can contain a mixture of different types of neighbor points. As shown in Figure 7, we can observe three types of points with regards to their previous occurrences.

Hubs are very frequent nearest neighbors, so lots of data is available to learn their occurrence tendencies from. This allows for better probability estimates for hub points in the occurrence models. Anti-hubs, on the other hand, pose a problem. If we wish to build an occurrence model that would account for any possible neighbor point, special care must be taken when dealing with anti-hubs, as they occur either very rarely as neighbors or, in some cases, they never occur. Most reasoning in the hubness-aware classification algorithms, including [Radovanović et al., 2009] is based on the neighbor k-occurrence profiles, as shown in Figure 8, where an occurrence profile of a neighbor point is shown. The occurrence profile for a point $x_i \in D$ represents an estimate of $p(y = c|x_i \in D_k(x))$ for all $c \in C$ based on all $x \in D$. Some occurrence profiles are *pure* in a sense that x_i occurs consistently as a neighbor to members of one class only. Other profiles are *heterogenous* as x_i acts as a neighbor to various points from many different classes in the data.

The probability estimates derived from the neighbor occurrence profiles on the training data are used to form the fuzzy voting schemes in the fuzzy hubness-aware kNN classifiers, namely h-FNN [Tomašev et al., 2011b][Tomašev et al., 2013b] and HIKNN [Tomašev and Mladenić, 2011c][Tomašev and Mladenić, 2012]. The Naive Hubness-Bayesian k-nearest neighbor classifier (NHBNN) is based on a similar, but slightly different idea. Before we proceed with the formal outline of the algorithm, we will intuitively explain the idea behind the approach.

Let x be the query point and let $D_k(x)$ denote the set of its k-nearest neighbors. Given the features of x, the training set D and the metric $d(.,.)$ the k-neighbor set is uniquely determined, as the process is entirely deterministic. Now imagine that the features and the metric are unknown to the observer and hidden inside a black box system. All that a user can observe in that case is the k-neighbor set for each query point and the labels of the training points. In that case, the k-neighbors of x could be interpreted as k random variables with a high dependency on the hidden label of the query point and the retrieval of

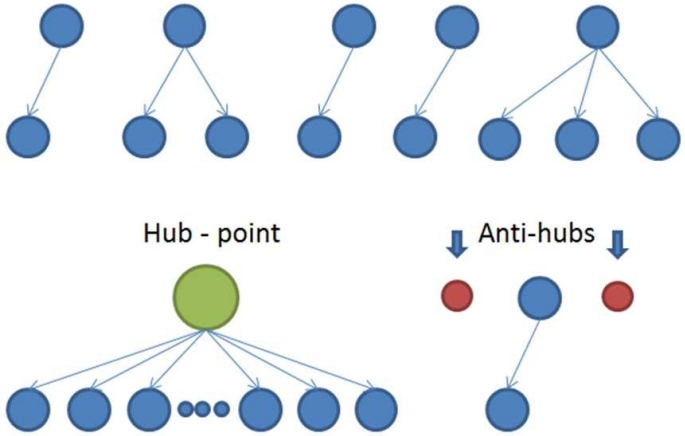


Figure 7: Regular neighbor points, hubs and outliers in high-dimensional data.

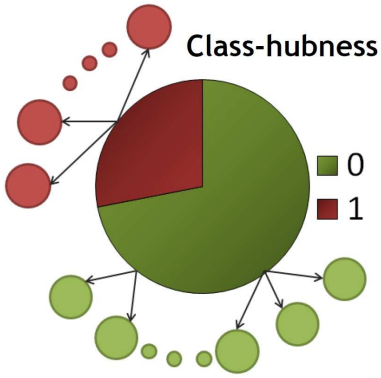


Figure 8: An occurrence profile of one neighbor point. The depicted profile shows that the point in question acts as a bad hub, as it is a neighbor to many points in different classes and therefore induces label mismatches thereby causing misclassification in the traditional kNN approaches.

the k-neighbor set could be interpreted as a random event (Figure 9). If such an abstraction is used to model the kNN process, the application of Bayesian techniques becomes possible and we have decided to use the simplest Naive Bayes [Han, 2005] (NB) method in order to build a proof-of-concept hubness-aware algorithm based on these assumptions.

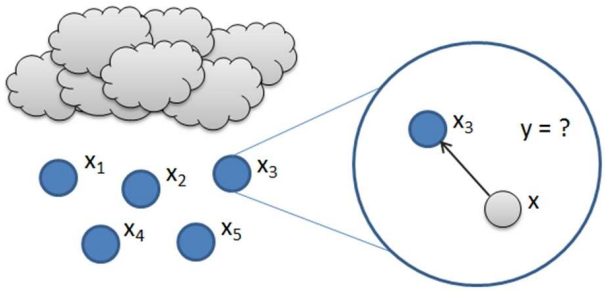


Figure 9: The idea behind the NHBNN approach - observe each neighbor occurrence as a random event.

It is almost as if the k neighbors of the query point are interpreted as a new set of its defining *features* and the kNN query process as *feature extraction*. This might be an intuitively easier explanation, though not entirely correct, for the reasons that will become apparent when examining the equations. Namely, a major problem would arise in case of *orphans* from the training data if they were observed in some k-neighbor sets on the test data. It would be as if observing a new feature value, one that has never appeared

before. Such a scenario is not unlikely in high-dimensional data. The traditional Naive Bayes method doesn't have the mechanisms to deal with this, so it was necessary to design an appropriate extension.

As mentioned above, the goal is to estimate the $p(y = c|D_k(x))$ for each $c \in C$. We view $D_k(x)$ as being the result of k random draws from D where each x_i has probability $p_t(x_i|D_t(x),x)$ of being drawn at time t, where $t \in \{1,2,\ldots,k\}$. Since the concepts of t-neighbor and $(t+1)$-neighbor are quite similar, we disregard the exact position of x_i in the list, which endows us with more occurrence information to work with.

Let $x_i \in X$ and x_{it}, for $t \in \{1,2,\ldots,k\}$, be the k nearest neighbors of x_i. We focus on a naive Bayesian estimate which takes the form shown in Eq. 4:

$$p(y_i = c|D_k(x_i)) \propto p(y_i = c)\prod_{t=1}^{k} p(x_{it} \in D_k(x_i)|y_i = c). \tag{4}$$

The independence assumption that is the basis of the Naive Bayes rule obviously does not hold in most cases. However, it was shown in the past that this is not necessary for the naive Bayesian approach to work [Rish, 2001], especially if there is in contrast strong functional correlation between the attributes. The dependencies tend to cancel each other out.

The main problem lies in estimating the probabilities on the right-hand side of Eq. 4, especially for anti-hubs, which will be treated as a special case. Also, each point was included to its own neighbor set at the 0th position, ensuring that each $N_k(x_i) \geq 1$.

For clarity, we shorten the notation by denoting $p(x_{it} \in D_k(x_i)|y_i = c)$ as $p_{x_i,c,k}(x_{it})$, and by n_c the number of elements of class c. Also, let $N_{k,c_1}(c_2)$ be the total number of k-occurrences of elements from class c_2 in neighborhoods of elements belonging to c_1.

Let (x,y) be an element from the neighborhood of x_i. Also, let $(x_g,y_g) \in \bigcup_{x_j:y_j=c} D_k(x_j)$. The derivation of the probability estimate is given in Eq. 5:

$$\begin{aligned} & p_{x_i,c,k}(x_{it}) \\ &= p_{x_i,k}(y = y_{it}|y_i = c)\cdot p_{x_i,k}(x = x_{it}|y = y_{it}) \\ &\approx p_{c,k}(y_g = y_{it})\cdot p_{c,k}(x_g = x_{it}|y_g = y_{it}) \\ &= \frac{N_{k,c}(y_{it})}{n_c\cdot(k+1)}\cdot\frac{N_{k,c}(x_{it})}{\sum_{j:y_j=y_{it}} N_{k,c}(x_j)} = \frac{N_{k,c}(x_{it})}{n_c\cdot(k+1)} \\ &\approx \frac{N_{k,c}(x_{it})+\lambda}{n_c\cdot(k+1)+\lambda|D|} = \bar{p}_{x_i,c,k}(x_{it}), \end{aligned} \tag{5}$$

where λ is the Laplace estimator, ensuring non-zero probabilities. When dealing with anti-hubs, however, the outlined approximation can not be expected to yield reliable probability estimates. This is why for anti-hubs we partly infer the probability from what is known about the typical points from the same class and their hubness, as shown in Eq. 6.

$$p_{x_i,c,k}(x_{it}) \approx \begin{cases} \bar{p}_{x_i,c,k}(x_{it}), & \text{if } N_k(x_{it}) > \theta, \\ u_{x_i,c,k}(x_{it}), & \text{if } N_k(x_{it}) \leq \theta. \end{cases}$$

$$u_{x_i,c,k}(x_{it}) = \alpha_h\cdot\bar{p}_{x_i,c,k}(x_{it}) + (1-\alpha_h)\cdot\acute{p}_{x_i,c,k}(x_{it}). \tag{6}$$

We proposed two approaches to defining $\acute{p}_{x_i,c,k}(x_{it})$. Both are based on the approximation given in Eq. 7:

$$N_{k,c}(x_{it}) \approx N_{k,c}(y_{it})/n_{y_{it}}. \tag{7}$$

This estimate will be referred to as the *global approximative approach*. Local information could also be taken into account by inferring $N_{k,c}(y_{it})$ from some k_{est}-neighborhood of x_{it} instead, as given in Eq. 8. This will be referred to as the *local approximative approach*. In the experiments in [Tomašev et al., 2011c], the default value of $k_{\text{est}} = 20$ was used.

$$N_{k,c}(x_{it}) \approx \frac{N_{x_i,k_{\text{est}},k,c}(y_{it})}{n_{k_{est},y_{it}}} = \frac{\sum_{j \in D_{k_{\text{est}}}(x_{it}): y_j = y_{it}} N_{k,c}(x_j)}{|x \in D_{k_{\text{est}}}(x_{it}) : y = y_{it}|}. \tag{8}$$

In the end, we obtain $\acute{p}_{x_i,c,k}(x_{it})$ from Eq. 6 by plugging either the local or the global class hubness estimate instead of the original $N_{k,c}(x_{it})$ into $\bar{p}_{x_i,c,k}(x_{it})$. In other words:

$$\acute{p}_{x_i,c,k}(x_{it}) = \begin{cases} \frac{N_{k,c}(y_{it})+\lambda}{n_c \cdot n_{y_{it}} \cdot (k+1)+\lambda C}, & \text{if global,} \\ \frac{N_{k_{\text{est}},k,c}(y_{it})+\lambda}{n_c \cdot n_{k_{est},y_{it}} \cdot (k+1)+\lambda C}, & \text{if local.} \end{cases}$$

Class affiliation of x is determined as $y = argmax_c p(y = c|D_k(x))$. In case of an unlikely tie, the assignment is made based on the prior class probabilities, i.e., by assigning to the majority class. Several parameters are required for the algorithm to work, but they can also be inferred from the training set by leave-one-out cross-validation.

The evaluation of NHBNN in [Tomašev et al., 2011c][N. and D., 2012][Tomašev and Mladenić, 2013] and Sections 2.2.3 and 2.2.4 demonstrates the effectiveness of the proposed method in various setups for high-dimensional data classification.

The algorithms that we have proposed are by no means the only possibilities for hubness-aware kNN classifier design and we hope that even better algorithms will be developed in the future. Also, it should be taken into account that, among the existing kNN classifiers that have not been designed with hubness in mind, some are more and some are less susceptible to the dimensionality curse and the hubness phenomenon. A thorough investigation of the impact of hubs on various kNN classifiers that are currently in use would be advisable. Such an examination was out of the scope of this thesis.

2.2.1 Fuzzy Class-Hubness Measures

This Section presents the paper titled *Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification* by Nenad Tomašev, Miloš Radovanović, Dunja Mladenić and Mirjana Ivanović. The paper was originally published at the MLDM (Machine Learning and Data Mining) conference in New York in 2011 [Tomašev et al., 2011b], where it was awarded the **best paper award**, and was further extended to its current form and accepted for publishing in the International Journal of Machine Learning and Cybernetics in November 2012 [Tomašev et al., 2013b].

The paper proposes a novel set of hubness-aware fuzzy measures to be used within the fuzzy k-nearest neighbor (FNN) classification framework [Keller et al., 1985], specifically tailored for high-dimensional data under the assumption of hubness.

The proposed fuzzy measures are based on the neighbor k-occurrence profiles observed on the training data. They are used as fuzzy votes during label assignment and reflect the uncertainty inherent in heterogenous hub neighbor occurrences. In a sense, the fuzzy measures obtained from the reverse nearest neighbors are conceptually similar to those obtained from the direct k-neighbor sets that have been previously used in FNN. However, the skewed neighbor occurrence distribution assures that the fuzzy probability estimates obtained via the proposed hubness-aware approach are derived from a larger average number of points, thereby reducing the expected misclassification rate on high-dimensional data.

The proposed hubness-fuzzy k-nearest neighbor algorithm (h-FNN) is the first hubness-aware classification algorithm based on the class-specific neighbor occurrences from the occurrence models. The weighting scheme that was initially proposed in hw-kNN [Radovanović et al., 2009] had only taken into account the bad hubness of neighbor points, without decomposing it into the class specific hubness tendencies.

Int. J. Mach. Learn. & Cyber. manuscript No.
(will be inserted by the editor)

Hubness-based fuzzy measures for high-dimensional *k*-nearest neighbor classification

Nenad Tomašev · Miloš Radovanović · Dunja Mladenić · Mirjana Ivanović

Received: 9. March 2012 / 19. November 2012.

Abstract Most data of interest today in data-mining applications is complex and is usually represented by many different features. Such high-dimensional data is by its very nature often quite difficult to handle by conventional machine-learning algorithms. This is considered to be an aspect of the well known *curse of dimensionality*. Consequently, high-dimensional data needs to be processed with care, which is why the design of machine-learning algorithms needs to take these factors into account. Furthermore, it was observed that some of the arising high-dimensional properties could in fact be exploited in improving overall algorithm design. One such phenomenon, related to nearest-neighbor learning methods, is known as *hubness* and refers to the emergence of very influential nodes (hubs) in *k*-nearest neighbor graphs. A crisp weighted voting scheme for the *k*-nearest neighbor classifier has recently been proposed which exploits this notion. In this paper we go a step further by embracing the *soft* approach, and propose several fuzzy measures for *k*-nearest neighbor classification, all based on hubness, which express fuzziness of elements appearing in *k*-neighborhoods of other points. Experimental evaluation on real data from the UCI repository and the image domain suggests that the fuzzy approach provides a useful measure of confidence in the predicted labels, resulting in improvement over the crisp weighted method, as well as the standard *k*NN classifier.

This is an extended version of the paper *Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification*, which was presented at the MLDM 2011 conference Tomašev et al. [2011b].

N. Tomašev and D. Mladenić
Institute Jožef Stefan, Artificial Intelligence Laboratory
Jožef Stefan International Postgraduate School
Jamova 39, 1000 Ljubljana, Slovenia
E-mail: nenad.tomasev@ijs.si, dunja.mladenic@ijs.si

M. Radovanović and M. Ivanović
University of Novi Sad, Department of Mathematics and Informatics
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
E-mail: radacha@dmi.uns.ac.rs, mira@dmi.uns.ac.rs

1 Introduction

High-dimensional data is ubiquitous in modern applications. It arises naturally when dealing with text, images, audio, data streams, medical records, etc. The impact of this high dimensionality is manyfold. It is a well known fact that many machine-learning algorithms are plagued by what is usually termed the *curse of dimensionality*. This comprises a set of properties that tend to become more pronounced as the dimensionality of data increases. First and foremost is the unavoidable sparsity of data. In high-dimensional spaces all data is sparse, meaning that there is not enough data to make reliable density estimates. Another detrimental influence comes from the concentration of distances, as all data points tend to become relatively more similar to each other as dimensionality increases. Such a decrease of contrast makes distinguishing between relevant and irrelevant points in queries much more difficult. This phenomenon has been thoroughly explored in the past [Aggarwal et al., 2001; François et al., 2007]. Usually, it only holds for data drawn from the same underlying probability distribution. Mixed data is not so severely affected [Houle et al., 2010], but the effects are still more pronounced than in the lower-dimensional scenarios. The difficulties arising from the influence

[0] Published in International Journal of Machine Learning & Cybernetics ©Springer Verlag
DOI: 10.1007/s13042-012-0137-1
Original article available at:
http://link.springer.com/article/10.1007%2Fs13042-012-0137-1
This is an extended version of the conference paper presented at the Machine Learning and Data Mining conference (MLDM 2011) Tomašev et al. [2011b]

of high dimensionality on distance measures even led some researchers to question the very notion of a point's nearest neighbor in high-dimensional feature spaces [Durrant and Kabán, 2009].

Regardless of the theoretical considerations above, methods based on nearest neighbors remain in very frequent use throughout various data-mining tasks, such as classification, clustering and information retrieval. This is hardly surprising, given the simplicity of the notion of nearest neighbors: the inferences about the current example are based on the most similar previously observed points. It is somewhat disheartening to see that even such simple ideas can be, at certain times, severely compromised by the dimensionality curse.

1.1 The hubness phenomenon

In this paper, we will focus only on one phenomenon of interest for nearest-neighbor methods operating in many dimensions. This phenomenon is known as *hubness*. The term was coined after *hubs*, very influential points which arise in high-dimensional spaces. Their influence is measured by the frequency with which they occur as neighbors to other points in the data. In a sense, they are very frequently 'consulted' during inference. If they are not compromised by noise and also contain accurate class-affiliation information, they exhibit a highly beneficial influence and all is well. If, on the other hand, there were some errors in feature values or the attached labels, such points would exhibit a highly detrimental influence and are known as *bad hubs* [Radovanović et al., 2009, 2010a,b]. Of course, real-world data is often noisy and not entirely reliable, whether it had been gathered automatically by sensors or input by human operators. Both ways of data acquisition are somewhat uncertain. Consequently, bad hubs are not an uncommon occurrence in practice.

There is more to hubness than just a few frequent neighbors. Denote by $N_k(x)$ the number of k-occurrences of x, i.e., the number of times x appears in k-nearest neighbor lists of other points in the data. The entire distribution of $N_k(x)$ becomes affected and an increasing skewness is usually observed. What this means is that most points very rarely occur as neighbors. Therefore, most of the time when we examine a queried k-neighbor set, it will contain some of the hub-points in the data. We will address these issues in more detail in Section 3. We should point out that hubness is a consequence of high *intrinsic* dimensionality of data (regardless of the nominal number of features in the chosen representation). It is a general property which stems from how the geometry of high-dimensional spaces affects the probability of each point being a k-neighbor (i.e., being among the k closest points to some other point in the data). More specifically, most data sets (approximately) appear as hyperspheres or unions of hyperspheres centered around some distribution means. This positioning renders points closer to the data centers more likely to be included in k-nearest neighbor lists of other data points. This tendency increases with dimensionality.

Hubness was first observed in music retrieval, when some songs were observed as being fetched very frequently in the queries and were determined not to be relevant on average, i.e. the calculated similarity in the feature spaces failed to capture the semantic similarity perceived by people [Aucouturier, 2006; Aucouturier and Pachet, 2004]. Even though we mentioned bad hubs as sometimes being caused by noise and errors in the data, it is not entirely so. Many data contain overlapping probability distributions, therefore bad hubness can arise even in error-free data sets. It is equally dangerous in both cases, so the underlying mechanics of hubness will not be given special attention in this paper, as it is a separate topic. What we will do is provide solutions to such cases when bad hubness does appear, as it can not always be avoided.

One simple solution to the problem has already recently been proposed, in form of a weighting scheme for the voting in the k-nearest neighbor (kNN) algorithm [Radovanović et al., 2010a; Radovanovic et al., 2010]. We will have a closer look at that weighting in Section 2.1, while we outline the motivation for our fuzzy approach.

Our idea is to extend the class-nonspecific crisp kNN weighting scheme described in [Radovanović et al., 2010a] to class-specific soft voting in the spirit of the fuzzy k-nearest neighbor (FNN) algorithm [Keller et al., 1985]. Introducing fuzziness is not only expected to enrich the classification by refining the confidence measures behind each decision but also often improves the overall accuracy. This makes it worth considering.

Other than in classification and retrieval [Tomašev and Mladenić, 2012], hubness has also been addressed in other data-mining tasks, as for example clustering [Tomašev et al., 2011c], anomaly detection [Tomašev and Mladenić, 2011], object recognition in images [Tomašev et al., 2011a] and instance selection (data reduction) [Buza et al., 2011].

The fact that hubness is among the most important aspects of the dimensionality curse in nearest-neighbor methods suggests that it certainly needs to be taken into account while designing new approaches. This is why we think that the hubness-aware design of the fuzziness measures for data labels in k-nearest neighbor classification might be advantageous and that is precisely what we will explore in this paper.

The rest of the paper is structured as follows. In Section 2 we present the related work, focused around two major points – the hubness-weighted kNN algorithm, and the FNN algorithm. While observing the former, we outline its

weak points and aim our proposed improvements in their direction. The respective hubness-based fuzzy membership functions are presented in Section 3. We go on to evaluate the proposed approach in Section 4. Finally, we give our final remarks and future research directions in Section 5.

2 Related work

2.1 Hubness-weighted kNN

Weighted voting in nearest-neighbor classifiers has become something of a common practice. Weights are usually either based on element position in the k-neighbor list or its distance to the observed data point. Some more robust approaches which take into account the correlation between these factors have also been recently developed [Zuo et al., 2008]. The hubness-weighting scheme which was first proposed in [Radovanović et al., 2009] is a bit more flexible, in a way that the weight associated to x_i is $w(x_i,k)$, meaning that each point in the training set has a unique associated weight, with which it votes whenever it appears in some k-neighbor list, regardless of its position in the list.

This weighting is based on the interpretation of how the hubness phenomenon affects kNN performance. As was mentioned before, hubness of an element x_i is the number of its k-occurrences in neighbor lists of other elements, and is denoted by $N_k(x_i)$. This can be decomposed into two parts: $N_k(x_i) = GN_k(x_i) + BN_k(x_i)$, where $GN_k(x_i)$ is the number of *good* k-occurrences and $BN_k(x_i)$ is the number of *bad* k-occurrences. Good occurrences are those when the label of x_i matches the label of the element in whose k-neighbor list x_i is observed. Bad occurrences are characterized by a mismatch of labels. Elements with high bad hubness are often found in neighbor lists of elements belonging to other categories in the data. This means that bad hubs exhibit a detrimental influence on k-nearest neighbor classification, because their vote often gives misleading information. Fig. 1 illustrates this point in a simple binary classification scenario. The aforementioned weighting scheme reduces these bad influences directly. Standardized bad hubness is defined as $h_b(x_i,k) = (BN_k(x_i) - \mu_{BN_k})/\sigma_{BN_k}$, where μ_{BN_k} is the mean bad hubness and σ_{BN_k} the standard deviation. The two parameters of the bad occurrence distribution are simply estimated from the training set as $\mu_{BN_k} = \frac{1}{N}\sum_{x_i \in D} BN_k(x_i)$ and $\sigma_{BN_k} = \sqrt{\frac{1}{N}\sum_{x_i \in D}(BN_k(x_i) - \mu_{BN_k})^2}$. The weight associated to x_i is then $w(x_i,k) = e^{-h_b(x_i,k)}$. It was shown that this often leads to significant improvements in high-dimensional settings where hubs naturally appear as an artefact of dimensionality. The amount of improvement depends on the distribution of bad hubness within the data.

What the described approach disregards completely is the structure of bad hubness. In non-binary classification, when a label mismatch occurs, it can occur in any of the class neighborhoods. Instead of observing $N_k(x_i)$ as a sum of good and bad hubness, we could decompose it with respect to individual classes into $N_k(x_i) = \sum_{c=1}^{n_c} N_{k,c}(x_i)$, where each $N_{k,c}(x_i)$ is the number of k-occurrences of x_i in neighborhoods of elements of class c, and n_c is the total number of classes. Good hubness is just the special case when $c = y_i$, y_i being the label of x_i in the data set. Therefore, instead of using the hubness information only to reduce the votes of bad hubs, it is possible to take into account the structure of bad hubness, which can be used to decompose the crisp vote given by x_i into a fuzzy vote relying on all $N_{k,c}(x_i)$. There already exists a framework that can assist in achieving this goal, referred to as the fuzzy nearest-neighbor classifier.

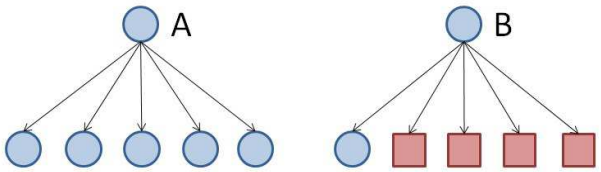


Fig. 1 An illustrative binary classification example. The instances of the two classes are depicted as circles and squares. An arrow indicates a nearest-neighbor relation, so that if it points from x_1 to x_2 this means that x_1 is a neighbor of x_2. We see that the two focal points, A and B, have the same overall hubness, $N_1(A) = N_1(B) = 5$. The nature of the influence of the two points is, on the other hand, entirely different. Point A seems to be a neighbor only to the points that share its own label, so we can be quite confident in its future votes. Point B behaves in quite the opposite way, and we can not be confident in its future votes. This is why the hubness-based weighting scheme is useful.

2.2 Fuzzy nearest neighbor algorithm

Fuzzy sets are based on a notion of inherent ambiguity in the data, meaning that a single element can be viewed as partially belonging to several different categories at the same time [A., 1965]. This ambiguity is often problem-specific and the set membership function is then provided by the domain experts. However, there are also ways of deducing some sort of fuzziness automatically from the data. Denote by $u_{ci} = u_c(x_i)$ the degree of membership of x_i in class c. The following properties must hold in order for u_c to define a fuzzy split on the data set:

$$\sum_{c=1}^{n_c} u_{ci} = 1,$$

$$0 < \sum_{i=1}^{n} u_{ci} < n,$$

$$u_{ci} \in [0,1].$$

The second and the third condition might seem equivalent, but in fact they are not, due to the strict inequality in

the second condition, which essentially guarantees that each class is non-empty. As for the first condition, that all class memberships for a given data point should sum up to 1, it is merely a convenience of scaling and u_c has been defined in such a way in previous work on using fuzzy labels in k-nearest neighbor classification [Keller et al., 1985]. We will, therefore, not argue with this definition, even though it is certainly possible to work with fuzzy measures where the first condition does not hold.

Let x be a newly observed data instance for which we wish to perform classification. Let $D_k(x) = x_1, ..x_k$ be its k nearest neighbors. The degree of membership of x in each class c is then defined as

$$u_c(x) = \frac{\sum_{i=1}^{k} u_{ci}(\|x - x_i\|^{-(2/(m-1))})}{\sum_{i=1}^{k}(\|x - x_i\|^{-(2/(m-1))})}, \quad (1)$$

where $\|\cdot\|$ denotes the Euclidean norm. The parameter m in Eq. 1 determines how heavily the distance is weighted when calculating contributions from each neighbor. For large values of m, neighbors are weighted more equally, while low values of m favor closer neighbors. The most commonly used default value for this parameter is $m = 2$, so that fuzzy votes are weighted by the reciprocal of the distance.

There exist many ways for automatically generating suitable fuzzy measures from the data. This is not only used for class membership fuzziness, but also for fuzzifying attributes. A range of techniques can be used, including genetic algorithms, clustering, neural networks, entropy, and others [Cintra et al., 2008]. In the original fuzzy nearest-neighbor article [Keller et al., 1985], some simple ways to achieve this were also proposed, one of which was to observe k nearest neighbors of x_i and count the percentages of them coming from any particular class. The final measure was a linear combination of the element's label and these percentages, normalized so as to fall in the desired $[0,1]$ range.

Apart from applying the fuzzy approach to specific domains [Cabello et al., 1991; Huang et al., 2007; Shen et al., 2006; Sim et al., 2005; Yu et al., 2002], most attention has been given lately to the issues of scalability in terms of achieving speedup in fuzzy nearest neighbor search [Babu and Viswanath, 2009; Zheng et al., 2010], as well as improving the weighting scheme [Pham, 2005].

3 Proposed hubness-based fuzzy measures

The basis of our motivation was already mentioned in Section 2.1 while discussing the properties of hubness-weighted kNN. Instead of using *good* and *bad* hubness, we propose to use *class hubness* $N_{k,c}(x_i)$ defined uniquely for each element in the training set. It is immediately apparent that this measure can be fit into the fuzzy nearest-neighbor framework. Contrary to the more usual fuzzy measures, it does not represent inherent fuzziness of an element's label, but instead measures the fuzziness of an *appearance* of elements in k-neighbor sets, based on the training data. Regardless of the semantic difference between the two, their form remains the same.

There are, however, some difficulties with using hubness as a fuzzy measure. For small values of k, there are many elements in the data which have zero hubness. This becomes even more pronounced in high dimensions due to the mentioned skew of the distribution of k-occurrences. Also, in non-binary classification problems, we need even more hubness information in order to be able to properly estimate the partial memberships for all the existing categories. This poses a serious limit on using class hubness for calculating fuzziness. We would be forced to use very high k values, which could be detrimental in cases when best kNN classification is achieved for smaller neighborhood sizes, as is often the case for non-noisy small or medium-sized data sets.

We propose to handle the problems outlined above by only using hubness of the elements which exhibit hubness greater than some predefined threshold. This in fact separates the data for which it is possible to make reliable fuzzy estimates from those that exhibit hubness too low to be of any use in such a manner. For the data below the threshold, we propose to use a different fuzzy estimate. We explore four such approaches and discuss the pros and cons of their use in the rest of this section, as well as analyze the fruitfulness of their application in Section 4 when presenting the results of the experimental evaluation. Let X be the training set and Y the set of corresponding labels. The hybrid fuzzy measure which we will be considering in the rest of the paper takes the following form:

$$u_c(x_i) = \begin{cases} p_k(y = c|x_i) \approx \frac{N_{k,c}(x_i)+\lambda}{N_k(x_i)+n_c\lambda}, & \text{if } N_k(x_i) > \theta, \\ f_k(c, x_i), & \text{if } N_k(x_i) < \theta. \end{cases}$$

The term $p_k(y = c|x_i)$ denotes the conditional probability of element x being of class c if element x_i appears in its k-neighbor set. For elements which exhibit hubness above a certain threshold, this can be estimated by dividing the class hubness by total hubness. The λ factor is a Laplace estimator, which is used for smoothing to prevent any probability from being estimated as zero. By observing the formula for the conditional probability, one can notice that the label y_i of x_i is not used at all when casting the vote of x_i! This is indeed a very peculiar property. Even though it is possible to work with fuzziness defined in such a way, we wanted to make the fuzziness also dependent on the element's label, so we included each x_i in its own neighbor list at the 0th position. For high-hubness elements, this does not make a large

difference, but by doing so we implicitly express a certain degree of confidence in label y_i.

The value of $f_k(c,x_i)$ for low-hubness elements should, ideally, represent a kind of estimate of the actual conditional probability. Since this is not easy to achieve, alternative nearest-neighbor based fuzzy estimates pose themselves as viable alternatives.

It should be noted that representing the neighbor occurrence fuzziness strictly in form of conditional probabilities is not entirely necessary. Fuzzy measures are in general not meant to be interpreted as probabilities. They are used to model uncertainty and are simply more adequate for modeling certain types of uncertainty than the probability theory [Singpurwalla and J.M., 2004][Wang et al., 2011]. In the context of neighbor occurrence models, this would imply that one can more easily extend the fuzzy framework, for instance by assigning different weights to different individual occurrences. The class specific weighted hubness then becomes $N_{k,c}(x_i) = \sum_{x:x_i \in D_k(x)} w_k(x,x_i)$ and the total weighted occurrence sum $N_k(x_i) = \sum_{c \in n_c} N_{k,c}(x_i)$. The weighting can be performed based on the distance between the neighbor points, as was recently demonstrated [Tomašev and Mladenić, 2011], but is not limited to that. Such weighted occurrence models are genuinely 'fuzzy', as they no longer try to estimate the $p_k(y = c|x_i)$.

We focused on four different ways of dealing with low hubness: a crisp estimate method, a global estimate method, as well as two different local estimates.

- What we refer to as the *crisp estimate* (CE) is the simplest and least flexible way of handling low hubness, which is not in itself necessarily bad – to use the element's own label. In this scenario, low-hubness elements vote the same way they would vote in kNN, with no attached fuzziness. Smoothing is performed by using the same λ value as before.
- *Global estimate* (GE) is more flexible, but introduces the risk of adding more fuzziness than necessary. We compute the GE of the conditional probability as defined in Eq. 2. The denominator represents the summation of $\sum_{(x,y)\in(X,Y)|y=y_i} \sum_{c=1}^{n_c} N_{k,c}(x)$. This is a sensible approach, but it remains questionable just how much is lost and how much is gained by employing it. Even though it does give a global conditional probability of elements from a particular class being included in neighbor sets of another class, there is no guarantee that locally, in the observed part of the data set, this estimate holds.

$$f_k(c,x_i) = \frac{\lambda + \sum_{(x,y)\in(X,Y)|y=y_i} N_{k,c}(x)}{n_c\lambda + \sum_{(x,y)\in(X,Y)|y=y_i} N_k(x)} \tag{2}$$

- If the global estimate fails to capture the proportions contained in the underlying conditional probability for a specific data instance, using a local fuzziness estimate is a possible alternative. Since we already have the k-neighbor lists, it seems natural to take advantage of this when trying to estimate an element's fuzziness. Here we depart from trying to estimate the actual conditional probability and experiment with a more usual approach. Let $\{x_{i1} \dots x_{ik}\}$ be the k nearest neighbors of x_i and for convenience denote x_i also as x_{i0}, since we insert each element into its neighbor list at the 0th position. The *local estimate* (LE_1) is then given by Eq. 3, where $\delta_{cy_{ij}}$ is Kronecker's delta function ($\delta_{cy_{ij}} = 1$ if $c = y_{ij}$ and 0 otherwise). This way, the $f_k(c,x_i)$ is defined as the proportion of examples from class c in the vicinity of the observed point, somewhat smoothed (λ). In a sense, it is a class density estimate. It is not entirely clear which value of k would work best in practice, as this depends on the local structure of the data. In our experiments we used a default neighborhood size of $k = 10$ when calculating the local estimate.

$$f_k(c,x_i) = \frac{\lambda + \sum_{j=0}^{k} \delta_{cy_{ij}}}{n_c\lambda + k + 1} \tag{3}$$

- There is an alternative way to define local fuzziness based on nearest neighbors and this was in fact one of the methods from the original FNN paper [Keller et al., 1985]. It is based on LE_1, but made so as to emphasize the label of an element, as in the CE method. In fact, it represents a linear combination of the two approaches. We will denote it LE_2, as defined in the following equation:

$$f_k(c,x_i) = \begin{cases} 0.51 + 0.49 \cdot \frac{\lambda + \sum_{j=1}^{k} \delta_{cy_{ij}}}{n_c\lambda + k + 1}, & \text{if } c = y_i, \\ 0.49 \cdot \frac{\lambda + \sum_{j=1}^{k} \delta_{cy_{ij}}}{n_c\lambda + k + 1}, & \text{if } c \neq y_i. \end{cases}$$

The factor of 0.51 was used for the label information simply to guarantee that $f_k(y_i,x_i) > f_k(y_j,x_i)$ for $i \neq j$. Any other $\alpha \in (0,1)$ could have in principle been used instead, whereas any $0.5 < \alpha < 1$ would have ensured that the majority of information comes from the label. This makes the LE_2 anti-hub estimate somewhat less fuzzy, but that is not necessarily a bad thing, as the primary goal is to ensure good classification accuracy.

Apart from testing these fuzzy measures separately, we have also merged them into a single hybrid hubness-based fuzzy nearest-neighbor algorithm which we present in Algorithm 1. Given the training data set, we use the leave-one-out procedure to try classifying each point x from the training data by observing the remaining $n-1$ elements. Such a classification is attempted for each element and for all the k values in a given range, as well as different threshold values and different $f_k(c,x_i)$. The configuration leading to the highest accuracy on the training data is then selected for use on the test set.

The time complexity of this approach is in fact completely comparable to the one of hubness-weighted kNN,

Algorithm 1 Hubness-based Fuzzy Nearest Neighbor: Training

```
int[][] NNs = computeNearestNeighborLists(k_min, k_max);
float[][][] classHubnessAllK = computeElementToClassHubness(NNs);
float[][][] GEAllK = computeGlobalEstimates(NNs);
float[][] LE_1 = computeLE1(NNs);
float[][] LE_2 = computeLE2(NNs);
float[][] CE = computeCE(NNs);
float maxAcc = 0;
int bestK, bestTheta;
for all θ = θ_min; θ ≤ θ_max; θ++ do
    for all k = k_min; k ≤ k_max; k++ do
        float GEAcc, LE1Acc, LE2Acc, CEAcc = 0;
        for all i = 0; i < trainingData.length; i++ do
            if votebyGE(x_i, GEAllK, ClassHubnessAllK, NNs) == x_i.label then
                GEAcc++;
            end if
            if votebyLE1(x_i, LE_1, ClassHubnessAllK, NNs) == x_i.label then
                LE1Acc++;
            end if
            if votebyLE2(x_i, LE_2, ClassHubnessAllK, NNs) == x_i.label then
                LE2Acc++;
            end if
            if votebyCE(x_i, CE, ClassHubnessAllK, NNs) == x_i.label then
                CEAcc++;
            end if
        end for
        updateMaxAccAndBestConfiguration(GEAcc, LE1Acc, LE2Acc, CEAcc);
    end for
end for
return The best parameter configuration and all the hubness estimates
```

with the bottleneck being the computation of k-neighbor sets. Fast approximate algorithms for calculating all k-neighbor sets do exist, one of the most recent being the one presented by Chen et al. [Chen et al., 2009]. This approximate algorithm runs in $\Theta(dn^{(1+\tau)})$ time, where $\tau \in (0,1]$ is a parameter used to set a trade-off between speed and accuracy. This makes hubness-based algorithms potentially feasible for use on large-scale data sets. We will present our initial results on the scalability of the proposed approach in Section 4.3.

We tested two versions of the algorithm shown in Algorithm 1. The first version uses the distance-based fuzzy vote weighting described in Eq. 1, which we denote by *dwh-FNN*. As an alternative we also tested a version of the algorithm where no distance-based weighting is performed, and fuzzy voting is achieved simply by summing all the respective u_{ci} for every class. This will be referred to as *h-FNN* in the rest of the text. The parameter m from Eq. 1 was set to 2 by default, this being the value which is most frequently used.

4 Experimental evaluation

This section presents the results of experiments that compare the standard k-nearest neighbor classifier and the hubness-weighted kNN with the two proposed hubness-based fuzzy approaches h-FNN and dwh-FNN. Section 4.1 deals with data sets of various dimensionalities from the established UCI repository, while Section 4.2 focuses on high-dimensional data from the image domain.

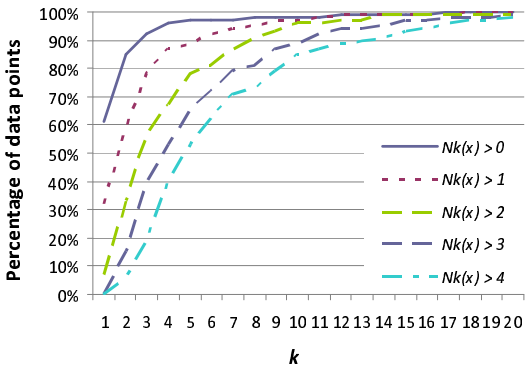
4.1 UCI data sets

Hubness-based fuzzy measures that we proposed are of a hybrid nature, since in each case they combine two different estimates. In order to see how different estimates might be applied, we calculated on each data set, for a range of neighborhood sizes, the percentage of data points which have hubness below/above a given threshold. For two of the used data sets, the plots of several lower thresholds for hubness can be seen in Fig. 2. Naturally, great variation of behavior can be observed across different data sets, since it is related to the aforementioned skew of the hubness distribution in high dimensions. In other words, we expect for highly skewed data sets the term $f_k(c,x_i)$ to play a more important role than in the case of low to medium-skewed data with respect to hubness. It is precisely for these data sets that the mentioned estimates of actual hubness may become as important as hubness itself. From Fig. 2, however, the difference becomes quite clear. For the less skewed data sets, if a good classification can be achieved for a neighborhood size of $k \in [10,20]$ or above, then there is probably enough hubness information to allow for its straightforward use as a fuzzy measure. If, on the other hand, the nature of the data is such that the best results are obtained for low k values, ranging maybe from 1 to 5, the situation is reversed. When dealing with highly skewed data, such as in the case of the Dexter data set, influence of $f_k(c,x_i)$ is non-negligible even when considering higher k values.

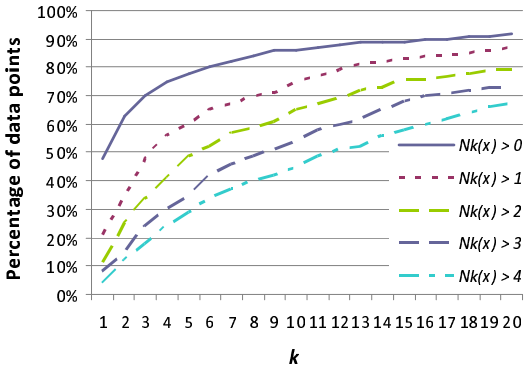
The first round of testing was performed on 15 data sets taken from the UCI data repository. The used data sets are of various sizes and dimensionalities, and are summarized in Table 1, with the first six columns denoting data-set name, size, dimensionality (d), number of classes (n_c), and the observed skewness of the distributions of N_1 and N_{10} (S_{N_1}, $S_{N_{10}}$).[1] For each data set, the skew of the distribution of k-occurrences was calculated for various k values, to indicate the degree of hubness of the data. Euclidean distance was used in all the experiments.

On the described UCI data sets, kNN, hubness-weighted kNN, h-FNN and dwh-FNN were tested. In all the algorithm tests, 10 runs of 10-fold cross-validation were performed. All algorithm parameters were set automatically, separately on each fold during the training phase, based on the training set. Neighborhood sizes were tested in the range $k \in [1,20]$ and thresholds $\theta \in [0,10]$. Classification accuracies achieved by the classifiers are given in Table 2. The corrected resampled t-test [Nadeau and Bengio, 2003] was used to test for statistical significance of differences in accuracy for each data set. Differences which were found to

[1] Skewness, the standardized 3rd moment of a probability distribution, is 0 if the distribution is symmetrical, while positive (negative) values indicate skew to the right (left).



(a) Iris data set



(b) Dexter data set

Fig. 2 Percentage of elements with hubness exceeding a certain threshold, for neighborhood sizes $k \in \{1..20\}$

Table 1 Summary of UCI datasets

Data set	size	d	n_c	S_{N_1}	$S_{N_{10}}$
colonTumor	62	2000	2	1.04	1.06
dexter	300	20000	2	2.95	3.33
diabetes	768	8	2	0.73	0.15
ecoli	336	7	8	0.62	0.37
glass	214	9	6	0.58	0.23
ionosphere	351	34	2	2.17	1.71
iris	150	4	3	0.46	0.03
isolet-1	1560	617	26	1.30	1.20
mfeat-fourrier	2000	76	10	1.20	0.75
ozone-eighthr	2534	72	2	1.31	0.70
page-blocks	5473	10	5	0.79	0.11
parkinsons	195	22	2	0.39	−0.19
segment	2310	19	7	0.70	0.16
vehicle	846	18	4	0.92	0.44
yeast	1484	8	10	0.78	0.27

be significant with $p < 0.01$ compared to dwh-FNN are denoted by symbols ○/● in the table.

The dwh-FNN classifier was selected as the baseline for statistical comparison in Table 2 since we determined that it generally outperformed all other classifiers. To provide a more detailed pairwise classifier comparison, Table 3 shows the number of wins of classifiers signified by the column label, over classifiers denoted by the row labels, with statistically significant wins given in parenthesis.

Table 2 Classification accuracy of kNN, hubness-weighted kNN (hw-kNN), h-FNN and dwh-FNN on UCI data sets. The symbols ○/● denote statistically significant better/worse performance than dwh-FNN

Data set	kNN	hw-kNN	h-FNN	dwh-FNN
colonTumor	65.1±19.6 ●	72.5±20.6	74.9±20.0	74.5±20.0
dexter	60.1±18.2 ●	72.5± 7.9 ○	68.6± 8.3	68.5± 8.3
diabetes	76.5± 4.1 ○	72.0± 4.6 ●	74.2± 4.9	74.2± 4.9
ecoli	85.4± 6.0	84.5± 6.4	83.6± 6.4	84.3± 6.3
glass	70.5± 9.3 ○	67.6±10.0 ○	65.4± 9.9 ○	63.8±10.0
ionosphere	89.7± 5.2	87.5± 5.7 ●	89.9± 5.5	90.0± 5.6
iris	96.9± 4.0 ○	95.3± 4.8	95.1± 4.7	94.7± 4.8
isolet-1	90.0± 2.6 ○	81.3± 3.4 ●	81.2± 3.8 ●	82.3± 3.6
mfeat-fourier	77.5± 2.9 ●	80.3± 2.6 ●	81.0± 2.6 ●	81.9± 2.6
ozone-eighthr	76.8± 2.5 ●	93.4± 1.8	93.4± 1.3	93.6± 1.3
page-blocks	93.5± 1.0 ●	96.0± 0.8	96.1± 0.8	96.2± 0.8
parkinsons	82.7± 7.7 ●	92.1± 5.8	92.5± 5.2	92.7± 5.2
segment	89.9± 1.7 ●	91.2± 1.7	90.8± 1.8 ●	91.2± 1.8
vehicle	60.7± 5.7 ●	66.6± 5.1	64.4± 4.9	65.2± 5.6
yeast	59.0± 4.1 ○	52.3± 4.1 ●	55.1± 3.8	55.5± 3.8
Average	78.29	80.34	80.41	80.57

Table 3 Pairwise comparison of classifiers on UCI data: number of wins (with the statistically significant ones in parenthesis)

	kNN	hw-kNN	h-FNN	dwh-FNN
kNN	–	8 (8)	9 (8)	9 (8)
hw-kNN	7 (6)	–	9 (4)	10 (5)
h-FNN	6 (6)	6 (2)	–	11 (3)
dwh-FNN	6 (5)	5 (2)	4 (1)	–

Overall improvement over kNN is apparent already from the shown average scores over all data sets in Table 2, as well as Table 3. Particular improvements vary and there do exist data sets for which none can be observed, as well as some where performance degradation is present. Hubness-weighted kNN, h-FNN and dwh-FNN exhibit similar improvement patterns, which makes sense given that they aim at exploiting the same phenomenon. Improvement over the standard kNN classifier signifies that there is a lot of usable bad-hubness information in the data. Fuzzy approaches appear to offer additional improvement over hw-kNN, justifying our approach and the need to differentiate between classes when employing bad hubness for nearest-neighbor classification. The cases where standard kNN is significantly better than hubness-based approaches most probably stem from the difficulties of estimating $p_k(y = c|x_i)$, which requires more data in the case of non-binary classification, as well as $f_k(c, x_i)$ occasionally being an inappropriate substitute in cases of low hubness.

It appears that the distance-based weighting from Eq. 1 does not bring drastic overall improvement to the hubness-based fuzzy membership functions that are used in the h-FNN algorithm, at least not for the default value of the m parameter. This is not all that surprising, though. As was stated in previous discussion, the semantics of hubness-based fuzziness differs slightly from that of more usual fuzzy measures. This is due to the fact that class hubness marks the fuzziness of the elementary event that point x_i appears in a k-neighbor set of an element of some specific category. This hubness is estimated by previous appearances of that element in k-neighbor sets of various other elements in the training data. Among these occurrences, x_i may be located at either place within each observed k-neighbor set. In other words, hubness is a measure which is for a fixed k independent of which positions in k-neighbor sets an element takes. If these lists were to undergo a random permutation, the hubness for that fixed neighborhood size would have remained unchanged.

Let us assume that we wish to determine the label of a new example x by using h-FNN. The contribution of those x_i closer to x stems not only from previous events when they were also close to the observed element, but also from previous events when they were much farther away. The same holds for farther elements in the k-neighbor set. This is why a linear combination of class hubness contributions is sufficient and any additional distance-based weighting seems superfluous. On the other hand, due to the fact that we can not calculate proper class-hubness probabilities for low-hubness elements, this is only partially true. In cases where fuzziness is estimated for low-hubness x_i, distance-based weighting might bring some improvement by emphasizing more important votes. In practice, most k-neighbor sets will probably contain a mixture of these cases.

Initial comparisons between the different hubness-based fuzzy membership functions proposed in Section 3 were also performed. Experiments were rerun without automatic parameter selection on the folds, so that the algorithms were trained once for every combination of $k \in [1,20]$ and $\theta \in [0,4]$, for every proposed fuzzy scheme. We extracted the parameter values from the range where the algorithms achieved highest accuracy scores, based again on the 10 times 10-fold cross-validation procedure, for every data set. Averages of k values for which the best results were obtained are shown for every used fuzzy scheme in Fig. 3. For each fuzzy approach, lower k values were selected on average if no distance-based vote weighting was performed. This suggests that if the distance weighting is performed, more neighbors are required to convey the same amount of information, due to some votes being downgraded. Different measures attain their best scores at different k-values, as suggested by the observed frequencies. In particular, the global hubness-based fuzziness (GE) finds its maximum at lower k-values than other measures. It is a useful property, as less time is required to perform all the computations when k is smaller. However, the average best accuracies for all the approaches were basically the same. This suggests that hubness itself is still the most important part of the hybrid fuzziness and that anti-hubs can be handled in any of the proposed ways, without significantly affecting the overall performance, at least in medium hubness data (UCI). We will re-evaluate the differences between the anti-hub estimates on high-hubness image data in Section 4.2. As for the threshold parameter, the average θ value for which the best accuracy was achieved was around 1.5 for all approaches. This means that more often than not, class hubness was to be preferred to any of the $f_k(c,x_i)$ terms, even when based only on 3 or 4 k-occurrences.

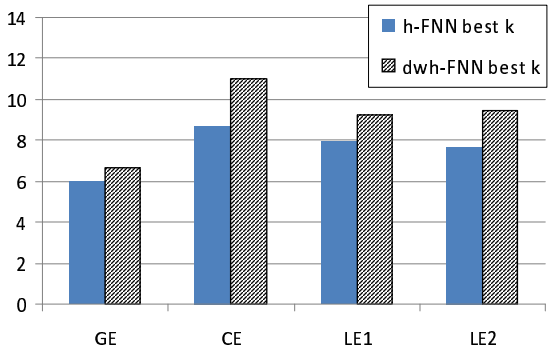


Fig. 3 Average best k values for different hubness-based fuzzy approaches, according to the results from tests on UCI data

The frequencies of the selected neighborhood size falling in one of the four ranges: $[1,5]$, $[6,10]$, $[11,15]$, $[16,20]$, are shown in Fig. 4. Two ranges are preferred more often, namely $k \in [1,5]$ and $k \in [11,15]$. By examining all the results, we found that in cases of the more tangible accuracy improvements, larger k values ($k > 10$) were selected, while lower k values usually signified equal or only slightly better performance. This can be seen as natural, since larger k values provide the algorithm with more hubness information and hence better probability estimates, on which the used fuzziness was based. However, not all data sets are such that high k values make sense, since in some it may induce a larger breach of locality. This is why hubness-based approaches are not expected to lead to an improvement over all data sets. This is their inherent limitation. Of course, this also depends heavily on the size of a particular data set. With more data, higher k values can be observed more safely. In high-dimensional spaces this is also affected by the curse of dimensionality because the data is always sparse.

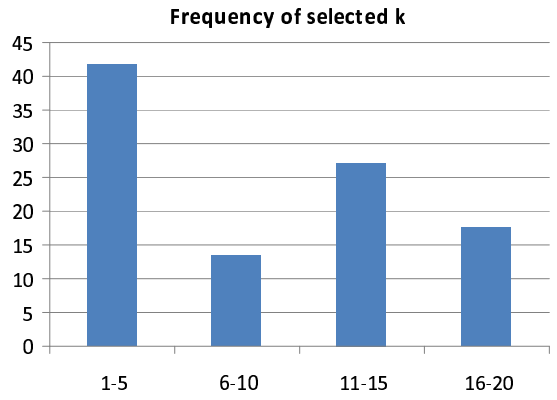


Fig. 4 Frequency of the selected best k values, based on the results from tests on UCI data

4.2 ImageNet data

The ImageNet database (`http://www.image-net.org/`) is a large repository containing over 12 million images organized in more than 17000 synsets (classes). Images are intrinsically high-dimensional data, and are therefore quite suitable for testing hubness-based approaches. Out of synsets from the ImageNet hierarchy we constructed five image data sets for testing, with the used classes summarized in Table 4. Some of them combine more easily distinguishable images, as *subs-3*, while some are made more difficult by containing several different plant types in different categories, as in *subs-6*. SIFT features and color histograms were extracted for each image [Zhang and Zhang, 2009]. A codebook of 400 most representative SIFT features was obtained by clustering from a large sample. Each image was thus represented by a 400-dimensional array of codebook frequencies, as well as a 16-dimensional color histogram. We used the Manhattan distance on this group of data sets. No feature weighting was performed, meaning that color and texture information was given equal significance. This may not be optimal, but we were not interested in performing optimal image classification, since our goal was only to compare the approaches under consideration on high-dimensional data. As in the previous section, Table 5 gives an overview of the obtained data sets. Note that this data exhibits a much higher skew of the distribution of k-occurrences than most UCI data sets from Table 1.

On each of the subsamples we performed 10 times 10-fold cross-validation. The value of k was chosen automatically from the range $k \in [1,10]$ on each fold. Average accuracies of the classifiers are given in Table 6. Statistically significant differences ($p < 0.05$) compared to dwh-FNN are denoted by symbols ∘/•. Pairwise classifier comparison is shown in Table 7.

Table 4 Class structure of the used ImageNet data subsamples

Data set	Classes
subs-3	sea moss, fire, industrial plant
subs-4	cloud, butterfly orchid, herbaceous plant, bird
subs-5	bird, fire, tracked vehicle, people, compass flower
subs-6	fish, industrial plant, wind turbine, compass flower, butterfly orchid, evergreen plant
subs-7	football, worm, sea star, night club, cloud, orchidaceous plant, mountain range

Table 5 Summary of ImageNet data sets

Data set	size	d	n_c	S_{N_1}	$S_{N_{10}}$
subs-3	2731	416	3	15.85	6.19
subs-4	6054	416	4	8.87	6.32
subs-5	6555	416	5	26.08	11.88
subs-6	6010	416	6	13.19	6.23
subs-7	8524	416	7	5.62	4.60

Table 6 Classification accuracy of kNN, hubness-weighted kNN (hw-kNN), h-FNN and dwh-FNN on ImageNet data sets for $k \in [1,10]$. The symbol • denotes statistically significant worse performance compared to dwh-FNN

Data set	kNN	hw-kNN	h-FNN	dwh-FNN
subs-3	78.29±2.38 •	81.51±3.34	82.16±2.26	82.34±2.23
subs-4	54.68±2.02 •	65.91±1.82	64.83±1.62	64.87±1.61
subs-5	50.80±2.08 •	58.06±3.80 •	61.54±1.93	61.81±1.95
subs-6	63.09±1.81 •	70.10±1.68	68.84±1.58	69.04±1.64
subs-7	46.71±1.63 •	51.99±4.68 •	58.85±1.60	59.04±1.59
Average	54.71	65.51	67.24	67.42

Table 7 Pairwise comparison of classifiers on ImageNet data: number of wins (with the statistically significant ones in parenthesis)

	kNN	hw-kNN	h-FNN	dwh-FNN
kNN	–	5 (5)	5 (5)	5 (5)
hw-kNN	0 (0)	–	3 (2)	3 (2)
h-FNN	0 (0)	2 (0)	–	5 (0)
dwh-FNN	0 (0)	2 (0)	0 (0)	–

Hubness-based algorithms show an obvious improvement on all subsets over the standard kNN classifier. As the number of classes increases, improvement of h-FNN and dwh-FNN over hubness-weighted kNN becomes more prominent, which is consistent with observations on UCI data.

In Section 4.1 we reported a brief comparison of the proposed fuzzy measures on medium-hubness UCI data, which revealed that all of them attain similar best accuracies, though for different k-values, when averaged over all the datasets. In Fig. 5 we focus on the comparison when varying the values of the threshold θ parameter. Higher θ values increase the influence of the $f_k(c,x_i)$ terms, while the lower threshold values emphasize the original point class-hubness fre-

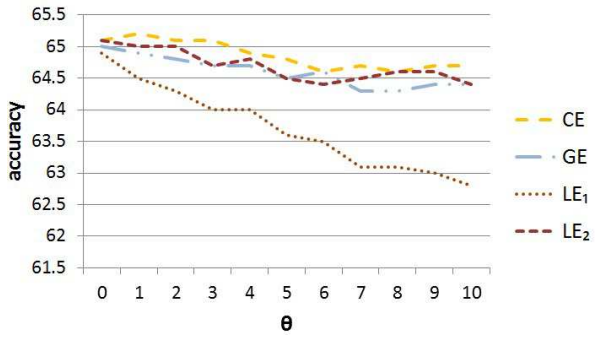


Fig. 5 A comparison between the fuzzy measures on subs-4.

quencies, even when derived from very few occurrences. A comparison is shown on subs-4, one of the high-hubness ImageNet datasets that we have analyzed.

As in the previous case of medium-hubness data, the best accuracies are observed for low θ parameter values and the best results for all measures are very similar. However, more differences can be observed as the θ is slowly increased and the $f_k(c,x_i)$ terms get more frequently used during voting. First of all, one notices that there is a clear difference between the performance of the two local estimates (LE_1 and LE_2) on this particular image dataset. In fact, LE_1 seems to be clearly inferior to LE_2, which is not surprising, given that it relies more on the crisp label than LE_1, and the crisp handling of the anti-hubs (CE) works best on this dataset.

The fact that all the measures achieve similar best scores and that this always takes place for low θ values makes the task of choosing the appropriate measure and the appropriate threshold much easier in practice. By setting $\theta = 0$ or $\theta = 1$ and by using either of the CE, GE or LE_2 estimate methods, one could hope to achieve very good results. This is important, as it essentially removes the need for performing cross-validation when doing the search for the best parameter configuration. It is a very time consuming step and removing it helps speed up the algorithm. This may not be very important for small datasets, but most real-world datasets are quite large and scalability is certainly important.

Noisy and compromised data, on the other hand, need to be handled somewhat more carefully. Most measurement errors, noisy points and outliers tend to be anti-hubs, though the reverse implication does not necessarily hold. This means that unreliable points would tend to have low hubness in most complex, real-world datasets. The negative influence of erroneous points could be reduced by setting a slightly higher threshold ($\theta > 1$) and relying more on the global class-to-class hubness estimate (GE) for handling such anti-hubs. It ought to be more reliable in noisy data scenarios than CE or LE_1 and LE_2, as the labels of such potentially incorrect data points are often wrong and the neighbors might be quite distant and less relevant for estimating the local occurrence fuzziness. If the data is not prohibitively large, it is still advisable to look for the best parameter configuration automatically during the training phase, as outlined in the algorithm 1.

4.3 Scalability

One of the most important issues in modern data-mining tasks is scalability, since we are mostly faced with problems involving big data. Algorithms that perform well in terms of accuracy, but scale poorly, are not really useful in most practical applications. This is why we decided to test how the proposed h-FNN and dwh-FNN perform under approximate *k*NN set calculation, which is used to speed up the procedures. We chose the approximate *k*NN graph construction algorithm described in [Chen et al., 2009], which is a divide and conquer method based on recursive Lanczos bisection. As mentioned before, the time complexity of the procedure is $\Theta(dn^{1+\tau})$, where $\tau \in (0,1]$ reflects the quality of the approximation. The main question which arises is: for which values of τ could we still retain the good performance observed on the actual *k*NN sets? Fig. 6 shows some encouraging results.

We see that the hubness-based approaches seem to be quite robust to approximate calculation of the *k*NN sets on the data, at least we could say that is the case for this particular employed approximate algorithm [Chen et al., 2009]. Improvements over the baseline *k*NN remain even for $\tau = 0$, which essentially means that the total overhead over the basic *k*NN can be reduced to linear time complexity, which is excellent. We also see that dwh-FNN remains better than hw-*k*NN in all approximate cases, which implies that the relative strengths and weaknesses of the approaches remain unchanged under such conditions.

The exact algorithm (with no approximations) is of the squared time complexity (and memory requirements) which makes it applicable to most medium-to-large real world

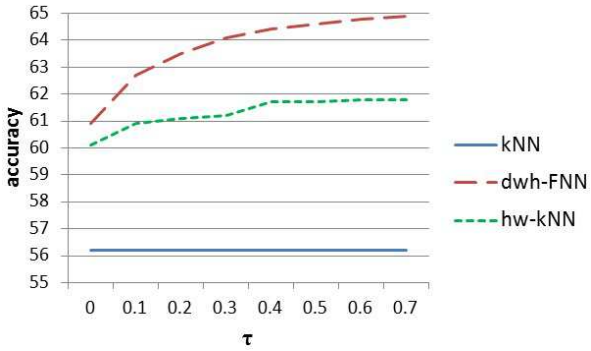


Fig. 6 The accuracy of the hubness-based approaches on subs-4 when the occurrence model is inferred from the approximate *k*NN graph generated by [Chen et al., 2009]. We see that there are significant improvements even for $\tau = 0$

datasets, though it may have difficulties handling *very* large datasets. On the other hand, constructing a *k*NN graph (in order to calculate the hubness scores) is among those tasks that can be easily solved by distributed computing. We are also using some initial multi-threaded implementations. As for the single-threaded implementation, the overall performance of h-FNN and dwh-FNN is essentially the same as in hw-*k*NN, since both algorithms spend most of the training time on calculating the distance matrix and all the *k*NN sets. The additional time required to summarize the class-hubness scores and/or make some local or global estimates for anti-hubs is negligible when compared to the two main sub-tasks.

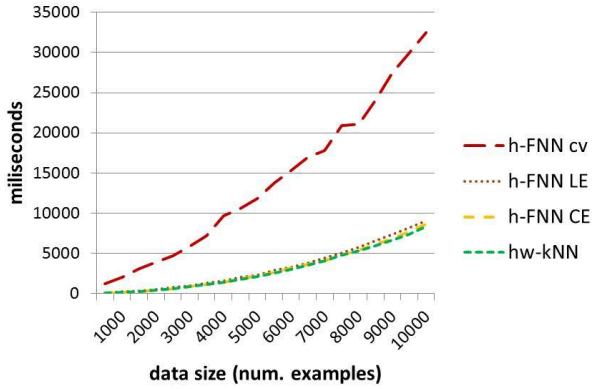


Fig. 7 The execution times of the training phase of hw-*k*NN, h-FNN employing CE/GE or LE_1/LE_2 and h-FNN performing cross-validation on the training set to decide on the best fuzzy measure and parameter set-up. $k = 5$ was used in the experiment. *k*NN is omitted, as it requires no training. All experiments were performed on a computer with an i7 Intel processor and 8Gb RAM.

In order to compare the approaches, we have generated a series of synthetic 100-dimensional Gaussian mixtures and we have measured the training time of each of the methods separately. According to Fig. 7, h-FNN and hw-*k*NN take almost the same amount of time for the training phase, while the most time consuming approach is to use the cross-validation in h-FNN or dwh-FNN in order to try and find the best fuzzy measure and the best parameter configuration (θ, M). Fortunately, as we have already discussed in Section 4.2, it seems that this is not really necessary in practice and that it is not so difficult to come up with good *default* parameters which ought to work well on most datasets. All curves in Fig. 7 do not intersect and the ordering remains the same as data size is increased: t(h-FNN cv) $>$ t(h-FNN LE) $>$ t(h-FNN CE) $>$ t(hw-*k*NN), though the differences between the last three are apparently minor. In other words, the improvement that h-FNN and dwh-FNN achieve over hw-*k*NN is essentially *free*, from the perspective of time complexity.

4.4 Probability landscapes

When estimating the potential benefits of using a particular classification algorithm, accuracy is not the only quantity of interest. We would also like the algorithm to be able to provide us with decent confidence measures behind its label assignments, which would provide the experts using the system with valuable additional information. Fuzzy approaches are, well, more 'fuzzy' and 'soft' to begin with, so they always do output some sort of a confidence measure alongside the final vote. The question remains: how *good* are these associated numbers?

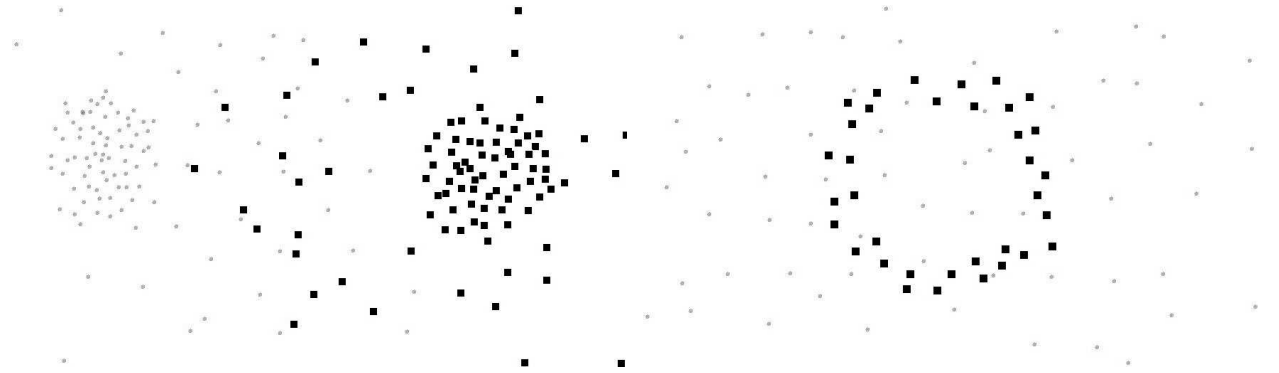
A complete analysis of the associated probabilities in all conceivable scenarios would be quite difficult and is certainly beyond the scope of this paper. We will, however, shed some light on the quality of the employed fuzzy measures by analyzing a couple of illustrative examples. We will consider the two-dimensional synthetic data sets shown in Fig. 8. We opted for 2D data so that we can easily visualize the results.

For the two data sets DS_1 and DS_2, shown in Fig. 8, we computed the probability landscapes in the following way: we performed a fifth-order Voronoi tessellation in the plane ($k = 5$) and then assigned a class probability to every pixel in each of the obtained cells by each of the considered algorithms (*k*NN, hw-*k*NN, h-FNN, dwh-FNN).

The probability landscapes generated for DS_1 are shown in Fig. 9. It is immediately apparent that *k*NN produces a fractured landscape, which indicates over-fitting. When there are many more points and a higher k value can be safely used, this is less of a problem. Real-world data, however, are not two-dimensional and are hence always sparse, much more so than in the considered DS_1 data set. This suggests that the basic *k*NN can not be expected to give reasonable probability estimates in such scenarios. The hubness-based weighting apparently helps, even though there is no hubness in two dimensions. However, it still reduces the votes of some less reliable borderline points. The hubness-based fuzzy approaches produce landscapes that are even more smooth, which seems like a nice property for a model of the data.

As for the second, ring-shaped data set, the associated probability landscapes are shown in Fig. 10. Once again we see that the basic *k*NN classifier over-fits on certain points and fails to detect a common theme. The hubness-based fuzzy *k*-nearest neighbor classifier (h-FNN) gives the most reasonably-looking result and hw-*k*NN lies somewhere in between the two.

The hubness-based *k*NN algorithms discussed in this paper are not designed to model the data directly, but are able to capture some of the underlying regularities in the data by virtue of building an occurrence model. We have observed some encouraging results on two-dimensional synthetic data sets. However, investigating the overall performance in the



(a) DS_1 (b) DS_2

Fig. 8 Two 2D synthetic binary datasets. The first one depicts a case of two overlapping Gaussian-like distributions with dense central regions and sparse border regions. The second example shows a ring-like distribution immersed into a rather uniform background distribution, which could even be interpreted as noise

general case is not as easy, therefore we can only assume for now that these observations may generalize to the high-dimensional case as well. In a sense, it can be considered a reasonable assumption, since both of these algorithms have been tailored specifically for high-dimensional data in the first place and the very fact that they perform very well in the low-dimensional case is an unexpected beneficial property of the algorithms.

5 Conclusions and future work

We have proposed several ways of incorporating hubness into fuzzy membership functions for data points in *k*NN classification. This was meant as a generalization of the previous hubness-weighted *k*NN approach. The fuzzy *k*-nearest neighbor classification offers better confidence measures for label assignments, which is a highly desirable property.

Several hybrid fuzzy membership functions were tested and evaluated. The fuzzy *k*-nearest neighbor classifier employing these fuzzy measures outperforms the basic *k*NN classifier and also offers improvement over the crisp hubness-weighted *k*NN. The accuracy improvement thus achieved may not be large on average, but the main advantage of the fuzzy approach lies in the mentioned interpretability of the results, and the fact that the approach takes advantage of high intrinsic dimensionality of the data instead of being hampered by it, taking a step closer to mitigating the curse of dimensionality.

The approach seems to be quite scalable when the approximate *k*NN sets are used instead. Most of the original accuracy is retained when opting for such speedup in computation.

These fuzzy measures represent but one way of exploiting the hubness phenomenon for classification. There are yet other paths to follow and we intend to address many of the related issues in our future work.

Acknowledgements This work was supported by the bilateral project between Slovenia and Serbia "Correlating images and words: Enhancing image analysis through machine learning and semantic technologies," the Slovenian Research Agency, the Serbian Ministry of Education and Science through project no. OI174023, "Intelligent techniques and their integration into wide-spectrum decision support," and the ICT Programme of the EC under PASCAL2 (ICT-NoE-216886) and PlanetData (ICT-NoE-257641).

References

A., Z. L. Fuzzy sets. *Information and Control* 338–353 (1965).

Aggarwal, C. C.; Hinneburg, A.; Keim, D. A. On the surprising behavior of distance metrics in high dimensional spaces. In: *Proc. 8th Int. Conf. on Database Theory (ICDT)*. 420–434 (2001).

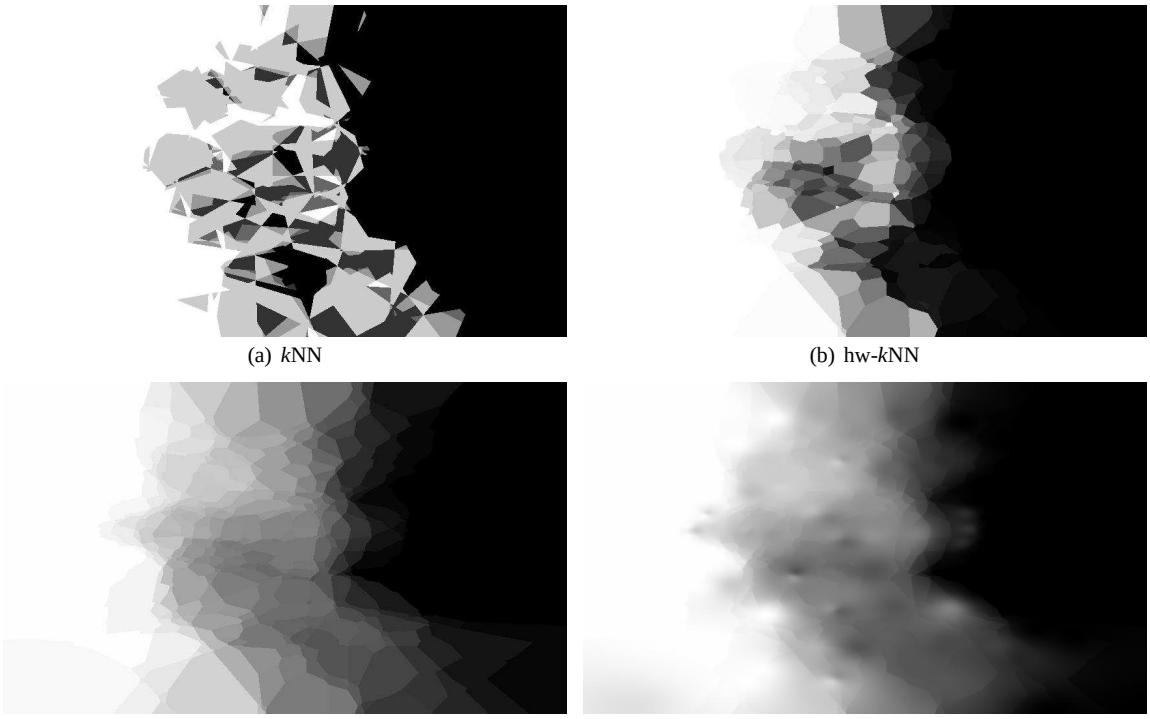
Aucouturier, J. Ten experiments on the modelling of polyphonic timbre. *Technical Report*, Docteral dissertation, University of Paris 6 (2006).

Aucouturier, J.; Pachet, F. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences* **1** (2004).

Babu, V. S.; Viswanath, P. Rough-fuzzy weighted k-nearest leader classifier for large data sets. *Pattern Recognition* **42**, 1719 – 1731 (2009).

Buza, K.; Nanopoulos, A.; Schmidt-Thieme, L. Insight: efficient and effective instance selection for time-series classification. In: *Proceedings of the 15th Pacific-Asia conference on Advances in knowledge discovery and data mining - Volume Part II*. 149–160, PAKDD'11 (Springer-Verlag, 2011).

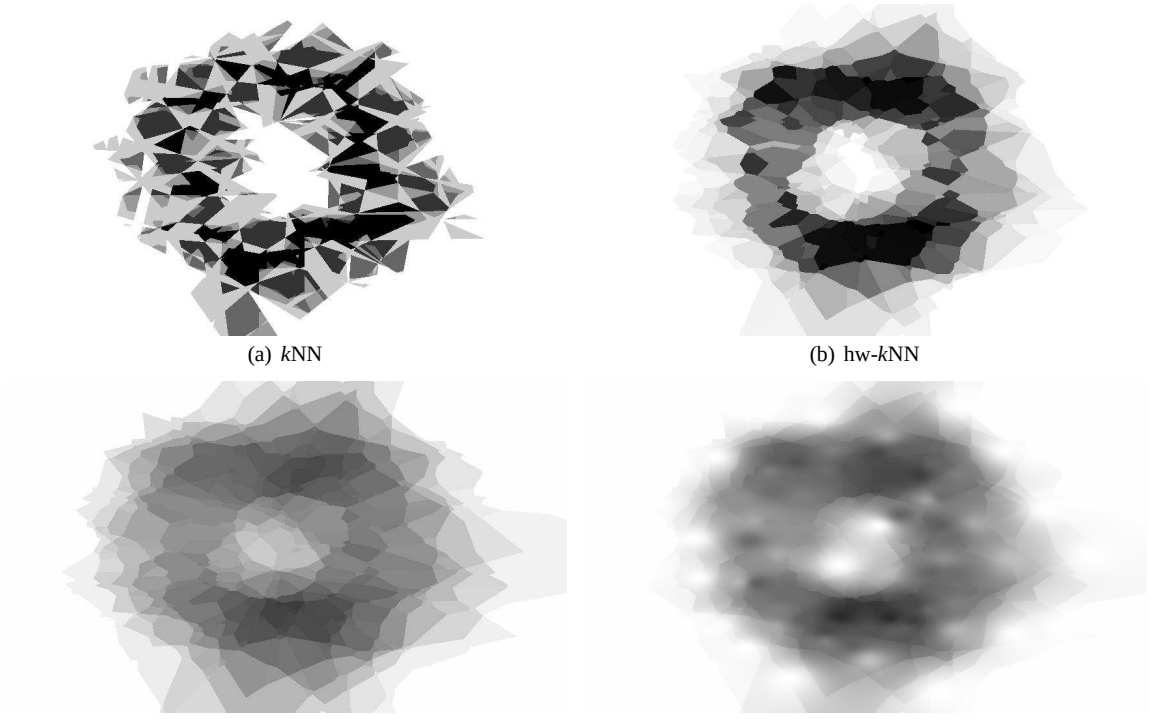
Cabello, D.; Barro, S.; Salceda, J. M.; Ruiz, R.; Mira, J. Fuzzy k-nearest neighbor classifiers for ventricular arrhythmia detection. *International Journal of Bio-Medical Computing* **27**, 77–93 (1991).



(a) *k*NN (b) hw-*k*NN

(c) h-FNN (d) dwh-FNN

Fig. 9 The probability landscapes of the analyzed algorithms on DS_1 for $k = 5$. The difference between different approaches is quite obvious, as both hw-*k*NN and the fuzzy hubness-based approach produce much smoother landscapes than the basic *k*NN



(a) *k*NN (b) hw-*k*NN

(c) h-FNN (d) dwh-FNN

Fig. 10 The probability landscapes of the analyzed algorithms on DS_2 for $k = 5$

Chen, J.; ren Fang, H.; Saad, Y. Fast approximate *k*NN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research* **10**, 1989–2012 (2009).

Cintra, M. E.; de Arruda Camargo, H.; Monard, M. C. A study on techniques for the automatic generation of membership functions for pattern recognition. In: *III Congresso da Academia Trinacional de Cincias* (2008).

Durrant, R. J.; Kabán, A. When is 'nearest neighbour' meaningful: A converse theorem and implications. *Journal of Complexity* **25**, 385–397 (2009).

François, D.; Wertz, V.; Verleysen, M. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering* **19**, 873–886 (2007).

Houle, M. E.; Kriegel, H.-P.; Kröger, P.; Schubert, E.; Zimek, A. Can shared-neighbor distances defeat the curse of dimensionality? In: *Proceedings of the 22nd international conference on Scientific and statistical database management*. 482–500, SSDBM'10 (Springer-Verlag, Berlin, Heidelberg, 2010).

Huang, W.-L.; Chen, H.-M.; Hwang, S.-F.; Ho, S.-Y. Accurate prediction of enzyme subfamily class using an adaptive fuzzy k-nearest neighbor method. *Biosystems* **90**, 405–413 (2007).

Keller, J. E.; Gray, M. R.; Givens, J. A. A fuzzy k-nearest-neighbor algorithm. In: *IEEE Transactions on Systems, Man and Cybernetics*. 580–585 (1985).

Nadeau, C.; Bengio, Y. Inference for the generalization error. *Machine Learning* **52**, 239–281 (2003).

Pham, T. D. An optimally weighted fuzzy k-nn algorithm. In: Singh, S.; Singh, M.; Apte, C.; Perner, P. (eds.) *Pattern Recognition and Data Mining*. **3686**, 239–247, 239–247, 239–247 (Springer Berlin / Heidelberg, 2005).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In: *Proc. 26th Int. Conf. on Machine Learning (ICML)*. 865–872 (2009).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* **11**, 2487–2531 (2010a).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. On the existence of obstinate results in vector space models. In: *Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 186–193 (2010b).

Radovanovic, M.; Nanopoulos, A.; Ivanovic, M. Time-series classification in many intrinsic dimensions. In: *SDM*. 677–688 (SIAM, 2010).

Shen, H.-B.; Yang, J.; Chou, K.-C. Fuzzy KNN for predicting membrane protein types from pseudo-amino acid composition. *Journal of Theoretical Biology* **240**, 9–13 (2006).

Sim, J.; Kim, S.-Y.; Lee, J. Prediction of protein solvent accessibility using fuzzy k-nearest neighbor method. *Bioinformatics* **21**, 2844–2849 (2005).

Singpurwalla, N.; J.M., B. Membership functions and probability measures of fuzzy sets. *Journal of the American Statistical Association* **99**, 867 – 877 (2004).

Tomašev, N.; Brehar, R.; Mladenić, D.; Nedevschi, S. The influence of hubness on nearest-neighbor methods in object recognition. In: *Proceedings of the 7th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. 367–374 (2011a).

Tomašev, N.; Mladenić, D. Exploring the hubness-related properties of oceanographic sensor data. In: *Proceedings of the SiKDD conference* (2011).

Tomašev, N.; Mladenić, D. The influence of weighting the k-occurrences on hubness-aware classification methods. In: *Proceedings of the SiKDD conference* (Institut "Jozef Stefan", Ljubljana, 2011).

Tomašev, N.; Mladenić, D. Nearest neighbor voting in high dimensional data: Learning from past occurrences. *Computer Science and Information Systems* **9**, 691–712 (2012).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. In: *Proc. MLDM* (2011b).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. In: *PAKDD (1)'11*. 183–195 (2011c).

Wang, X.-Z.; He, Y.-L.; Dong, L.-C.; Zhao, H.-Y. Particle swarm optimization for determining fuzzy measures from data. *Information Sciences* **181**, 4230 – 4252 (2011).

Yu, S.; Backer, S. D.; Scheunders, P. Genetic feature selection combined with composite fuzzy nearest neighbor classifiers for hyperspectral satellite imagery. *Pattern Recognition Letters* **23**, 183–190 (2002).

Zhang, Z.; Zhang, R. *Multimedia Data Mining* (Chapman and Hall, 2009), 1st edition.

Zheng, K.; Fung, P. C.; Zhou, X. K-nearest neighbor search for fuzzy objects. In: *Proceedings of the 2010 international conference on Management of data*. 699–710, SIGMOD '10 (ACM, New York, NY, USA, 2010).

Zuo, W.; Zhang, D.; Wang, K. On kernel difference-weighted k-nearest neighbor classification. *Pattern Analysis and Applications* **11**, 247–257 (2008).

2.2.2 An Information-Theoretic Perspective

In the previously discussed hubness-based fuzzy k-nearest neighbor algorithm (h-FNN) [Tomašev et al., 2013b], all fuzzy votes are given the same weight. This is not necessarily the best approach and this observation was exploited for in the development of the hubness information k-nearest neighbor classifier (HIKNN) [Tomašev and Mladenić, 2011c][Tomašev and Mladenić, 2012] that is discussed below. This Section presents the results of the paper titled *Nearest Neighbor Voting in High Dimensional Data: Learning from Past Occurrences* that was published in *Computer Science and Information Systems* journal in 2012 [Tomašev and Mladenić, 2012].

Some class affiliation information in the point of interest is contained in the labels of its nearest neighbors and some can be derived from the class-conditional k-occurrence profiles, as demonstrated in h-FNN [Tomašev et al., 2013b]. The idea behind the HIKNN classifier is that rarely occurring points hold information that is somewhat more locally relevant for estimating the class distribution in the point of interest. This is a geometric consequence of hubs being somewhat closer to distribution centers that exhibit 'average' properties. On the other hand, the k-occurrence profiles can not be reliably estimated for those rarely occurring points, while good estimates are available for hubs occurrence profiles. This has lead us to propose a hybrid approach where the trade-off between label information and occurrence profile information is set based on the self-information *surprise factors* of the observed neighbor occurrences. The experimental evaluation shows that this might indeed be a promising approach.

UDC, DOI: N/A

Nearest Neighbor Voting in High Dimensional Data: Learning from Past Occurrences

Nenad Tomašev and Dunja Mladenić

Artificial Intelligence Laboratory, Jožef Stefan Institute and
Jožef Stefan International Postgraduate School
1000 Ljubljana, Slovenia
nenad.tomasev@ijs.si, dunja.mladenic@ijs.si

Abstract. *Hubness* is a recently described aspect of the *curse of dimensionality* inherent to nearest-neighbor methods. This paper describes a new approach for exploiting the hubness phenomenon in k-nearest neighbor classification. We argue that some of the neighbor occurrences carry more information than others, by the virtue of being less frequent events. This observation is related to the hubness phenomenon and we explore how it affects high-dimensional k-nearest neighbor classification. We propose a new algorithm, Hubness Information k-Nearest Neighbor (HIKNN), which introduces the k-occurrence informativeness into the hubness-aware k-nearest neighbor voting framework. The algorithm successfully overcomes some of the issues with the previous hubness-aware approaches, which is shown by performing an extensive evaluation on several types of high-dimensional data.

1. Introduction

Supervised learning (classification) is one of the most fundamental machine learning tasks, often encountered in various practical applications. It involves assigning a label to a new piece of input data, where the label is one out of several predefined categories. Many algorithmic approaches to performing automatic classification have been explored in the past. This includes, among others, Bayesian learning methods, support vector machines (SVM), decision trees and nearest neighbor methods [Witten and Frank, 2005].

The k-nearest neighbor algorithm is one of the simplest pattern classification algorithms. It is based on a notion that instances which are judged to be similar in the feature space often share common properties in other attributes, one of them being the instance label itself. The basic algorithm was first proposed

[0] Published by the ComSIS Consortium in Computer Science and Information Systems
DOI: 10.2298/CSIS111211014T
Original article available at: http://www.comsis.org/archive.php?show=ppr406-1112
This is an extended version of the conference paper presented at the PhD forum of ICDM conference 2011 [Tomašev and Mladenić, 2011]

in [Fix and Hodges, 1951]. The label of a new instance is determined by a majority vote of its k-nearest neighbors (kNN) from the training set. This simple rule has some surprising properties which go in its favor. For instance, when there is no overlap between the classes, 1-nearest neighbor is asymptotically optimal [T.M.Cover and P.E.Hart, 1967][Devroye, 1981]. As for the kNN rule, it has been shown to be universally consistent under some strong assumptions, namely $k \to \infty$ and $k/n \to 0$ [C.J.Stone, 1977][L. Devroye and Lugosi, 1994].

Let $D = (x_1, y_1), (x_2, y_2), ..(x_n, y_n)$ be the data set, where each $x_i \in R^d$. The x_i are feature vectors which reside in some high-dimensional Euclidean space, and $y_i \in c_1, c_2, ..c_C$ are the labels. It can be shown that in the hypothetical case of an infinite data sample, the probability of a nearest neighbor of x_i having label c is asymptotically equal to the posterior class probability in point x_i, namely $p(c|x_i) = \lim_{n\to\infty} p(c|\text{NN}(x_i))$. Real-world data is usually very sparse, so the point probability estimates achieved by kNN in practice are much less reliable. However, this is merely one aspect of the well known *curse of dimensionality*.

Concentration of distances [Aggarwal et al., 2001; François et al., 2007] is another phenomenon of interest, since all nearest-neighbor approaches require a similarity measure. In high-dimensional spaces, it is very difficult to distinguish between relevant and irrelevant points and the very concept of nearest neighbors becomes much less meaningful.

Hubness is a recently described aspect of the dimensionality curse, related specifically to nearest neighbor methods [Radovanović et al., 2009][Tomašev et al., 2011d]. The term is coined to reflect the emergence of *hubs*, very frequent nearest neighbors. As such, these points exhibit a substantial influence on the classification outcome. Two types of hubs can be distinguished: *good hubs* and *bad hubs*, based on the proportion of label matches/mismatches in their k-occurrences. The phenomenon of *hubness* will be explained in more detail in Section 2.2, and the previous approaches for exploiting hubness in kNN classification will be outlined in Section 2.3.

The issue of data dimensionality needs to be emphasized because most real world data sets are in fact high-dimensional, for example: textual documents, images, audio files, data streams, medical histories, etc.

1.1. Contributions

This paper aims at further clarifying the consequences of hubness in high dimensional kNN classification, by focusing on one specific aspect of the phenomenon - the difference in the information content of the individual k-occurrences. Here we summarize the main contributions of the paper:

- When there is hubness, some points occur much more frequently in k-neighbor sets. We claim that some occurrences hence become much less informative than others, and are consequently of much lower value for the kNN classification process.
- We propose a new hubness-aware approach to k-nearest neighbor classification, Hubness Information k-Nearest Neighbor (HIKNN). The algorithm

exploits the notion of occurrence informativeness, which leads to a more robust voting scheme.
– We provide a thorough experimental evaluation for the approach, by testing it both on low-to-medium hubness data and also high-hubness data from two different domains: images and text. The experiments are discussed in Section 5, while Section 7 takes a deeper look into the class probabilities which the algorithm returns.

2. Related work

2.1. kNN classification

The k-nearest neighbor method is among the most influential approaches in machine learning, due to its simplicity and effectiveness. Many extensions to the basic method have been proposed, dealing with various different aspects - including attribute weighting [Han et al., 2001], adaptive distances [Wang et al., 2007][Song et al., 2007], fuzzy labels [Keller et al., 1985][Jensen and Cornelis, 2008][Shang et al., 2006], evidence-theoretic approaches [Wang et al., 2008], and many more. Some advanced algorithms have been proposed recently, including the large margin kNN classifier which learns the Mahalanobis distance matrices via semidefinite programming [Weinberger et al., 2006][Min et al., 2009].

2.2. Hubs, frequent nearest neighbors

The emergence of *hubs* as prominent points in k-nearest neighbor methods had first been noted in analyzing music collections [Aucouturier and Pachet, 2004][Aucouturier, 2006]. The researchers discovered some songs which were similar to many other songs (i.e. frequent neighbors). The conceptual similarity, however, did not reflect the expected perceptual similarity.

The phenomenon of *hubness* was further explored in [Radovanović et al., 2009][Radovanović et al., 2010a], where it was shown that hubness is a natural property of many inherently high-dimensional data sets. Not only do some very frequent points emerge, but the entire distribution of k-occurrences exhibits very high *skewness*. In other words, most points occur very rarely in k-neighbor sets, less often than what would otherwise have been expected. We refer to these rarely occurring points as *anti-hubs*.[Radovanović et al., 2010b]

Denote by $N_k(x_i)$ the number of k-occurrences of x_i and by $N_{k,c}(x_i)$ the number of such occurrences in neighborhoods of elements from class c. The latter will also be referred to as the *class hubness* of instance x_i. A k-neighborhood of x_i is denoted by $D_k(x_i)$.

The skewness of the $N_k(x)$ distribution in high dimensional data can sometimes be very severe [Radovanović et al., 2010a]. Let us illustrate this point by plotting the $N_k(x)$ distribution for one of the datasets which we used for the experiments, namely the Acquis data. This is shown in Figure 1, for $k = 5$. Such a

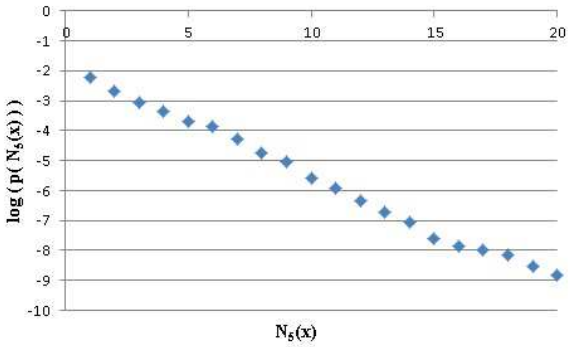


Fig. 1. The hubness distribution of the Acquis data is given for the 5-occurrence probabilities of $N_5(x) \in \{1..20\}$. We see that the distribution apparently forms a straight line on the logarithmic scale, so it is in fact exponential.

drastic shift in the distribution shape must certainly be taken into account when designing kNN algorithms for high dimensional data.

Hubness-aware algorithms have recently been proposed for clustering [Tomašev et al., 2011d], instance selection [Buza et al., 2011], outlier and anomaly detection [Radovanović et al., 2010a][Tomašev and Mladenić, 2011] and classification [Radovanović et al., 2009][Tomašev et al., 2011b][Tomašev et al., 2011c][Tomašev et al., 2011a], which we will discuss below.

2.3. Hubness-aware classification

Hubs, as frequent neighbors, can exhibit both *good* and *bad* influence on kNN classification, based on the number of label *matches* and *mismatches* in the respective k-occurrences.The number of good occurrences will be denoted by $GN_k(x_i)$ and the number of bad ones by $BN_k(x_i)$, so that $N_k(x_i) = GN_k(x_i) + BN_k(x_i)$.

All three previously proposed approaches deal with bad hubs in seemingly similar, but radically different ways. We will refer to these algorithms as hubness-weighted kNN (hw-kNN) [Radovanović et al., 2009], hubness fuzzy kNN (h-FNN) [Tomašev et al., 2011b] and naive hubness Bayesian kNN (NHBNN) [Tomašev et al., 2011c]. We discuss these ideas below, outlining their respective strengths and weaknesses.

hw-kNN

– **Idea:** When a point exhibits bad hubness, give its vote lesser weight. This has been achieved by calculating the standardized bad hubness as $h_B(x_i) = \frac{BN_k(x_i) - \mu_{BN_k}}{\sigma_{BN_k}}$, where μ_{BN_k} and σ_{BN_k} denote the mean and standard deviation of bad hubness, respectively. Each x_i is then assigned a voting weight of $w_i = e^{-h_B(x_i)}$.

- **Strengths:**
  - Reduces the influence of bad hubs
  - Very simple and easy to implement
- **Weaknesses:**
  - Each element still votes by its own label, which means that bad hubs still exhibit some detrimental influence
  - Some information is left unexploited, since class hubness is ignored
  - It is equivalent to kNN for $k = 1$

h-FNN

- **Idea:** Decompose bad hubness into fuzzy class-specific hubness-based votes as $u_c(x_i) \propto N_{k,c}(x_i)/N_k(x_i)$. This is only possible for points with $N_k(x_i) > 0$ and only sensible for points with $N_k(x_i) > \theta$, where θ is some predefined threshold parameter. Anti-hubs are thus considered to be special cases. Their fuzzy votes are approximated by average class-to-class fuzzy votes. This algorithm is otherwise based on the fuzzy nearest neighbor (FNN) framework [Keller et al., 1985], with distance weighting included.
- **Strengths:**
  - Generalizes the hw-kNN approach by taking class hubness into account
  - Combines fuzzy votes with distance weighting
- **Weaknesses:**
  - No clear way of dealing with anti-hubs, approximations need to be used instead
  - Uses a threshold parameter θ for determining anti-hubs, which is difficult to set in practice. If learned automatically from the data, it can lead to over-fitting.

NHBNN

- **Idea:** Observe each k-occurrence as a random event and use the Naive Bayes rule to calculate the posterior class affiliation probabilities, as shown in Equation 1. The x_{it}, $t = \{1, 2..k\}$ represent the k nearest neighbors of x_i. As in h-FNN, anti-hubs are a special case and one needs to estimate their class hubness scores via local or global approximative approaches.

$$p(y_i = c|D_k(x_i)) \approx \frac{p(y_i = c)\prod_{t=1}^{k} p(x_{it} \in D_k(x_i)|y_i = c)}{\sum_{c \in C} p(y_i = c)\prod_{t=1}^{k} p(x_{it} \in D_k(x_i)|y_i = c)} \quad (1)$$

- **Strengths:**
  - Generalizes the hw-kNN approach by taking class hubness into account
  - Rephrasing the problem in Bayesian terms allows for further improvements and extensions based on the known ways for improving Bayesian classifiers

– **Weaknesses:**
  - Strong dependencies between occurrences in the same k-neighbor set greatly restrict the applicability of the approach in larger k-neighborhoods
  - Due to these dependencies, class affiliation probabilities tend to be close to 0 or 1 in many cases.
  - Additionally, both weaknesses of h-FNN hold for NHBNN as well

3. The motivation

3.1. Casting a vote: label vs class hubness

Before we delve into the specific ideas behind our proposed approach, the reasons for using the class hubness scores need to be further elucidated. For simplicity, let us begin by focusing on the 1-NN rule. It was already mentioned in the introduction that $p(c|x_i) = \lim_{n\to\infty} p(c|\text{NN}(x_i))$. If the data were not sparse and if there was no overlap between the classes and no noise, 1-NN would work really well. Of course, none of these conditions are met in real world data.

So, what happens is that nearest neighbors sometimes have different labels and this can already be seen on the training set. Observe an illustrative low-dimensional example displayed in Figure 2.

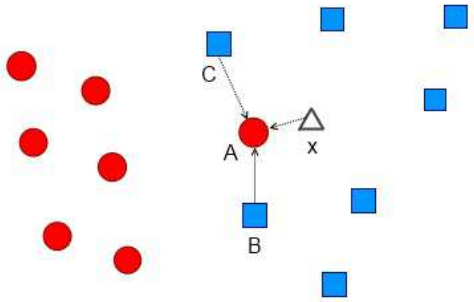


Fig. 2. Illustrative example of a binary classification case. The first class is given by the red circles, the second by the blue squares. A triangle represents an instance yet to be classified. An arrow is drawn from an instance to its nearest neighbor.

The point x is about to be classified. Let's say that the circles represent class 0 and squares represent class 1. According to the 1-NN rule, x would be assigned to class 0, since this is the label of $\text{NN}(x) = A$. But, we also have $\text{NN}(B) = A$ and $\text{NN}(C) = A$, and points B and C are of class 1. If we were to try approximating $p(y = c|A \in D_1(x)) \approx \frac{N_{1,c}(A)}{N_1(A)}$ for $c = 0, 1$, we would get

$p(y = 0|A \in D_1(x)) = 0$ and $p(y = 1|A \in D_1(x)) = 1$. So, according to class hubness, x should be assigned to class 1, which seems more plausible when looking at the data.

Two-dimensional data does not exhibit hubness, so Figure 2 can only serve as a simplified model. A more general case is presented in Figure 3. Two examples are given, with class hubness scores shown on the right. In both examples, the label of x is 0 (the red circle).

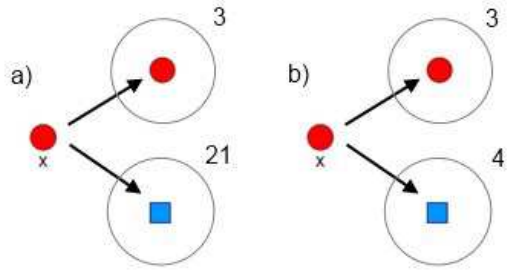


Fig. 3. A more general binary classification case. Class hubness is shown for point x towards both classes. Two examples are depicted, example 'a' where there is a big difference in previous k-occurrences, and example 'b' where there is nearly no observable difference.

In the first example, $N_{1,0}(x) = 3$ and $N_{1,1}(x) = 21$, which indicates high bad hubness. Therefore, if x is a neighbor to the point of interest, it is certainly beneficial to base the vote on the class hubness scores, instead of its label. It would reduce the probability of error.

In the second example, $N_{1,0}(x) = 3$ and $N_{1,1}(x) = 4$, which makes for a very small difference in class hubness scores. Even though $N_{1,1}(x) > N_{1,0}(x)$, the label of x is 0, so it is not entirely clear how x should vote. What needs to be evaluated is how much trust should be placed in the neighbor's label and how much in the occurrence information. If there had been no previous occurrences of x on the training data (an anti-hub), there would be no choice but to use the label. On the other hand, for high hubness points we should probably rely more on their occurrence tendencies. It is precisely the points in between which need to be handled more carefully.

Anti-hubs While discussing the relevance of hubness for kNN classification, we must keep in mind that most points are in fact anti-hubs, when the inherent dimensionality of the data is high. This is illustrated in Figure 4, where the percentage of points exceeding certain k-occurrence thresholds is given. The Dexter data (from the UCI repository) exhibits some hubness, so that even for k as large as 10, there is still around 15% of instances that never occur as neighbors.

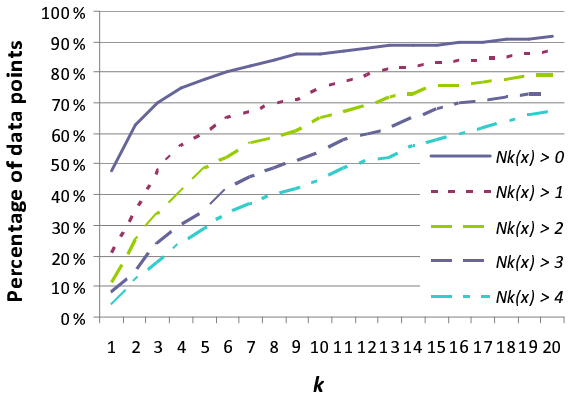


Fig. 4. Percentage of elements with hubness over a certain threshold, for $k = 1$ to $k = 20$ on Dexter data. Each line corresponds to one threshold.

Both previously proposed class-hubness based approaches (h-FNN and NHBNN) have failed to provide an easy and consistent way of handling anti-hubs, which is probably their most pronounced weakness. In Section 4 we propose a new way of dealing with such low hubness points.

3.2. Informativeness

The basics What is the information content of an observed event? Intuitively, the more surprised we are about the outcome, the more information the outcome carries. We're all quite used to the sun coming up every morning and by observing this over and over again we don't gain any novel insights. If, however, the sun fails to appear on the sky someday, such a peculiar event would be much more informative, though unfortunate.

This is where information theory comes in. The event self-information is equal to the negative of the logarithm of its probability (i.e. the logarithm of the inverse of the probability) [MacKay, 2002]. It is often possible to estimate the event probabilities directly by observing the frequencies in previous occurrences, which is what we will be doing with the neighbor points.

Hubs Suppose that there is a data point $x_i \in D$ which appears in *all* k-neighbor sets of other $x_j \in D$. Assume then that we are trying to determine a label of a new data point x and that x_i also appears in this neighborhood, $D_k(x)$. This would not be surprising at all, since x_i appears in *all* other previously observed neighborhoods. Since such an occurrence *carries no information*, x_i should not be allowed to cast a vote. By going one step further, it is easy to see that less frequent occurrences may in fact be more informative and that such neighbors might be more local to the point of interest. This is exploited in our proposed approach.

Going back to the *always-a-neighbor* example, we can see that both the traditional kNN voting scheme and the fuzzy scheme proposed in the h-FNN algorithm fail to handle the extreme case properly. The fact is that whichever point x we observe, $x_i \in D_k(x)$, so there is no correlation between x_i being in $D_k(x)$ and the class affiliation of x. In case of the original kNN procedure, x_i would vote by its label, y_i. If, on the other hand, we were to vote by the class hubness induced fuzziness as in h-FNN, we would in fact be voting by class priors. This is, of course, the lesser evil, but it is still the wrong thing to do. Since there is no information that can be derived from the occurrence of x_i, its vote should be equal to zero.

This scenario does seem quite far-fetched. When reviewing the experimental results, though, it will become clear that such pathological cases are not only theoretically possible - they occasionally take place in real world data.

Anti-hubs Most high-dimensional points are anti-hubs and suppose that x_i is one such point that never occurs in k-neighborhoods on D, i.e. $N_k(x_i) = 0$. Let us say that we are trying to determine the label of a new point x and x_i is found among the neighbors, i.e. $x_i \in D_k(x)$. Such an occurrence would be highly informative. We could be fairly certain that the point x_i carries some important *local* information to the point of interest, since it is not a shared neighbor with many other points.

Of course, not all points are hubs and anti-hubs, as many points will fall somewhere between the two extremes. Any approach designed to handle the informativeness hubs and anti-hubs needs to be applicable to the entire spectrum of possible occurrence frequencies, so that these medium-hubness points are processed in an appropriate way.

It is quite surprising that these simple observations have before gone unnoticed. Previous kNN algorithms have not been taking occurrence informativeness explicitly into consideration.

This is very significant for high dimensional data, where hubs appear. The skewness in the $N_k(x)$ distribution induces the skewness in the distribution of self-information among individual neighbor occurrences. In the following Section we will propose an information-based voting procedure which exploits this fact.

4. The algorithm

Let now x_i be the point of interest, to which we wish to assign a label. Let x_{it}, $t = \{1, 2..k\}$ be its k nearest neighbors. We calculate the informativeness of the occurrences according to Equation 2. In all our calculations, we assume each data point to be its own 0^{th} nearest neighbor, thereby making all $N_k(x_i) \geq 1$. Not only does this give us some additional data, but since it makes all k-occurrence frequencies non-zero, we thereby avoid any pathological cases in our calculations.

$$p(x_{it} \in D_k(x_i)) \approx \frac{N_k(x_{it})}{n}$$
$$I_{x_{it}} = \log \frac{1}{p(x_{it} \in D_k(x_i))} \tag{2}$$

We proceed by defining relative and absolute normalized informativeness. We will also refer to them as *surprise values*.

$$\alpha(x_{it}) = \frac{I_{x_{it}} - \min_{x_j \in D} I_{x_j}}{\log n - \min_{x_j \in D} I_{x_j}}, \quad \beta(x_{it}) = \frac{I_{x_{it}}}{\log n} \tag{3}$$

As we have been discussing, one of the things we wish to achieve is to combine the class information from neighbor labels and their previous occurrences. In order to do this, we need to make one more small observation. Namely, as the number of previous occurrences ($N_k(x_{it})$) increases, two things happen simultaneously. First of all, the informativeness of the current occurrence of x_{it} drops. Secondly, class hubness gives us a more accurate estimate of $p_k(y_i = c|x_{it} \in D_k(x_i))$. Therefore, when the hubness of a point is high, more information is contained in the class hubness scores. Also, when the hubness of a point is low, more information is contained in its label.

$$\bar{p}_k(y_i = c|x_{it} \in D_k(x_i)) = \frac{N_{k,c}(x_{it})}{N_k(x_{it})} = \bar{p}_{k,c}(x_{it})$$
$$p_k(y_i = c|x_{it}) \approx \begin{cases} \alpha(x_{it}) + (1 - \alpha(x_{it})) \cdot \bar{p}_{k,c}(x_{it}), & y_{it} = c \\ (1 - \alpha(x_{it})) \cdot \bar{p}_{k,c}(x_{it}), & y_{it} \neq c \end{cases} \tag{4}$$

The α factor controls how much information is contributed to the vote by the instance label and how much by its previous occurrences. If x_{it} never appeared in a k-neighbor set apart from its own, i.e. $N_k(x_{it}) = 1$, then it votes by its label. If, on the other hand, $N_k(x_{it}) = \max_{x_j \in D} N_k(x_j)$, then the vote is cast entirely according to the class hubness scores.

The fuzzy votes are based on the $p_k(y_i = c|x_{it})$, which are approximated according to Equation 4. These probabilities are then weighted by the absolute normalized informativeness $\beta(x_{it})$. This is shown in Equation 5.

$$u_c(x_i) \propto \sum_{t=1}^{k} \beta(x_{it}) \cdot d_w(x_{it}) \cdot p_k(y_i = c|x_{it}) \tag{5}$$

Additional distance weighting has been introduced for purposes of later comparison with the h-FNN algorithm [Tomašev et al., 2011b], since it also employs distance weighting. It is not an essential part of the algorithm. We opted for the same distance weighting scheme used in h-FNN, which was in turn first proposed in FNN [Keller et al., 1985]. It is given in Equation 6.

$$d_w(x_{it}) = \frac{\|x_i - x_{it}\|^{-2}}{\sum_{t=1}^{k} (\|x_i - x_{it}\|^{-2})} \tag{6}$$

Equations 2, 3, 4 and 5 represent our proposed solution for exploiting the information contained in the past k-occurrences on the training data and we will refer to this new algorithm as Hubness Information k-Nearest Neighbor (HIKNN). It embodies some major improvements over the previous approaches:

- Unlike h-FNN and NHBNN, it is essentially parameter-free, one only needs to set the neighborhood size (k).
- Anti-hubs are no longer a special case. They are, however, handled appropriately via the information-based framework.
- Label information is combined with information from the previous k-occurrences, so that both sources of information are exploited for the voting.
- Total occurrence informativeness is taken into account

The training phase of the algorithm is summarized in (1). The voting is simply done according to (5) and requires no further explanations.

Algorithm 1 HIKNN: Training

Input: (X,Y,k)
training set $T = (X, Y) \subset R^{d \times 1}$
number of neighbors $k \in \{1, 2 \dots n-1\}$

Train:
kNeighbors = findNeighborSets(T, k)
for all $(x_i, y_i) \in (X, Y)$ **do**
  $N_k(x_i) = 0$
  for all $c = 1 \dots C$ **do**
    count $N_{k,c}(x_i)$
    $N_k(x_i) += N_{k,c}(x_i)$
  end for
  calculate $\alpha(x_i)$ and $\beta(x_i)$ by Eq. 3
  for all $c = 1 \dots C$ **do**
    calculate $p_k(y = c|x_i)$ by Eq. 4
  end for
end for

The time complexity of HIKNN, as with all other hubness-based approaches, is asymptotically the same as constructing a kNN graph. Fast algorithms for constructing approximate kNN graphs exist, like the algorithm by [Chen et al., 2009]. This particular procedure runs in $\Theta(dn^{1+\tau})$ time, where $\tau \in (0, 1]$ is a parameter which is used to set a trade off between speed and graph construction accuracy.

5. Experiments

We compared our proposed HIKNN algorithm to the existing related algorithms: kNN, hw-kNN, h-FNN and NHBNN - on 32 classification problems. We had three

test cases: low-to-medium hubness data of lower intrinsic dimensionality, high-hubness textual data and high-hubness image data. In all cases, 10-times 10-fold cross validation was performed. Corrected resampled t-test was used to check for statistical significance. All experiments were performed for $k = 5$, which is a standard choice. Default values described in the respective papers were used for the parameters in h-FNN and NHBNN. The detailed results are given in Table 3 and the basic properties of the datasets are discussed in Table 2.

5.1. The data

Low and medium hubness data Datasets from the well known UCI data repository (http://archive.ics.uci.edu/ml/datasets.html) are usually of low or medium hubness. Since such datasets are less interesting from the perspective of hubness-aware algorithms, we present here the results on a sample of 10 UCI datasets. The datasets were picked so that they correctly reflect the results on the entire repository. The Manhattan distance was used for this data, as well as for the image data. All features were normalized prior to classification.

Text The Acquis aligned corpus data (http://langtech.jrc.it/JRC-Acquis.html) represents a set of more than 20000 documents in several different languages. In the experiments we only used the English documents. The data was preprocessed and represented as a bag-of-words (term frequencies). On top of this data, 14 different binary classification problems were examined. We used the cosine similarity.

Images We used several datasets in the experiments which were subsets taken from the ImageNet online repository (http://www.image-net.org/). These datasets were selected to match some of the ones used in [Tomašev et al., 2011a]. All datasets are quantized feature representations. Representations iNet3-iNet7 are based on SIFT features and were also appended color information.

On the other hand, in case of iNet3Err100, iNet3Err150 and iNet3Err1000 - Haar wavelet features were used. These three representations have one interesting property. Due to an I/O error during feature extraction, 5 images were accidentally assigned empty representations (zero vectors). Normally, this would have probably gone unnoticed. In this case, however, the hubness of zero vectors increased drastically with the representation dimensionality. Since all 5 of these points were of the minority class, the classification results were affected greatly and this a prime example of how bad the bad hubness can get in high dimensional data.

5.2. The results

The results in Table 3 show that the hubness-aware algorithms clearly outperform the basic kNN algorithm. Also, HIKNN seems to be the overall best ap-

Table 1. Pairwise comparison of classifiers: number of wins (with statistically significant ones in parenthesis)

	kNN	hw-kNN	h-FNN	NHBNN	HIKNN	Total
kNN	–	0 (0)	1 (0)	10 (8)	0 (0)	11 (8)
hw-kNN	32 (27)	–	14 (1)	18 (13)	1 (0)	65 (41)
h-FNN	31 (27)	17 (9)	–	25 (16)	5 (1)	79 (53)
NHBNN	22 (19)	14 (9)	4 (1)	–	3 (1)	43 (30)
HIKNN	**32 (31)**	**29 (14)**	**27 (9)**	**27 (20)**	–	**115 (74)**

proach, with a clear edge on the textual and UCI data, while performing more or less equal as h-FNN and NHBNN on image datasets. The detailed comparison between the algorithms is shown in Table 1. By comparing both the total number of wins and also the number of wins between pairs of algorithms, we see that HIKNN is to be preferred to the second-best algorithm in the experiments, h-FNN - since it beats it quite convincingly 27(9) : 5(1) in direct comparison and 115(74) : 79(53) overall.

In further comparisons on the image data, we examined the entire range of k-values to see how the algorithms are influenced by neighborhood size. The results on iNet6 are shown in Figure 5. We see that an increase in k separates the algorithms and makes distinctions easier. HIKNN achieves the best results for $k > 5$, where the highest accuracies are achieved. It is not surprising that the accuracy gain over h-FNN increases with k, since the number of large hubs also increases - and the payoff from taking their informativeness into account becomes more substantial. Also, we see that NHBNN simply fails to work when the dependencies between neighbors become too strong, in this case for $k > 15$. The accuracy graphs for the other datasets depict the same general tendencies.

The results for the three iNet3Err representations require special attention. As mentioned in the data description, 5 points in the dataset ended up being zero vectors representing the minority class. We see how an increase in the representation dimensionality causes an amazing increase in bad hubness, which in turn completely disables the basic kNN classifier, as well as the hw-kNN approach. On this 3-category dataset kNN ends up being worse than zero-rule! Keep in mind that such a great drop in accuracy was caused by no more than 5 erroneous instances, out of 2731 total. In the end, 80% of the 5-occurrences were label mismatches. On the other hand, the algorithms based on class hubness: h-FNN, NHBNN and HIKNN - even though affected, retained a much more decent accuracy: 60% compared to the mere 21% by kNN. These five points occur in nearly all k-neighborhoods and this dataset shows how some pathological cases of very bad hubness also occasionally emerge in practical situations. Even if the erroneous points were not of the minority class, they would still have caused significant misclassification. Also, note that the major hub in the 1000-dimensional case appears in 86.5% of all k-neighbor sets. Its

Table 2. Overview of the datasets. Each dataset is described by its size, dimensionality, the number of categories, skewness of the N_5 distribution (S_{N_5}), proportion of bad 5-occurrences BN_5, as well as the maximal achieved number of occurrences on the dataset.

Data set	size	d	C	S_{N_5}	BN_5	$\max N_5$
dexter	300	20000	2	6.64	30.5%	219
diabetes	768	8	2	0.19	32.3%	14
glass	214	9	6	0.26	35.0%	13
ionosphere	351	34	2	2.06	12.5%	34
isolet1	1560	617	26	1.23	28.7%	30
page-blocks	5473	10	5	0.31	5.0%	16
segment	2310	19	7	0.33	5.3%	15
sonar	208	60	2	1.28	21.3%	22
vehicle	846	18	4	0.64	36.0%	14
vowel	990	10	11	0.60	9.7%	16
Acquis1	23412	254963	2	62.97	19.2%	4778
Acquis2	23412	254963	2	62.97	8.7%	4778
Acquis3	23412	254963	2	62.97	27.3%	4778
Acquis4	23412	254963	2	62.97	12.2%	4778
Acquis5	23412	254963	2	62.97	5.7%	4778
Acquis6	23412	254963	2	62.97	7.6%	4778
Acquis7	23412	254963	2	62.97	18.1%	4778
Acquis8	23412	254963	2	62.97	9.3%	4778
Acquis9	23412	254963	2	62.97	7.6%	4778
Acquis10	23412	254963	2	62.97	21.4%	4778
Acquis11	23412	254963	2	62.97	23.4%	4778
Acquis12	23412	254963	2	62.97	9.8%	4778
Acquis13	23412	254963	2	62.97	16.4%	4778
Acquis14	23412	254963	2	62.97	6.9%	4778
iNet3Err100	2731	100	3	20.56	10.2%	375
iNet3Err150	2731	150	3	25.1	34.8%	1280
iNet3Err1000	2731	1000	3	23.3	79.7%	2363
iNet3	2731	416	3	8.38	21.0%	213
iNet4	6054	416	4	7.69	40.3%	204
iNet5	6555	416	5	14.72	44.6%	469
iNet6	6010	416	6	8.42	43.4%	275
iNet7	10544	416	7	7.65	46.2%	268

Table 3. Overview of the experiments. Classification accuracy is given for kNN, hubness-weighted kNN (hw-kNN), hubness-based fuzzy nearest neighbor (h-FNN) and hubness information k-nearest neighbor (HIKNN). All experiments were performed for $k = 5$. The symbols •/○ denote statistically significant worse/better performance ($p < 0.05$) compared to HIKNN. The best result in each line is in bold.

Data set	kNN	hw-kNN	h-FNN	NHBNN	**HIKNN**
dexter	57.2 ± 7.0 •	67.7 ± 5.4	67.6 ± 4.9	**68.0 ± 4.9**	**68.0 ± 5.3**
diabetes	67.8 ± 3.7 •	75.6 ± 3.7	75.4 ± 3.2	73.9 ± 3.4	**75.8 ± 3.6**
glass	61.5 ± 7.3 •	65.8 ± 6.7	67.2 ± 7.0	59.1 ± 7.5 •	**67.9 ± 6.7**
ionosphere	80.8 ± 4.5 •	87.9 ± 3.6	90.3 ± 3.6 ○	**92.2 ± 3.2** ○	87.3 ± 3.8
isolet1	75.2 ± 2.5 •	82.5 ± 2.1 •	83.8 ± 1.8 •	83.0 ± 2.0 •	**86.8 ± 1.5**
page-blocks	95.1 ± 0.6 •	95.8 ± 0.6 •	96.0 ± 0.6	92.6 ± 0.6 •	**96.2 ± 0.6**
segment	87.6 ± 1.5 •	88.2 ± 1.3 •	88.8 ± 1.3 •	87.8 ± 1.3 •	**91.2 ± 1.1**
sonar	82.7 ± 5.5	83.4 ± 5.3	82.0 ± 5.8	81.1 ± 5.6 •	**85.3 ± 5.5**
vehicle	62.5 ± 3.8 •	65.9 ± 3.2	64.9 ± 3.6 •	63.7 ± 3.5 •	**67.2 ± 3.6**
vowel	87.8 ± 2.2 •	88.2 ± 1.9 •	91.0 ± 1.8 •	88.1 ± 2.2 •	**93.6 ± 1.6**
Acquis1	78.7 ± 1.0 •	87.5 ± 0.8 •	88.8 ± 0.7	88.4 ± 0.7 •	**89.4 ± 0.6**
Acquis2	92.4 ± 0.5 •	93.6 ± 0.5	93.3 ± 0.5	92.5 ± 0.5 •	**93.7 ± 0.5**
Acquis3	72.7 ± 0.9 •	78.7 ± 0.9	79.5 ± 0.9	78.9 ± 0.9	**79.6 ± 0.9**
Acquis4	89.8 ± 0.6 •	90.6 ± 0.6	90.5 ± 0.6	87.4 ± 0.7 •	**91.0 ± 0.5**
Acquis5	97.3 ± 0.3 •	97.6 ± 0.3	97.5 ± 0.3	95.1 ± 0.4 •	**97.7 ± 0.3**
Acquis6	93.6 ± 0.4 •	94.4 ± 0.5	94.0 ± 0.5 •	92.5 ± 0.5 •	**94.6 ± 0.5**
Acquis7	82.9 ± 0.8 •	86.3 ± 0.7 •	86.1 ± 0.6 •	85.7 ± 0.7 •	**87.0 ± 0.7**
Acquis8	92.3 ± 0.5 •	93.0 ± 0.5	93.1 ± 0.5	91.0 ± 0.5 •	**93.5 ± 0.5**
Acquis9	93.0 ± 0.5 •	**94.8 ± 0.4**	94.2 ± 0.5 •	93.4 ± 0.5 •	**94.8 ± 0.4**
Acquis10	83.1 ± 1.6 •	88.8 ± 0.7 •	88.7 ± 0.6 •	87.4 ± 0.7 •	**89.7 ± 0.5**
Acquis11	77.7 ± 0.9 •	81.8 ± 0.8	82.4 ± 0.6	81.9 ± 0.7	**82.5 ± 0.5**
Acquis12	91.9 ± 0.6 •	**92.8 ± 0.5**	92.6 ± 0.5	90.7 ± 0.6 •	**92.8 ± 0.5**
Acquis13	85.6 ± 0.7 •	87.5 ± 0.6	87.1 ± 0.7 •	85.2 ± 0.7 •	**88.0 ± 0.7**
Acquis14	94.2 ± 0.4 •	94.9 ± 0.4	94.6 ± 0.5	92.5 ± 0.5 •	**95.0 ± 0.5**
iNet3Err100	92.4 ± 0.9 •	93.6 ± 0.9 •	97.5 ± 0.9	97.5 ± 0.9	**97.6 ± 0.9**
iNet3Err150	80.0 ± 2.0 •	88.7 ± 2.0 •	94.6 ± 0.9	94.6 ± 0.9	**94.8 ± 0.9**
iNet3Err1000	21.2 ± 2.0 •	27.1 ± 11.2 •	59.5 ± 3.2	**59.6 ± 0.9**	**59.6 ± 3.2**
iNet3	72.0 ± 2.7 •	80.8 ± 2.3	**82.4 ± 2.2**	81.8 ± 2.3	82.2 ± 2.0
iNet4	56.2 ± 2.0 •	63.3 ± 1.9 •	**65.2 ± 1.7**	64.6 ± 1.9	64.7 ± 1.9
iNet5	46.6 ± 2.0 •	56.3 ± 1.7 •	**61.9 ± 1.7**	61.8 ± 1.9	60.8 ± 1.9
iNet6	60.1 ± 2.2 •	68.1 ± 1.6 •	69.3 ± 1.7	69.4 ± 1.7	**69.9 ± 1.9**
iNet7	43.4 ± 1.7 •	55.1 ± 1.5 •	**59.2 ± 1.5**	58.2 ± 1.5	56.9 ± 1.6
AVG	76.72	81.13	83.09	81.86	**83.60**

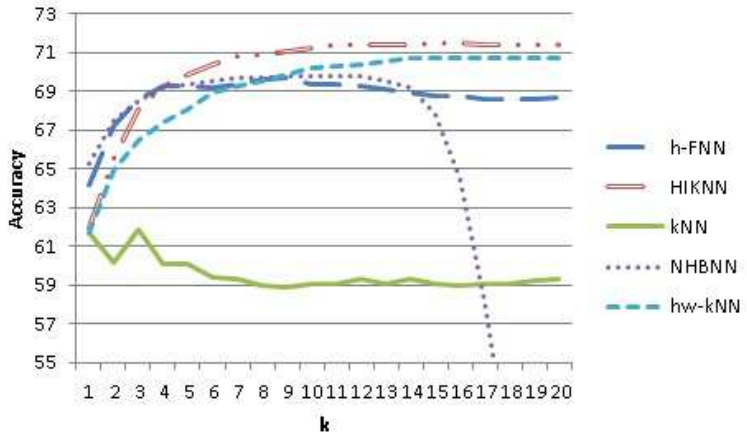


Fig. 5. Classifier accuracies over a range of neighborhood sizes $k \in 1..20$ on iNet6 dataset.

occurrence is, therefore, not very informative - and this further justifies the discussion presented in Section 3.2.

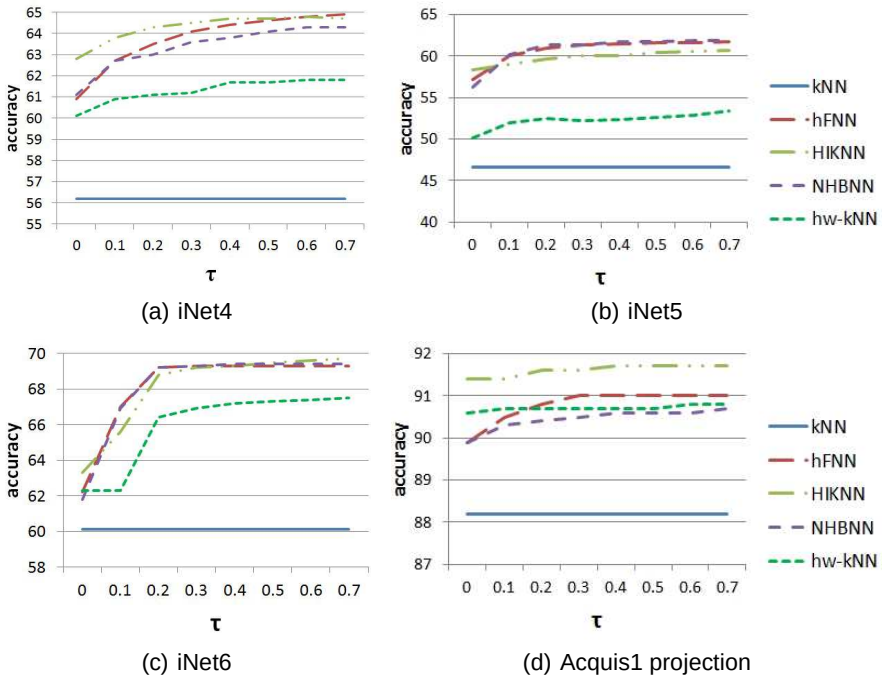
Bad hubness of the data is closely linked to the error of the kNN classification. The Pearson correlation coefficient comparing the kNN error with bad hubness percentages on the datasets in our experiments gives 0.94, which indicates strong positive correlation between the two quantities. HIKNN bases its votes on expectations derived from the previous k-occurrences, so it is encouraging that the correlation between the accuracy gain over kNN and bad hubness of the data is also very strong: 0.87 according to the Pearson coefficient.

6. The approximate implementation

Computing all the k-neighbor sets on the training data in order to build an occurrence model could be overly time-consuming in large-scale data collections. Hubness-aware approaches would be applicable in large-scale scenarios only if it were possible to retain the previously observed improvements while working with some sort of approximate kNN sets.

Many approximate kNN algorithms have been proposed in the literature, either for speeding-up individual queries or constructing an entire kNN graph. It is the latter that is of interest for building an occurrence model. Many of these procedures had been proposed specifically for handling high-dimensional data, which is where hubness-aware classification has been shown to be useful.

In our experiments we focused on one such approach [Chen et al., 2009]. It is a divide and conquer method based on recursive Lanczos bisection. The time complexity of the procedure is $\Theta(dn^{1+\tau})$, where $\tau \in (0, 1]$ reflects the quality of the approximation. There are two ways to implement the recursive division and



(a) iNet4 (b) iNet5

(c) iNet6 (d) Acquis1 projection

Fig. 6. The accuracy of the hubness-aware approaches when the occurrence model is inferred from the approximate kNN graph generated by [Chen et al., 2009]. We see that there are significant improvements even for $\tau = 0$.

we have chosen the GLUE method, as it has proven to be significantly faster than the OVERLAP method, though the quality of the resulting graph is only slightly inferior in comparison [Chen et al., 2009]. The question that we would like to answer is: *for which values of τ are we able to retain the improvements observed on actual kNN sets*?

Re-running all the experiments for all τ values would be beyond the scope of this paper. We did, however, examine the full spectrum of τ-values for the four datasets previously used in the experiments. We report the results for the iNet4, iNet5, iNet6 and Acquis1 datasets in Figure 6. The original Acquis data had too many features for our approximate kNN graph implementation to be able to handle it properly in reasonable time, so we considered a projection onto a 400-dimensional feature space. The data was projected via canonical correlation analysis procedure onto a common semantic space obtained by correlating the English and French aligned versions of documents from the dataset [Hardoon et al., 2004][Hotelling, 1935]. It is one of the standard dimensionality reduction techniques used in text mining and its details are beyond the scope of this paper.

The results shown in Figure 6 are indeed very encouraging. They suggest that significant improvements over the kNN baseline are possible even when the graph is constructed in linear time (w.r.t. number of instances). Moreover, the quality level of $\tau = 0.2$ or $\tau = 0.3$ already seems good enough to capture most of the original occurrence information, as the resulting accuracies are quite close to the ones achieved in the original experiments.

The accuracy curves for different algorithms sometimes intersect. This can be seen for iNet5, iNet6 and Acquis1 in Figure 6. In general, the approximate results correspond rather well to the non-approximate results, but the correlation between the two can vary depending on the particular choice of τ.

In these initial findings HIKNN appears to be quite robust to the employed approximate kNN graph construction method for $\tau = 0$. This is a very nice property, as it allows for obtaining usable results in reasonable time. If better approximations are required, $\tau = 0.3$ should suffice.

A comparison between the results shown in Figure 6(d) and those previously summarized in Table 3 reveals that dimensionality reduction may sometimes significantly affect the classification process and improve the overall classification accuracy. Even though hubness is practically unavoidable in most high-dimensional data mining tasks, its severity does depend on the particular choice of feature representation and/or similarity measure. It is, therefore, not surprising that the dimensionality reduction of the Acquis data helped the kNN classifiers by reducing data hubness. The hubness was not entirely eliminated and this is why all the hubness-aware classification methods still managed to outperform the kNN baseline for all the τ values.

These initial experiments suggest that hubness-aware methods are applicable even to large datasets, as the scalable, approximate kNN graph construction methods are able to deliver good hubness estimates. More experiments are needed to reach the final verdict, on different types of high-dimensional data.

7. Estimating class probabilities

Most frequently, in classification, we are simply interested in assigning a label to a point of interest. What this label suggests is that we are entirely certain that a point belongs to a given class. However, this is just a special case of a more general problem. We would in fact like to be able to assign a 'fuzzy' label to each object, so that it belongs to several classes at the same time. This 'belonging' marks our confidence in any particular atomic label choice.

There are cases, however, when the classes overlap. This happens very frequently in real-world data. There exist points then, in these overlapping regions, that could belong to either of the neighboring categories. In such cases it is meaningless to assign a simple 'crisp' label to each point - what we would like to be able to do is to predict the actual class probability at each point, for every given class. This probability reflects the relative density of each class probability distribution at that point.

The HIKNN algorithm was made to be fuzzy and in the following experiments we wished to determine how well the predicted class probabilities reflect our intuition about the data. The basic kNN algorithm can also be used for point class probability estimates and it is a useful baseline for comparison.

In order to check if the predicted values make sense or not, we examined the algorithm output on synthetic 2D data. The fact that data has only 2 dimensions allows us to draw a *probability map*, where each pixel is 'classified' by the examined algorithms and assigned a probability of belonging to each class. We have generated several such datasets and here we discuss one of them. The dataset is simple, representing 2 categories with overlapping border regions. We have used HIKNN without the distance weighting. The resulting probability maps can be seen in Figure 7.

We see that the probability map generated by HIKNN looks much more natural in the overlapping region. The gradient between the classes should be more or less smooth if the model is able to generalize well. kNN produces a fractured landscape, essentially over-fitting on the training data. These maps suggest that the votes based on previous occurrences may offer better estimates of the underlying class probabilities, which we intend to explore more thoroughly in our future work.

8. Conclusion

In this paper we presented a novel approach for handling high-dimensional data in k-NN classification, Hubness Information k-Nearest Neighbor (HIKNN). It is a hubness-aware approach which is based on evaluating the informativeness of individual neighbor occurrences. Rare neighbors (anti-hubs) are shown to carry valuable information which can be well exploited for classification.

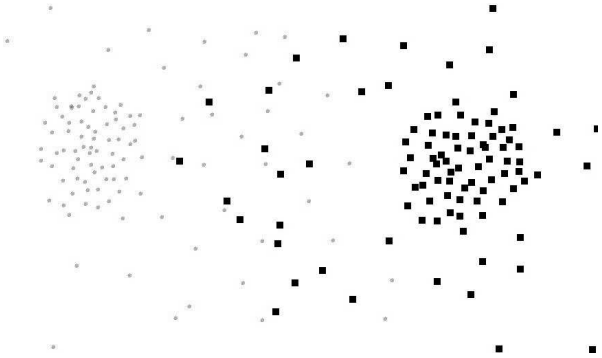
The algorithm is parameter-free, unlike the previous class-hubness based hubness-aware classification algorithms. The danger of over-fitting is thereby greatly reduced.

The algorithm was compared to the three recently proposed hubness-aware approaches (hw-kNN, h-FNN, NHBNN), as well as the kNN baseline on 32 classification problems. Our proposed approach had an overall best performance in the experiments.

Since HIKNN modifies only the voting, it is easily extensible and could be combined with various sorts of metric learning or dynamic k-neighbor sets. We intend to explore these directions thoroughly in our future work.

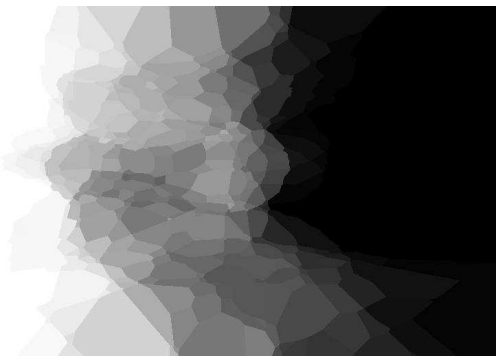
Acknowledgment

This work was supported by the Slovenian Research Agency, the IST Programme of the EC under PASCAL2 (IST-NoE-216886) and the Serbian Ministry of Education and Science project no. OI174023.



(a) The synthetic data set

(b) kNN probability map



(c) HIKNN probability map

Fig. 7. Probability maps inferred from kNN and HIKNN on synthetic data, for $k = 5$. Each pixel was classified by the algorithms and assigned a probability value of belonging to each of the two classes.

Bibliography

Aggarwal, C. C.; Hinneburg, A.; Keim, D. A. On the surprising behavior of distance metrics in high dimensional spaces. In: *Proc. 8th Int. Conf. on Database Theory (ICDT)*. 420–434 (2001).

Aucouturier, J. Ten experiments on the modelling of polyphonic timbre. *Technical Report*, Docteral dissertation, University of Paris 6 (2006).

Aucouturier, J.; Pachet, F. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences* **1** (2004).

Buza, K.; Nanopoulos, A.; Schmidt-Thieme, L. Insight: efficient and effective instance selection for time-series classification. In: *Proceedings of the 15th Pacific-Asia conference on Advances in knowledge discovery and data mining - Volume Part II*. 149–160, PAKDD'11 (Springer-Verlag, 2011).

Chen, J.; ren Fang, H.; Saad, Y. Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research* **10**, 1989–2012 (2009).

C.J.Stone. Consistent nonparametric regression. *Annals of Statistics* **5**, 595–645 (1977).

Devroye, L. On the inequality of cover and hart. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **3**, 75–78 (1981).

Fix, E.; Hodges, J. Discriminatory analysis, nonparametric discrimination: consistency properties. *Technical Report*, USAF School of Aviation Medicine, Randolph Field (1951).

François, D.; Wertz, V.; Verleysen, M. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering* **19**, 873–886 (2007).

Han, E.-H.; Karypis, G.; Kumar, V. Text categorization using weight adjusted k-nearest neighbor classification. In: Cheung, D.; Williams, G.; Li, Q. (eds.) *Advances in Knowledge Discovery and Data Mining*. **2035**, 53–65, 53–65, 53–65 (Springer Berlin / Heidelberg, 2001).

Hardoon, D. R.; Szedmák, S.; Shawe-Taylor, J. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation* **16**, 2639–2664 (2004).

Hotelling, H. The most predictable criterion. *Journal of Educational Psychology* **26**, 139–142 (1935).

Jensen, R.; Cornelis, C. A new approach to fuzzy-rough nearest neighbour classification. In: Chan, C.-C.; Grzymala-Busse, J.; Ziarko, W. (eds.) *Rough Sets and Current Trends in Computing*. **5306**, 310–319, 310–319, 310–319 (Springer Berlin / Heidelberg, 2008).

Keller, J. E.; Gray, M. R.; Givens, J. A. A fuzzy k-nearest-neighbor algorithm. In: *IEEE Transactions on Systems, Man and Cybernetics*. 580–585 (1985).

L. Devroye, A. K., L. Gyorfi; Lugosi, G. On the strong universal consistency of nearest neighbor regression function estimates. *Annals of Statistics* **22**, 1371–1385 (1994).

MacKay, D. J. C. *Information Theory, Inference and Learning Algorithms* (Cambridge University Press, New York, NY, USA, 2002).

Min, M. R.; Stanley, D. A.; Yuan, Z.; Bonner, A. J.; Zhang, Z. A deep non-linear feature mapping for large-margin knn classification. In: *ICDM*. 357–366 (2009).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In: *Proc. 26th Int. Conf. on Machine Learning (ICML)*. 865–872 (2009).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* **11**, 2487–2531 (2010a).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. On the existence of obstinate results in vector space models. In: *Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 186–193 (2010b).

Shang, W.; Huang, H.; Zhu, H.; Lin, Y.; Qu, Y.; Dong, H. An adaptive fuzzy knn text classifier. In: Alexandrov, V.; van Albada, G.; Sloot, P.; Dongarra, J. (eds.) *Computational Science ICCS 2006*. **3993**, 216–223, 216–223, 216–223 (Springer Berlin / Heidelberg, 2006).

Song, Y.; Huang, J.; Zhou, D.; Zha, H.; Giles, C. L. Iknn: Informative k-nearest neighbor pattern classification. In: *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*. 248–264, PKDD 2007 (Springer, 2007).

T.M.Cover; P.E.Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* **IT-13**, 21–27 (1967).

Tomašev, N.; Brehar, R.; Mladenić, D.; Nedevschi, S. The influence of hubness on nearest-neighbor methods in object recognition. In: *Proceedings of the 7th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. 367–374 (2011a).

Tomašev, N.; Mladenić, D. Exploring the hubness-related properties of oceanographic sensor data. In: *Proceedings of the SiKDD conference* (2011).

Tomašev, N.; Mladenić, D. Nearest neighbor voting in high-dimensional data: Learning from past occurrences. In: *ICDM PhD Forum* (2011).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. In: *Proc. MLDM* (2011b).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. A probabilistic approach to nearest neighbor classification: Naive hubness bayesian k-nearest neighbor. In: *Proceeding of the CIKM conference* (2011c).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. In: *PAKDD (1)'11*. 183–195 (2011d).

Wang, J.; Neskovic, P.; Cooper, L. N. Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recogn. Lett.* **28**, 207–213 (2007).

Wang, L.; Khan, L.; Thuraisingham, B. An effective evidence theory based k-nearest neighbor (knn) classification. In: *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent*

Agent Technology - Volume 01. 797–801 (IEEE Computer Society, Washington, DC, USA, 2008).

Weinberger, K. Q.; Blitzer, J.; Saul, L. K. Distance metric learning for large margin nearest neighbor classification. In: *In NIPS* (MIT Press, 2006).

Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques* (Morgan Kaufmann Publishers, 2005), second edition.

2.2.3 Prototype Hubness in Instance Selection

In this section, we will discuss the interplay of hubness and instance selection in high-dimensional data, within the context of k-nearest neighbor classification. A new selection framework is proposed that allows for the hubness-aware classification methods to be used in conjunction with instance selection.

Despite its popularity, the basic kNN implementation also suffers from some serious drawbacks. Most importantly, there are issues with scalability, due to its high storage requirements and relatively slow classification response. Its high specificity bias, which is useful in imbalanced data classification, also makes it more prone to noise and erroneous/mislabeled data.

One of the most promising research directions in addressing these issues is data reduction. Reducing the size of the training set speeds up subsequent classification and reduces the storage requirements, while it may additionally eliminate outliers and noisy examples. There are two types of data reduction algorithms, one where the prototypes are generated through some internal models and the other where they are selected from among the existing data points. We will have a closer look at several common data reduction strategies and examine how they handle hub points.

Hubness is doubtlessly an important phenomenon in high-dimensional kNN classification, but it was not the focus of study in the domain of instance selection. The most detailed reviews of many existing instance selection methods have failed to take the data dimensionality into account and did not consider the implications of selecting or failing to select data hubs [Olvera-López et al., 2010][Garcia et al., 2012]. Up until now, only two simple instance selection methods that take hubness into account have been proposed [Buza et al., 2011][Dai and Hsu, 2011]. We have included them in our analysis.

Prototype selection for k-nearest neighbor classification is a frequently used data preparation technique and many methods have been proposed over the years [Garcia et al., 2012][Liu, 2010][Liu and Motoda, 2002]. *Edition* methods try to eliminate noise in the data by removing noisy instances and *wrapper* methods try to preserve the classifier accuracy by removing superfluous examples. Many methods are hybrid, as they try to achieve both goals, to some degree. This division reflects some fundamental differences in prototype selection strategies, as the edition methods seek to remove the border points, while the wrappers usually perform condensation by keeping precisely such points which are close to decision boundaries [Garcia et al., 2012]. According to what is reported in the literature, good results can be obtained either by keeping or rejecting the border points and either by keeping or rejecting the central points. There is no unified approach and it is clear that the best strategy is data-dependent.

Regardless of the border point selection/rejection strategies, the methods which seek to safely reduce the data size often in fact aim at maximizing the *coverage* of points by their selected k-nearest prototypes [Buza et al., 2011]. Set coverage is an NP-complete problem. The prototype selection problem was shown to be equivalent to the set coverage problem, suggesting that one should apply approximate and heuristic methods.

We have considered several well-known selection strategies, as well as a few very recent ones. Random sampling will be used as a baseline. Any complex, time-consuming method ought to perform at least as well as random sampling if it were to justify its use. Additionally, random sampling is unbiased, which fits the purpose of our comparisons quite well.

The other approaches that we have considered are ENN [Wilson, 1972], CNN [PE, 1968], GCNN [Chou et al., 2006], RT3 [Wilson and Martinez, 1997], AL1 [Dai and Hsu, 2011] and INSIGHT [Buza et al., 2011]. We briefly describe each of them.

ENN: One of the first proposed approaches was the *edited nearest neighbor* (ENN) [Wilson, 1972]. It keeps the examples which are correctly classified by the kNN rule on the training data, k usually being set to 3 or 5. In high hubness data, there is a clear distinction between the role that some points play as neighbors and reverse neighbors.

There is no guarantee that good hubs will be correctly classified or that bad hubs won't. There is certainly some correlation between the two, but it becomes weaker with increasing dimensionality. Therefore, this method may select points which cause severe misclassification.

CNN: The *condensed nearest neighbor* (CNN) [PE, 1968] method is an incremental procedure which retains at each step an instance if it is misclassified by the current prototype set. As outliers are often misclassified, this procedure retains most of the noise in the data and its reduction rate is not so high. Therefore, it is not well suited for big data analysis.

GCNN: A generalized CNN approach was later proposed, applying a strong absorption rule [Chou et al., 2006]. GCNN retains more examples than CNN, usually leading to a better accuracy. As both ENN and CNN, there is no guarantee that the selected points would exhibit good hubness.

RT3: Another classic instance pruning technique is the RT3 rule presented in [Wilson and Martinez, 1997]. In the first pass, noisy instances are removed by a rule similar to ENN. The remaining points are sorted by the distance to their *nearest enemy* and then iteratively removed if their removal does not increase misclassification in the set of their reverse nearest neighbors. RT3 achieves very good data reduction, as only a small set of prototypes is retained. However, as it uses ENN-like noise filtering approach, it can lead to suboptimal selection sets, as some good hubs might be filtered out in the first pass. Additionally, the order in which the instances are pruned may pose problems in high-dimensional data, due to a concentration of distances, which causes the sorting by the distance to the nearest enemy point to be much less stable and less reliable.

AL1: Unlike the above outlined methods, AL1 [Dai and Hsu, 2011] is a selection rule based on reverse-neighbor sets. A point x_i is retained if it is a reverse neighbor to at least one other point, assuming that x_i had not previously been covered by an already selected point. Even though this rule is obviously affected by the hubness phenomenon, the implications of this have not been considered by the authors in the original paper.

INSIGHT: A hubness-aware selection strategy for time series classification was recently proposed [Buza et al., 2011]. INSIGHT takes into account the good and bad k-occurrences of each instance, and then chooses a previously specified number of instances as prototypes. In some applications, it is an advantage if one can specify the exact size of the selected set, but it is also a disadvantage that the method requires this parameter to be set a priori, as it is sometimes unclear which values would work good in practice.

Of course, many other selection strategies exist in the literature and what we have analyzed is no more than a subset of techniques. Genetic algorithms are another common approach [Kim, 2006][Cano et al., 2003][Derrac et al., 2009]. The influence of individual selected instances on future query quality had been estimated in [Zhu and Wu, 2006], though in a radically different way from what we propose here. Some special selection techniques have been quite effective in learning under class imbalance [Pérez-Rodríguez et al., 2011]. Alternatively, instance selection methods can be used for boosting in classifier ensemble construction. As there is no single best selection method, combining several techniques into hybrid selection strategies has also recently been considered [Caises et al., 2011].

All the methods which we consider base their selection criteria on information obtained by analyzing k-neighbor sets. This allows us to implement the hubness-aware components with minimal / negligible overhead in terms of time-complexity. This is not an unreasonable requirement, as these methods are mostly tailored precisely for kNN, so it certainly makes sense that the k-neighbor information is used in the selection criterion.

2.2.3.1 Hubness-aware Instance Selection Framework

During instance selection, the original training set D is decomposed into two disjoint subsets, the set of selected and rejected examples, S and R respectively. We will use α to denote the selection rate $\alpha = \frac{|S|}{|D|}$. Traditionally, only S is used in the subsequent classifier training, while R is disregarded completely. What we essentially propose is to use $D = S \cup R$ for prototype occurrence modeling, i.e. hubness-aware classifier training, while only considering the prototypes $x \in S$ as potential neighbors. There is a way to do this with minimal overhead, in those selection methods which rely on k-nearest neighbor sets.

Hubness-aware classification has never before been used in conjunction with prior instance selection. We believe that these methods can significantly improve the overall system performance.

The proposed selection process is outlined in Figure 10, where the instance selection phase is extended by including the unbiased prototype hubness estimation, followed by hubness-aware k-nearest neighbor classification.

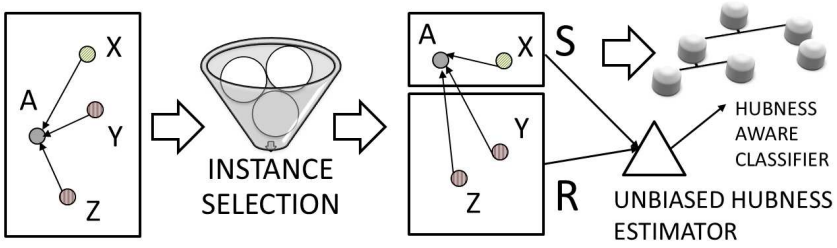


Figure 10: The modified instance selection pipeline. An unbiased prototype occurrence profile estimator is included between the instance selector and a huness-aware classifier. It ought to provide more reliable hubness estimates to the hubness-aware occurrence models. In the example we see that point A is a neighbor to three other points (X,Y,Z), but only one of them gets selected. Hence, some occurrence information is irretrievably lost.

Let **prototype hubness** for a given selected set S be the relative neighbor occurrence frequency of its selected prototype points when only $x \in S$ are permitted as neighbor points. For each instance $x \in S \cup R$, its nearest neighbors from S are calculated.

Note that this is not the same as the hubness of those same prototype points within the training set prior instance selection. The rejected points $x_i \in R$ are put in a tabu-list and are not considered as potential neighbors. Let $x_i \in S$ be a prototype point. Denote by $N_k^P(x_i)$, $N_{k,c}^P(x_i)$, $GN_k^P(x_i)$ and $BN_k^P(x_i)$ the unbiased hubness quantities: prototype occurrence frequency, prototype class hubness, prototype good hubness and prototype bad hubness, respectively. They are inferred from the graph of k-nearest prototypes on D.

As the prototype hubness quantities are derived from prototype occurrences on all of D, the classifiers themselves would be unable to calculate them if provided only with S, the reduced dataset. This is why, in the proposed framework, the instance selection methods need to output a separate prototype hubness estimate array.

Similarly, let the **prototype pseudo-hubness** be the *biased estimate* inferred only from S. The $N_k^S(x_i)$, $N_{k,c}^S(x_i)$, $GN_k^S(x_i)$ and $BN_k^S(x_i)$ will denote the pseudo-hubness, class-specific pseudo-hubness, pseudo-good hubness and pseudo-bad hubness of $x_i \in S$, respectively.

We are interested in estimating how the selected prototype points would occur on the test data, whether they would be hubs or orphans, good neighbors or bad neighbors. This is only possible if the estimate is performed on a sample which follows the same distribution as the test data. The training set usually does, if the data has been sampled properly, so we will refer to the prototype hubness estimates on D as *unbiased prototype hubness estimates.* On the other hand, the distribution of points in S does not necessarily follow the original data distribution.

The only case in which the pseudo-hubness quantities are themselves unbiased is when

the instance selection is entirely random. Even though it is certainly possible to simply use random sampling for instance selection, it is arguably not the best approach, as it selects both relevant and irrelevant points. Additionally, even though random sampling is unbiased, there is still the issue of the reliability of the restricted prototype estimates, as they are inferred from a smaller sub-sample. The *standard error* of a probability estimate p is $\sqrt{\frac{p(1-p)}{n}}$, where n is the number of observations it is derived from. When estimating the class-specific occurrence profiles, $p(y=c|x_i \in D_k(x))$ is required, and there the number of observations is actually $n = N_k(x_i)$. In other words, the expected error is proportional to the reciprocal of the square root of the point hubness. However, $\sum_{x_i \in S} N_k^S(x_i) = k|S|$ and $\sum_{x_i \in S} N_k^P(x_i) = k|D|$. Therefore, $E(N_k^S(x_i)) = k$, while $E(N_k^P(x_i)) = k\frac{|D|}{|S|}$. In other words, even when the selection bias is not the major issue, we would expect the $N_k^P(x_i)$ prototype hubness scores to deliver better estimates by reducing the expected error by a factor of $\sqrt{\frac{|D|}{|S|}}$.

Note that it is impossible to avoid the bias by simply selecting all the hubs, as their hubness also depends on the rejected points. That way it would only be possible to discern the hubness of prototypes towards other hubs, while the test data contains both hubs and anti-hubs.

It was already mentioned that many kNN instance selection methods build an entire kNN graph on the training data during the instance selection phase. In order to calculate all the $N_k^P(x_i)$ and $N_{k,c}^P(x_i)$, these neighbor lists need to be modified so that they only contain members of S, the selected prototypes. This is easily achieved. First, all $x \in R$ are removed from the neighbor sets, which are then shifted to the left. The remaining positions in each $D_k(x)$ are then filled by considering all $\{x : x \in S \setminus D_k(x)\}$. This is illustrated in Figure 11. The worst case scenario would come if no prototypes were in the original kNN graph as neighbors, but this is hardly the case, as the instance selection methods try to pick the most relevant points for kNN classification. Even if, hypothetically, such a situation were to occur, calculating the prototype-restricted kNN graph is still $\frac{|D|}{|S|}$ times faster than calculating the entire training kNN graph, so the additional computational requirements do not increase the overall complexity.

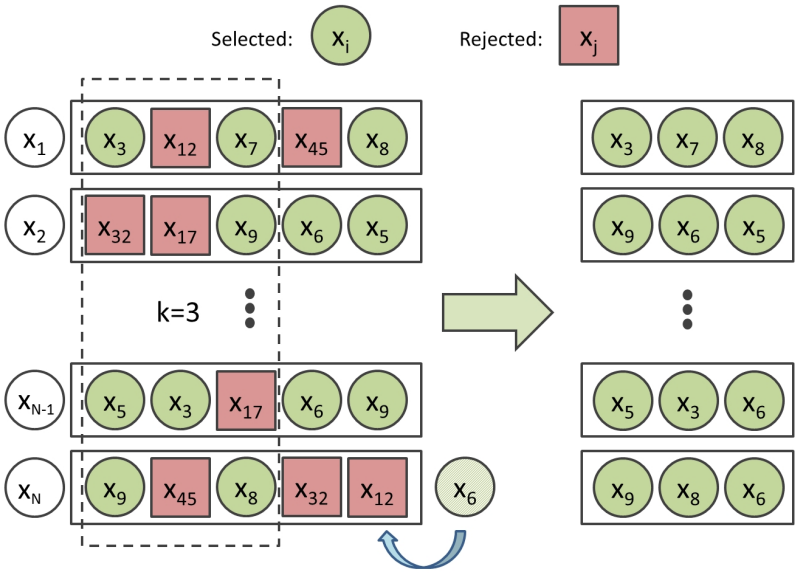


Figure 11: The existing k-nearest neighbor lists on the training set $D = S \bigcup R$ are easily modified to obtain the unbiased prototype hubness estimates. The rejected examples are removed from the neighbor sets and the remaining neighbors are shifted to the left. It is possible to use different neighborhood sizes for instance selection and classification, which would significantly reduce the number of remaining calculations. In some cases, partial nearest neighbor queries might be needed to fill in the last few remaining positions.

An additional benefit is that the considered hubness-aware kNN classifiers [Tomašev et al., 2011c][Tomašev and Mladenić, 2012][Tomašev et al., 2011b] *do not require any training* once provided with all the $N^P_{k,c}(x_i)$ values by the unbiased hubness estimator.

In principle, it would not be possible to build the complete kNN graph on very big data, where there are millions of examples. There exist, however, fast approximate methods which can be used to construct fairly accurate approximations in reasonable time. It is possible to use either a generic approach [Chen et al., 2009] or some metric-specific approximation method based on locality-sensitive hashing [Kulis and Grauman, 2011][Pauleve et al., 2010][Haghani et al., 2009]. The data which we had available for our experiments was not prohibitively big, so we only report the results on the accurate, complete, k-nearest neighbor graphs. It is worth noting, however, that it was already shown that the hubness-aware classification methods are quite robust with regards to the approximate neighbor sets [Tomašev and Mladenić, 2012] and that there is usually no significant decrease in accuracy even for nearly linear graph construction time complexities.

2.2.3.2 Test Data

We have compared the selected instance pruning methods and evaluated our proposed approach on several types of data sets. In our benchmark, we have included difficult image data, time series data and synthetic overlapping high-dimensional class imbalanced Gaussian mixtures. The overview of some hubness-relevant properties of the data is given in Table 1. Manhattan distance was used on the quantized image data, Euclidean on the Gaussian mixtures and dynamic time warping (DTW) on time series data. Image and Gaussian data exhibited high hubness, while these particular time series datasets did not.

The image datasets in the experiments were subsets taken from the ImageNet online repository (http://www.image-net.org/) [Tomašev et al., 2011a]. They are quantized and normalized SIFT feature representations [Lowe, 2004][Zhang and Zhang, 2008], enriched by the color histogram information. They exhibit high overall hubness, as well as high bad hubness, which is not unusual for quantized image data. They have already been discussed in more detail in previous sections.

The Gaussian mixture data was generated with a specific intent to pose great difficulties for k-nearest neighbor methods. It was generated by a fairly complex stochastic process. Let μ_c and σ_c be the d-dimensional mean and standard deviation vectors of a hyper-spherical Gaussian class $c \in 1..C$ on a synthetic Gaussian mixture data set. The covariance matrices of the generated classes were diagonal for simplicity, i.e. the attributes were independent and the i-th entry in σ_c signifies the independent dispersion of that synthetic feature. For the first class, the mean vector was set to zeroes and the standard deviation vector was generated randomly. Each subsequent class c was randomly '*paired*' with one prior Gaussian class, which we will denote by $\bar{c}$, so that some overlap between the two was assured. For each dimension $i \in 1..d$ independently, μ_c was set to $\mu_c \approx \mu_{\bar{c}} \pm \beta \cdot \sigma_{\bar{c}}$ with equal probability, where $\beta = 0.75$. Additionally, dispersion was updated by the following rule: $\sigma_c = \gamma \cdot \sigma_{\bar{c}} + (\gamma - \beta) \cdot Z \cdot \sigma_{\bar{c}}$, where $\gamma = 1.5$ and Z is a uniform random variable defined on $[0,1]$. Each class was set to be either a minority class or a majority class and the class sizes ranged from 20 to 1000, each being randomly determined either in the upper $[700,1000]$ or the lower $[20,170]$ interval of the range. All 10 compared synthetic datasets were set to be 100-dimensional and to contain 20 different classes.

Instance selection methods are potentially very useful in the time series domain, as it takes a lot of time to calculate the dynamic time warping (DTW) distance (Figure 12) between a pair of time series. Instance selection reduces the number of distance calculations in future queries, which helps in speeding up the process. DTW can be interpreted as an edit distance [Levenshtein, 1966]. This means that we can conceptually consider the calculation of the DTW distance of two time series x_1 and x_2 of length l_1 and l_2 respectively as the

Table 1: Overview of the datasets. Each dataset is described by its size, dimensionality, the number of categories, skewness of the N_k distribution (S_{N_k}), proportion of bad k-occurrences BN_k, the number of hubs ($|H_k^D|$), as well as the degree of the major hub. The neighborhood size of $k = 1$ was used for time series and $k = 10$ for images and synthetic Gaussian data.

Data set	size	d	C	$S_{N_{10}}$	BN_{10}	$\lvert H_k^D \rvert$	$\max N_{10}$
iNet3	2731	416	3	4.61	26.1%	76	750
iNet4	6054	416	4	10.77	48.1%	137	906
iNet5	6555	416	5	7.42	50.3%	170	1635
iNet6	6010	416	6	4.32	56.9%	245	1834
iNet7	10544	416	7	5.56	55.0%	343	1638
GM_1	10785	100	20	4.40	41.4%	439	272
GM_2	8849	100	20	5.12	45.6%	319	274
GM_3	8102	100	20	5.35	40.0%	315	323
GM_4	11189	100	20	5.97	45.0%	509	338
GM_5	9859	100	20	5.32	49.2%	361	306
GM_6	10276	100	20	9.19	42.9%	291	500
GM_7	12572	100	20	6.80	45.3%	434	420
GM_8	8636	100	20	8.33	48.5%	256	517
GM_9	9989	100	20	5.26	53.0%	375	289
GM_{10}	9330	100	20	6.12	45.4%	320	357
Data set	size	d	C	S_{N_1}	BN_1	$\lvert H_k^D \rvert$	$\max N_1$
50words	905	270	50	1.26	19.6%	40	5
Adiac	781	176	37	1.15	33.5%	20	6
Cricket X	780	300	12	1.26	16.7%	20	6
Cricket Y	780	300	12	1.25	18.2%	30	6
Cricket Z	780	300	12	0.99	15.9%	19	5
ECGFiveDays	884	136	2	0.68	1%	50	4
Haptics	463	1092	5	1.36	53.6%	16	6
InlineSkate	650	1882	7	1.11	43.1%	12	6
ItalyPowerDemand	1096	24	2	1.30	4.5%	44	6
MedicalImages	1141	99	10	0.78	18.2%	73	4

process of transforming x_1 into x_2. Two types of editing steps are allowed: elongation and replacement, both of them being associated with a cost. The cost of transforming the time series x_1 into x_2 is the cost of *sum* of the costs of all the necessary editing steps. In general, there are many possibilities to transform x_1 into x_2, DTW calculates the one with minimal cost. This minimal cost serves as the distance between the two time series. In practical applications, the dynamic time warping distance is often used in conjunction with k-nearest neighbor classification [Pogorelc and Gams, 2012][Zhang et al., 2012a].

In order to evaluate our approach on time-series data, we used publicly available real-world datasets from the UCR repository[1], a collection that was used by many authors, see e.g. [Csatári and Prekopcsák, 2010], [Ratanamahatana and Keogh, 2004]. Here, we only report the results on 10 representative datasets, namely: 50words, Adiac, Cricket X, Cricket Y, Cricket Z, ECGFiveDays, Haptics, InlineSkate, ItalyPowerDemand, MedicalImages. Similar trends can be observed on the other datasets, as well.

The 50words dataset is associated with a handwriting recognition task [Rath and Manmatha, 2003]. Scanned images of handwritten documents were turned into time series in order to allow to index the documents. The Adiac dataset is associated with a biological shape recognition problem, namely the automatic identification of diatoms (single-celled algae with silica shells) [Jalba et al., 2005]. The Cricket datasets were captured by accelerometers worn by people "mimicking the 12 gestures of a cricket umpire" [Ko et al.,

[1] http://www.cs.ucr.edu/~eamonn/time_series_data/

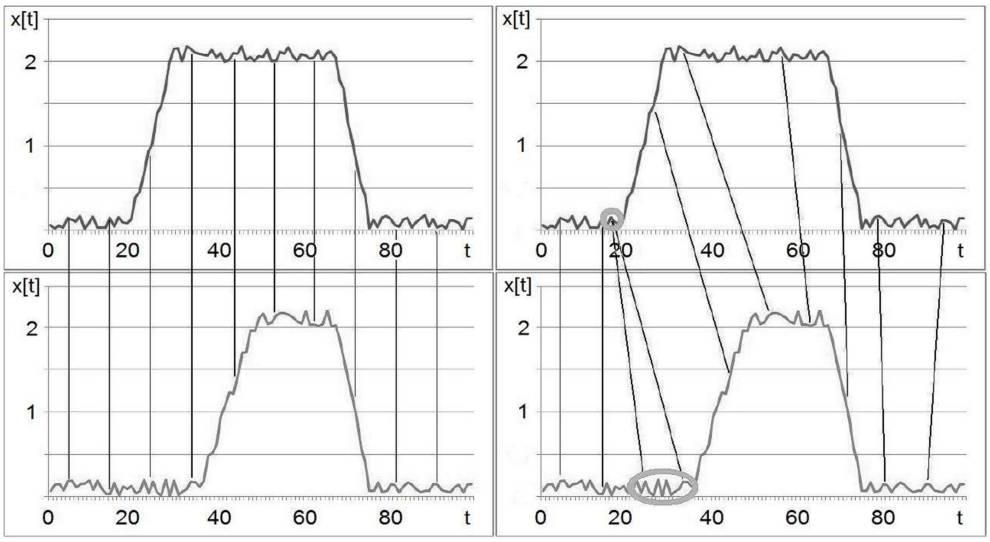


Figure 12: Euclidean Distance vs. Dynamic Time Warping: Euclidean Distance compares always the k-th positions of the both time series with each other (left), while DTW allows for elongation, and therefore when calculating the distance of two time series with DTW, the k-th position of the first time series is not necessarily matched to the k-th position of the second time series (right). This matching is shown by the roughly-vertical lines in both cases.

2008]. The classification problem is to identify which of the gestures was mimicked. The ECGFiveDays dataset was obtained from physionet.org, and it contains 884 ECG signals of length 136 taken from a 67 year old male. Two classes correspond to two different days, 12th and 17th November 1990. The Haptics dataset contains graphical passwords entered on a touch screen [Malek et al., 2006]. In the InlineSkate dataset, muscular activities of in-line speed skaters are recorded over time [Mörchen et al., 2004]. In the classification task associated with the ItalyPowerDemand dataset, days in the summer has to be distinguished from the days in winter based on the power consumption over time during that day [Keogh et al., 2006]. In the MedicalImages dataset, time-series correspond "histograms of pixel intensity of medical images. The classes are different human body regions."[2]

These time series datasets do not exhibit very high hubness, which can be seen from the fact that there are no major hubs in the data, the maximal degree of any single node in the kNN graph is not excessive, as shown in Table 1, unlike in image and Gaussian data. This might seem initially surprising, as this data is undeniably highly multivariate, but this apparent high dimensionality is quite deceiving. In time series, neighboring measurements are often highly correlated, so the intrinsic dimensionality of the data, disregarding redundancies, is usually much lower [Radovanović et al., 2010].

Most of the datasets in Table 1 are quite difficult and challenging for kNN classification even prior to instance selection, especially image and Gaussian data that have about 50% bad k-occurrences, i.e. label mismatches in k-neighbor sets.

The degree of the major hub shows us how some individual points permeate surprisingly many k-neighbor sets and exhibit high influence on k-nearest neighbor classification. This is especially apparent in iNet5-iNet7 datasets. In iNet6, the major hub appears in 30.5% of query results. It often occurs in k-neighbor sets of classes other than its own and makes the subsequent classification task more difficult. This is a good example of why it is important to always consider hub-points when dealing with high-dimensional data.

The inclusion of time series datasets ought to demonstrate the usefulness of the proposed approach even in the context of only mild skewness of the occurrence distribution.

[2] users.eecs.northwestern.edu/~hdi117/listfile/VLDB08_datasets.ppt

2.2.3.3 Experimental Setup

As mentioned before, some instance pruning algorithms automatically determine the size of the prototype set by some intrinsic criteria, but some require this to be specified in advance. Additionally, some methods require parameters.

In the experiments,the selection rate for INSIGHT and baseline random sub-sampling was set to $\alpha = 0.1$. The average data size in practical applications is constantly increasing, so the interested lies mainly in such instance selection methods that can significantly reduce the data size. Additionally, it will be possible to test if the proposed hubness-aware approach of using neighbor occurrence models coupled with an unbiased hubness estimate can overcome such an information loss.

The generalized condensed nearest neighbor rule requires a parameter for its strong absorbtion rule. The proposed default value mentioned in the original paper did not meet our needs in high-hubness data, as it ended up selecting almost the entire dataset. The value of 0.1 was used instead, after some initial trials, which produced a more reasonable reduction. As for INSIGHT, it allows for different heuristic relevance rankings. In our experiments we report only the results based on the good hubness ranking, as the Xi index turned out to be much less effective.

The evaluation of classification under instance selection was performed as 10-times 10-fold cross-validation. Statistical significance was tested using the corrected re-sampled t-test to compensate for dependencies between the runs.

The first part of the instance selection experiments, presented in Section 2.2.3.4 deals with the interplay between hubs and various instance selection methods, showing that different approaches yield different hub selection rates. Section 2.2.3.7 illustrates the difference between the biased and unbiased hubness estimates. Finally, Section 2.2.3.8 demonstrates the effectiveness of our proposed approach in practical use in k-nearest neighbor classification.

2.2.3.4 Hubs and Instance Selection

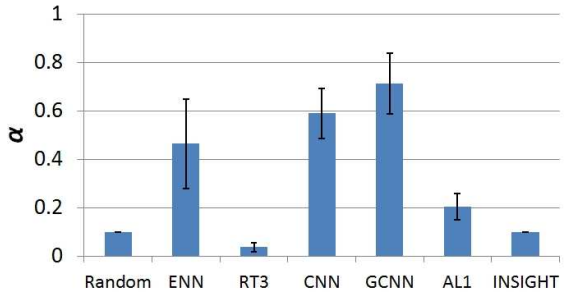
Hubs are the centers of influence in k-nearest neighbor classification. Examining how different selection strategies handle hubs is, therefore, of high importance. Those methods that select many hubs from the original data will tend to preserve the structure of influence. Those that fail to select the original hubs will drastically change the structure and distribution of influence, consequences of which can be either good or bad, depending on the data.

The average results for the image datasets and the synthetic Gaussian mixtures will be presented, as time series data does not exhibit high hubness and the time series hub-points have much less coverage/influence over the neighbor sets.

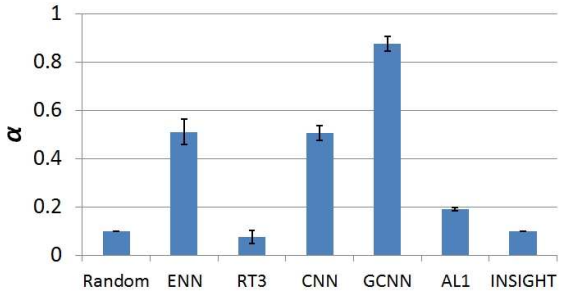
Before considering the selection rate of hubs, the overall selection rates for all points in general should be briefly addressed. This is shown in Figure 13.

The most compact method seems to be RT3, as it doesn't select many points as prototypes. Random sub-sampling and INSIGHT have been pre-set to fixed selection rates of $\alpha = 0.1$. The generalized condensed absorbtion rule, GCNN, achieves only a mild reduction of the data, as it retains most of the original points. ENN and CNN achieve selection rates of about 50% on this particular data.

Most compared instance pruning methods manage to select a non-negligible number of hub-points, but many hubs also get rejected in the process, as shown in Figure 14. We will not discuss the nature of the selected and rejected hubs here, as the impact of the selection process on bad hubness will be closely examined in Section 2.2.3.7. The highest proportion of hubs get selected in INSIGHT and GCNN, while Random and RT3 select the fewest among the original hubs.

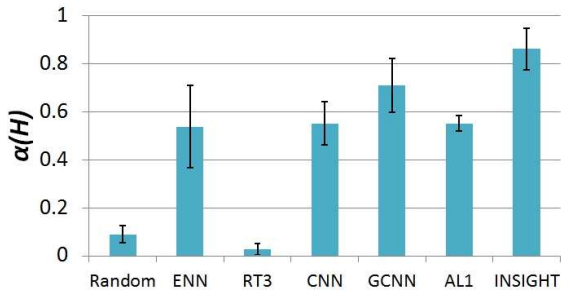


(a) iNet

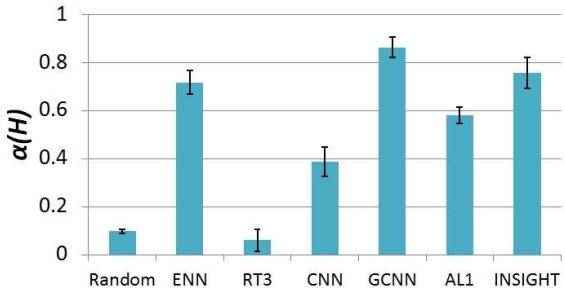


(b) Gaussian Mixtures

Figure 13: Average selection rate α of the examined instance selection methods.



(a) iNet

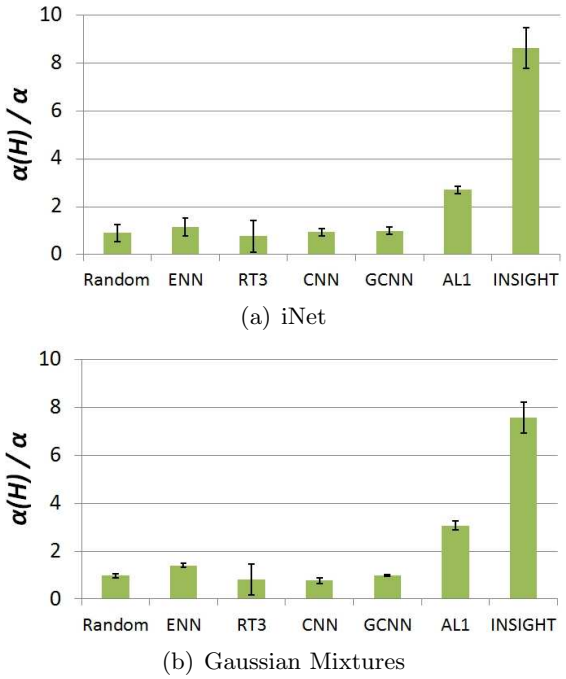


(b) Gaussian Mixtures

Figure 14: Average hub selection rate $\alpha(H)$ of different instance selection methods. A higher rate implies a preservation of the distribution of influence.

However, selecting many hubs is not the same as favoring them over other points. In order to determine which methods favor or penalize hubs, the quantities must be normalized by the general selection rates. This way, it is easy to see if the hub selection rate is higher or lower to that of average points. It also allows a comparison to the bias-less random sub-sampling approach. These normalized results are summarized in Figure 15.

Not surprisingly, INSIGHT and AL1 select a much higher proportion of hubs than any other tested selection method. They are based on examining reverse neighbor sets, while



(a) iNet

(b) Gaussian Mixtures

Figure 15: Averaged normalized hub selection rate $\alpha(H)$ of different instance selection methods. A number close to 1 implies that the hub selection rate does not differ from that of random subsampling.

all other approaches focus on k-neighbor lists directly. This reverse reasoning seems to have payed off, at least with respect to hub point selection. ENN is the only remaining method which achieves a hub selection rate significantly higher than random, on Gaussian mixtures, about 1.4. On the examined Gaussian data, the hub selection rate of CNN is even significantly *lower* than random.

Putting things in context, it was already mentioned that some selection strategies rely on selecting and some on rejecting either the borderline points or the central points. In terms of local data topology, hubs tend to arise near sub-cluster centers. Therefore, a high hub selection rate might be interpreted as a bias towards central regions in the data space and a low hub selection rate as a bias towards less central regions and outliers.

However, there is an important distinction between the selection of hubs and the selection of instances in central regions. Borderline regions between different classes are not necessarily regions of low density, so hubs also arise in borderline regions. Topological centrality is not the same as class-centrality, which is what the supervised instance selection methods focus on. A preference for hubs is a bias towards the centers of k-neighbor influence throughout the data space, in *both* the interior and the borderline class regions.

2.2.3.5 Dependency on Neighborhood Size

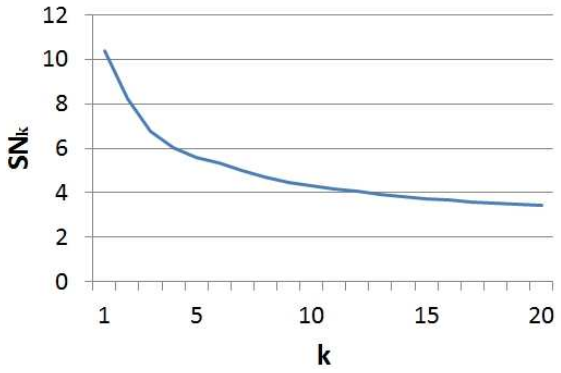
Selecting the optimal neighborhood size has always been an issue in k-nearest neighbor methods. Ideally, one should try and choose a k that is large enough to compensate for noise and allow for good and reliable class density estimates, provided that such a k does not breach the locality assumption somewhere in the feature space, most notably in the minority class regions.

This issue is addressed in various ways in different k-nearest neighbor methods. Sometimes the optimal neighborhood size is determined via cross-validation on each separate fold [Paik and Yang, 2004]. Other algorithms might use some internal heuristics to try and guess the appropriate value of k. Some advanced approaches even define different neighborhood sizes around different points [Ougiaroglou et al., 2007][Wang et al., 2006] or approach

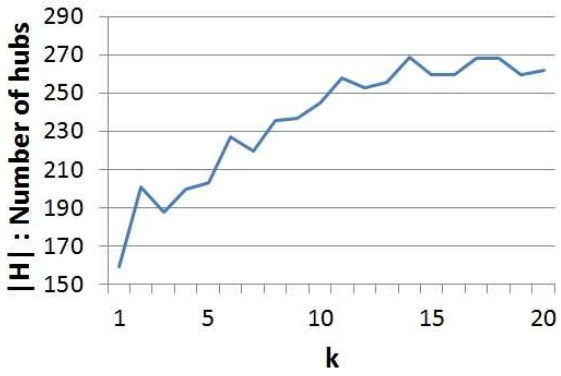
the problem via meta-models [Buza et al., 2010].

As most of the experiments in Section 2.2.3.7 and Section 2.2.3.8 are given for a fixed, pre-determined value of k, it is interesting to take a brief look at how things may potentially change for different neighborhood sizes, in general.

Figure 16 shows the change in hubness on a particular image dataset as the neighborhood size increases. As a rule, the skewness of the occurrence distribution slowly decreases. This rule is not without exceptions and reverse trends can sometimes be observed in low-dimensional low-hubness data. On the other hand, a decrease in the overall occurrence skewness does not necessarily imply a decrease in the number of prominent hubs. Their number slowly increases with increasing k until it reaches a stable point. Naturally, if we continue to increase k, their number also starts decreasing at some later point.



(a) Skewness of the occurrence distribution



(b) The number of hubs

Figure 16: The change in hubness over a set of different neighborhood sizes on iNet6 dataset. The skewness decreases with increasing k, but the number of hubs increases until it reaches a plateau.

The robustness of the data reduction methods with respect to k is shown in Figure 17, in particular we depicted the change in the hub point selection rates. The number of selected hubs does not vary greatly and is rather stable in INSIGHT, CNN, GCNN and RT3. The fluctuations in ENN most probably stem from tie resolution in kNN classification and could vary depending on the tie resolution strategy. As for AL1, its hub selection rate decreases monotonously with k, as more and more hubs get covered by other hubs and are therefore rejected by the algorithm. Regardless, similar k values tend to produce similar results. It is also apparent that the relative ordering of the methods with respect to hub selection remains invariant.

This brief example shows that varying neighborhood sizes does not change the bias towards or against hubs substantially for the examined approaches.

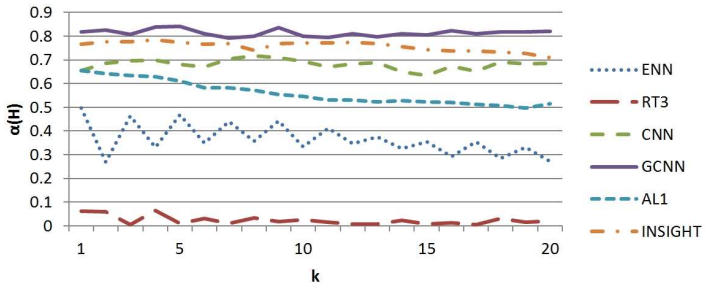


Figure 17: The stability of hub selection rates of different instance selection methods under changing neighborhood sizes, calculated on the iNet6 dataset.

2.2.3.6 Prototype Occurrence Skewness

The skewness of the prototype k-occurrence distribution is not necessarily the same as the overall hubness of the data and different selection methods induce different degrees of prototype hubness. The average prototype skewness over different groups of datasets and different selection methods is given in Figure 18.

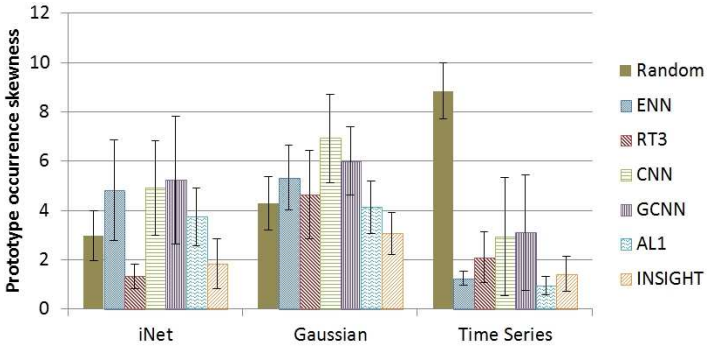


Figure 18: Average unbiased skewness in the prototype occurrence distributions, SN_k^P, given for different instance selection methods.

CNN and GCNN induce the highest prototype set hubness among the compared approaches in all three data groups. Only in the otherwise low-hubness, time series data does random sub-sampling induce a higher skew in the k-occurrence distribution. The prototype sets obtained by INSIGHT seem to be of lowest occurrence skewness, on average. This is a consequence of its hub selection strategy, as it does not select orphan points, so the overall skewness is reduced.

Not all approaches are consistent with respect to the induced occurrence skewness, as for instance ENN and RT3. Sometimes they induce high skewness prototype sets and sometimes low skewness prototype sets. This means that it would in principle be impossible to separate the instance pruning techniques into low-hubness and high-hubness types. The resulting hubness depends on the data and its distribution.

2.2.3.7 Biased and Unbiased Hubness Estimates

As instance selection methods incorporate a selection bias, calculating the hubness of the selected prototypes within the selected subset alone ($N_k^S(x_i)$, $N_{k,c}^S(x_i)$, $GN_k^S(x_i)$,$BN_k^S(x_i)$) yields biased pseudo-hubness estimates. Up until recently, k-nearest neighbor methods did not use these estimates for classification, so no attention was given to this fact, not even in the two explicitly hubness-based instance selection methods that we are considering here, INSIGHT [Buza et al., 2011] and AL1 [Dai and Hsu, 2011].

On the other hand, the recently proposed hubness-aware k-neighbor occurrence models rely directly on neighbor occurrence estimates, so it becomes important to explore how the

selection bias influences the estimates on the training set. Figure 19 shows high regularity in estimating label mismatch percentages on image data, i.e. bad hubness. ENN, RT3 and INSIGHT consistently underestimate the actual bad influence of their prototypes, while CNN and GCNN consistently overestimate the bad influence of their prototype sets. The best estimate of the nature of future prototype influence is achieved in case of AL1, where the results are quite encouraging.

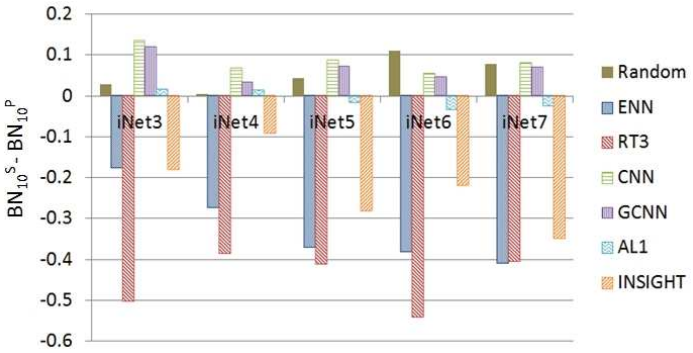


Figure 19: The difference between the pseudo-bad hubness estimated on the set of selected instances S and the actual prototype bad hubness estimated on the entire training set.

This is an important finding, as it allows us to interpret how these selection rules would work when coupled with the hubness-aware k-occurrence models. In a sense, underestimating bad hubness is potentially much more dangerous than overestimating it, as it would cause the models to favor certain points that might actually turn out to be bad hubs. This could cause significant misclassification. On the other hand, overestimating bad hubness would cause the models to disregard certain otherwise reliable points, which could also cause misclassification to occur.

Figure 20 demonstrates the severity of mis-estimating the probability of label mismatches for individual neighbor points on ImageNet data. CNN and GCNN display the lowest error average, though this can also be contributed to the fact that they select large prototype sets, so orphan points reduce the average point-wise estimation error, even if they don't actually influence the classification outcomes. On the other hand, RT3 displays a very high bad hubness estimation error rate. This is not altogether surprising, given that it retains very few prototypes, which makes any estimate quite difficult. Yet, even INSIGHT and ENN exhibit non-negligible estimation error rates on several datasets, up to 0.3 in probability. These estimates were calculated for $k = 10$, which means that a substantial amount of error can accumulate with 10 normalized fuzzy votes that are approximately 0.3 off from what they ought to be. This example clearly shows why it would be dangerous to estimate neighbor occurrence probabilities from the reduced part of the training set alone. It might cause severe misclassification, which will become even more apparent in Section 2.2.3.8 in presenting the results of actual classification tests.

Even though the absolute and relative $BN_k^S(x)$ differ notably from $BN_k^P(x)$, prototype neighbor points mostly retain their general class hubness tendencies. There is a high average correlation between $N_{k,c}^S(x)$ and $N_{k,c}^P(x)$ for $c \in C$. This can be seen in Figure 21. The correlation is only low in case of RT3. Pearson correlation in class hubness is highest for CNN and GCNN, while it remains in the range $[0.6, 0.9]$ for most other approaches. The fact remains that there is some difference in class hubness structure between the actual and the pseudo-scores. They are, however, highly correlated, which is not surprising.

Unlike bad hubness, no regularity can be seen in underestimating or overestimating the skewness of the prototype occurrence distribution itself, as given in Figure 22. Same algorithms both underestimate and overestimate the skewness, depending on the dataset. This shows that a reliable prediction of the actual occurrence skewness can not be made

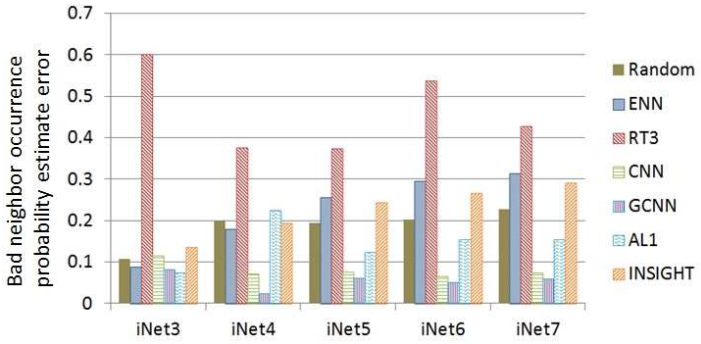


Figure 20: The average absolute difference in estimating the bad 10-occurrence probabilities of individual prototype points on ImageNet data, in other words $Err_{AVG}^{p(BN_{10}^S)} = E_{\{x:N_{10}^S(x)>0 \vee N_{10}^P(x)>0\}}(|\frac{BN_{10}^S(x)}{N_{10}^S(x)} - \frac{BN_{10}^P(x)}{N_{10}^P(x)}|)$.

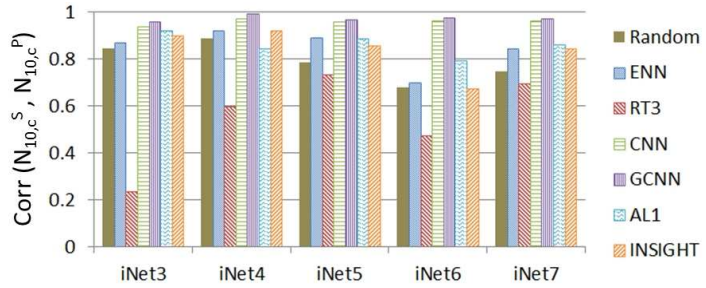


Figure 21: Average Pearson correlation between class hubness tendencies of prototype neighbor points for the compared selection methods on ImageNet data.

based on the retained prototype set alone. Therefore, SN_k^S is not a viable substitute for SN_k^P.

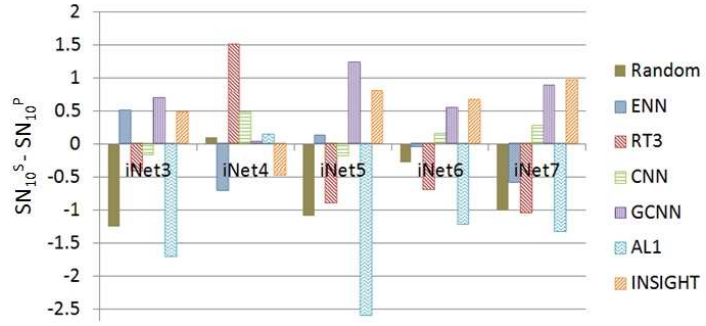


Figure 22: The difference between the pseudo-hubness estimated on S and the prototype occurrence skewness estimated on the entire training set. There is no apparent regularity, which means that very little can be discerned from observing pseudo-hubness of prototypes on a single dataset, as one can not even know with certainty whether the estimate exceeds the actual data hubness or underestimates it instead.

2.2.3.8 Classification

Even though hubness selection bias plays an important role in the classification outcome, it should be stressed that the absence of an apparent bias is no guarantee of good performance. Data reduction entails an information loss. Instance selection methods try to compensate for this loss by selecting more relevant points. A mild hubness estimation bias doesn't tell us anything about the actual relevance of the retained points $x_i \in S$. Therefore, instance

selection can only be confidently evaluated within a certain context and the context which we chose to explore is k-nearest neighbor classification.

The biased and unbiased hubness estimates have been tested within several different hubness-aware classifiers and occurrence models: hw-kNN [Radovanović et al., 2009], h-FNN [Tomašev et al., 2011b], NHBNN [Tomašev et al., 2011c] and HIKNN [Tomašev and Mladenić, 2012]. Most results are reported primarily for HIKNN, as similar improvement trends can be observed in other hubness-aware classifiers, as well.

The results presented here are summarized in Tables 2–Table 4. The accuracy of kNN classifier under different instance selection schemes is given in Table 2. All selection approaches perform rather poorly on this data, which is not surprising, since the data was intentionally selected in such a way that kNN classification gets difficult, even without information loss due to instance selection. GCNN is the best instance selection approach here, though it doesn't perform well in time series classification. The two selection strategies which favor hubs, AL1 and INSIGHT perform worse than random sub-sampling, which suggests that they do not select hubs in a proper way.

Table 2: Cross-validated classification accuracy of k-nearest neighbor classifier under several different selection strategies. ○ and • denote significantly better or worse result ($p < 0.01$) than kNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test.

classifier: kNN								
	Instance selection method							
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	85.1±1.6	80.4±1.8•	82.2±1.6•	52.5±5.8•	82.6±1.5•	84.2±1.3•	82.1±1.8•	80.8±1.5•
iNet4	68.8±1.4	62.3±1.4•	64.6±1.5•	41.3±2.8•	68.2±1.4	68.9±1.5	65.1±1.4•	63.6±1.3•
iNet5	63.7±1.4	56.0±1.6•	55.4±1.4•	25.5±3.1•	63.4±1.4	64.8±1.2○	59.3±1.4•	58.0±1.6•
iNet6	66.3±1.5	57.7±1.8•	56.5±1.7•	33.0±2.9•	63.8±1.5•	65.0±1.3•	59.9±1.6•	56.2±1.8•
iNet7	61.2±1.1	52.9±1.5•	40.4±1.3•	37.2±3.5•	58.3±1.3•	60.1±1.2•	49.6±1.5•	48.8±2.2•
AVG_{img}	69.0	61.9	59.8	37.9	67.3	68.6	63.2	61.5
GM_1	68.1±0.8	50.3±3.0•	28.0±1.8•	12.1±1.3•	43.8±4.1•	68.5±1.2	30.9±2.9•	29.6±4.9•
GM_2	64.4±1.1	43.4±3.9•	32.5±2.0•	24.9±4.3•	53.0±2.1•	61.0±1.5•	29.3±2.5•	37.6±4.3•
GM_3	70.6±1.1	51.7±3.4•	49.2±2.7•	14.2±4.3•	65.6±1.9•	62.7±1.4•	35.2±3.2•	52.7±5.0•
GM_4	67.3±1.0	46.7±3.7•	30.2±2.0•	16.2±2.3•	44.8±2.2•	54.2±1.7•	38.4±3.5•	31.3±3.2•
GM_5	58.4±0.8	38.4±3.3•	34.8±2.1•	8.5 ±3.3•	43.5±3.9•	58.6±1.3	24.6±1.8•	40.3±3.8•
GM_6	66.5±1.2	48.7±2.9•	21.5±1.5•	22.9±4.6•	43.9±2.8•	60.8±1.3•	40.8±3.5•	27.4±3.1•
GM_7	67.1±1.0	48.4±4.5•	9.9 ±0.8•	15.0±1.1•	23.6±3.0•	66.1±1.0•	12.2±1.8•	11.5±1.2•
GM_8	59.8±1.1	43.4±3.4•	40.0±1.7•	16.9±3.3•	60.8±1.5○	58.7±1.2•	35.9±2.8•	48.5±1.9•
GM_9	55.8±1.0	37.7±2.6•	14.1±1.0•	12.4±2.4•	26.2±2.6•	55.9±1.1	19.0±1.7•	16.2±2.1•
GM_{10}	65.9±1.1	44.1±4.2•	37.3±2.3•	12.8±2.4•	57.1±2.2•	65.3±1.2	25.9±3.2•	40.0±3.4•
AVG_{gm}	64.4	45.3	29.8	15.6	46.2	61.2	29.2	33.5
50words	79.7±3.2	21.2±5.8•	29.9±5.4•	11.0±4.6•	17.1±4.7•	16.9±4.9•	32.5±8.4•	18.9±5.7•
Adiac	65.1±3.5	8.5 ±3.3•	15.2±3.9•	5.2 ±2.5•	12.9±4.0•	11.9±3.3•	11.0±4.8•	8.0 ±3.6•
Cricket X	82.8±2.9	28.0±7.5•	56.5±5.2•	16.0±5.4•	29.1±6.9•	30.0±7.1•	56.5±7.3•	31.5±7.9•
Cricket Y	81.5±2.9	30.6±6.3•	50.9±6.1•	20.2±5.3•	24.2±6.4•	23.2±6.7•	60.1±6.3•	31.2±8.2•
Cricket Z	83.1±2.8	28.2±7.1•	67.6±4.5•	14.2±5.2•	46.9±8.2•	49.1±8.3•	56.4±7.8•	35.8±7.0•
ECGFiveD	99.0±0.8	88.7±3.6•	98.7±0.9	48.9±4.8•	91.4±3.8•	91.0±4.2•	96.5±1.8•	89.9±3.5•
Haptics	45.9±4.5	30.7±5.8•	38.7±5.2•	28.9±5.7•	40.0±4.8•	39.9±5.2•	31.5±4.4•	33.4±6.7•
InlineSkate	55.7±3.9	26.5±5.0•	46.7±4.5•	20.6±4.6•	49.9±4.5•	49.2±4.6•	36.3±5.4•	33.4±4.7•
ItalyPower	95.5±1.2	89.9±3.6•	95.3±1.5	59.0±8.1•	88.9±4.5•	90.4±3.6•	92.7±2.1•	92.3±2.5•
MedicalImg	80.9±2.4	51.4±4.9•	70.5±3.3•	41.8±4.3•	66.2±3.8•	66.1±4.2•	63.4±5.3•	50.1±5.1•
AVG_{ts}	76.9	40.4	57.0	26.6	46.7	46.8	53.7	39.5
AVG	70.3	46.7	46.7	24.5	50.6	56.9	45.8	41.5

In fact, there is no guarantee that the hubs that are being favored by AL1 and INSIGHT are beneficial for the classification process. AL1 simply tries to select a very small number of points that maximize coverage and the selected hubs might exhibit extremely detrimental neighbor occurrence profiles, causing many label mismatches and inducing misclassification. The same goes for INSIGHT, even though it tries to select *good hubs* primarily, it does so by

focusing on the total number of beneficial occurrences and disregards the bad ones. Unfortunately, on this data, most major hubs have both many good and many bad occurrences, so that particular selection strategy fails to achieve its purpose. Alternative ranking measures have been proposed for INSIGHT, but our initial tests did not show promising results.

Similar trends can be observed in HIKNN classification, when the biased hubness estimate is used, as shown in Table 3. The relative ordering of selection methods based on their performance remains the same, though there is already significant improvement over basic kNN classification.

Table 3: Cross-validated classification accuracy of Hubness Information k-nearest neighbor classifier (HIKNN) under several different selection strategies. The model is trained on the prototype set only, which means that the *biased* hubness estimate is used. ○ and • denote significantly better or worse result ($p < 0.01$) than HIKNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test.

classifier: HIKNN, hubness estimate: BIASED								
	Instance selection method							
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	85.8±1.5	80.4±2.0•	83.0±1.5•	51.6±6.1•	81.8±1.6•	83.2±1.4•	82.9±1.9•	80.9±1.5•
iNet4	69.9±1.3	63.5±1.4•	65.2±1.6•	41.9±3.2•	69.4±1.3	70.1±1.4	66.6±1.4•	64.2±1.4•
iNet5	67.6±1.3	59.7±1.4•	57.8±1.3•	25.4±2.5•	65.7±1.4•	67.4±1.2	61.8±1.4•	60.1±1.5•
iNet6	68.6±1.5	60.0±1.6•	58.9±1.6•	32.8±2.8•	65.1±1.6•	66.8±1.3•	62.9±1.9•	58.1±1.7•
iNet7	64.3±1.0	55.4±1.5•	42.5±1.3•	37.7±3.6•	59.8±1.3•	62.5±1.0•	51.5±1.6•	50.9±2.1•
AVG_{img}	71.2	63.8	61.5	37.9	68.4	70.0	65.2	62.8
GM_1	78.3±0.9	65.6±3.5•	31.2±2.5•	13.2±1.5•	52.6±4.8•	79.0±0.9○	34.7±3.3•	32.5±6.1•
GM_2	77.3±1.2	61.7±4.5•	39.1±2.2•	26.1±4.7•	65.3±2.0•	74.4±1.3•	34.8±3.4•	45.7±5.2•
GM_3	81.3±0.9	65.8±4.3•	57.1±3.0•	15.6±4.7•	75.2±1.8•	71.6±1.5•	40.6±3.8•	59.9±5.9•
GM_4	78.2±0.9	60.6±4.4•	32.3±2.3•	19.3±3.0•	55.7±1.9•	64 ±1.9•	43.3±4.5•	33.8±3.9•
GM_5	75.2±0.9	60.8±4.5•	45.1±2.4•	9.0 ±3.4•	57.4±4.3•	75.2±1.1	19.3±1.4•	50.5±4.1•
GM_6	79.4±0.9	65.0±3.6•	26.4±1.6•	22.9±5.3•	50.2±3.5•	69.6±1.5•	49.2±4.8•	33.1±3.8•
GM_7	81.1±0.8	65.4±5.6•	11.2±0.8•	16.3±1.1•	28.4±3.7•	80.5±0.9	10.2±1.5•	13.2±1.1•
GM_8	77.0±1.0	63.8±3.7•	52.2±1.7•	17.6±3.9•	76.3±1.8	75.6±1.1•	41.2±4.5•	60.9±2.2•
GM_9	70.6±1.1	56.1±3.4•	13.5±0.9•	14.7±2.8•	34.3±3.2•	71.4±0.9○	17.4±1.4•	16.6±2.7•
GM_{10}	80.3±0.9	64.7±6.0•	46.8±3.2•	13.8±2.8•	70.9±2.5•	80.0±1.0	32.1±4.3•	49.1±4.7•
AVG_{gm}	77.9	63.0	35.5	16.8	56.6	74.1	32.3	39.5
50words	79.7±3.1	21.6±5.8•	30.7±5.4•	11.1±4.7•	17.1±4.7•	16.9±4.9•	32.9±8.3•	19.5±5.9•
Adiac	65.1±3.6	8.3 ±3.3•	15.2±3.9•	5.2 ±2.6•	12.8±4.0•	11.7±3.3•	11.0±4.7•	7.9 ±3.6•
Cricket X	82.9±2.9	27.8±7.5•	57.2±5.2•	16.0±5.4•	28.6±6.7•	29.5±7.0•	56.5±7.3•	31.6±8.1•
Cricket Y	81.5±2.8	30.3±6.3•	51.4±6.3•	20.2±5.3•	24.3±6.3•	23.1±6.6•	60.3±6.4•	31.2±8.2•
Cricket Z	83.1±2.8	28.4±7.1•	68.3±4.3•	14.3±5.2•	46.5±8.0•	48.8±8.1•	56.6±7.9•	35.9±7.2•
ECGFiveD	99.0±0.8	88.9±3.4•	98.7±0.9	48.9±4.6•	88.6±4.8•	88.7±4.9•	96.7±1.7•	90.0±3.5•
Haptics	47.0±4.9	30.7±6.2•	39.1±5.1•	29.2±5.8•	41.8±4.8•	41.3±5.3•	31.5±4.4•	33.9±6.8•
InlineSkate	55.8±3.9	27.0±4.8•	46.8±4.6•	20.5±4.7•	50.5±4.5•	49.7±4.5•	36.4±5.6•	33.0±4.9•
ItalyPower	95.5±1.2	90.4±3.4•	95.3±1.4	58.8±8.2•	89.9±4.4•	91.2±3.7•	92.9±1.9•	92.7±2.5•
MedicalImg	81.3±2.4	51.6±4.9•	70.6±3.2•	42.2±4.5•	66.3±3.9•	66.3±4.1•	63.6±5.4•	50.6±4.9•
AVG_{ts}	77.1	40.5	57.3	26.6	46.6	46.7	53.8	42.6
AVG	76.2	54.2	49.4	30.0	55.0	62.3	47.5	45.4

Using the proposed unbiased hubness estimation significantly improves the performance of HIKNN *regardless of the underlying instance selection strategy*, as shown in Table 4 and Figure 23, while Figure 24 shows that the same trend holds for other hubness-aware classifiers. This confirms our initial hypothesis that the instance selection bias reflects negatively on neighbor occurrence models in hubness-aware k-nearest neighbor classifiers and that using an unbiased estimate leads to better results. In most cases the improvement is achieved with little to no computational overhead.

The benefits of using the unbiased hubness estimates with the HIKNN classifier are most pronounced in INSIGHT and AL1, where there is an average absolute 20% accuracy improvement over the use of biased estimate, which is itself still better than simply using

kNN with instance selection. The absolute improvements in other instance selection methods are about 10% and there is even an improvement of 3.5% in random sub-sampling which has no selection bias. As we have argued before, the unbiased estimates calculated on the entire training set are of higher quality by the very virtue of being based on more observed occurrences, even when no bias is present.

Table 4: Cross-validated classification accuracy of Hubness Information k-nearest neighbor classifier (HIKNN) under several different selection strategies. The neighbor occurrence profiles in the model are derived by observing the unrestricted prototype occurrences on the entire training set, leading to an *unbiased* hubness estimate. ∘ and • denote significantly better or worse result ($p < 0.01$) than HIKNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test. Accuracies higher than in the biased case are given in **bold**.

classifier: HIKNN, hubness estimate: UNBIASED								
		Instance selection method						
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	85.8±1.5	79.7 ± 1.6•	**83.6±1.4**•	50.3 ± 2.1•	**83.0±1.5**•	**84.5±1.6**•	**83.7±1.4**•	**81.6±1.4**•
iNet4	69.9±1.3	**65.0±1.3**•	**65.9±1.2**•	**43.5±2.9**•	**69.6±1.2**	69.8 ± 1.3	**67.7±1.4**•	**65.1±1.4**•
iNet5	67.6±1.3	**60.8±1.3**•	**60.1±1.3**•	**39.0±3.4**•	**66.7±1.2**•	**67.7±1.2**	**64.2±1.3**•	**61.9±1.2**•
iNet6	68.6±1.5	**62.0±1.3**•	**61.9±1.4**•	**33.2±3.0**•	**66.9±1.5**•	**67.6±1.3**•	**65.2±1.4**•	**61.6±1.3**•
iNet7	64.3±1.0	**58.4±1.0**•	**48.8±1.4**•	**38.5±3.6**•	**62.6±1.1**•	**63.8±1.1**	**59.3±1.3**•	**57.5±1.1**•
AVG_{img}	71.2	**65.2**	**64.1**	**40.9**	**69.8**	**70.7**	**68.0**	**65.5**
GM_1	78.3±0.9	**73.5±1.1**•	**63.6±1.3**•	**46.7±3.2**•	**73.7±1.5**•	**79.5±0.9**∘	**72.2±1.0**•	**74.9±1.0**•
GM_2	77.3±1.2	**70.2±2.0**•	**61.9±1.4**•	**60.6±3.1**•	**74.5±1.4**•	**76.7±1.1**	**70.2±1.3**•	**73.6±1.1**•
GM_3	81.3±0.9	**76.4±1.5**•	**67.1±1.6**•	**53.2±6.9**•	**80.3±1.5**•	**77.2±1.5**•	**73.0±1.2**•	**77.5±1.2**•
GM_4	78.2±0.9	**68.9±1.8**•	**58.9±1.7**•	**44.8±2.6**•	**72.0±1.9**•	**74.3±1.0**•	**68.3±1.3**•	**69.8±1.2**•
GM_5	75.2±0.9	**70.9±1.5**•	**60.9±1.0**•	**18.1±1.4**•	**70.7±1.8**•	**76.0±0.9**∘	**64.6±1.2**•	**72.5±1.1**•
GM_6	79.4±0.9	**73.0±1.7**•	**56.1±1.2**•	**46.8±2.5**•	**73.1±1.5**•	**76.0±1.2**•	**71.6±1.3**•	**74.7±1.1**•
GM_7	81.1±0.8	**75.6±1.3**•	**46.7±1.4**•	**48.4±3.1**•	**61.9±2.7**•	**81.3±0.9**	**66.6±1.5**•	**75.8±1.1**•
GM_8	77.0±1.0	**71.5±2.2**•	**61.2±1.3**•	**51.5±3.4**•	**79.3±1.2**∘	**76.9±1.1**	**64.9±2.1**•	**72.4±1.4**•
GM_9	70.6±1.1	**66.0±1.7**•	**46.9±1.2**•	**30.1±2.9**•	**57.6±2.7**•	**71.5±0.9**∘	**62.2±1.3**•	**68.5±1.1**•
GM_{10}	80.3±0.9	**77.4±1.6**•	**66.1±1.9**•	**36.0±3.3**•	**79.7±1.0**	**81.1±0.9**∘	**71.1±1.1**•	**77.8±1.1**•
AVG_{gm}	77.9	**72.3**	**58.9**	**43.6**	**72.3**	**77.1**	**68.5**	**73.8**
50words	79.7±3.1	20.9 ± 5.7•	**42.8±5.8**•	**42.8±4.9**•	**39.0±4.2**•	**40.2±4.4**•	**61.6±4.6**•	**55.5±4.0**•
Adiac	65.1±3.6	**8.4 ±4.1**•	**24.9±4.2**•	**22.4±5.4**•	**29.5±4.0**•	**29.2±4.4**•	**37.4±4.6**•	**35.8±4.8**•
Cricket X	82.9±2.9	27.3 ± 7.5•	**64.7±4.7**•	**36.1±7.1**•	**57.4±5.7**•	**58.1±4.6**•	**69.9±4.1**•	**58.8±5.0**•
Cricket Y	81.5±2.8	**32.1±7.2**•	**59.1±5.0**•	**39.2±6.8**•	**53.9±5.7**•	**52.6±5.0**•	**71.8±4.2**•	**57.7±5.0**•
Cricket Z	83.1±2.8	28.0 ± 7.3•	**71.5±4.2**•	**35.0±6.8**•	**65.9±4.7**•	**66.9±4.8**•	**71.0±4.2**•	**59.9±4.8**•
ECGFiveD	99.0±0.8	87.6 ± 3.9•	**98.8±0.8**	**52.6±4.9**•	**95.1±3.4**•	**96.0±1.9**•	**97.9±1.2**•	**93.4±1.8**•
Haptics	47.0±4.9	**31.5±6.1**•	**42.2±5.2**•	26.6 ± 5.5•	**42.7±5.2**•	**42.0±5.1**•	**38.7±4.4**•	**41.2±6.0**•
InlineSkate	55.8±3.9	26.8 ± 5.2•	**46.9±4.3**•	**22.6±4.0**•	49.9 ± 4.3•	**51.6±4.5**•	**43.8±4.7**•	**38.7±4.3**•
ItalyPower	95.5±1.2	**90.6±3.0**•	**95.4±1.3**	56.5 ± 8.2•	**91.2±3.8**•	**93.3±2.4**•	**94.8±1.5**•	**95.8±1.3**
MedicalImg	81.3±2.4	50.9 ± 5.1•	**72.7±3.2**•	**53.0±3.4**•	**66.4±3.9**•	**66.6±3.6**•	**71.6±3.2**•	**62.5±4.0**•
AVG_{ts}	77.1	39.4	**63.7**	**38.7**	**59.1**	**59.7**	**65.9**	**59.9**
AVG	76.2	**57.7**	**61.9**	**41.1**	**66.5**	**68.9**	**67.4**	**66.6**

Similar performance improvements stemming from the better quality of hubness estimates in the neighbor occurrence models can also be seen in other examined hubness-aware classifiers, hw-kNN (Tables 7 and 8), h-FNN (Tables 9 and 10) and NHBNN (Tables 5 and 6). In some cases, the combination of instance selection and hubness-aware classification can yield consistent and significant improvements over classification without prior selection. On the tested Gaussian data, the improvement is visible in the unbiased case for NHBNN when coupled with CNN, hw-kNN when coupled with GCNN and h-FNN when coupled with either CNN, GCNN or AL1. The summaries are given in Figure 25.

Out of all the considered hubness-aware k-nearest neighbor classifiers (HIKNN, NHBNN, hw-kNN, h-FNN), the smallest increase in performance was noted in hw-kNN, which is not surprising, as it is based on the aggregate mislabeling estimates, unlike the other three algorithms that consider the detailed occurrence profiles. Even though this makes hw-kNN somewhat less sensitive to the effects of the selection bias, we can not recommend its use,

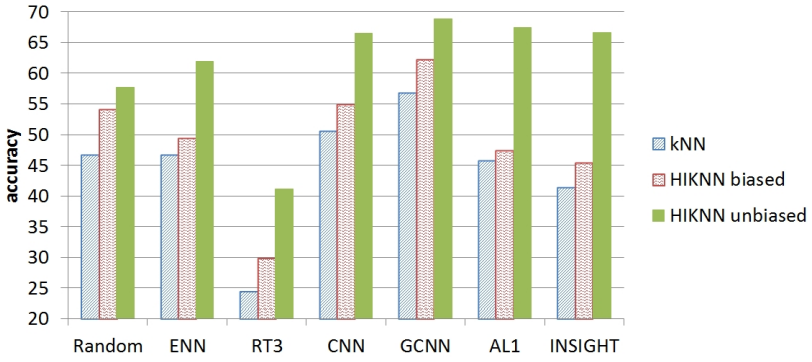


Figure 23: The overall accuracy improvement achieved by using the unbiased hubness estimate in HIKNN. Significant improvements are achieved for every instance selection method.

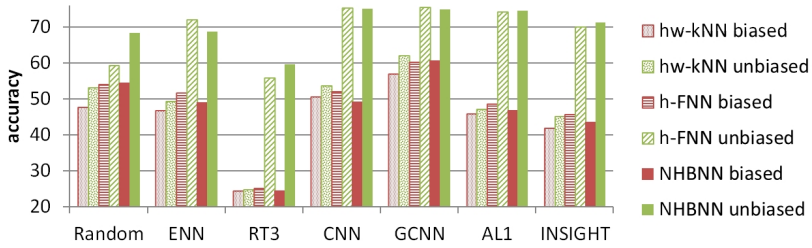


Figure 24: The accuracy improvements obtained by using the unbiased prototype hubness estimation in hw-kNN, h-FNN and NHBNN.

as its overall performance is still worse than in HIKNN, NHBNN or h-FNN. This confirms what was previously documented in earlier studies where these classifiers were compared.

We can conclude that the existing results suggest that hubs play an important role in kNN-based instance selection methods and that they should be taken into account when designing future instance selection strategies for data reduction in high-dimensional data.

Table 5: Cross-validated classification accuracy of Naive Hubness-Bayesian k-nearest neighbor classifier (NHBNN) under several different selection strategies. The model is trained on the prototype set only, which means that the *biased* hubness estimate is used. ∘ and • denote significantly better or worse result ($p < 0.01$) than NHBNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test.

classifier: NHBNN, hubness estimate: BIASED								
	Instance selection method							
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	84.2±1.6	78.6±78.6•	81.8±1.6•	56.8±8.6•	81.5± 1.7•	83.4± 1.4•	81.5±1.6•	79.6±1.8•
iNet4	68.4±1.2	63.1± 1.3•	65.2±1.3•	42.8±2.9•	66.4± 1.3•	67.4± 1.4•	66.2±1.2•	64.0±1.5•
iNet5	64.3±1.3	56.5± 1.5•	57.8±1.4•	27.4±4.8•	60.9± 1.4•	62.5± 1.2•	60.6±1.3•	56.7±1.3•
iNet6	66.8±1.4	57.5± 1.5•	56.2±1.5•	33.2±2.8•	62.9± 1.3•	64.5± 1.3•	60.7±1.5•	54.2±1.5•
iNet7	63.2±0.9	53.3± 1.3•	39.0±1.3•	37.4±3.3•	58.0± 1.0•	60.9± 1.0•	50.6±2.1•	48.8±2.8•
AVG_{img}	69.4	61.9	60.0	39.5	66.0	67.8	64.0	60.7
GM_1	87.0±0.7	69.0± 4.7•	26.4±2.1•	12.7±1.4•	48.8± 6.1•	86.2± 0.7•	29.5±3.2•	26.9±5.9•
GM_2	85.1±0.9	65.3± 5.7•	36.2±2.1•	20.7±3.0•	67.8± 2.7•	81.5± 1.2•	32.5±3.1•	41.0±4.9•
GM_3	88.8±0.8	68.0± 5.9•	55.2±3.7•	13.1±5.0•	77.6± 2.7•	75.1± 2.3•	38.6±3.5•	55.9±6.0•
GM_4	85.0±0.7	64.0± 5.0•	30.5±1.9•	17.1±2.5•	60.0± 2.6•	69.9± 2.4•	42.3±4.9•	30.7±3.4•
GM_5	85.8±0.7	65.5± 5.4•	44.8±2.5•	10.1±3.0•	56.0± 4.1•	84.5± 0.8•	23.5±2.0•	46.3±4.7•
GM_6	87.9±0.7	67.9± 4.7•	24.1±1.6•	22.1±4.7•	47.3± 4.1•	71.8± 1.9•	48.0±4.9•	30.4±3.7•
GM_7	88.0±0.6	69.3± 6.4•	12.2±0.6•	14.9±1.1•	23.0± 3.2•	87.7± 0.6	13.3±1.7•	13.3±1.0•
GM_8	86.8±0.7	66.8± 4.8•	56.8±2.0•	13.8±3.8•	80.8± 2.2•	85.4± 0.8•	39.8±5.1•	59.9±2.2•
GM_9	82.5±0.8	62.4± 4.1•	16.0±0.8•	12.2±2.7•	34.4± 3.6•	81.9± 0.8	20.0±1.6•	17.5±2.8•
GM_{10}	87.4±0.7	67.7± 6.8•	45.9±3.7•	13.2±2.2•	71.4± 3.3•	85.5± 0.9•	32.5±4.1•	44.7±4.4•
AVG_{gm}	86.4	66.6	34.8	15.0	56.7	80.9	32.0	36.7
50words	79.1±3.2	22.5± 5.8•	36.0±5.3•	12.7±5.0•	15.7± 3.8•	15.2± 4.0•	34.4±7.5•	21.6±6.1•
Adiac	63.9±3.6	6.7 ± 3.0•	15.2±3.7•	4.9 ±2.6•	9.6 ± 3.2•	9.2 ± 3.3•	9.6 ±3.9•	7.4 ±3.4•
Cricket X	82.6±2.9	24.3± 6.7•	57.2±5.1•	15.4±5.4•	24.8± 5.8•	25.3± 6.6•	54.8±7.6•	28.4±7.1•
Cricket Y	81.4±2.7	27.1± 5.2•	51.1±6.5•	19.4±5.4•	21.8± 5.6•	20.7± 5.5•	59.4±6.2•	29.3±6.9•
Cricket Z	83.0±2.9	25.3± 5.8•	67.9±4.3•	13.4±4.7•	35.9± 7.0•	38.1± 6.8•	56.2±7.9•	33.0±6.5•
ECGFiveD	99.0±0.8	88.5± 3.6•	98.7±0.9	49.0±5.2•	36.1±11.1•	53.1±10.9•	96.5±1.7•	89.9±3.4•
Haptics	46.6±4.6	28.5± 5.9•	38.5±4.8•	28.4±5.8•	35.1± 4.8•	34.4± 4.1•	31.0±4.4•	33.0±6.5•
InlineSkate	55.8±3.7	23.3± 4.5•	46.5±4.5•	20.2±4.4•	38.7± 4.3•	39.0± 3.8•	36.7±5.5•	30.0±4.6•
ItalyPower	95.6±1.2	90.1± 3.4•	95.3±1.4	56.8±7.9•	62.8±12.7•	79.7± 7.8•	92.9±1.8•	92.5±2.5•
MedicalImg	81.2±2.4	53.2± 4.0•	72.6±2.9•	43.9±4.5•	57.9± 4.1•	58.4± 3.7•	64.5±4.1•	54.1±4.3•
AVG_{ts}	76.8	38.9	57.9	26.4	33.8	37.3	53.6	41.9
AVG	79.2	54.6	49.1	24.5	49.2	60.8	47.0	43.6

Table 6: Cross-validated classification accuracy of Naive Hubness-Bayesian k-nearest neighbor classifier (NHBNN) under several different selection strategies. The neighbor occurrence profiles in the model are derived by observing the unrestricted prototype occurrences on the entire training set, leading to an *unbiased* hubness estimate. ∘ and • denote significantly better or worse result ($p < 0.01$) than NHBNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test. Accuracies higher than in the biased case are given in **bold**.

classifier: NHBNN, hubness estimate: UNBIASED								
		Instance selection method						
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	84.2±1.6	**80.3±1.5**•	**83.4±1.3**•	**79.9± 1.9**•	**82.1±1.5**•	83.1 ± 1.3•	**82.4±1.5**•	**81.1±1.4**•
iNet4	68.4±1.2	**64.6±1.3**•	**67.3±1.1**•	**62.6± 1.5**•	**67.5±1.1**•	**67.8±1.2**	**66.6±1.3**•	**65.4±1.2**•
iNet5	64.3±1.3	**57.2±1.4**•	**60.6±1.2**•	**51.6± 1.5**•	**62.1±1.0**•	**63.4±1.3**•	59.8 ± 1.4•	**57.3±1.2**•
iNet6	66.8±1.4	**59.3±1.2**•	**63.9±1.3**•	32.7 ± 1.8•	**64.4±1.4**•	**65.6±1.1**•	**62.8±1.3**•	**59.6±1.3**•
iNet7	63.2±0.9	**57.1±0.9**•	**60.2±0.9**•	**38.0± 4.4**•	**61.4±0.9**•	**62.3±0.9**•	**60.1±1.1**•	**58.0±1.0**•
AVG_{img}	69.4	**63.7**	**67.1**	**53.0**	**67.6**	**68.5**	**66.4**	**64.3**
GM_1	87.0±0.7	**82.7±0.7**•	**82.0±0.7**•	**77.3± 1.0**•	**87.8±0.6**∘	**87.0±0.6**	**85.1±0.7**•	**84.0±0.7**•
GM_2	85.1±0.9	**81.3±0.9**•	**79.6±0.9**•	**82.6± 1.0**•	**87.1±0.7**∘	**85.2±0.8**	**83.7±0.7**•	**82.2±0.9**•
GM_3	88.8±0.8	**85.2±0.8**•	**84.2±0.8**•	**80.1± 1.5**•	**90.0±0.8**∘	**89.1±0.8**	**87.7±0.7**•	**86.0±0.7**•
GM_4	85.0±0.7	**80.5±0.8**•	**80.3±0.7**•	**73.3± 1.5**•	**85.5±0.7**	**84.9±0.7**	**83.1±0.7**•	**82.4±0.7**•
GM_5	85.8±0.7	**81.9±0.8**•	**78.8±0.7**•	**70.7± 1.9**•	**88.1±0.7**∘	**86.5±0.7**∘	**84.0±0.7**•	**83.4±0.8**•
GM_6	87.9±0.7	**83.8±0.9**•	**82.8±0.7**•	**81.7± 1.3**•	**89.1±0.6**∘	**87.9±0.8**	**86.7±0.6**•	**85.4±0.7**•
GM_7	88.0±0.6	**84.6±0.7**•	**83.5±0.7**•	**79.0± 0.9**•	**88.8±0.5**∘	**88.0±0.5**	**86.5±0.6**•	**86.4±0.7**•
GM_8	86.8±0.7	**83.3±0.9**•	**79.6±0.7**•	**80.5± 1.2**•	**88.5±0.7**∘	**86.8±0.7**	**85.2±0.8**•	**84.2±0.8**•
GM_9	82.5±0.8	**79.0±0.9**•	**75.8±0.8**•	**72.0± 1.0**•	**84.5±0.8**∘	**82.6±0.8**	**81.0±0.8**•	**79.7±0.8**•
GM_{10}	87.4±0.7	**83.9±1.0**•	**81.7±0.9**•	**78.3± 1.0**•	**89.8±0.7**∘	**88.2±0.7**∘	**86.1±0.7**•	**85.5±0.7**•
AVG_{gm}	86.4	**82.6**	**80.8**	**77.6**	**87.9**	**86.6**	**84.9**	**83.9**
50words	79.1±3.2	**48.6±4.0**•	**47.6±3.5**•	**47.6± 3.9**•	**63.6±3.6**•	**63.7±3.7**•	**69.2±3.8**•	**55.0±3.9**•
Adiac	63.9±3.6	**34.5±4.7**•	**26.7±3.4**•	**27.1± 4.5**•	**41.1±3.9**•	**40.7±3.7**•	**47.3±4.0**•	**37.4±4.4**•
Cricket X	82.6±2.9	**53.0±4.5**•	52.1 ± 5.2•	**44.4± 5.4**•	**69.8±4.5**•	**69.9±4.0**•	**69.0±3.8**•	**63.6±4.5**•
Cricket Y	81.4±2.7	**52.9±4.2**•	46.8 ± 5.1•	**45.4± 6.1**•	**67.2±4.2**•	**67.4±3.8**•	**70.9±3.9**•	**60.4±4.5**•
Cricket Z	83.0±2.9	**53.3±4.4**•	51.4 ± 5.1•	**43.8± 5.1**•	**69.2±3.9**•	**69.0±3.3**•	**70.7±3.8**•	**64.0±4.2**•
ECGFiveD	99.0±0.8	**91.9±2.5**•	**98.7±0.7**	**53.4± 4.8**•	**97.5±1.2**•	**97.4±1.3**•	94.8 ± 4.8•	**93.6±1.7**•
Haptics	46.6±4.6	**37.7±5.6**•	**42.3±5.6**•	**37.0± 5.5**•	**43.0±5.0**•	**42.4±5.0**•	**42.2±4.9**•	**40.8±5.4**•
InlineSkate	55.8±3.7	**34.6±4.6**•	44.5 ± 4.7•	**27.9± 4.1**•	**45.0±4.8**•	**45.4±3.9**•	**49.3±4.1**•	**40.3±3.9**•
ItalyPower	95.6±1.2	**94.0±1.9**•	95.1 ± 1.3	**65.5±10.6**•	**94.9±1.5**	**95.0±1.3**	**95.3±1.8**	**95.8±1.2**
MedicalImg	81.2±2.4	**63.3±3.3**•	69.6 ± 2.9•	**56.8± 3.2**•	**69.7±3.0**•	**69.9±2.9**•	**72.2±3.1**•	**68.5±3.3**•
AVG_{ts}	76.8	**56.4**	57.5	**44.9**	**66.1**	**66.1**	**68.1**	**62.0**
AVG	79.2	**68.3**	**68.7**	**59.6**	**75.1**	**74.8**	**74.5**	**71.2**

Table 7: Cross-validated classification accuracy of hw-kNN under several different selection strategies. The model is trained on the prototype set only, which means that the *biased* hubness estimate is used. ∘ and • denote significantly better or worse result ($p < 0.01$) than hw-kNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test.

classifier: hw-kNN, hubness estimate: BIASED								
	Instance selection method							
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	85.6±1.6	79.9±3.0•	81.9±2.5•	52.3±6.2•	76.6±3.1•	81.2±2.9•	81.6±2.4•	80.5± 1.5•
iNet4	69.5±1.4	61.5±1.8•	63.9±1.3•	41.6±3.2•	65.9±1.4•	67.9±1.5•	63.4±1.6•	63.5± 1.6•
iNet5	67.2±1.4	56.6±2.3•	57.1±1.4•	25.4±4.3•	61.3±1.7•	63.1±1.3•	56.8±1.7•	58.4± 1.4•
iNet6	67.8±1.5	57.5±1.9•	56.0±1.6•	31.5±4.2•	61.1±1.6•	62.4±1.4•	58.4±1.6•	56.5± 1.8•
iNet7	64.3±1.1	52.2±1.8•	40.8±1.3•	36.5±3.1•	55.1±1.5•	58.3±1.4•	48.9±1.8•	48.5± 3.2•
AVG_{img}	70.9	61.5	59.9	37.5	64.0	66.6	61.8	61.5
GM_1	68.7±4.6	52.3±4.5•	28.5±1.8•	12.0±1.5•	45.1±4.7•	69.2±3.8	31.4±3.3•	30.0± 4.9•
GM_2	64.4±4.9	46.8±6.5•	32.9±2.2•	25.7±4.2•	55.5±2.8•	62.4±4.2•	31.5±3.2•	38.3± 4.5•
GM_3	71.0±4.4	54.8±6.6•	48.9±2.8•	13.6±4.1•	68.3±2.6•	63.8±2.9•	37.2±3.6•	52.3± 7.2•
GM_4	67.1±4.1	47.7±5.3•	29.4±1.9•	16.4±2.7•	47.0±2.8•	54.1±2.9•	40.2±4.2•	31.3± 3.2•
GM_5	59.6±6.1	41.3±6.7•	36.9±2.6•	8.7 ±3.3•	45.1±3.6•	58.2±5.0	21.0±2.2•	41.8± 4.7•
GM_6	66.9±5.1	51.0±5.7•	22.2±1.7•	22.8±4.6•	45.5±3.0•	60.4±2.9•	44.2±4.7•	28.2± 3.3•
GM_7	69.0±5.5	49.3±5.7•	10.0±0.7•	15.4±1.1•	23.6±3.7•	68.9±4.9	12.0±2.2•	11.6± 1.3•
GM_8	61.7±6.4	48.4±6.3•	41.7±2.6•	16.2±3.6•	62.0±3.6	61.3±5.9	37.4±3.9•	51.1± 4.6•
GM_9	57.3±5.5	41.5±5.4•	13.5±0.9•	12.3±2.9•	26.2±3.0•	57.0±4.7	17.5±1.6•	16.5± 2.1•
GM_{10}	67.1±5.3	47.7±6.7•	38.7±2.8•	12.7±2.6•	58.6±3.4•	66.3±4.5	28.6±4.0•	41.7± 3.7•
AVG_{gm}	65.3	48.1	30.3	15.6	47.7	62.2	30.1	34.3
50words	79.7±3.2	21.1±5.8•	29.8±5.3•	10.9±4.5•	16.8±4.8•	16.6±4.7•	32.4±8.3•	18.8± 5.6•
Adiac	65.1±3.5	8.4 ±3.3•	15.2±3.9•	5.2 ±2.5•	12.9±4.0•	11.9±3.3•	11.0±4.7•	7.9 ± 3.5•
Cricket X	82.8±2.9	28.0±7.5•	56.5±5.1•	15.9±5.3•	29.1±6.9•	29.9±7.0•	56.5±7.2•	31.5± 7.9•
Cricket Y	81.5±2.9	30.5±6.3•	50.8±6.1•	20.1±5.2•	24.2±6.3•	23.2±6.6•	60.1±6.3•	31.2± 8.2•
Cricket Z	83.1±2.8	28.4±7.0•	67.5±4.5•	14.2±5.1•	46.8±8.2•	49.1±8.3•	56.3±7.7•	35.8± 6.9•
ECGFiveD	99.0±0.8	88.7±3.5•	94.9±9.1•	50.1±3.9•	91.4±3.7•	90.9±4.2•	96.4±1.7•	89.9± 3.4•
Haptics	45.9±4.5	30.6±5.7•	38.6±5.1•	28.8±5.7•	39.7±5.0•	39.9±5.1•	31.5±4.4•	33.4± 6.6•
InlineSkate	55.7±3.9	27.0±4.8•	46.6±4.5•	20.7±4.5•	49.8±4.4•	49.2±4.5•	36.3±5.4•	33.3± 4.6•
ItalyPower	95.5±1.2	89.0±5.6•	95.2±1.4•	58.1±7.6•	88.8±4.5•	90.4±3.5•	92.6±2.0•	63.0±13.8•
MedicalImg	80.9±2.4	51.3±4.8•	70.4±3.2•	41.7±4.3•	66.1±3.8•	66.1±4.1•	63.4±5.2•	50.0± 5.1•
AVG_{ts}	76.9	40.3	56.6	26.6	46.6	46.7	53.6	39.5
AVG	71.1	47.7	46.7	24.4	50.5	56.9	45.8	41.8

Table 8: Cross-validated classification accuracy of hw-kNN under several different selection strategies. The neighbor occurrence profiles in the model are derived by observing the unrestricted prototype occurrences on the entire training set, leading to an *unbiased* hubness estimate. ∘ and • denote significantly better or worse result ($p < 0.01$) than hw-kNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test. Accuracies higher than in the biased case are given in **bold**.

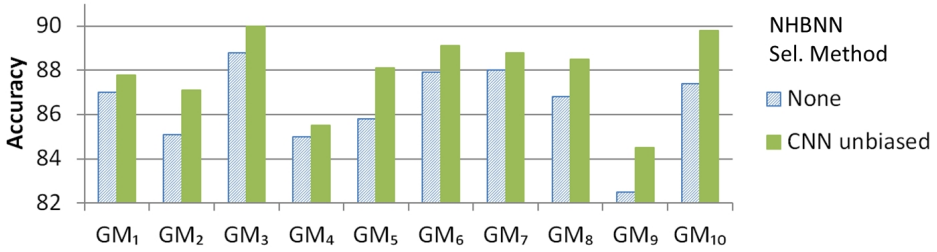
classifier: hw-kNN, hubness estimate: UNBIASED								
		Instance selection method						
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	85.6±1.6	**80.7±1.6**•	**83.6±1.2**•	49.3 ± 6.7•	**82.8±1.6**•	**84.1±1.6**•	**82.9±1.5**•	**80.7±1.6**•
iNet4	69.5±1.4	**63.4±1.6**•	**66.4±1.2**•	**42.7±3.1**•	**68.3±1.2**•	**68.8±1.2**	**66.6±1.4**•	**63.6±1.7**•
iNet5	67.2±1.4	**58.9±1.5**•	**58.6±1.2**•	24.4 ± 3.9•	**63.8±1.3**•	**65.7±1.2**•	**61.3±1.7**•	**60.6±1.2**•
iNet6	67.8±1.5	**59.7±1.5**•	**58.6±1.4**•	**32.9±2.9**•	**64.4±1.5**•	**65.7±1.2**•	**61.9±1.5**•	**58.0±1.6**•
iNet7	64.3±1.1	**55.1±1.4**•	**41.4±1.3**•	**37.0±3.5**•	**57.9±1.2**•	**60.3±1.1**•	**50.2±1.8**•	**48.7±3.3**•
AVG_{img}	70.9	**63.6**	**61.7**	**37.2**	**67.5**	**68.9**	**64.6**	**62.3**
GM_1	68.7±4.6	**63.1±3.3**•	**33.4±2.3**•	**12.5±1.6**•	**48.9±4.4**•	**78.6±0.8**∘	**32.2±3.3**•	28.1 ± 4.4•
GM_2	64.4±4.9	**58.3±5.1**•	**37.4±2.6**•	25.3 ± 4.2•	**59.4±2.3**•	**73.9±1.3**∘	**33.7±3.4**•	**45.1±5.6**•
GM_3	71.0±4.4	**66.1±3.8**•	**56.7±2.9**•	**16.6±6.2**•	**71.9±2.1**	**70.9±1.8**	**39.5±3.8**•	**59.7±4.6**•
GM_4	67.1±4.1	**57.3±4.5**•	**33.0±2.4**•	16.4 ± 3.1•	**54.7±1.8**•	**61.5±1.9**•	**43.9±3.8**•	**38.1±4.6**•
GM_5	59.6±6.1	**58.8±3.5**	**45.0±2.2**•	6.3 ± 1.8•	**54.4±3.6**•	**75.5±1.0**∘	**24.1±2.1**•	**49.2±3.6**•
GM_6	66.9±5.1	**61.9±3.9**•	**24.9±1.8**•	**24.9±5.8**•	**47.3±3.1**•	**69.6±1.4**∘	**45.9±4.8**•	**31.6±3.4**•
GM_7	69.0±5.5	**61.4±5.4**•	**11.5±0.6**•	**16.4±1.2**•	**27.3±3.0**•	**79.6±0.9**∘	**15.3±2.4**•	**13.6±1.6**•
GM_8	61.7±6.4	**60.4±4.6**	**49.2±2.0**•	**22.8±4.8**•	**73.3±2.0**∘	**76.1±1.2**∘	**39.9±3.9**•	**57.4±3.1**•
GM_9	57.3±5.5	**52.5±4.1**•	**15.4±0.9**•	**13.2±3.1**•	**31.9±3.2**•	**70.3±1.0**∘	**17.9±1.8**•	**17.5±1.8**•
GM_{10}	67.1±5.3	**63.9±5.4**•	**44.7±3.0**•	**13.3±1.9**•	**67.9±2.4**	**80.4±0.9**∘	**34.1±3.6**•	**50.2±4.7**•
AVG_{gm}	65.3	**60.4**	**35.1**	**16.8**	**53.7**	**73.7**	**32.6**	**39.0**
50words	79.7±3.2	**21.6±5.7**•	29.2 ± 4.9•	10.2 ± 4.4•	16.5 ± 4.6•	**17.5±4.9**•	**32.5±8.2**•	17.7 ± 6.9•
Adiac	65.1±3.5	8.3 ± 4.0•	14.9 ± 4.5•	4.8 ± 2.5•	**13.0±3.9**•	**13.0±3.7**•	10.6 ± 5.8•	**9.6 ±4.0**•
Cricket X	82.8±2.9	27.4 ± 7.4•	**58.1±5.4**•	14.7 ± 5.6•	**29.2±7.3**•	29.7 ± 6.1•	55.4 ± 7.8•	**31.7±8.7**•
Cricket Y	81.5±2.9	**33.2±7.2**•	**52.7±5.3**•	**20.6±6.0**•	**24.9±6.7**•	**23.8±5.9**•	58.9 ± 6.7•	**32.4±8.4**•
Cricket Z	83.1±2.8	**28.4±7.3**•	67.2 ± 4.8•	14.2 ± 4.2•	**47.9±8.6**•	**49.3±9.4**•	55.8 ± 7.2•	35.2 ± 7.9•
ECGFiveD	99.0±0.8	87.9 ± 3.9•	**98.7±0.8**	**52.5±4.9**•	91.0 ± 3.9•	90.4 ± 4.0•	96.3 ± 1.8•	89.5 ± 3.0•
Haptics	45.9±4.5	**32.1±5.9**•	**39.9±4.9**•	25.2 ± 5.3•	**40.9±4.9**•	39.8 ± 5.0•	31.3 ± 4.3•	**34.9±6.4**•
InlineSkate	55.7±3.9	**27.0±5.2**•	45.6 ± 4.3•	**20.9±3.9**•	48.8 ± 4.4•	**50.3±4.5**•	34.5 ± 5.7•	33.0 ± 4.8•
ItalyPower	95.5±1.2	**90.8±3.0**•	95.0 ± 1.3	**59.0±8.1**•	**90.1±3.7**•	**91.6±2.7**•	91.6 ± 2.5•	**91.4±2.9**•
MedicalImg	80.9±2.4	50.9 ± 5.0•	70.2 ± 3.1•	**42.7±4.5**•	65.6 ± 3.6•	66.0 ± 3.7•	61.5 ± 4.8•	**51.2±4.7**•
AVG_{ts}	76.9	**40.7**	**57.2**	26.5	**46.8**	**47.1**	52.9	**42.7**
AVG	71.1	**53.2**	**49.3**	**24.8**	**53.7**	**62.1**	**47.1**	**45.1**

Table 9: Cross-validated classification accuracy of h-FNN under several different selection strategies. The model is trained on the prototype set only, which means that the *biased* hubness estimate is used. ∘ and • denote significantly better or worse result ($p < 0.01$) than h-FNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test.

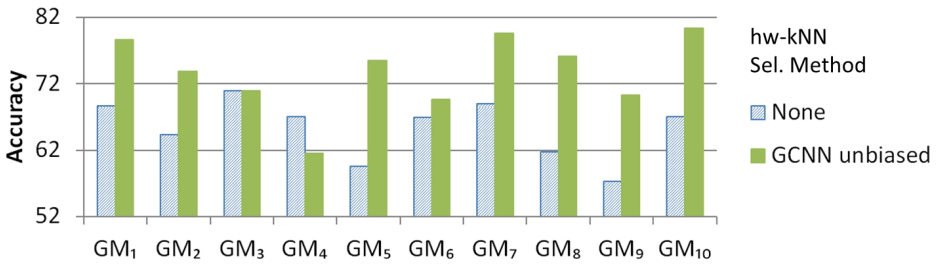
classifier: h-FNN, hubness estimate: BIASED								
		Instance selection method						
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	84.4±1.5	79.3±1.8•	81.6±1.6•	51.7±6.3•	80.1±1.7•	81.1±1.7•	81.7±1.6•	80.3±1.6•
iNet4	68.6±1.4	62.2±1.3•	65.3±1.2•	42.8±3.0•	67.0±1.3•	67.8±1.4	65.9±1.3•	63.6±1.5•
iNet5	66.5±1.7	57.9±1.6•	55.9±1.6•	25.4±4.5•	61.6±1.6•	63.9±1.1•	58.9±1.5•	58.5±1.4•
iNet6	66.9±1.5	58.1±1.6•	60.8±1.5•	31.7±4.4•	60.9±1.4•	62.7±1.3•	61.1±1.3•	57.5±1.5•
iNet7	62.7±1.1	53.3±1.3•	42.9±1.5•	37.1±3.7•	56.2±1.2•	59.2±1.2•	50.6±1.6•	49.6±2.5•
AVG_{img}	69.8	62.2	61.3	37.4	65.2	66.9	63.6	61.9
GM_1	78.8±3.0	68.4±3.2•	34.9±3.2•	14.3±1.5•	54.0±5.1•	79.0±1.7	36.6±3.9•	33.6±6.7•
GM_2	77.7±2.4	62.1±5.6•	44.0±3.1•	24.9±4.5•	68.1±1.9•	77.2±1.5	36.9±3.9•	47.4±5.8•
GM_3	82.7±1.7	66.6±4.6•	64.1±3.7•	16.0±5.3•	74.3±1.7•	74.0±1.6•	43.9±4.3•	61.8±5.9•
GM_4	79.9±1.6	62.5±3.9•	34.0±2.7•	20.2±3.1•	60.7±2.6•	67.0±2.4•	46.9±5.3•	33.1±4.7•
GM_5	76.2±3.4	62.2±5.4•	53.6±2.8•	8.8 ±3.3•	60.8±4.2•	76.0±3.2	28.0±1.7•	53.2±3.9•
GM_6	80.8±2.4	65.4±4.0•	31.3±1.6•	21.0±5.7•	49.9±3.7•	68.6±1.7•	53.9±5.1•	34.1±4.3•
GM_7	80.7±4.5	67.6±6.0•	11.7±1.0•	16.6±1.3•	23.4±3.5•	81.2±3.5	12.7±2.3•	13.3±1.2•
GM_8	76.6±4.3	64.1±4.8•	58.0±2.1•	19.2±4.4•	74.3±2.0	78.3±2.6	42.5±5.3•	61.8±2.5•
GM_9	75.0±3.3	60.1±4.4•	13.9±1.0•	15.3±3.1•	35.0±3.6•	75.7±3.0	18.8±2.0•	17.1±2.8•
GM_{10}	80.1±3.5	65.1±6.6•	56.2±4.6•	14.3±3.2•	67.6±2.2•	79.5±3.4	35.5±4.9•	51.8±5.8•
AVG_{gm}	78.9	64.4	40.2	17.1	56.8	75.7	35.6	40.7
50words	78.3±3.1	21.4±6.0•	35.7±5.6•	12.9±4.8•	15.9±3.7•	15.6±4.1•	35.0±7.8•	21.9±5.6•
Adiac	62.4±3.7	7.3 ±3.2•	15.1±3.7•	5.0 ±2.6•	10.9±3.7•	9.8 ±2.9•	10.2±4.0•	7.9 ±3.5•
Cricket X	78.9±3.1	24.9±6.9•	53.5±5.7•	16.0±5.4•	23.8±6.0•	24.7±6.5•	56.0±7.5•	27.7±8.0•
Cricket Y	80.1±2.7	28.9±6.1•	57.9±6.7•	20.2±5.5•	20.6±6.1•	18.9±5.4•	59.0±5.9•	32.0±7.2•
Cricket Z	81.9±2.8	26.3±6.3•	66.4±4.7•	14.4±4.6•	37.8±6.9•	40.6±6.9•	57.3±7.2•	33.4±6.9•
ECGFiveD	98.6±0.9	87.9±3.7•	98.6±0.9	49.9±4.7•	66.4±5.3•	71.8±5.3•	95.8±1.9•	89.1±3.7•
Haptics	45.4±4.5	29.7±5.9•	40.9±5.1•	30.9±6.0•	37.0±4.6•	36.9±4.7•	32.1±5.0•	35.7±6.3•
InlineSkate	52.8±3.5	24.1±5.0•	46.7±4.7•	21.1±4.3•	43.9±4.5•	43.8±4.5•	37.7±5.7•	30.5±4.5•
ItalyPower	95.3±1.4	90.7±3.7•	95.3±1.4	59.6±9.1•	86.3±4.6•	88.3±4.4•	93.1±1.8•	93.3±2.0•
MedicalImg	80.4±2.4	52.5±4.3•	71.9±3.1•	43.2±4.4•	62.0±4.0•	62.5±3.6•	64.7±4.5•	52.9±4.5•
AVG_{ts}	78.4	39.4	58.2	27.3	40.5	41.3	54.1	42.4
AVG	76.9	54.0	51.6	25.2	52.0	60.2	48.6	45.6

Table 10: Cross-validated classification accuracy of h-FNN under several different selection strategies. The neighbor occurrence profiles in the model are derived by observing the unrestricted prototype occurrences on the entire training set, leading to an *unbiased* hubness estimate. ∘ and • denote significantly better or worse result ($p < 0.01$) than h-FNN with no instance selection, trained on the entire training set, based on the corrected re-sampled t-test. Accuracies higher than in the biased case are given in **bold**.

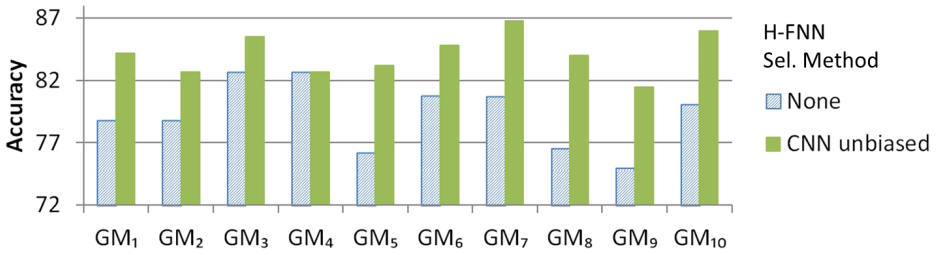
classifier: h-FNN, hubness estimate: UNBIASED								
		Instance selection method						
Data set	None	Random	ENN	RT3	CNN	GCNN	AL1	INSIGHT
iNet3	84.4±1.5	**79.6±1.6**•	**83.6±1.3**	**77.7± 1.8**•	**81.1±1.6**•	**83.1±1.7**•	81.6 ± 1.4•	80.3 ± 1.4•
iNet4	68.6±1.4	**63.5±1.3**•	**66.4±1.1**•	**61.7± 1.3**•	**67.5±1.2**•	**68.0±1.2**	**66.3±1.3**•	**64.9±1.2**•
iNet5	66.5±1.7	**60.3±1.2**•	**63.8±1.3**•	**54.9± 1.3**•	**64.9±1.2**•	**65.9±1.2**	**62.9±1.4**•	**61.4±1.2**•
iNet6	66.9±1.5	**59.9±1.1**•	**64.5±1.3**•	**33.2± 3.3**•	**65.2±1.5**•	**66.1±1.2**•	**63.3±1.5**•	**61.3±1.1**•
iNet7	62.7±1.1	**56.9±1.0**•	**59.2±0.9**•	**38.8± 3.6**•	**61.3±1.0**•	**62.3±1.0**	**59.8±1.1**•	**57.3±1.0**•
AVG_{img}	69.8	**64.1**	**67.5**	**53.3**	**68.0**	**69.1**	**66.8**	**65.0**
GM_1	78.8±3.0	**76.5±0.8**•	**77.3±0.9**•	**68.7± 1.0**•	**84.2±0.7**∘	**83.5±0.7**∘	**79.7±0.8**∘	**78.0±0.9**
GM_2	77.7±2.4	**73.3±1.2**•	**74.9±1.1**•	**73.2± 1.4**•	**82.7±0.9**∘	**82.2±0.8**∘	**78.1±0.9**	**76.0±1.1**•
GM_3	82.7±1.7	**77.9±1.2**•	**79.6±1.0**•	**68.9± 2.1**•	**85.5±0.9**∘	**86.0±0.9**∘	**82.1±0.9**	**79.3±1.0**•
GM_4	79.9±1.6	**73.4±0.9**•	**75.7±0.8**•	**67.5± 1.4**•	**82.7±0.7**∘	**82.8±0.8**∘	**78.0±0.8**•	**76.2±0.9**•
GM_5	76.2±3.4	**73.0±1.0**•	**73.8±0.9**•	**59.1± 2.1**•	**83.2±0.8**∘	**82.2±0.8**∘	**76.7±0.9**	**76.1±0.9**
GM_6	80.8±2.4	**77.2±1.1**•	**78.3±0.9**•	**74.6± 1.9**•	**84.8±0.7**∘	**85.4±0.8**∘	**81.6±0.8**	**79.7±0.9**•
GM_7	80.7±4.5	**80.2±0.8**	**80.8±0.7**	**69.5± 1.2**•	**86.8±0.6**∘	**86.8±0.6**∘	**83.6±0.7**∘	**83.2±0.6**∘
GM_8	76.6±4.3	**75.6±1.0**	**74.2±1.0**•	**68.0± 1.9**•	**84.0±0.9**∘	**84.0±0.8**∘	**78.4±1.1**∘	**76.5±1.1**
GM_9	75.0±3.3	**72.7±1.2**•	**70.7±0.9**•	**61.9± 1.1**•	**81.5±0.8**∘	**79.8±0.7**∘	**76.2±0.9**	**73.4±0.9**•
GM_{10}	80.1±3.5	**79.0±1.0**•	**79.3±0.9**	**65.7± 1.2**•	**86.0±0.7**∘	**85.5±0.8**∘	**81.3±0.8**∘	**79.6±0.9**
AVG_{gm}	78.9	**75.9**	**76.5**	**67.7**	**84.1**	**83.8**	**79.6**	**77.8**
50words	78.3±3.1	20.9 ± 5.6•	**65.5±3.4**•	**54.4± 3.6**•	**68.8±3.2**•	**69.0±3.7**•	**74.5±3.4**•	**62.5±3.3**•
Adiac	62.4±3.7	**8.1 ±4.0**•	**50.9±4.0**•	**32.2± 4.3**•	**52.8±3.8**•	**53.1±3.4**•	**56.8±3.5**•	**44.8±4.0**•
Cricket X	78.9±3.1	**27.1±7.4**•	**71.3±3.5**•	**45.3± 6.1**•	**72.1±4.3**•	**72.6±3.6**•	**75.2±3.1**•	**65.7±4.1**•
Cricket Y	80.1±2.7	**32.1±7.2**•	**72.0±3.5**•	**47.5± 5.8**•	**71.9±3.6**•	**71.4±3.0**•	**75.9±3.4**•	**62.4±4.0**•
Cricket Z	81.9±2.8	**27.9±7.3**•	**74.9±3.3**•	**44.9± 5.5**•	**73.6±3.2**•	**73.9±3.2**•	**75.9±3.2**•	**65.4±4.0**•
ECGFiveD	98.6±0.9	87.4 ± 4.0•	98.4 ± 0.8	**52.0± 4.7**•	**97.7±1.1**•	**97.7±1.1**•	**98.2±0.9**	**93.6±1.7**•
Haptics	45.4±4.5	**31.7±6.0**•	**44.6±5.5**	**31.3± 5.0**•	**42.3±5.1**•	**42.4±5.5**•	**43.4±4.7**•	**43.3±5.7**•
InlineSkate	52.8±3.5	**26.7±5.1**•	**48.9±4.4**•	**26.9± 4.2**•	**50.9±4.4**•	**51.4±4.6**•	**51.9±4.3**•	**42.5±3.9**•
ItalyPower	95.3±1.4	90.4 ± 3.0•	95.1 ± 1.3	**60.6±10.3**•	**95.0±1.4**	**95.0±1.3**	**95.8±1.3**	**95.9±1.2**
MedicalImg	80.4±2.4	50.9 ± 5.0•	**76.6±2.5**•	**59.0± 3.4**•	**75.9±2.6**•	**75.7±3.0**•	**76.4±2.4**•	**69.7±3.2**•
AVG_{ts}	78.4	**40.3**	**69.8**	**45.4**	**70.1**	**70.2**	**72.4**	**64.6**
AVG	76.9	**59.3**	**72.0**	**55.9**	**75.3**	**75.4**	**74.2**	**70.0**



(a) NHBNN



(b) hw-kNN



(c) h-FNN

Figure 25: The accuracy improvement in hubness-aware k-nearest neighbor classification under instance selection with unbiased hubness estimates when compared to the baseline case where no selection was done (i.e. where the model was trained on the entire training set). The average biased accuracy over $GM_1 - GM_{10}$ is much smaller in every case, merely 56.7% for NHBNN, 62.2% for hw-kNN and 56.8% for h-FNN. This shows how the unbiased hubness estimate might in some cases entirely prevent a decrease in model performance and even lead to better and more robust classification models.

2.2.4 Hubs in Class Imbalanced Data and the Robustness of Hubness-aware Classification

Learning under class imbalance is an important problem in supervised and unsupervised learning methods [Holte et al., 1989]. The relative difference in class densities in borderline regions might induce severe misclassification. The minority classes are under-represented and more difficult to model.

In this section the role of hubs under the assumption of class imbalance is investigated [Tomašev and Mladenić, 2013]. It will be shown that high-dimensional data exhibit some surprising properties that need to be taken into account. Even though hubness and class imbalance seem to be unrelated phenomena, there is some interplay in the difficulties they pose for the k-nearest neighbor classifiers. Apart from analyzing the connections between class imbalance and hubness, this Section also deals with the overall robustness of the proposed hubness-aware approaches. The results are reported for the analysis under severe mislabeling rates and significant class overlap.

Many standard classifiers are not very effective when dealing with class imbalanced data. Algorithms which induce classification models usually adopt the maximum generality bias [Holte et al., 1989]. Data sets with significant class imbalance often pose difficulties for learning algorithms [Weiss, 2004], especially those with high generality bias. Such algorithms tend to over-generalize on the majority class, which in turn leads to a lower performance on the minority class. Designing good methods capable of coping with highly imbalanced data still remains a daunting task. In contrast, the k-nearest neighbor classifier exhibits high specificity bias, since it retains all the examples. The specificity bias is considered a desired property of algorithms designed for handling highly imbalanced data. Not surprisingly, kNN has been advocated as one way of handling such imbalanced data sets [van den Bosch et al., 1997][Holte et al., 1989]. As the extensions of kNN are frequently used in class imbalanced classification scenarios, examining the role of hubs in such classification seems to be quite relevant.

Certain concerns have recently been raised about the applicability of the basic kNN approach in imbalanced scenarios [Garcia et al., 2008][Hand and Vinciotti, 2003]. The method requires high densities to deliver good probability estimates. These densities are often closely related to class size, which makes kNN somewhat sensitive to the imbalance level. The difference among the densities between the classes becomes critical in the overlap regions. Data points from the denser class (usually the *majority class*) are often encountered as neighbors of points from the less dense category (usually the *minority class*). In high-dimensional data, the task is additionally complicated by the well-known *curse of dimensionality*.

The problem of learning from imbalanced data has recently attracted attention of both industry and academia alike. Many classification algorithms used in real-world systems and applications fail to meet the performance requirements when faced with severe class distribution skews [He and Garcia, 2009][Fernandez et al., 2011] and overlapping data distributions [Prati et al., 2004]. Various approaches have been developed in order to deal with this issue, including some forms of class under-sampling or over-sampling [Chawla et al., 2002][He et al., 2008][Liu et al., 2006][Zhang and Mani, 2003][Batista et al., 2004][Kubat and Matwin, 1997], misclassification cost-sensitive techniques [McCarthy et al., 2005][Zhou and Liu, 2006][Ting, 2002], decision trees [Liu et al., 2010], kernel methods [Wu and Chang, 2005][Hong et al., 2007] or active learning [Ertekin et al., 2007b][Ertekin et al., 2007a].

As mentioned before, high specificity bias makes kNN one of the prime candidates for various extensions aimed at handling imbalanced data. Probably the simplest way of introducing some imbalanced class distribution awareness is to assign some appropriate instance weights [Tan, 2005a][Wang et al., 2010b]. This hopes to compensate for the minority class being under-represented in the data by assigning higher voting weights to its members. Such tweaks are of limited scope, especially as it is known that classifier performance mostly de-

pends on the class imbalance in the overlap regions, which does not directly correspond to the overall imbalance used to generate the instance weights. Moreover, the degree of imbalance can vary across different class borders, rendering such a unified correction severely flawed. Some other approaches seem to be more promising, as for instance the examplar-based kNN [Li and Zhang, 2011]. It introduces the concept of detecting pivot minority points, which are then expanded to Gaussian balls. This makes pivot points closer to other minority class examples, hence improving the classification. Another interesting research direction has been outlined in [Liu and Chawla, 2011], where the authors argue that the problem in handling imbalanced data with kNN arises from trying to estimate *prior* class probabilities in points of interest. They suggest that a more complex probabilistic model built on neighbor attributes might be able to circumvent some of the difficulties that are frequently encountered with prior estimates. Nearest-neighbor methods have also occasionally been used for oversampling/undersampling, as in [Zhang and Mani, 2003].

2.2.4.1 Bad Hubness Under Class Imbalance: the Hypothesis

The usual interpretation of the bad influence of imbalanced data on kNN classification is that the majority class points would often become neighbors of the minority class examples, due to the relative difference in densities between different categories. As neighbors, they would often cause misclassification of the minority class. Consequently, the methods which are being proposed for imbalanced data classification and are focused primarily on rectifying this by improving the overall classifier performance on the minority class. Naturally, something has to be sacrificed in return and usually it is the recall of the majority class.

This is certainly reasonable. In many real-world problems the misclassification cost is much higher for the minority class. Some well-known examples include cancer detection, oil spill recognition, earthquake prediction, terrorist detection, etc. However, things are not so simple as they might seem. Often enough, the cost of misclassifying the majority class is almost equally high. In fraud detection [Ezawa et al., 1996][Ezawa and Schuermann, 1995], accusing innocent people of fraud might lose customers for the companies involved and incur a significant financial loss. Even in breast cancer detection it has recently been shown that the current diagnostic techniques lead to significant over-diagnosis of cancer cases [Kalager et al., 2012]. This leads to many otherwise healthy women being admitted for treatment and subjected to various drug courses and/or operating procedures.

The extreme pathological case of major bad image hubs arising from careless data preprocessing that was described in [Tomašev et al., 2011a] already shows how things may go awry if the minority instances turn into bad hubs. In that particular case, the culprit was 'noise', i.e. the feature extraction system which failed to output representations for the five images in question, as well as the data preparation/preprocessing module which failed to anticipate such a possibility. So, we might be inclined to think that if there is no noise in the data, all is fine. Such thinking is, in fact, highly misleading.

What that example strongly suggests is that when we are dealing with high-dimensional data, we should actually be more concerned about the *minority class hubs causing misclassification of the majority class points* instead of the other way around. This is exactly the opposite of what most imbalanced data classification algorithms are trying to solve. It is a very important observation, especially because most of the data that is being automatically processed and mined is in fact high-dimensional and exhibits hubness, whether it is text, images, video, time series, etc. [Radovanović et al., 2009][Radovanović et al., 2010a][Tomašev et al., 2011a][Radovanović et al., 2010]

Such a phenomenon is easy to overlook, as it is highly counterintuitive. In lower dimensional data, most misclassification in imbalanced data sets occurs in border regions where classes overlap and have different densities. Naturally, the class with a lower local density (which is usually the minority class) gets misclassified more often. This intuition breaks due

to the influence of curse of dimensionality. All data is sparse, distances concentrate, and most importantly - most misclassification is caused by *bad hubs*.

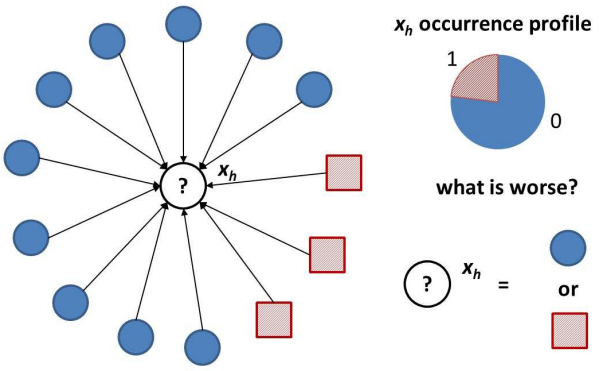


Figure 26: An illustrative example. x_h is a hub, neighbor to many other points. There is a certain label distribution among its reverse nearest neighbors, defining the occurrence profile of x_h. It is obvious that most damage would be done to the classification process by x_h if it were to share the label of the minority part of its reverse neighbor set. On average, we would expect this to equal the overall minority class in the data. This suggests that minority hubs might have a higher average tendency to become bad hubs and that this might prove to be, in general, quite detrimental to classifier performance.

At first glance, it might not be clear how this changes the rules of the game, but the key lies hidden in understanding the mechanisms behind the hubness phenomenon. [Radovanović et al., 2009][Radovanović et al., 2010a][Radovanović et al., 2010b] As we have mentioned earlier, general hubness of the data arises from its intrinsically high dimensionality and is either enhanced or diminished by the particular choice of feature representation and similarity measure. What this means is that the hubness of *particular points* has a geometric interpretation rather than a strictly semantic one. In fact, bad hubs are by their very definition points where the semantics of the nearest-neighbor relation is most severely compromised and the similarity measure fails to capture the intended semantic similarity between neighbor points. The more compromised the semantic relation is, the more the occurrence profile of bad hubs tends to take shape of either the global prior class distribution or some local prior distribution.

Of course, as misclassification in nearest-neighbor methods is caused exclusively by label mismatch in neighbor-sets, 1-NN misclassification rate for a particular hub-point is trivially maximized if its label matches the minority class in its occurrence profile. This is illustrated in Figure 26. In the more general case of kNN, this claim does not necessarily hold in every point (as a single bad occurrence does not always entail misclassification), but it should hold on average. Due to the very nature of hubness, we would expect that the minority class in the occurrence profile would quite often match the overall minority class in the data.

2.2.4.2 Data and the Experimental Setup

In order to test the above stated hypotheses, an extensive experimental evaluation was performed. In our experiments we have used both low hubness data sets (mostly balanced) and high-hubness image data sets (mostly imbalanced). The former were taken from the UCI repository (http://archive.ics.uci.edu/ml/datasets.html), the latter from the ImageNet public collection (http://www.image-net.org/). The image data was represented as a quantized SIFT representation extended by a color histogram. More details can be found in [Tomašev et al., 2011b][Tomašev et al., 2011a].

From the first five image data sets a random subset of instances was removed from all the minority classes in order to make the data even more imbalanced for the experiments.

The relevant properties of the data sets are given in Table 11. The listed UCI data sets were mostly not imbalanced and we included them in Table 12 for later tests involving induced mislabeling in determining the robustness of the hubness-aware approaches.

Table 11: Summary of data sets. Each data set is described by the following set of properties: size, number of features (d), number of classes (c), skewness of the 5-occurrence distribution (S_{N_5}), the percentage of *bad* 5-occurrences (BN_5), the degree of the largest hub-point ($\max N_5$), relative imbalance of the label distribution (RImb) and the size of the majority class ($p(c_M)$)

Data set	size	d	C	S_{N_5}	BN_5	$\max N_5$	RImb	$p(c_M)$
diabetes	768	8	2	0.19	32.3%	14	0.30	65.1%
ecoli	336	7	8	0.15	20.7%	13	0.41	42.6%
glass	214	9	6	0.26	25.0%	13	0.34	35.5%
iris	150	4	3	0.32	5.5%	13	0	33.3%
mfeat-factors	2010	216	10	0.83	7.8%	25	0	10%
mfeat-fourrier	2000	76	10	0.93	19.6%	27	0	10%
ovarian	2534	72	2	0.50	15.3%	16	0.28	64%
segment	2310	19	7	0.33	5.3%	15	0	14.3%
sonar	208	60	2	1.28	21.2%	22	0.07	53.4%
vehicle	846	18	4	0.64	35.9%	14	0.02	25.8%
iNet3	2731	416	3	8.38	21.0%	213	0.40	50.2%
iNet4	6054	416	4	7.69	40.3%	204	0.14	35.1%
iNet5	6555	416	5	14.72	44.6%	469	0.20	32.4%
iNet6	6010	416	6	8.42	43.4%	275	0.26	30.9%
iNet7	10544	416	7	7.65	46.2%	268	0.09	19.2%
iNet3Imb	1681	416	3	3.48	17.2%	75	0.72	81.5%
iNet4Imb	3927	416	4	7.39	38.2%	191	0.39	54.1%
iNet5Imb	3619	416	5	9.35	41.4%	258	0.48	58.7%
iNet6Imb	3442	416	6	4.96	41.3%	122	0.46	54%
iNet7Imb	2671	416	7	6.44	42.8%	158	0.46	52.1%

Some experiments on synthetic Gaussian data were also performed and the details are available in Section 2.2.4.4.

All classification tests were performed as 10-times 10-fold cross-validation. Corrected re-sampled t-test was used to detect statistical significance. Manhattan metric was used in all real-world experiments, while the Euclidean distance was used for dealing with Gaussian mixtures. All feature values in UCI and ImageNet data were normalized to the $[0,1]$ range. All the hubness-aware algorithms were tested under their default parameter configurations.

2.2.4.3 The Curse of Minority Hubs

While analyzing the connection between hubness and class imbalance we will focus on the image datasets shown in the lower half of Table 11. To measure the imbalance of a particular dataset, we will observe two quantities: $p(c_M)$, which is the relative size of the majority class - and relative imbalance (RImb) of the label distribution which we define as the normalized standard deviation of the class probabilities from the absolutely homogenous mean value of $1/c$ for each class. In other words, $\mathrm{RImb} = \sqrt{(\sum_{c \in C}(p(c) - 1/C)^2)/((C-1)/C)}$. In our experiments we will not be dealing with binary imbalanced data (one very frequent majority class, one very rare minority class), so by this measure we are trying to quantify imbalance in class distribution in a more general sense.

Class-to-class *k-occurrence matrices* are an excellent way to gain a quick insight into the kNN structure of the data. We will discuss one such matrix for iNet7Imb data set in Table 13. Each row contains average outgoing hubness from one category to another. Each column contains expressed hubness towards a particular category by all other classes in the data. On the diagonal we are able to see the percentage of occurrences of points from each category in neighborhoods of points from the same category (i.e. good hubness). We see

Table 12: Experiments on UCI and ImageNet data. Classification accuracy is given for kNN, hubness-weighted kNN (hw-kNN), hubness-based fuzzy nearest neighbor (h-FNN), naive hubness-Bayesian k-nearest neighbor (NHBNN) and hubness information k-nearest neighbor (HIKNN). All experiments were performed for $k = 5$. The symbols $\bullet/\circ$ denote statistically significant worse/better performance ($p < 0.05$) compared to kNN. The best result in each line is in bold.

Data set	kNN	hw-kNN	h-FNN	NHBNN	**HIKNN**
diabetes	67.8 ± 3.7	75.6 ± 3.7 ∘	75.4 ± 3.2 ∘	73.9 ± 3.4 ∘	**75.8 ± 3.6** ∘
ecoli	82.7 ± 4.2	86.9 ± 4.1 ∘	**87.6 ± 4.1** ∘	86.5 ± 4.1 ∘	87.0 ± 4.0 ∘
glass	61.5 ± 7.3	65.8 ± 6.7	67.2 ± 7.0 ∘	59.1 ± 7.5	**67.9 ± 6.7** ∘
iris	95.3 ± 4.1	**95.8 ± 3.7**	95.3 ± 3.8	95.6 ± 3.7	95.4 ± 3.8
mfeat-factors	94.7 ± 1.1	96.1 ± 0.8 ∘	95.9 ± 0.8 ∘	95.7 ± 0.8 ∘	**96.2 ± 0.8** ∘
mfeat-fourier	77.1 ± 2.2	81.3 ± 1.8 ∘	82.0 ± 1.6 ∘	**82.1 ± 1.7** ∘	**82.1 ± 1.7** ∘
ovarian	91.4 ± 3.6	92.5 ± 3.5	93.2 ± 3.5	93.5 ± 3.3	**93.8 ± 2.9**
segment	87.6 ± 1.5	88.2 ± 1.3	88.8 ± 1.3 ∘	87.8 ± 1.3	**91.2 ± 1.1** ∘
sonar	82.7 ± 5.5	83.4 ± 5.3	82.0 ± 5.8	81.1 ± 5.6	**85.3 ± 5.5**
vehicle	62.5 ± 3.8	65.9 ± 3.2 ∘	64.9 ± 3.6	63.7 ± 3.5	**67.2 ± 3.6** ∘
iNet3	72.0 ± 2.7	80.8 ± 2.3 ∘	**82.4 ± 2.2** ∘	81.8 ± 2.3 ∘	82.2 ± 2.0 ∘
iNet4	56.2 ± 2.0	63.3 ± 1.9 ∘	**65.2 ± 1.7** ∘	64.6 ± 1.9 ∘	64.7 ± 1.9 ∘
iNet5	46.6 ± 2.0	56.3 ± 1.7 ∘	**61.9 ± 1.7** ∘	61.8 ± 1.9 ∘	60.8 ± 1.9 ∘
iNet6	60.1 ± 2.2	68.1 ± 1.6 ∘	69.3 ± 1.7 ∘	69.4 ± 1.7 ∘	**69.9 ± 1.9** ∘
iNet7	43.4 ± 1.7	55.1 ± 1.5 ∘	**59.2 ± 1.5** ∘	58.2 ± 1.5 ∘	56.9 ± 1.6 ∘
iNet3Imb	72.8 ± 2.4	87.7 ± 1.7 ∘	87.6 ± 1.6 ∘	84.9 ± 1.9 ∘	**88.3 ± 1.6** ∘
iNet4Imb	63.0 ± 1.8	68.8 ± 1.5 ∘	69.9 ± 1.4 ∘	69.4 ± 1.5 ∘	**70.3 ± 1.4** ∘
iNet5Imb	59.7 ± 1.5	63.9 ± 1.8 ∘	64.7 ± 1.8 ∘	63.9 ± 1.8 ∘	**65.5 ± 1.8** ∘
iNet6Imb	62.4 ± 1.7	69.0 ± 1.7 ∘	**70.9 ± 1.8** ∘	68.4 ± 1.8 ∘	70.2 ± 1.8 ∘
iNet7Imb	55.8 ± 2.2	63.4 ± 2.0 ∘	64.1 ± 2.3 ∘	63.1 ± 2.1 ∘	**64.3 ± 2.1** ∘
AVG	69.77	75.40	76.38	75.23	**76.75**

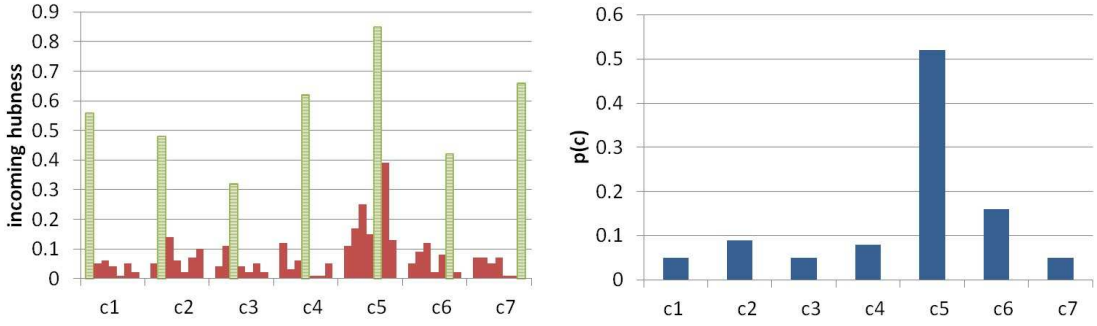
that in iNet7Imb the majority class has highest relative good hubness. It also seems that most of the bad hubness expressed by the minority classes is directed towards the majority class. We can see this more clearly by observing the graph of *incoming hubness* (i.e. visualize Table 13 column by column). This is displayed in Figure 27. So it seems that, at least in this particular image data set, most bad hubness is generated by the minority classes and most of this bad influence is directed towards the majority class (c5).

Table 13: Class-to-class hubness between different classes in iNet7Imb for $k = 5$. Each row contains the outgoing occurrence rate towards other categories. For instance, in the first row we see that only 56% of all neighbor occurrences of points from the first class are in the neighborhoods of elements from the same class. The diagonal elements (self-hubness) are given in bold, as well as the majority class.

	p(c)	c1	c2	c3	c4	c5	c6	c7
c1	0.05	**0.56**	0.05	0.04	0.12	0.11	0.05	0.07
c2	0.08	0.05	**0.48**	0.11	0.03	0.17	0.09	0.07
c3	0.05	0.06	0.14	**0.32**	0.06	0.25	0.12	0.05
c4	0.08	0.04	0.06	0.04	**0.62**	0.15	0.02	0.07
c5	**0.52**	0.01	0.02	0.02	0.01	**0.85**	0.08	0.01
c6	0.17	0.05	0.07	0.05	0.01	0.39	**0.42**	0.01
c7	0.05	0.02	0.10	0.02	0.05	0.13	0.02	**0.66**

Even though misclassification is induced by the cumulative influence of various sorts of bad hubness, the two are not the same and not even necessarily highly correlated when $k > 1$, even though we would expect to see strong correlation in most real-world data sets. Any single bad occurrence does not entail misclassification. For misclassification to occur in kNN, a majority of neighbors must belong to a particular class which does not match the label in the point of interest. So, for $k = 5$, even if the point x had two neighbors with the

correct label and three other neighbors with incorrect labels - but each having a different one - the point x would be properly classified by kNN even though most of its neighbors do not share its label. Therefore, it is not sufficient to check the incoming and expressed hubness distributions, one must also take into account the kNN confusion matrix. The confusion matrix for iNet7Imb data is given in Table 14, generated by averaging after 10 runs of 10-fold cross-validation.



(a) incoming hubness (b) class distribution

Figure 27: The *incoming hubness* towards each category expressed by other categories in the data shown for iNet7Imb data set. The 7 bars in each group represent columns of the class-to-class k-occurrence Table 13. Neighbor sets were computed for $k = 5$. We see that most hubness expressed by the minority classes is directed towards the majority class. This gives some justification to our hypothesis that in high-dimensional data with hubness it is mostly the minority class instances that cause misclassification of the majority class and not the other way around.

Table 14: The average 5-NN confusion matrix for iNet7Imb data after 10-times 10-fold cross-validation. Each row displays how elements of a particular class were assigned to other classes by the 5-NN classifier. The overall number of false negatives (FN) and false positives (FP) for each category is calculated. The results for the majority class are in bold.

	p(c)	c1	c2	c3	c4	c5	c6	c7	FN
c1	0.05	42.9	13.5	3.8	11.8	**6.2**	60.7	1.1	97.1
c2	0.08	22.8	48.0	15.3	8.9	**54.9**	77.1	0.0	179.0
c3	0.05	8.9	21.0	13.0	3.3	**25.6**	55.2	0.0	114.0
c4	0.08	44.0	6.0	2.0	100.5	**15.5**	43.0	0.0	110.5
c5	**0.52**	**78.5**	**36.7**	**25.9**	**21.9**	**1028.1**	**200.9**	**0.0**	**363.9**
c6	0.17	16.9	19.1	10.2	4.3	**142.9**	254.6	0.0	193.4
c7	0.05	17.9	8.3	6.1	12.1	**41.0**	36.9	3.7	122.3
	FP	189.0	104.6	63.3	62.3	**286.1**	473.8	1.1	

Several things in Table 14 are worth noting. First of all, the majority class FP rate is lower than its FN rate, which means that more errors are made on average by misclassifying the majority class points than by misclassifying the minority class points into the majority class. Also, the highest FP rate is not achieved by the majority class, but rather by one of the minority classes - $c6$. Both of these observations are very important, as we have already mentioned that there are various scenarios where the cost of misclassifying the majority class points is quite high. [Ezawa et al., 1996][Ezawa and Schuermann, 1995][Kalager et al., 2012]

The previously discussed correlation between relative class size and bad hubness can be established also by inspecting a collection of imbalanced data sets (iNet3Imb-iNet7Imb) at the same time. Pearson correlation between class size and class-specific bad hubness is -0.76 when taken for $k = 5$. This implies that there might be a very strong negative correlation between the two quantities and that the minority classes indeed exhibit high bad hubness relative to their size. A plot of all $(\frac{p(c)}{p(c_M)}, BN_5(c))$ is shown in Figure 28.

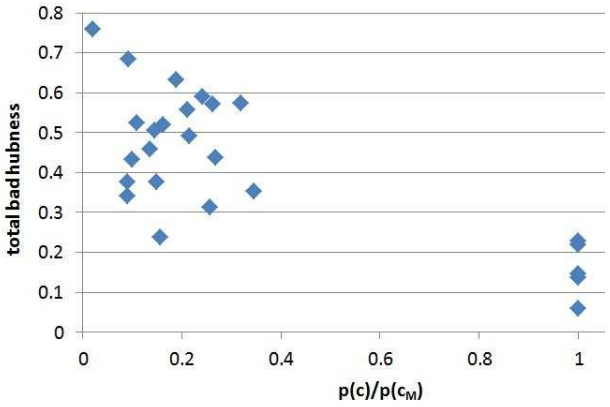


Figure 28: Average bad hubness exhibited by each class from data sets iNet3Imb-iNet7Imb plotted against relative class size ($p(c)/p(c_M)$). We see that the minority classes exhibit on average much higher bad hubness than the majority classes.

This poses an entirely new challenge for imbalanced data classification algorithm design. We could formulate this new requirement in the following way: *Imbalanced data classification algorithms tailored for high-dimensional data ought to be able to simultaneously improve the recall of both the minority AND the majority class.* This is a much more difficult problem, as most current imbalanced data classification algorithms aim at a certain trade-off: giving away some majority class recall in order to increase the minority class recall as much as possible. On the contrary, in data that exhibits hubness, we should seek algorithms capable of tackling both problems at once. In order to achieve this, we need a better understanding of the problem at hand.

In Section 2.2.4.1, it was conjectured that bad hubs among the minority points are expected to have higher bad hubness on average. In order to check this hypothesis, class distributions among different types of points were examined, namely: hubs, anti-hubs and bad hubs. As before, hubs were defined as those points whose occurrence frequency was more than two standard deviations greater than the mean (k), as suggested in [Radovanović et al., 2010a]. A similar criterion was used for determining bad hubs, namely: $\{x : BN_k(x) > \mu_{BN_k(x)} + 2 \cdot \sigma_{BN_k(x)}\}$. We took as many anti-hubs as hub-points, by taking those with least occurrences from the ordered list. Class distributions among these types of points can be compared to the prior distribution over all data points. The comparison for iNet7Imb data is shown in Figure 29. Similar trends are present in the rest of the image data sets, as well. We see that the class distribution is entirely different for different types of points. This needs to be taken into account when modeling the data. Most importantly, we see that in this data set, all top bad hubs come from the minority classes, which further justifies our initial hypothesis. In the rest of the examined image data sets the situation is very similar, though the majority class is naturally not always at 0% among the top hubs, but it is always less frequent than among all points combined.

Considering the anti-hub distribution in Figure 29 can reveal some characteristics of the outlier structure of the data. Previous research [Radovanović et al., 2009][Radovanović et al., 2010a][Radovanović et al., 2010b][Tomašev et al., 2011d] suggests that outliers tend to be anti-hubs in the data, though anti-hubs are not always outliers. The two distributions are, however, correlated. The fact that class $c1$ contributes so much to anti-hubs suggests that this particular minority class consists mostly of outliers.

This purely geometrical interpretation of what it means to be an outlier is not best suited for working with labeled data, as it can easily be refined. We could consider as outliers only those points which are atypical for their own class (i.e. locally) instead of within the entirety of data (as measured by hubness). A simple and effective characterization of points within class-imbalanced data has recently been proposed [Napierala and Stefanowski, 2012]. The

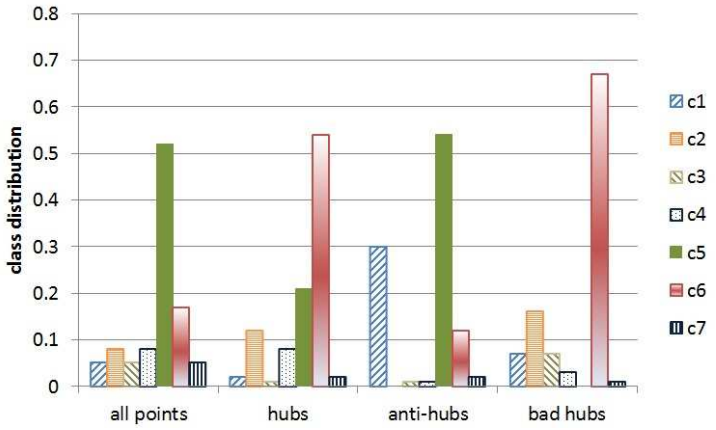


Figure 29: Distribution of classes among different types of points in iNet7Imb data: hubs, anti-hubs and bad hubs. We see that there are nearly no majority class points among the top bad hubs in the data. Data points of class c6 exhibit highest bad hubness, which explains the high FP rate observed in Table 14

authors have proposed a way to distinguish between 4 different point types: *safe points*, *borderline examples*, *rare points* and *outliers*. The distinction is made according to the number of matched labels within the point 5-NN sets. Their initial research investigates only binary classification cases and only deals with the distribution of these point types within a single minority class. The authors conclude that the distribution of minority point types varies across different data sets and determines the difficulty that data poses for classification algorithms.

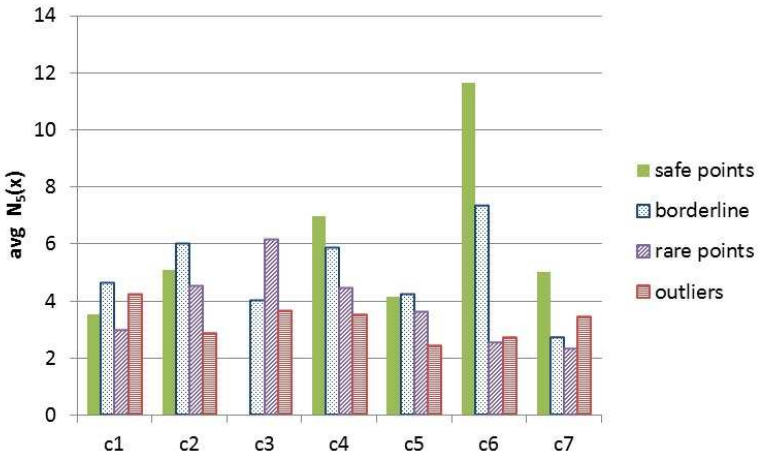


Figure 30: Average hubness of different point types in different categories. Safe points are not consistently the points of highest hubness. Quite frequently borderline examples and even rare points of the minority classes end up being neighbors to other points. This also means that less typical points exhibit a substantial influence on the classification process.

In Figure 30 we can see the distribution of occurrence frequencies among different point types given separately for each category of the iNet7Imb data set. The results indicate a strong violation of the cluster assumption, as point hubness is closely linked to within-cluster centrality [Tomašev et al., 2011e][Tomašev et al., 2013c]. High hubness of borderline points (by the above definition) indicates that data clusters are not homogenous with respect to the label space. Indeed, our initial tests have shown that this data does not cluster well. Another thing worth noting is that points that we usually think of as reliable might have a detrimental influence on the classification process, which is clear from examining the hubness/bad hubness distribution across different point types for $c6$, which has a high overall

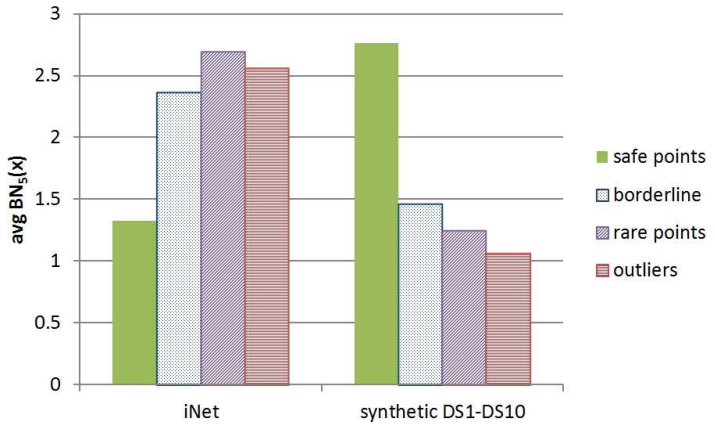


Figure 31: Average 5-NN bad hubness of different point types shown both for iNet and high-dimensional synthetic Gaussian mixtures. We give both bad hubness distributions here for easier comparison. It is clear that they are quite different. In the analyzed image data, most bad influence is exhibited by atypical class points (borderline examples, rare points, outliers), while most bad influence in the Gaussian mixture data is generated by safe points. The latter is quite counterintuitive, as we usually expect for such typical points to be located in the inner regions of class distributions.

bad hubness and FP rate. It is precisely the safe points that exhibit both highest hubness (AVG. 11.66) and highest bad hubness (AVG. 6.63). This is yet another good illustration of the differences between low-dimensional and high-dimensional data. Intuitively, we would expect the safe points to be located in the innermost part of the class distribution space. Such points ought not to become neighbors to *many* other points from different categories. This is precisely what happens here and is yet another counterintuitive result and the point characterization framework [Napierala and Stefanowski, 2012] ought to be used while keeping this in mind.

Bad occurrence distributions summarized in Figure 31 illustrate that different underlying bad hub structures exist in different types of data. In the analyzed image data (iNet3-7, iNetImb3-7), the previously described pathological case of safe/inner points arising as top bad hubs in the data is still more an exception than a rule, while in high-dimensional Gaussian mixtures it becomes a dominating feature.

Of course, we must be tentative when drawing conclusions, as all this still does not imply that the stated correlations hold for all imbalanced high-dimensional data sets. What it shows, though, is that *at least some* real-world (and synthetic) imbalanced high-dimensional data are susceptible to high bad hubness of the minority classes. This peculiar property of high-hubness data has before gone unnoticed and we believe that it should be taken into account in future algorithm design. Bad hubness in the data is apparently closely linked with class imbalance, so a hubness-aware approach to kNN classification should in principle be able to mitigate the influence of bad hubs and improve overall classifier performance.

Classification accuracy is, admittedly, not the best performance measure in class imbalance classification scenarios, so not much can be inferred from the general results in Table 12. Accuracy is only applicable to such scenarios where all classes share very similar misclassification costs. The overall accuracies in Table 12 are given mostly for comparison with experiments in Section 2.2.4.4 and we will base our discussion on algorithm performance under class imbalance on a detailed examination of precision and recall separately.

It is quite clear, however, that unbalancing the class distribution in iNet data did not increase the difficulty of the data sets, as the algorithm performance remained pretty much the same, regardless of the increase in relative imbalance. The same can be said for the average bad hubness shown in Table 11, which has not increased. This is not altogether surprising, as we have already explained that the total bad hubness in the data is caused by an interplay of various contributing factors, so it is only partly caused by class imbalance

Table 15: Precision and recall for each class and each method separately on iNet7Imb data set. Values greater or equal to the score achieved by kNN are given as bold. The last column represents the Spearman correlation between the improvement over kNN in precision or recall and the size of the class. In other words, $\text{corrImp} = corr(\frac{p(c)}{\max p(c)}, \text{improvement})$.

method	measure	c_1	c_2	c_3	c_4	c_5	c_6	c_7		
priors:		0.05	0.08	0.05	0.08	0.52	0.17	0.05	AVG	corrImp
kNN	precision	0.20	0.32	0.18	0.62	0.78	0.35	0.31	0.39	
	recall	0.31	0.21	0.10	0.47	0.74	0.57	0.03	0.35	
hw-kNN	precision	**0.46**	**0.39**	**0.28**	**0.72**	**0.79**	**0.41**	**0.58**	**0.52**	-0.96
	recall	0.30	**0.30**	**0.19**	**0.73**	**0.81**	**0.59**	**0.17**	**0.44**	-0.43
h-FNN	precision	**0.65**	**0.46**	**0.37**	**0.72**	0.69	**0.44**	**0.76**	**0.58**	-0.86
	recall	0.18	0.19	0.09	**0.73**	**0.92**	0.43	**0.12**	**0.38**	-0.07
NHBNN	precision	**0.36**	**0.37**	**0.22**	**0.62**	**0.79**	**0.47**	**0.45**	**0.47**	-0.39
	recall	**0.43**	**0.22**	**0.22**	**0.80**	**0.81**	0.50	**0.20**	**0.45**	-0.68
HIKNN	precision	**0.55**	**0.45**	**0.30**	**0.74**	**0.78**	0.40	**0.67**	**0.55**	-0.75
	recall	0.24	**0.23**	**0.14**	**0.74**	**0.84**	**0.61**	**0.17**	**0.42**	0.0

itself. Also, iNetImb data sets were selected via random undersampling and it is always difficult to predict the effects of data reduction on hubness. Removing anti-hubs makes nearly no difference, but removing hub-points certainly does. After a hub is removed and all neighbor lists are recalculated, the occurrence profiles of many other hub-points change, as they fill in the thereby released 'empty spaces' in neighbor lists where the removed hub participated.

To gain a better understanding of how hubness-aware algorithms handle minority classes in the data, we examined the precision and recall for each class in all of the imbalanced data sets. An example is shown in Table 15 where iNet7Imb data set is discussed. We see that all hubness-aware algorithms improve on average both precision and recall for most individual categories. To further analyze the structure of this improvement, we tested for correlation between class size and the improvement in precision or recall which was achieved by each individual algorithm. As it turns out, hubness-aware algorithms improve precision much more consistently than recall - and this improvement has high negative correlation with relative class size. In other words, *hubness-aware classification improves the precision of minority class categorization*, and the improvement grows for smaller and smaller classes. Actually, NHBNN is an exception, as it soon becomes clear that it behaves differently. A closer examination reveals that the recall of the majority class is improved in all the imbalanced data sets, except when NHBNN is used. This is shown in Figure 32. On the contrary, NHBNN is best at improving the minority class recall, which is not always improved by other hubness-aware algorithms, as shown in Figure 33.

HIKNN is essentially an extension of the basic h-FNN algorithm, so it is interesting to observe such a clear difference between the two. h-FNN is always better at improving the majority class recall, while HIKNN achieves better overall minority class recall. Both algorithms rely on occurrence models, but HIKNN derives more information directly from a neighbor's label and in that sense it certainly has a higher specificity bias, which is reflected in the results. The results of NHBNN, on the other hand, are not so easy to interpret. It seems that the Bayesian modeling of the neighbor-relation differs from the fuzzy model in some subtle way.

Observing precision and recall separately does not allow us to rank the algorithms according to their relative performance, so we will rank them according to the F_1-measure scores [Witten and Frank, 2005a]. We report the micro- and macro-averaged F_1-measure (F_1^μ and F_1^M, respectivelly) for each algorithm over the imbalanced data sets in Table 16. Micro-averaging is affected by class imbalance, so the macro-averaged F_1 scores ought to be preferred. In this case it makes no difference. The results show that all of the hubness-

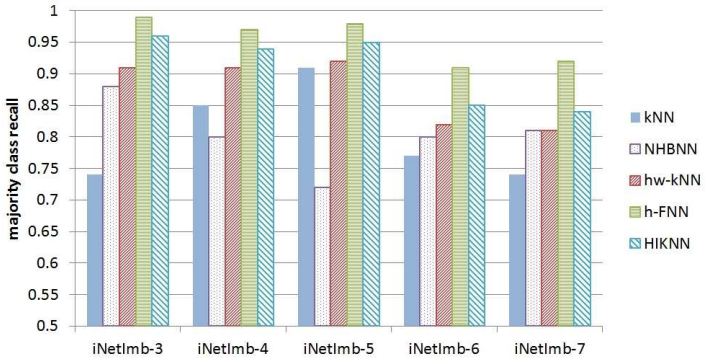


Figure 32: A comparison of majority class recall achieved by both kNN and the hubness-aware classification algorithms on five imbalanced image data sets. Improvements are clear in hw-kNN, h-FNN and HIKNN.

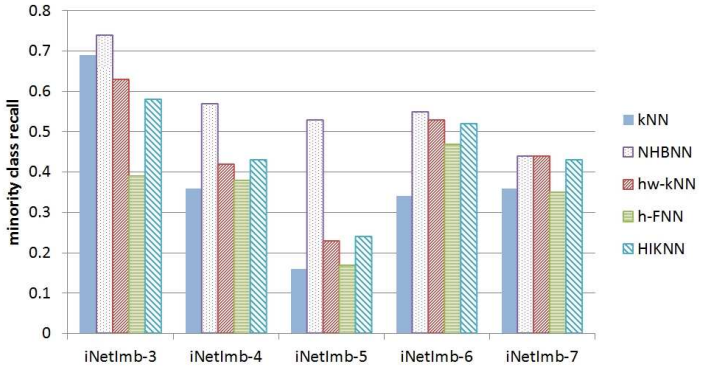


Figure 33: A comparison of the cumulative minority class recall (micro-averaged) achieved by both kNN and the hubness-aware classification algorithms on five imbalanced image data sets. NHBNN seems undoubtedly the best in raising the minority class recall. Other hubness-aware algorithms offer some improvements on iNetImb4-7, but under-perform at iNet3Imb data. In this case, HIKNN is better than h-FNN on all data sets, just as h-FNN was constantly slightly better than HIKNN when raising the majority class recall.

aware approaches improve on the basic kNN in terms of both F_1^μ and F_1^M. NHBNN achieves the best F_1-score, followed by HIKNN and hw-kNN, while h-FNN is, in this case, the least balanced of all the considered hubness-aware approaches.

Table 16: Micro- and macro-averaged F_1 scores of the classifiers on the imbalanced data sets. The best score in each line is in bold.

	kNN	hw-kNN	h-FNN	NHBNN	HIKNN
F_1^μ	0.61	0.68	0.66	**0.70**	0.69
F_1^M	0.43	0.52	0.47	**0.57**	0.53

The property of hw-kNN, h-FNN and HIKNN of significantly raising the recall of the majority class is a very useful one. Especially since they are able to do so without harming the minority class recall, on average. We have already seen that a significant portion of bad hubness in the data is caused by minority class hubs and that it's mostly directed towards the majority class, reducing its recall substantially. These three hubness-aware approaches are there to rectify some of that damage by exploiting the bad hubness information captured by their occurrence models. The advantages are twofold. They are able to generalize and build a model without significantly compromising the classification of the minority classes in the data, which is exactly the opposite of what most imbalanced data algorithms are trying to do. Most importantly, they represent easily extensible voting frameworks. This

designates them as a good basis for constructing hybrid approaches. Indeed, including over-sampling/under-sampling [Chawla et al., 2002][He et al., 2008][Liu et al., 2006][Zhang and Mani, 2003][Batista et al., 2004][Kubat and Matwin, 1997], instance weighting [Tan, 2005b] or examplar-based learning [Li and Zhang, 2011] would not be difficult. Also, unlike the basic kNN and most of its common extensions, hubness-aware kNN algorithms can in principle support cost-sensitive learning. This is made possible by the occurrence model, as not every occurrence has to be given the same weight when calculating $N_{k,c}(x)$. Distance-weighted occurrence models were already analyzed [Tomašev and Mladenić, 2011b], but cost-sensitive occurrence models should also be considered.

2.2.4.4 Robustness: Mislabeling and Class Overlap

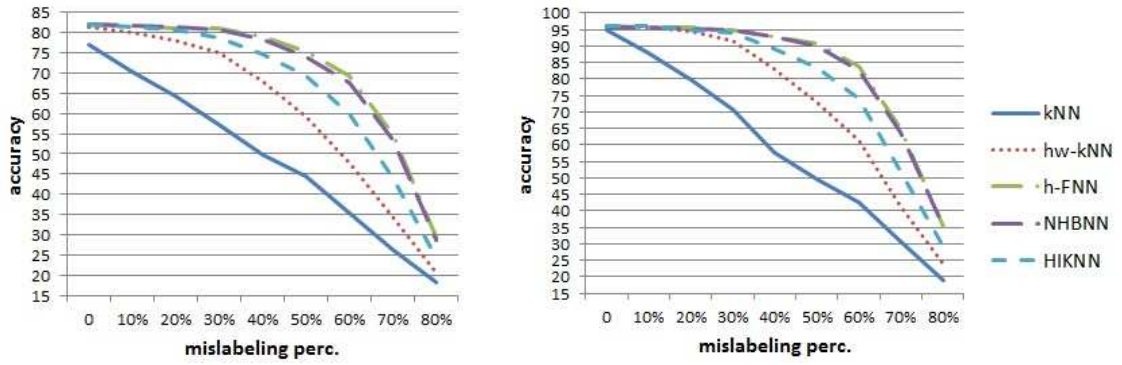
Mislabeled examples are not uncommon in large, complex systems. Detecting and correcting such data points is not an easy task and many correction algorithms have been proposed in an attempt to solve the problem [Hayashi, 2012][Guan et al., 2011][Valizadegan and Tan, 2007]. Regardless, some errors always remain in the data. This is why robustness to mislabeling is very important in classification algorithms. Instance mislabeling is not unrelated to class imbalance. [Hulse et al., 2007] Algorithm performance depends on the distribution of mislabeling across the categories in the data. Even more importantly, the impact of mislabeling on algorithm performance in high-dimensional data depends heavily on the average hubness of mislabeled examples. Mislabeling anti-hubs makes no difference whatsoever. Mislabeling even a couple of hub-points should be enough to cause significant misclassification. Yet, we have predicted that hubness-aware classification algorithms should be able to implicitly deal with mislabeling via occurrence modeling.

Table 17: Experiments on mislabeled data. 30 % mislabeling was artificially introduced to each data set at random. All experiments were performed for $k = 5$. The symbols $\bullet/\circ$ denote statistically significant worse/better performance ($p < 0.05$) compared to kNN. The best result in each line is in bold.

Data set	kNN	hw-kNN	**h-FNN**	NHBNN	HIKNN
diabetes	54.1 ± 3.7	64.7 ± 3.9 ∘	**66.2 ± 3.4** ∘	66.1 ± 3.4 ∘	65.4 ± 3.9 ∘
ecoli	68.1 ± 5.6	80.2 ± 4.7 ∘	**85.8 ± 4.1** ∘	79.3 ± 4.8 ∘	81.7 ± 4.6 ∘
glass	50.6 ± 7.3	61.6 ± 7.3 ∘	**62.8 ± 6.8** ∘	56.8 ± 6.6	61.5 ± 6.7 ∘
iris	71.1 ± 8.5	88.2 ± 6.0 ∘	90.7 ± 5.4 ∘	**93.2 ± 4.6** ∘	87.8 ± 6.3 ∘
mfeat-factors	70.7 ± 2.3	91.4 ± 1.5 ∘	**94.9 ± 1.1** ∘	94.7 ± 1.2 ∘	93.9 ± 1.2 ∘
mfeat-fourier	57.1 ± 2.5	75.0 ± 2.1 ∘	**81.0 ± 1.7** ∘	80.7 ± 1.9 ∘	78.7 ± 1.7 ∘
ovarian	58.1 ± 6.6	76.3 ± 6.1 ∘	**81.1 ± 5.6** ∘	79.4 ± 5.6 ∘	78.3 ± 5.5 ∘
segment	62.7 ± 2.2	81.1 ± 1.9 ∘	**84.3 ± 1.7** ∘	83.8 ± 1.6 ∘	80.8 ± 1.7 ∘
sonar	61.5 ± 7.7	70.8 ± 6.8 ∘	72.4 ± 6.4 ∘	**72.9 ± 6.3** ∘	71.4 ± 6.8 ∘
vehicle	48.2 ± 3.9	57.5 ± 3.9 ∘	58.1 ± 4.0 ∘	56.8 ± 4.0 ∘	**59.2 ± 3.8** ∘
iNet3	51.0 ± 2.3	69.9 ± 2.2 ∘	**81.2 ± 1.8** ∘	80.6 ± 1.6 ∘	75.3 ± 2.0 ∘
iNet4	44.6 ± 1.4	52.5 ± 1.3 ∘	**63.3 ± 1.3** ∘	63.1 ± 1.2 ∘	57.6 ± 1.3 ∘
iNet5	40.0 ± 1.4	47.2 ± 1.4 ∘	**60.6 ± 1.2** ∘	60.0 ± 1.2 ∘	53.1 ± 1.3 ∘
iNet6	49.5 ± 1.7	55.1 ± 1.4 ∘	**68.0** ± 1.3 ∘	67.4 ± 1.3 ∘	62.8 ± 1.4 ∘
iNet7	33.1 ± 1.1	44.8 ± 1.1 ∘	**57.6 ± 1.1** ∘	56.8 ± 1.1 ∘	51.0 ± 1.1 ∘
iNet3Imb	56.7 ± 3.0	78.7 ± 2.2 ∘	**87.0 ± 1.6** ∘	81.1 ± 2.2 ∘	83.2 ± 2.1 ∘
iNet4Imb	51.8 ± 1.7	55.0 ± 1.7 ∘	**68.7 ± 1.7** ∘	67.3 ± 1.8 ∘	63.9 ± 1.7 ∘
iNet5Imb	50.7 ± 2.1	53.5 ± 2.0 ∘	**64.2 ± 2.0** ∘	60.5 ± 1.8 ∘	60.6 ± 1.2 ∘
iNet6Imb	54.7 ± 2.1	55.8 ± 2.0 ∘	**69.7 ± 1.7** ∘	66.6 ± 1.9 ∘	62.8 ± 2.0 ∘
iNet7Imb	33.1 ± 2.3	52.0 ± 1.9 ∘	**62.9 ± 1.9** ∘	61.1 ± 1.9 ∘	58.6 ± 1.7 ∘
AVG	53.37	65.57	**73.03**	71.41	69.38

In our experiments, mislabeling was distributed uniformly across different categories and only the training data on each cross-validation fold was mislabeled. The algorithms were evaluated by consulting the original labels. An overview of algorithm performance under

30% mislabeling rate is shown in Table 17. The results are pretty convincing and reveal that the hubness-aware algorithms exhibit *much higher robustness* than kNN.



(a) mfeat-fourier (b) mfeat-fact

Figure 34: The drop in accuracy as the mislabeling rate increases. The kNN accuracy drops linearly, but that is not the case with hubness-aware approaches, which retain good performance even under high mislabeling rates.

Out of the compared hubness-aware algorithms, h-FNN dominates in this experimental setup. On many datasets h-FNN is no more than 1-2% less accurate than before, which is astounding considering the level of mislabeling in the data. On the other hand, the hubness-weighting approach (hw-kNN) fails in this case and is not able to cope with such high mislabeling rates. This was to be expected, since it is not based on class hubness scores and instances still vote by their own label instead of their occurrence profile.

Of course, we have experimented with various noise levels and Figure 34 depicts the drop in accuracy as mislabeling is slowly introduced in the data. These graphs are interesting, as they show kNN performance decreasing at a linear rate with increasing noise. At the same time, hubness-aware approaches retain most of their accuracy as the mislabeling rate goes all the way up to $40\% - 50\%$. This amazing result has an intuitive explanation. In hubness-aware classification, hub-points are not voting by their labels, so mislabeling does not affect them directly in any way. They vote by their occurrence profiles. As the mislabeling was uniformly distributed, really high mislabeling levels were required to sufficiently compromise the occurrence profiles of major hubs.

These results show that it is actually possible to *exploit the curse of dimensionality* and hubness in particular in order to improve algorithm performance. All that is required is an algorithm design that is implicitly aware of the underlying structure of the data.

Class imbalance is by itself usually not enough to cause serious misclassification. It has to be coupled with some overlap between different class distributions. Such overlap is often present in complex, real-world data. It is, however, very difficult to measure or even detect when the data is very sparse and high-dimensional. Also, in order to study specifically the impact of class overlap on classification performance, we must be able to observe it independently. This is easiest to achieve in synthetic data, as it is the only way that we can be really confident that there is no mislabeling present.

Assuring substantial overlap between classes in high-dimensional data is not as easy as it sounds, as everything tends to be spread far apart. A degree of overlap high enough to induce severe misclassification was required, since we wanted to make the data hard for nearest-neighbor methods (not necessarily for other types of classifiers). We generated a series of 10 synthetic data sets as random 100-dimensional 10-category Gaussian mixtures. High overlap degree was achieved by placing each distribution center (for a given feature) randomly within a certain multiple of the standard deviation from some other randomly chosen, previously determined, distribution center. As shown in Table 18, all the data sets

Table 18: Classification accuracies on synthetic Gaussian mixture data for $k = 10$. For each data set, the skewness of the N_{10} distribution is given along with the bad occurrence rate (BN_{10}). The symbols $\bullet/\circ$ denote statistically significant worse/better performance ($p < 0.01$) compared to kNN. The best result in each line is in bold.

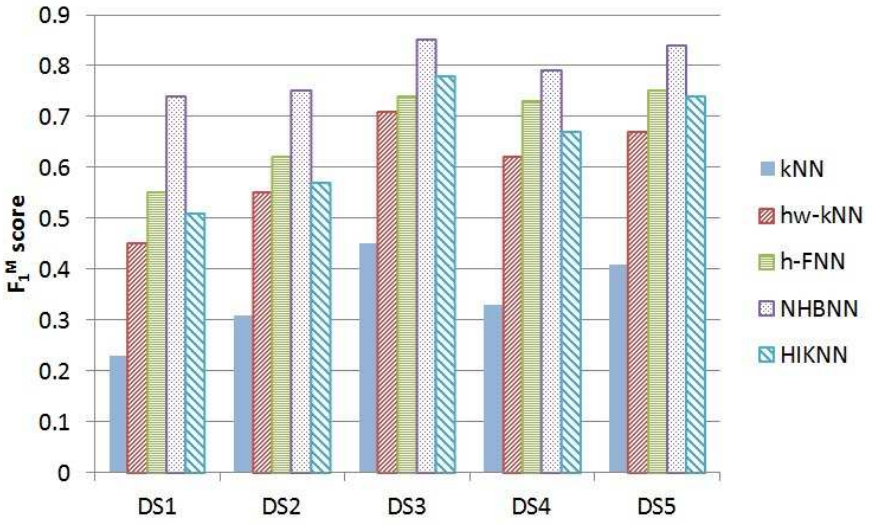
Data set	size	SN_{10}	BN_{10}	kNN	hw-kNN	h-FNN	**NHBNN**	HIKNN
DS_1	1244	6.68	53.5%	43.8 ± 3.1	64.4 ± 5.3 ∘	72.6 ± 2.8 ∘	**80.7 ± 2.4** ∘	65.8 ± 3.0 ∘
DS_2	1660	4.47	49.2%	48.4 ± 2.8	73.6 ± 6.9 ∘	79.3 ± 2.2 ∘	**83.9 ± 2.2** ∘	73.1 ± 2.5 ∘
DS_3	1753	5.50	42.0%	67.3 ± 2.3	85.3 ± 2.6 ∘	86.8 ± 1.7 ∘	**90.0 ± 1.4** ∘	86.7 ± 1.9 ∘
DS_4	1820	3.45	51%	52.2 ± 2.6	72.8 ± 2.3 ∘	78.4 ± 2.2 ∘	**81.9 ± 2.0** ∘	72.2 ± 2.3 ∘
DS_5	1774	4.39	46.3%	59.2 ± 2.7	80.2 ± 3.4 ∘	84.6 ± 1.8 ∘	**87.2 ± 1.5** ∘	81.1 ± 2.1 ∘
DS_6	1282	3.98	45.6%	58.6 ± 3.3	80.0 ± 3.3 ∘	81.7 ± 2.5 ∘	**86.6 ± 2.2** ∘	79.4 ± 2.5 ∘
DS_7	1662	4.64	41.5%	65.0 ± 2.4	84.6 ± 2.4 ∘	85.4 ± 1.9 ∘	**90.1 ± 1.5** ∘	84.5 ± 2.0 ∘
DS_8	1887	4.19	40.0%	71.0 ± 2.3	82.7 ± 2.5 ∘	85.9 ± 1.9 ∘	**88.4 ± 1.8** ∘	83.9 ± 2.3 ∘
DS_9	1661	5.02	47.5%	57.9 ± 2.7	76.3 ± 3.3 ∘	82.3 ± 2.0 ∘	**87.5 ± 1.7** ∘	77.7 ± 2.4 ∘
DS_{10}	1594	4.82	46.9%	57.5 ± 2.9	78.1 ± 3.3 ∘	81.1 ± 2.3 ∘	**85.5 ± 1.9** ∘	77.7 ± 2.2 ∘
AVG				58.09	77.80	81.81	**86.18**	78.21

exhibited very high hubness and very high bad hubness. Imbalance level in the data was moderate. There were no clear majority or minority classes, but some overall imbalance was present, with RImb≈ 0.2 in most data sets. As in previous experiments, we performed 10-times 10-fold cross- validation and the corrected re-sampled t-test was used to verify the statistically significant differences. For this round of experiments, we have opted for setting the neighborhood size to $k = 10$, as we thought that the algorithms might benefit from more information in borderline regions. Euclidean distance was used for dissimilarity, as it makes more sense than the Manhattan distance when dealing with Gaussian data.

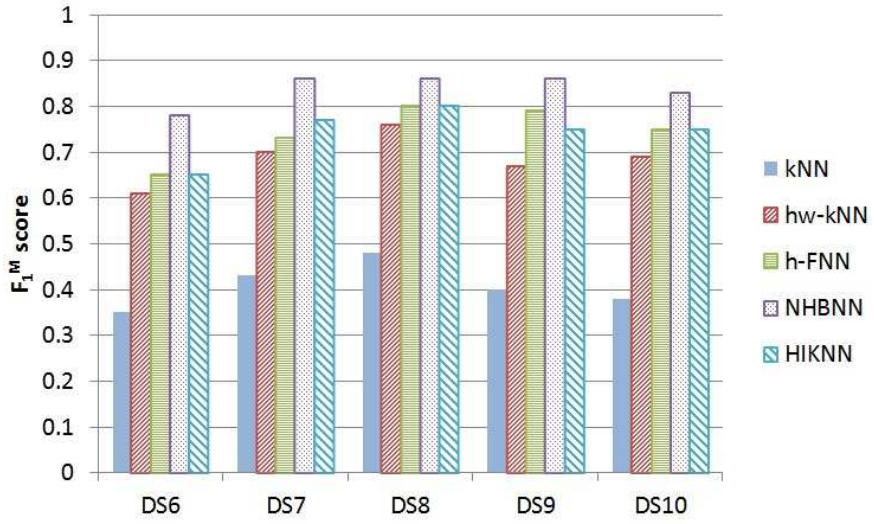
The results are given in Table 18. The baseline kNN is on average only able to achieve 58.09% accuracy, while NHBNN stands best among the hubness-aware methods with an impressive average accuracy of 86.18%. Not only NHBNN, but all hubness-aware approaches clearly and convincingly outperform kNN in this experimental setup. The weighted approach (hw-kNN) was the least successful among the hubness-aware approaches. Even though all differences seem very obvious, we also report the macro-averaged F_1 measure for each algorithm on each data set in Figure 35. The difference in F_1 score is even more convincing that in accuracy, so we can safely say that hubness-aware voting helps in successfully dealing with class distribution overlap.

Data points located in overlap regions can vary in difficulty that they pose for the kNN method. This depends mostly on how far they lie from their own distribution mean and how close they are to other clusters. We can take advantage of the already discussed point characterization scheme [Napierala and Stefanowski, 2012] and use it to analyze and better understand the nature of the observed improvements. Figure 36 shows the precision that each of the algorithms achieves on safe points, borderline examples, rare points and outliers, separately. The charts are given for DS_0 and DS_1 but are very similar for other data sets, as well.

Not surprisingly, kNN is completely incapable of dealing with rare points and outliers - and performs badly even on borderline points. We should point out that the reason why the precision isn't 100% on safe points is that $k = 5$ is used (as described in [Napierala and Stefanowski, 2012]) to determine point types, but here we are observing 10-NN classification. Hubness-aware methods achieve higher precision on all point types, safe points included. The difference in performance is most pronounced for more difficult point types and this is where most of the improvement stems from. Also, we are able to see why NHBNN scores better than the other hubness-aware algorithms on this data. It performs better when classifying



(a)

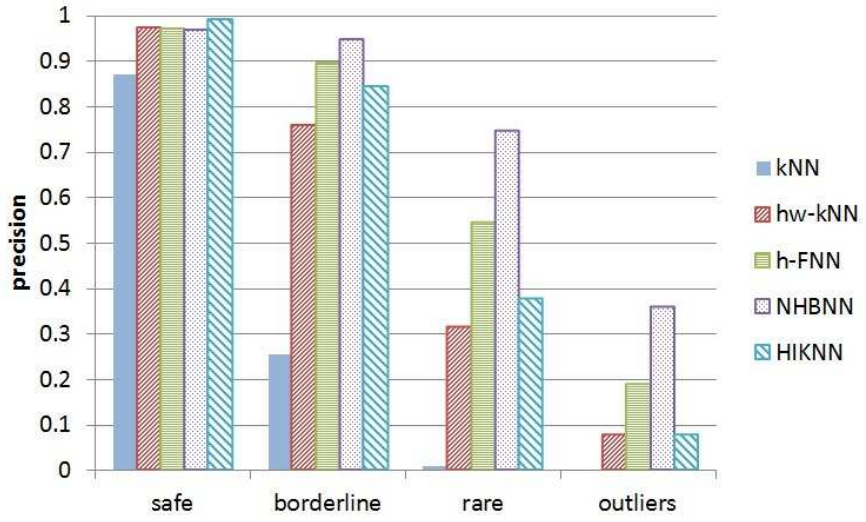


(b)

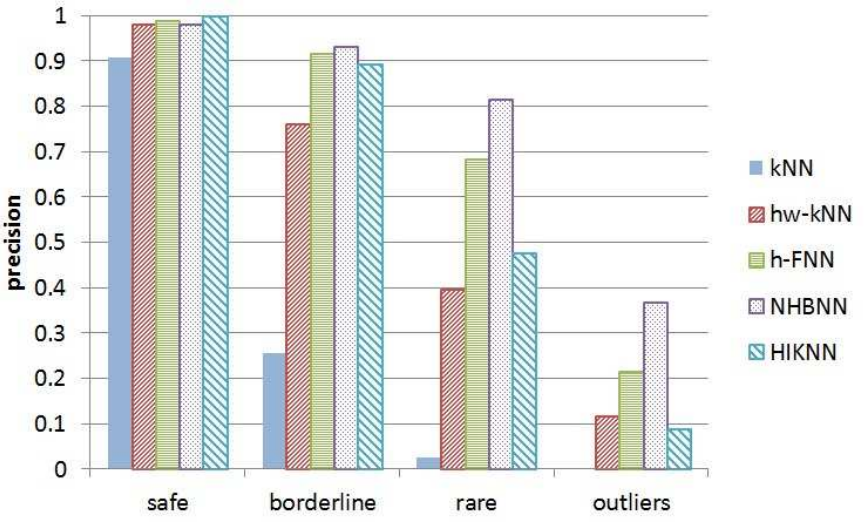
Figure 35: Macro-averaged F_1 score on overlapping Gaussian mixture data.

all the difficult point types in the overlap regions. On average, NHBNN manages to correctly assign the labels to more than 90% of borderline points, about $75-80$% of rare points and 40% of outliers. We have verified that this is indeed true for all 10 examined Gaussian mixtures. It is interesting to note that the same trend is not detected in iNet data that was previously discussed. Bad hubness in iNet data is not exclusively due to class overlap, so it is a different story altogether. In any case, these results show that the Bayesian way of building the occurrence model remains promising. This was not anticipated, as it is in contrast with what some earlier experiments had implied [Tomašev and Mladenić, 2011c][Tomašev and Mladenić, 2012][N. and D., 2012].

As a final remark, we report the performance of some other well-known algorithms on class overlap data. Table 19 contains a summary of results given for the fuzzy k-nearest-neighbor (FNN) [Keller et al., 1985], probabilistic nearest neighbor (PNN) [Holmes and Adams, 2002], neighbor-weighted kNN (NWKNN) [Tan, 2005a], adaptive kNN (AKNN) [Wang et al., 2007], J48 (a WEKA [Witten and Frank, 2005a] implementation of the Quinlan's C4.5 algorithm [Quinlan, 1993]), random forest classifier [Breiman, 2001] and Naive Bayes [Mitchell, 1997]. The first thing to notice is that FNN scores much worse than its hubness-aware counterpart h-FNN. This shows that there is a large difference in semantics



(a) DS_0



(b) DS_1

Figure 36: Classification precision on certain types of points: safe points, borderline points, rare examples and outliers. We see that the baseline kNN is completely unable to deal with rare points and outliers and this is precisely where the improvements in hubness-aware approaches stem from.

between the fuzziness derived from direct and reverse k-nearest neighbor sets. The best performance among all the tested hubness-unaware kNN methods is attained by the adaptive kNN (AKNN), which is not surprising since it was designed specifically for handling class-overlap data [Wang et al., 2007]. Its performance is still, however, somewhat inferior to that of NHBNN, at least in this experimental setup.

Decision trees, on the other hand, seem to have been heavily affected by the induced class overlap, as using either C4.5 or random forest classifiers results in low overall accuracy rates. Naive Bayes was the best among the tested approaches on these Gaussian Mixtures.

What this comparison reveals is that the currently available hubness-aware k-nearest neighbor approaches rank rather well when compared to the other kNN-based methods, but there is also some room for improvement.

Table 19: Classification accuracy of a selection of algorithms on Gaussian mixture data. The results are given for fuzzy k-nearest-neighbor (FNN), probabilistic nearest neighbor (PNN), neighbor-weighted kNN (NWKNN), adaptive kNN (AKNN), J48 implementation of the Quinlan's C4.5 algorithm, random forest classifier and Naive Bayes, respectivelly. A neighborhood size of $k = 10$ was used in the nearest-neighbor-based approaches, where applicable. Results better than than the ones of NHBNN in Table 18 are given in bold.

Data set	FNN	PNN	NWKNN	AKNN	J48	R. Forest	Naive Bayes
DS_1	36.6 ± 3.0	39.8 ± 3.5	46.5 ± 3.3	79.5 ± 2.6	42.4 ± 4.3	59.5 ± 3.7	**95.6 ± 1.3**
DS_2	40.5 ± 2.9	35.9 ± 3.2	54.0 ± 2.6	82.7 ± 2.1	47.3 ± 3.9	65.4 ± 3.9	**97.1 ± 0.9**
DS_3	61.5 ± 2.7	71.3 ± 2.4	67.4 ± 2.5	88.7 ± 1.7	48.9 ± 3.9	69.2 ± 3.1	**98.6 ± 0.2**
DS_4	46.6 ± 2.4	43.4 ± 4.6	56.5 ± 2.9	84.7 ± 1.7	44.0 ± 3.7	59.7 ± 3.7	**98.4 ± 0.2**
DS_5	52.3 ± 2.9	54.1 ± 4.3	61.8 ± 2.6	83.2 ± 2.1	45.6 ± 2.9	64.1 ± 3.2	**98.3 ± 0.1**
DS_6	51.5 ± 3.0	51.5 ± 3.5	62.2 ± 3.0	78.6 ± 3.2	52.1 ± 4.2	67.2 ± 3.1	**97.3 ± 1.1**
DS_7	59.0 ± 2.7	60.0 ± 4.0	66.9 ± 2.6	90.1 ± 1.5	51.0 ± 3.7	70.7 ± 2.6	**98.3 ± 0.7**
DS_8	67.8 ± 2.6	72.6 ± 2.6	71.5 ± 2.5	85.2 ± 1.9	50.2 ± 3.7	67.1 ± 3.1	**98.7 ± 0.4**
DS_9	51.9 ± 2.7	48.9 ± 4.6	61.7 ± 2.6	84.5 ± 2.0	43.9 ± 3.6	64.5 ± 3.7	**98.3 ± 0.7**
DS_{10}	51.0 ± 2.7	47.8 ± 4.2	62.1 ± 2.5	79.6 ± 2.0	46.2 ± 3.8	64.0 ± 3.1	**97.9 ± 0.8**
AVG	51.87	52.53	61.06	83.68	47.16	65.14	**97.85**

2.3 Hubness-aware Metric Learning

This section presents a paper titled *Hubness-aware Shared Neighbor Distances for High-dimensional k-Nearest Neighbor Classification* by Nenad Tomašev and Dunja Mladenić. The brief version of the paper was first published at the Hybrid Artificial Intelligence Systems conference in Salamanca, Spain, in 2012 [N. and D., 2012]. This initial manuscript has been greatly extended by introducing an in-depth analysis of the metric components, as well as including new benchmarks and data types. The final manuscript was published in the Knowledge and Information Systems journal in 2013 [Tomašev and Mladenić, 2013].

The notion of working with the k-nearest neighbors is based on the ability to produce a ranking for each query point, that reflects some explicit or implicit underlying similarity measure defined for all pairs of points in the feature space. The effectiveness of k-nearest neighbor methods depends on the effectiveness of the underlying metric. For some types and distributions of data it is easy to find an appropriate metric, even among the standard ones like the Euclidean, cosine, Jaccard, Manhattan or fractional [Han, 2005]. However, these standard metrics become highly inappropriate for many forms of very high-dimensional data due to the distance concentration phenomenon [François et al., 2007]. Therefore, learning the appropriate metric directly from the data instead of defining it a priori is sometimes justified, regardless of the increase in computational complexity.

Shared neighbor distances are a class of secondary pseudo-metrics derived from the k-nearest neighbor topology of the data and are often used in high-dimensional data clustering applications. In our paper we have proposed a novel secondary similarity measure, $simhub_s$, derived from the original shared neighbor similarity score. The proposed approach was evaluated on a wide range of high-dimensional datasets and was shown to significantly outperform the baseline in the context of k-nearest neighbor classification. The best results were thus obtained by using a combination of hubness-aware metric learning and hubness-aware k-nearest neighbor classification.

Hubness-aware Shared Neighbor Distances for High-dimensional k-Nearest Neighbor Classification

Nenad Tomašev[1] and Dunja Mladenić[1]

Artificial Intelligence Laboratory, Jožef Stefan Institute and
Jožef Stefan International Postgraduate School
1000 Ljubljana, Slovenia
nenad.tomasev@ijs.si, dunja.mladenic@ijs.si

Abstract. Learning from high-dimensional data is usually quite challenging, as captured by the well known phrase *curse of dimensionality*. Data analysis often involves measuring the similarity between different examples. This sometimes becomes a problem, as many widely used metrics tend to concentrate in high-dimensional feature spaces. The reduced contrast makes it more difficult to distinguish between close and distant points, which renders many traditional distance-based learning methods ineffective. Secondary distances based on shared neighbor similarities have recently been proposed as one possible solution to this problem. However, these initial metrics failed to take *hubness* into account. Hubness is a recently described aspect of the dimensionality curse and it affects all sorts of k-nearest neighbor learning methods in severely negative ways. This paper is the first to discuss the impact of hubs on forming the shared neighbor similarity scores. We propose a novel, hubness-aware secondary similarity measure $simhub_s$ and an extensive experimental evaluation shows it to be much more appropriate for high-dimensional data classification than the standard $simcos_s$ measure. The proposed similarity changes the underlying kNN graph in such a way that it reduces the overall frequency of label mismatches in k-neighbor sets and increases the purity of occurrence profiles, which improves classifier performance. It is a hybrid measure, which takes into account both the supervised and the unsupervised hubness information. The analysis shows that both components are useful in their own ways and that the measure is therefore properly defined. This new similarity does not increase the overall computational cost and the improvement is essentially 'free'.

Keywords: hubs, metric learning, curse of dimensionality, k-nearest neighbor, classification, shared neighbor distances

[0] *Received*: Jul 05, 2012
Revised: Sep 26, 2012
Accepted: Nov 11, 2012

This paper was published in the Knowledge and Information Systems journal in 2013 as an extended version of the paper *Hubness-aware Shared Neighbor Distances for High-dimensional k-Nearest Neighbor Classification*, which was presented at the Data Mining: Data Preparation and Analysis special session of the Hybrid Artificial Intelligence conference (HAIS 2012) [Tomašev and Mladenić, 2012a]

1 Introduction

Machine learning in many dimensions is often rendered very difficult by an interplay of several detrimental factors, commonly referred to as the *curse of dimensionality*. In high-dimensional spaces, all data is sparse, as the requirements for proper density estimates rise exponentially with the number of features. Empty space predominates [Scott and Thompson, 1983] and data lies approximately on the surface of hyper-spheres around cluster means, i.e. in distribution tails. Relative contrast between distances on sample data is known to decrease with increasing dimensionality, as the distances concentrate [Aggarwal et al., 2001][François et al., 2007]. The expectation of the distance value increases, but the variance remains constant. It is therefore much more difficult to distinguish between close and distant points. This has a profound impact on nearest neighbor methods, where inference is done based on the k examples most similar (relevant) to the point of interest. The very concept of a nearest neighbor was said to be much less meaningful in high-dimensional data [Durrant and Kabán, 2009].

Difficulty in distinguishing between relevant and irrelevant points is, however, not the only aspect of the dimensionality curse which burdens k-nearest neighbor based inference. The recently described phenomenon of *hubness* is also considered to be highly detrimental. The term was coined after *hubs*, very frequent neighbor points which dominate among all the occurrences in the k-neighbor sets of inherently high-dimensional data [Radovanović et al., 2009][Radovanović et al., 2010b]. Most other points either never appear as neighbors or do so very rarely. They are referred to as *anti-hubs*.

The skewness of the k-occurrence distribution has a geometric interpretation and does not reflect the underlying semantics of the data. This was first noticed in music retrieval applications [Aucouturier and Pachet, 2004][Aucouturier, 2006] and is still an unresolved issue [Flexer et al., 2010][Schedl M., 2012][Flexer A., 2012] [Schnitzer et al., 2011][Gasser M., 2010]. Some songs were very frequently retrieved by the recommendation systems, but were in fact irrelevant for the considered queries. Their occurrences were simply an artifact of the employed similarity measures, when applied to high-dimensional audio data.

There is no easy way out, as demonstrated in [Radovanović et al., 2010a], since dimensionality reduction techniques fail to eliminate the neighbor occurrence distribution skewness for any reasonable dimensionality of the projection space. The skewness decreases only when the data is projected onto spaces below the intrinsic dimensionality of the data, where some potentially relevant information is irretrievably lost. It is therefore necessary to work under the assumption of hubness when using nearest neighbor methods for analyzing high-dimensional data.

Different metric spaces exhibit different degrees of hubness, so choosing a proper distance measure becomes a non-trivial task. The apparent inadequacy of many common metrics (Manhattan, Euclidean, etc.) in high-dimensional data has motivated some researchers to start using higher-order secondary distances based on shared nearest neighbor similarity scores. This approach has frequently been used in clustering applications [Jarvis and Patrick, 1973][Ertz et al., 2001][Yin et al., 2005][Moëllic et al., 2008]
[Patidar et al., 2012][Zheng and Huang, 2012]. The basic idea is that the similarity between two points can be measured by the number of k-nearest neighbors that they have

in common. This is somewhat similar to collaborative filtering, where the purchase set intersections are used to determine similarity between different customers.

Turning a similarity score into a distance is a trivial task. We will address the details in Section 2.1. Shared neighbor distances are considered by some as a potential cure for the curse of dimensionality [Houle et al., 2010].

Even though the shared neighbor distances have mostly been considered in the context of clustering, we will focus on the supervised learning case and show their usefulness in k-nearest neighbor (kNN) classification.

Hubness exhibits a dual influence on shared neighbor distances. As the secondary metric is introduced, the overall hubness in the data must also change. We will show that even though the skewness in the k-occurrence distribution is somewhat reduced, some non-negligible hubness still remains and using the hubness-aware classification methods yields definite improvements. More importantly, the hubness in the original metric space has a profound impact on how the shared neighbor similarity scores are formed in the first place. Hubs are very frequent neighbors so they become very frequently shared neighbors as well. As we have already mentioned, hubs are usually points where the semantics of the similarity measure is most severely compromised, so relying on them when defining a secondary distance is not a very wise choice. This is why we have proposed a new *hubness-aware* method for calculating shared neighbor similarities/distances [Tomašev and Mladenić, 2012a].

The paper is structured as follows. In Section 2 we outline the basic concepts in defining the shared neighbor distances and discuss some recent findings in learning under the assumption of hubness. We proceed by considering how the two might be successfully combined and propose a new way to define shared neighbor similarities in Section 3. In Section 4 we test our hypothesis on several high-dimensional synthetic and image datasets and examine the findings.

2 Related work

2.1 Shared neighbor distances

Regardless of the skepticism expressed in [Durrant and Kabán, 2009], nearest neighbor queries have been shown to be meaningful in high-dimensional data under some natural assumptions [Bennett et al., 1999], at least when it comes to distinguishing between different clusters of data points. If the clusters are pairwise stable, i.e. inter-cluster distances dominate intra-cluster distances, the neighbors will tend to belong to the same cluster as the original point. An obvious issue with this line of reasoning is that cluster assumption violation is present to different degrees in real world data, so that sometimes the categories do not correspond well to the aforementioned clusters. Nevertheless, this observation motivated the researchers to consider using secondary distances based on the ranking induced by the original similarity measure [Houle et al., 2010]. A common approach is to count the number of shared nearest neighbors (SNN) between pairs of points for a given, fixed neighborhood size.

Let $D = (x_1, y_1), (x_2, y_2), ..(x_n, y_n)$ be the data set, where each $x_i \in \mathbb{R}^d$. The x_i are feature vectors which reside in some high-dimensional Euclidean space, and

$y_i \in c_1, c_2, ..c_C$ are the labels. Denote by $D_k(x_i)$ the k-neighborhood of x_i. A shared neighbor similarity between two points is then usually defined as:

$$simcos_s(x_i, x_j) = \frac{|D_s(x_i) \cap D_s(x_j)|}{s} \tag{1}$$

where we have used s to denote the neighborhood size, since we will use these similarity measures to perform k-nearest neighbor classification and the neighborhood sizes in these two cases will be different. The $simcos_s$ similarity can easily be transformed into a distance measure in one of the following ways [Houle et al., 2010]:

$$\begin{aligned} dinv_s(x_i, x_j) &= 1 - simcos_s(x_i, x_j) \\ dacos_s(x_i, x_j) &= \arccos(simcos_s(x_i, x_j)) \\ dln_s(x_i, x_j) &= -\ln(simcos_s(x_i, x_j)) \end{aligned} \tag{2}$$

All three of the above given distance measures produce the same ranking, so they are essentially equivalent when being used for k-nearest neighbor inference. We based all our subsequent experiments on $dinv_s(x_i, x_j)$.

In shared neighbor distances, all neighbors are treated as being equally relevant. We argue that this view is inherently flawed and that its deficiencies become more pronounced when the dimensionality of the data is increased. Admittedly, there have been some previous experiments on including weights into the SNN framework for clustering [Ayad and Kamel, 2003], but these weights were associated with the positions in the neighbor list, not with neighbor objects themselves. In Section 3 we will discuss the role of hubness in SNN measures.

2.2 Hubs: very frequent nearest neighbors

High dimensionality gives rise to *hubs*, influential objects which frequently occur as neighbors to other points. Most instances, on the other hand, are very rarely included in k-neighbor sets, thereby having little or no influence on subsequent classification. What this change in the k-occurrence distribution entails is that potential errors, if present in the hub points, can easily propagate and compromise many k-neighbor sets. Furthermore, hubness is a geometric property of inherently high-dimensional data, as the points closer to the centers of hyper-spheres where most of the data lies tend to become very similar to many data points and are hence often included as neighbors [Radovanović et al., 2010b]. This means that hubness of a particular point has little to do with its semantics. Hubs are often not only neighbors to objects of their own category, but also neighbors to many points from other categories as well. In such cases, they exhibit a highly detrimental influence and this is why hubness of the data usually hampers k-nearest neighbor classification.

Hubness has only recently come into focus, but some hubness-aware algorithms have already been proposed for clustering [Tomašev et al., 2011d], instance selection [Buza et al., 2011], outlier and anomaly detection [Radovanović et al., 2010a] [Tomašev and Mladenić, 2011] and classification [Radovanović et al., 2009] [Tomašev et al., 2011b][Tomašev et al., 2011c][Tomašev and Mladenić, 2011b]

[Tomašev and Mladenić, 2012b][Tomašev and Mladenić, 2011a], which we will discuss below.

Let us introduce some notation. Denote by $R_k(x_i)$ the reverse neighbor set of x_i, so the number of k-occurrences is then $N_k(x_i) = |R_k(x_i)|$. This total number of neighbor occurrences includes both the *good* occurrences, where the labels of points and their neighbors match and the *bad* occurrences where there is a mismatch between them. Formally, $N_k(x_i) = GN_k(x_i) + BN_k(x_i)$, the former being referred to as the good hubness and the latter as the bad hubness of x_i. The bad hubness itself can be viewed as a composite quantity, comprising all the class-specific k-occurrences where label mismatch occurs. Let $N_{k,c}(x_i) = |x \in R_k(x_i) : y = c|$ denote such class-specific hubness. The total occurrence frequency is then simply $N_k(x_i) = \sum_{c \in C} N_{k,c}(x_i)$. Calculating all the $N_k(x_i)$ equals building an occurrence model, which can be used to somehow estimate all the implied posterior class probabilities in the point of interest $p(y = c|x_i \in D_k(x))$. This observation served as the basis for several recent hubness-aware approaches [Tomašev et al., 2011b][Tomašev et al., 2011c][Tomašev and Mladenić, 2011b]
[Tomašev and Mladenić, 2012b].

2.3 Hubness-aware classification methods

The basic k-nearest neighbor method [Fix and Hodges, 1951] is very simple, but has nevertheless been proven to exhibit certain beneficial asymptotic properties [C.J.Stone, 1977][L. Devroye and Lugosi, 1994][T.M.Cover and P.E.Hart, 1967] [Devroye, 1981]. A label in the point of interest is decided upon by a majority vote of its nearest neighbors. Many extensions of the basic algorithm have been developed over the years, improving the original approach in various ways. [Keller et al., 1985] [Jensen and Cornelis, 2008][Song et al., 2007][Hodge and Austin, 2005] [Ougiaroglou et al., 2007][Zhang et al., 2006][Triguero et al., 2011] [Ambert and Cohen, 2012] The k-nearest neighbor classification is still widely used in many practical applications, with a recent focus on time series analysis [Xing et al., 2009]
[Chaovalitwongse et al., 2007] and imbalanced data classification [Holte et al., 1989] [van den Bosch et al., 1997][Li and Zhang, 2011][Tan, 2005] [Wang et al., 2010][Van Hulse and Khoshgoftaar, 2009].

Hubness in high-dimensional data, nevertheless, affects kNN in some severely negative ways [Radovanović et al., 2009][Radovanović et al., 2010a][Radovanović et al., 2010b]. This is why several hubness-aware classification algorithms have recently been proposed. An effective vote weighting scheme was first introduced in [Radovanović et al., 2009], assigning to each neighbor a weight inversely correlated with its bad hubness. More specifically, $w_k(x_i) = e^{-h_b(x_i,k)}$, where $h_b(x_i, k) = (BN_k(x_i) - \mu_{BN_k})/\sigma_{BN_k}$ is the standardized bad hubness. We will refer to this approach as hubness-weighted k-nearest neighbor (hw-kNN).

Fuzzy measures based on $N_{k,c}(x_i)$ have been introduced in [Tomašev et al., 2011b], where the fuzzy k-nearest neighbor voting framework was extended to include hubness information (h-FNN). This was further refined in [Tomašev and Mladenić, 2011b] by

considering the self-information of each individual occurrence. Anti-hubs were therefore treated as more informative. Intuitively, such neighbor points are more local to the point of interest, as they are not frequent neighbors. The algorithm was named hubness information k-nearest neighbor (HIKNN).

Along with the fuzzy approaches, a naive Bayesian model was desribed in [Tomašev et al., 2011c], where the algorithm naive hubness-Bayesian kNN (NHBNN) was proposed for probabilistic k-nearest neighbor classification in high dimensional data.

We will see in Section 4.3 that these hubness-aware algorithms are in fact well suited for dealing with the secondary SNN distances.

3 Hubness-aware shared-neighbor distances

Since hubness affects the distribution of neighbors, it must also affect the distribution of neighbors shared between different points. Each x_i is shared between $N_s(x_i)$ data points and participates in $\binom{N_s(x_i)}{2}$ similarity scores. Hub points, by the virtue of being very frequent neighbors, are expected to arise quite frequently as shared neighbors in pairwise object comparisons. What this means, however, is that sharing a hub s-neighbor is quite common and not very informative. This is consistent with observations in [Tomašev and Mladenić, 2011b]. Rarely shared neighbors (anti-hubs), on the other hand, carry information more local to the points of interest and should be given preference when calculating similarities. Figure 1 outlines this observation.

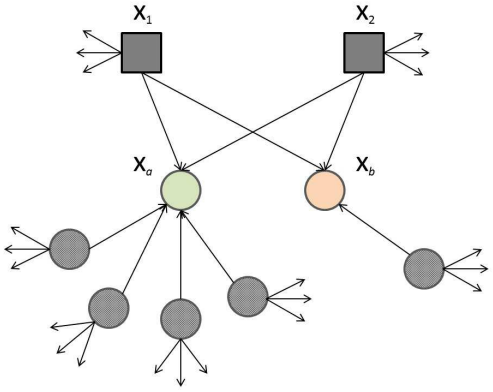


Fig. 1. An illustrative example. x_1 and x_2 share two neighbors, $D_s(x_1) \cap D_s(x_2) = \{x_a, x_b\}$. The two shared neighbors are not indicative of the same level of similarity, as x_b is a neighbor only to x_1, x_2 and one other point, while x_a is a more frequently shared neighbor.

Each neighbor point can, depending on its hubness, contribute to many pairwise similarities. Some of these similarities will be between the elements from the same

class and some between the elements from different classes. Therefore, we can expect some neighbors to contribute more to the intra-class similarities and some more to the inter-class similarities, depending on the class distribution in their occurrence profiles. Clearly, hubs which occur almost exclusively as neighbors to points from a single category ought to be preferred to those which occur inconsistently among various categories in the data. This is illustrated in Figure 2. The purity of the reverse neighbor sets can clearly be exploited for improving class separation.

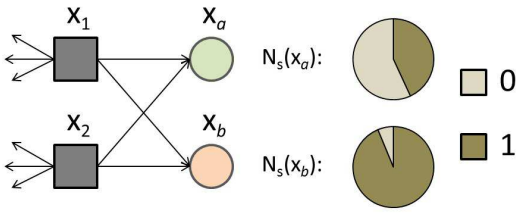


Fig. 2. A binary example where the shared neighbors have significantly different occurrence profiles. x_a is equally often present as a neighbor to objects from both categories, while x_b is almost exclusively in s-neighbor sets of the second class. By favoring x_b over x_a in the similarity score, the average intra-class similarity is expected to increase and the inter-class similarity decreases.

In order to refine the basic shared neighbor similarity, we will give preference to less frequent and good neighbor points and reduce the influence of bad hubs. We propose a new SNN similarity measure:

$$simhub_s(x_i, x_j) = \frac{\sum_{x \in D_s(x_i) \cup D_s(x_j)} I_n(x) \cdot (\max H_s - H(R_s(x)))}{s \cdot \max H_s \cdot \max I_n} \tag{3}$$

$$I_n(x) = \log \frac{n}{(N_s(x))}; \qquad \max I_n = \log n \tag{4}$$

$$H(R_s(x)) = H(Y|x \in D_s) = -\sum_{c \in C} \frac{N_{s,c}(x)}{N_s(x)} \log \frac{N_{s,c}(x)}{N_s(x)}; \qquad \max H_s = \log c \tag{5}$$

Though it may seem slightly complicated, $simhub_s$ is in fact very simple and intuitive. The denominator merely serves the purpose of normalization to the $[0, 1]$ range. Each shared neighbor is assigned a weight which is a product of two quantities. Occurrence informativeness ($I_n(x)$) increases the voting weights of rare neighbors. The reverse neighbor set entropy ($H(R_s(x))$) measures the non-homogeneity (inconsistency) in occurrences. When subtracted from the maximum entropy ($\max H_s$), it represents the *information gain* from observing the occurrence of x, under the uniform label assumption. The labels are, of course, not uniformly distributed, but it is convenient to have $(\max H_s - H(R_s(x))) \geq 0$. For the purposes of calculating $I_n(x)$ and $H(R_s(x))$, x is treated as its own 0th nearest neighbor, in order to avoid zero divisions for points which haven't previously occurred as neighbors on the training data. In other words,

$N_s(x) := N_s(x) + 1$, $N_{s,y}(x) := N_{s,y}(x) + 1$, where y is the label of x. The $simhub_s$ similarity can be turned into a distance measure in the same way as the $simcos_s$, as previously demonstrated in Eq. 2.

What is great about this new way of defining similarity is that the extra computational cost is negligible, since all the s-neighbor sets need to be calculated anyway. One only has to count the occurrences, which is done in $O(s \cdot n)$ time. Calculating all the $D_s(x)$ neighbor sets accurately takes $\Theta(d \cdot n^2)$ time in high dimensional data, where d is the number of features (since usually $d > s$), which is the time required to compute the distance matrix in the original metric. An approximate algorithm exists, however, which does the same in $\Theta(d \cdot n^{1+t})$, $t \in [0, 1]$ [Chen et al., 2009]. It is a divide and conquer method based on recursive Lanczos bisection. In our initial experiments, very good estimates are obtained even for $t = 0$ (so, in linear time!), provided that the stop criterion for subset division is set high enough, since the accurate s-neighborhoods are computed in the leaves of the split.

It is possible to observe the $simhub_s$ similarity in terms of its constituents, as it is jointly based on two different quantities - neighbor informativeness and neighbor occurrence purity. These factors can be considered separately, as given in Equation 6 and Equation 7.

$$simhub_s^{\mathrm{IN}} = \frac{\sum_{x \in D_s(x_i) \cup D_s(x_j)} I_n(x)}{s \cdot \max I_n} \tag{6}$$

$$simhub_s^{\mathrm{PUR}} = \frac{\sum_{x \in D_s(x_i) \cup D_s(x_j)} (\max H_s - H(R_s(x)))}{s \cdot \max H_s} \tag{7}$$

In some of the experiments we will examine the influence of each of the two constituent measures on the final $simhub_s$ similarity score and the overall classification performance.

4 Experiments and Discussion

4.1 Overview of the data

The analysis was performed on both synthetic and real-world data. In synthetic data, we were interested only in such datasets that would pose significant difficulties for kNN-based methods, as this fits well with the analysis of hubness and the rest of the experimental setup. To that purpose, we have generated 10 difficult 100-dimensional Gaussian mixtures with a significant class overlap, each comprising 10 different categories. The overlap was achieved by randomly placing each distribution center for each feature within a certain range of another already generated center, constraining the distance between them to a certain multiple of their standard deviations. This is well reflected in Table 1, where we can see that these data sets (DS_1-DS_{10}) exhibit substantial bad hubness.

Most of the analysis was done on high-dimensional image representations, but some brief comparisons were also performed on relatively low-dimensional data, in order to gain further insights into the potential applicability of the similarity measures (Table 2, Section 4.9).

Ten image datasets were selected for the basic high-dimensional experiments, as the image data is known to exhibit significant hubness [Tomašev et al., 2011a]. They represent different subsets of the public ImageNet repository (http://www.image-net.org/). We have selected the same subsets that were used in classification benchmarks in previous papers on hubness-aware classification [Tomašev et al., 2011b][Tomašev et al., 2011a]
[Tomašev and Mladenić, 2011b][Tomašev and Mladenić, 2012a], to simplify the comparisons.

Table 1. The summary of high-hubness datasets. Each dataset is described both by a set of basic properties (size, number of features, number of classes) and some hubness-related quantities for two different neighborhood sizes, namely: the skewness of the k-occurrence distribution (S_{N_k}), the percentage of *bad* k-occurrences (BN_k), the degree of the largest hub-point ($\max N_k$). Also, the relative imbalance of the label distribution is given, as well as the size of the majority class (expressed as a percentage of the total)

(a) ImageNet data, L_1 distance

Data set	size	d	C	S_{N_5}	BN_5	$\max N_5$	$S_{N_{50}}$	BN_{50}	$\max N_{50}$	RImb	$p(c_M)$
iNet3	2731	416	3	8.38	21.0%	213	3.10	25.0%	665	0.40	50.2%
iNet4	6054	416	4	7.69	40.3%	204	3.56	46.2%	805	0.14	35.1%
iNet5	6555	416	5	14.72	44.6%	469	6.10	51.1%	1420	0.20	32.4%
iNet6	6010	416	6	8.42	43.4%	275	3.60	51.0%	836	0.26	30.9%
iNet7	10544	416	7	7.65	46.2%	268	4.21	54.3%	1149	0.09	19.2%
iNet3Imb	1681	416	3	3.48	17.2%	75	1.45	21.2%	271	0.72	81.5%
iNet4Imb	3927	416	4	7.39	38.2%	191	3.47	43.2%	750	0.39	54.1%
iNet5Imb	3619	416	5	9.35	41.4%	258	4.61	47.4%	995	0.48	58.7%
iNet6Imb	3442	416	6	4.96	41.3%	122	2.64	48.0%	534	0.46	54%
iNet7Imb	2671	416	7	6.44	42.8%	158	2.72	50.4%	551	0.46	52.1%
AVG	4723.4	416	7.5	5	37.64%	223.3	3.55	43.8%	797.6	0.36	46.8%

(b) Gaussian mixture data, L_2 distance

Data set	size	d	C	$S_{N_{10}}$	BN_{10}	$\max N_{10}$	$S_{N_{50}}$	BN_{50}	$\max N_{50}$	RImb	$p(c_M)$
DS_1	1244	100	10	6.68	53.5%	291	3.87	58.8%	802	0.21	20.2%
DS_2	1660	100	10	4.47	49.2%	234	3.42	55.4%	705	0.19	16.7%
DS_3	1753	100	10	5.50	42.0%	253	3.19	50.9%	783	0.16	16.8%
DS_4	1820	100	10	3.45	51.0%	174	2.63	59.5%	560	0.13	15.6%
DS_5	1774	100	10	4.39	46.3%	177	3.15	55.0%	565	0.13	16.6%
DS_6	1282	100	10	3.97	45.6%	149	2.90	55.1%	482	0.21	20.7%
DS_7	1662	100	10	4.64	41.5%	209	3.64	50.3%	738	0.16	16.7%
DS_8	1887	100	10	4.19	39.9%	210	3.14	49.1%	622	0.14	15.3%
DS_9	1661	100	10	5.02	47.5%	259	3.11	56.0%	748	0.10	14.7%
DS_{10}	1594	100	10	4.82	46.9%	217	3.24	56.2	655	0.14	17.7%
AVG	1633.7	100	10	4.71	46.34%	217.3	3.23	54.63%	666.0	0.16	17.1%

The images in these ten datasets (iNet3-iNet7, iNet3Imb-iNet7Imb) were represented as 400-dimensional quantized SIFT feature vectors [Lowe, 2004][Zhang and Zhang, 2008] extended by 16-bin color histograms. SIFT features are commonly used in object recognition systems, as they exhibit invariance to scale, rotation and translation. Each part of the representation was normalized separately. This particular image representation may not be the best choice for the given datasets [Tomašev et al., 2011a], but is nevertheless a natural choice and quite challenging for kNN classification, which makes it a good benchmark. It can be seen in Table 1 that these image datasets exhibit substantial bad hubness.

As implied by the names, there is a correspondence between the first (iNet3..iNet7) and the second five datasets (iNet3Imb..iNet7Imb). The latter have been obtained from the former via random sub-sampling of the minority classes in order to increase the imbalance in the data. The difference is easily seen in Table 1 by considering the relative imbalance factor: $\mathrm{RImb} = \sqrt{(\sum_{c \in C} (p(c) - 1/C)^2)/((C-1)/C)}$, which is merely the normalized standard deviation of the class probabilities from the absolutely homogenous mean value of $1/c$.

We will not focus on class imbalance in this paper. We will, however, use one recently proposed framework for imbalanced data analysis [Napierala and Stefanowski, 2012] to outline the most important differences between the analyzed metrics. This will be discussed in Section 4.6.

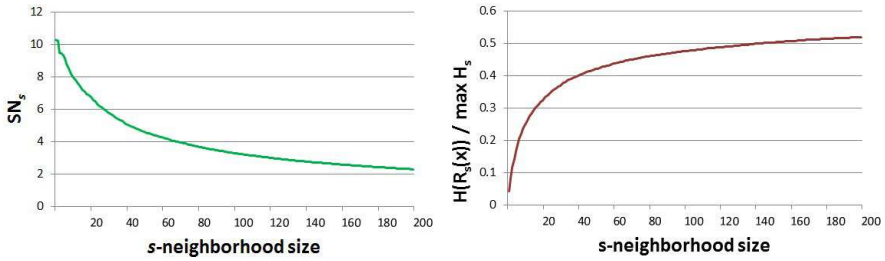
Additionally, three partially faulty quantized Haar feature representations [Lienhart and Maydt, 2002] of iNet3 (iNet3Err:100, iNet3Err:150, iNet3Err:1000) were presented in Section 4.7 as a pathological special case where erroneous hub-points rendered the k-nearest neighbor classification completely ineffective. It will be shown that the secondary shared neighbor similarities are able to reduce the negative consequences of hubness in the data and that the proposed $simhub_s$ measure does so more effectively than $simcos_s$.

Table 1 shows the properties of the data when the primary metrics are used. Since the images have been represented in a form of coupled probability distributions, Manhattan distance (L_1) is used in experiments on image data, as it represents the integral of the absolute difference between the distributions. The Euclidean distance (L_2) was used when analyzing Gaussian data, as it induces hyper-spherical neighborhoods, which are well suited for modeling Gaussian clusters. In our initial experiments, the difference between the two metrics (L_1,L_2) was not so big, but we have nevertheless opted for the more natural choice in both cases.

The reason why Table 1 shows the statistics for several different neighborhood sizes ($k = 5$ and $k = 50$ for the image data and $k = 10$ and $k = 50$ for the synthetic data) is that we will be performing $5 - NN$ classification of the image data and $10 - NN$ classification of the synthetic data, while using the shared neighbor distances based on 50-neighbor sets. The neighborhood size for the image data is chosen for comparison with previous work, while a larger k is beneficial in Gaussian mixtures, as it allows for better estimates in the borderline regions. In Section 4.4 we will show that the difference between the examined metrics actually holds regardless of the particular choice of k.

An increase in neighborhood size somewhat reduces the skewness of the occurrence distribution, since more points become hubs. Bad hubness increases, as well as the non-

homogeneity of reverse neighbors sets. This is illustrated in Figure 3 for iNet5Imb dataset. The increase is not always smooth as in the given figure, but the same general trend exists in all the datasets that we have analyzed in our experiments.



(a) Occurrence skewness (hubness) (b) Reverse neighbor set entropy

Fig. 3. s-occurrence skewness and reverse neighbor set entropy over a range of neighborhood sizes for iNetImb5 dataset.

The degree of major hubs is quite high for $s = 50$ neighborhood size which will be used to calculate the secondary SNN distances. In some of the datasets, the major hub appears in approximately 20% of all neighbor lists. This shows why it might be important to take the hubness into account when deducing the secondary distances for high-dimensional data. Likewise, high reverse neighbor set entropies indicate that good hubs are a rarity when using large neighborhoods - so their influence on similarity should be emphasized, whenever possible.

Even though both $simcos_s$ and $simhub_s$ were designed primarily for high-dimensional data, it is prudent to perform some comparisons on low-to-medium dimensional data as well. We have selected 10 such datasets from the UCI repository (http://archive.ics.uci.edu/ml/). The summary of low dimensional datasets is given in Table 2. We see that the skewness of the occurrence distribution is even negative in some datasets, so there is no hubness to speak of. The comparisons on this data are given in Section 4.9.

4.2 Hubness in the shared neighbor metric space

Switching to secondary distances induces a change in the hubness of the data. As the similarities are recalculated, so are the k-nearest neighbor sets and this affects the structure of the kNN graph. The change can be either beneficial or detrimental to the following classification process. The impact on the kNN classification can already be estimated by observing the change in the total number of bad occurrences on the data. This is summarized in Figure 4, for both the synthetic and the ImageNet data.

As mentioned in Section 2.1, we are using the $dinv_s(x_i, x_j)$ method of converting a similarity into a distance measure, which essentially means that we are subtracting the normalized similarity score from 1 to obtain the normalized distance score. Therefore, the primary distances in Figure 4 are compared to the $dinv_s(x_i, x_j)$ distances based

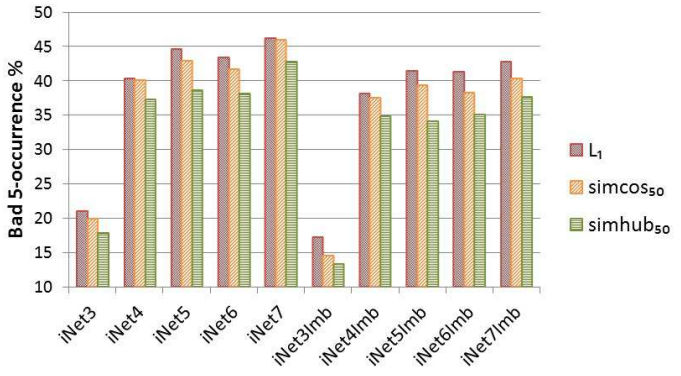
Table 2. The summary of low-to-medium dimensional datasets from the UCI repository. The same properties are shown as in Table 1. This data does not exhibit hubness and is briefly discussed in Section 4.9.

Data set	size	d	C	S_{N_5}	BN_5	$\max N_5$	$S_{N_{50}}$	BN_{50}	$\max N_{50}$	RImb	$p(c_M)$
diabetes	768	8	2	0.34	33.7%	13	0.03	36.0%	112	0.30	0.65
wpbc	198	33	2	-0.09	33.7%	10	-0.80	35.4%	75	0.52	0.76
wdbc	569	30	2	0.09	8.9%	13	-0.86	11.6%	82	0.25	0.63
yeast	1484	8	10	0.40	51.3%	16	0.28	56.4%	132	0.37	0.31
wine	178	13	3	0.04	31.9%	10	-0.99	38.3%	71	0.11	0.40
page-blocks	5473	10	5	0.25	5.2%	14	-0.12	7.8%	108	0.87	0.90
segment	2310	19	7	0.32	6.8%	14	-0.06	23.4%	96	0	0.14
ecoli	336	7	8	0.43	20.4%	15	0.10	29.3%	118	0.41	0.43
mfeat-fourier	2000	76	10	0.87	18.5%	24	0.43	27.5%	145	0	0.1
ozone	2534	72	2	0.76	9.6%	25	0.70	10.2%	157	0.87	0.93
AVG	1585	27.6	5.1	0.34	25.0%	15.4	-0.13	27.6%	109.6	0.37	0.53

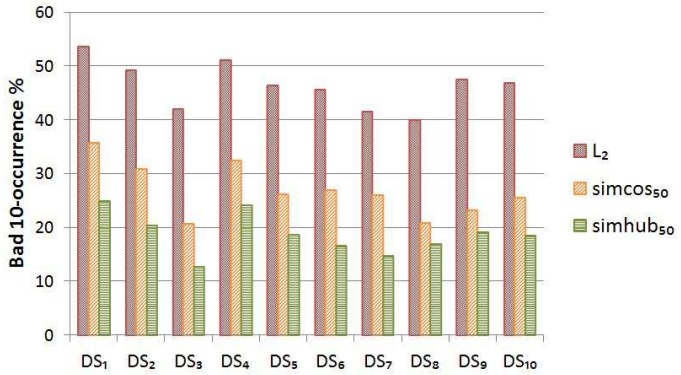
on the $simcos_s$ and $simhub_s$ similarity scores. To simplify the notation in Figures and Tables we will be using the $simcos_s$ and $simhub_s$ interchangeably throughout the following sections to denote either similarity or the implied dissimilarity, depending on the context.

The comparison between the bad occurrence percentages in Figure 4 reveals that both secondary distances achieve a significant reduction in the overall bad hubness of the data. The proposed hubness-aware $simhub_{50}$ similarity score clearly outperforms the standard $simcos_{50}$ similarity, as it produces fewer bad occurrences on every single analyzed dataset. The reduction in both similarity measures is more pronounced in the synthetic data, both for $k = 5$ and $k = 10$ (though only the latter is shown in Figure 4). As mentioned before, two different neighborhood sizes will be used for classifying the image data and the Gaussian mixtures, so the analysis here is also aligned with the following classification experiments in Section 4.3.

Both similarity measures significantly reduce the skewness in k-occurrences on the analyzed data, which is shown in Figure 5. The reduction rates are similar, though the $simcos_{50}$ induces somewhat less hubness in the secondary metric space. This is an important property of both shared neighbor similarity scores. Reducing the hubness in the data partly resolves the implications of the curse of dimensionality in k-nearest neighbor inference. This result reaffirms the previous claims regarding the usefulness of shared neighbor distances [Houle et al., 2010]. Nevertheless, it should be noted that the remaining occurrence skewness is non-negligible. On synthetic data, it amounts to 1.62 and 1.75 on average for $simcos_{50}$ and $simhub_{50}$, respectively. This remaining hubness implies that even though the shared neighbor similarities are doubtlessly helpful in redefining the metric space, the subsequent classification should probably be performed in a hubness-aware way as well. In other words, these similarity scores reduce but do not entirely eliminate the consequences of the dimensionality curse.



(a) ImageNet data

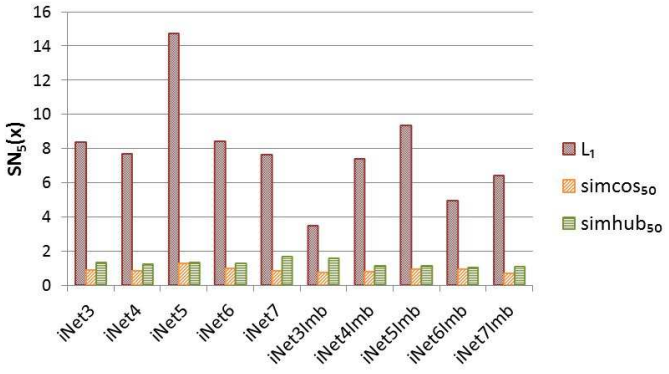


(b) Gaussian mixtures

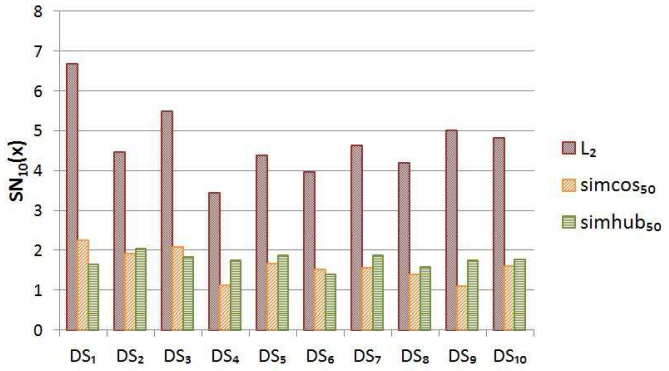
Fig. 4. Bad occurrence percentages in each of the examined metrics. The standard shared-neighbor similarity measure $simcos$ manages to reduce the overall bad hubness in the data, but the proposed hubness-aware $simhub$ similarity reduces the frequency of bad occurrences even more, on all of the analyzed datasets.

Figures 4,5 have shown us how $simcos_{50}$ and $simhub_{50}$ change the overall nature of hubness in the data. However, the average occurrence skewness and the average bad occurrence percentage cannot tell us everything about the change in the kNN graph structure. What needs to be seen is if the good/bad hub-points are invariant to this particular change of metric. Figure 6 gives the pointwise Pearson correlations in the total occurrence frequencies ($N_k(x)$) and bad occurrence frequencies ($BN_k(x)$) between the kNN graphs in the primary and secondary metric spaces, on synthetic data. Similar trends are present in the ImageNet data as well.

The two comparisons in Figure 6 reveal a major difference between the standard $simcos_{50}$ and the proposed $simhub_{50}$ similarity measure. Namely, there exists low-to-moderate positive correlation between hubs and bad hubs in the primary metric space and the metric space induced by $simcos_{50}$. Some primary hubs remain secondary hubs



(a) ImageNet data

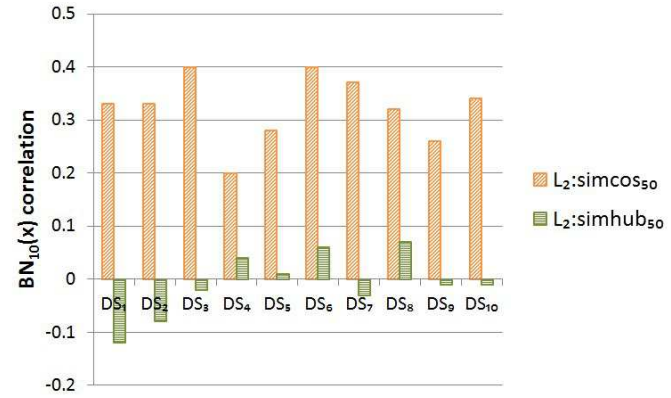


(b) Gaussian mixtures

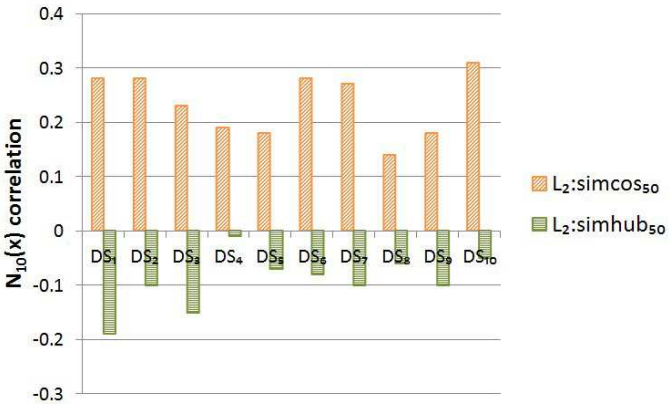
Fig. 5. Overall hubness (k-occurrence skewness) in each of the examined metrics. Both secondary similarity measures significantly reduce the hubness of the data, which should be beneficial for the ensuing classification.

and even more importantly - some primary bad hubs remain secondary bad hubs. On the other hand, $simhub_{50}$ changes the kNN graph structure more drastically, as there is nearly no correlation in bad hubness between the two metric spaces. The correlation in $N_k(x)$ is even slightly negative both in Gaussian mixtures and in ImageNet data. This may be a part of the reason why $simhub_{50}$ succeeds in reducing the overall bad occurrence percentage much more effectively than $simcos_{50}$ - as it is able to reconfigure the kNN graph enough to rectify most of the semantic similarity breaches.

An illustrative example is given in Figure 7, showing how the neighbor occurrence profile of a particular image changes when the secondary similarities are introduced. The number of the reverse 5-nearest neighbors of X_{14} in each category is written above the arrow connecting it to the image. This example is taken from the iNet3 dataset, the simplest among the examined ImageNet subsets. It consists of three different categories: sea moss, fire and industrial plant. Not surprisingly, most misclassifications



(a) Pointwise correlation in bad hubness



(b) Pointwise correlation in hubness

Fig. 6. The Pearson correlation in point hubness ($N_k(x)$) and point bad hubness ($BN_k(x)$) between the primary metric space and the secondary metric spaces induced by $simcos_{50}$ and the proposed $simhub_{50}$ shared neighbor similarity.

occur between the fire and sea moss image categories. Many images of fire were taken in the dark and most sea moss images taken at considerable depth also have a dark background. Some sea mosses are yellow or reddish in color. Also, it is clear from the selected photo in Figure 7 how sometimes the shape of the flames could be confused with leaf-like organic objects.

The example in Figure 7 nicely illustrates both properties of the secondary similarity measures that were discussed in this Section. Due to a reduction in the overall hubness of the iNet3 data, a hub point X_{14} is reduced to being slightly above average in number of occurrences under $simcos_{50}$ and below average under the proposed $simhub_{50}$ similarity score. Both secondary measures significantly reduce its number of

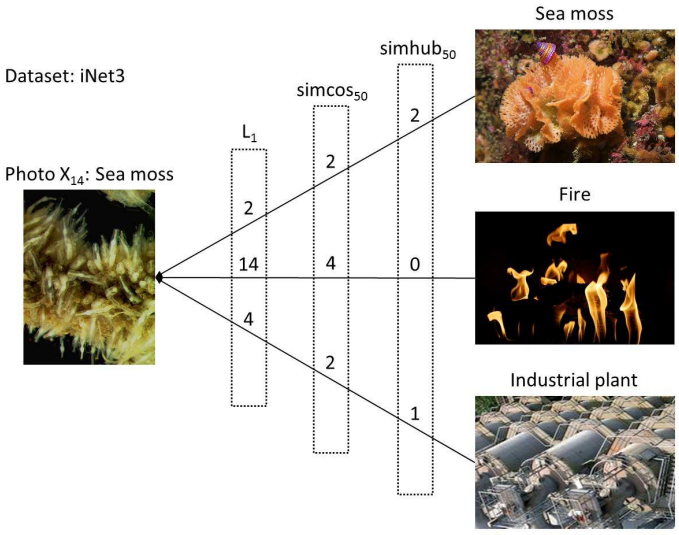


Fig. 7. The change in the neighbor occurrence profile of point X_{14} in iNet3 dataset, as the secondary similarities are introduced. The iNet3 data contain three image categories: sea moss, fire, industrial plant. In the primary metric space, image X_{14} is above average in terms of its occurrence frequency. However, 90% (18/20) of its occurrences are bad, it acts as a neighbor to points in other categories. We see how the secondary similarity scores gradually resolve this issue.

bad occurrences $BN_5(X_{14})$, but $simhub_{50}$ performs better than $simcos_{50}$ by allowing only one remaining X_{14} bad occurrence into the kNN graph.

4.3 Classification with the secondary metrics

The analysis outlined in Section 4.2 suggests that the hubness-aware definition of shared-neighbor similarities might prove more useful for the kNN classification when compared to the standard approach. In order to test this hypothesis, we have compared $simhub_{50}$ with $simcos_{50}$ in the context of k-nearest neighbor classification both on synthetic and image data.

The choice of parameters was the same as before: the shared neighbor similarities were derived from the 50-neighbor sets and the values of $k = 5$ and $k = 10$ were used for ImageNet data and the Gaussian mixtures, respectively. Other parametrizations are certainly possible and Section 4.4 deals precisely with the impact of different neighborhood sizes on the classification process.

As some hubness remains even in the shared neighbor metric space, the similarity measures were compared both in the basic kNN and across a range of hubness-aware k-nearest neighbor classification methods (hw-kNN [Radovanović et al., 2009], h-FNN [Tomašev et al., 2011b], NHBNN [Tomašev et al., 2011c], HIKNN [Tomašev and Mladenić, 2012b]).

Table 3. Algorithm performance when using the primary metrics. Classification accuracy is given for kNN, hubness-weighted kNN (hw-kNN), hubness-based fuzzy nearest neighbor (h-FNN), naive hubness-Bayesian kNN (NHBNN) and hubness information k-nearest neighbor (HIKNN). The symbols •/◦ denote statistically significant worse/better performance ($p < 0.05$) compared to kNN. The best result in each line is in bold.

(a) ImageNet data, L_1 distance, $k = 5$

Data set	kNN	hw-kNN	**h-FNN**	NHBNN	HIKNN
iNet3	72.0 ± 2.7	80.8 ± 2.3 ◦	**82.4 ± 2.2** ◦	81.8 ± 2.3 ◦	82.2 ± 2.0 ◦
iNet4	56.2 ± 2.0	63.3 ± 1.9 ◦	**65.2 ± 1.7** ◦	64.6 ± 1.9 ◦	64.7 ± 1.9 ◦
iNet5	46.6 ± 2.0	56.3 ± 1.7 ◦	**61.9 ± 1.7** ◦	61.8 ± 1.9 ◦	60.8 ± 1.9 ◦
iNet6	60.1 ± 2.2	68.1 ± 1.6 ◦	69.3 ± 1.7 ◦	69.4 ± 1.7 ◦	**69.9 ± 1.9** ◦
iNet7	43.4 ± 1.7	55.1 ± 1.5 ◦	**59.2 ± 1.5** ◦	58.2 ± 1.5 ◦	56.9 ± 1.6 ◦
iNet3Imb	72.8 ± 2.4	87.7 ± 1.7 ◦	87.6 ± 1.6 ◦	84.9 ± 1.9 ◦	**88.3 ± 1.6** ◦
iNet4Imb	63.0 ± 1.8	68.8 ± 1.5 ◦	69.9 ± 1.4 ◦	69.4 ± 1.5 ◦	**70.3 ± 1.4** ◦
iNet5Imb	59.7 ± 1.5	63.9 ± 1.8 ◦	64.7 ± 1.8 ◦	63.9 ± 1.8 ◦	**65.5 ± 1.8** ◦
iNet6Imb	62.4 ± 1.7	69.0 ± 1.7 ◦	**70.9 ± 1.8** ◦	68.4 ± 1.8 ◦	70.2 ± 1.8 ◦
iNet7Imb	55.8 ± 2.2	63.4 ± 2.0 ◦	64.1 ± 2.3 ◦	63.1 ± 2.1 ◦	**64.3 ± 2.1** ◦
AVG	59.20	67.64	**69.52**	68.55	69.31

(b) Gaussian mixture data, L_2 distance, $k = 10$

Data set	kNN	hw-kNN	h-FNN	**NHBNN**	HIKNN
DS_1	43.8 ± 3.1	64.4 ± 5.3 ◦	72.6 ± 2.8 ◦	**80.7 ± 2.4** ◦	65.8 ± 3.0 ◦
DS_2	48.4 ± 2.8	73.6 ± 6.9 ◦	79.3 ± 2.2 ◦	**83.9 ± 2.2** ◦	73.1 ± 2.5 ◦
DS_3	67.3 ± 2.3	85.3 ± 2.6 ◦	86.8 ± 1.7 ◦	**90.0 ± 1.4** ◦	86.7 ± 1.9 ◦
DS_4	52.2 ± 2.6	72.8 ± 2.3 ◦	78.4 ± 2.2 ◦	**81.9 ± 2.0** ◦	72.2 ± 2.3 ◦
DS_5	59.2 ± 2.7	80.2 ± 3.4 ◦	84.6 ± 1.8 ◦	**87.2 ± 1.5** ◦	81.1 ± 2.1 ◦
DS_6	58.6 ± 3.3	80.0 ± 3.3 ◦	81.7 ± 2.5 ◦	**86.6 ± 2.2** ◦	79.4 ± 2.5 ◦
DS_7	65.0 ± 2.4	84.6 ± 2.4 ◦	85.4 ± 1.9 ◦	**90.1 ± 1.5** ◦	84.5 ± 2.0 ◦
DS_8	71.0 ± 2.3	82.7 ± 2.5 ◦	85.9 ± 1.9 ◦	**88.4 ± 1.8** ◦	83.9 ± 2.3 ◦
DS_9	57.9 ± 2.7	76.3 ± 3.3 ◦	82.3 ± 2.0 ◦	**87.5 ± 1.7** ◦	77.7 ± 2.4 ◦
DS_{10}	57.5 ± 2.9	78.1 ± 3.3 ◦	81.1 ± 2.3 ◦	**85.5 ± 1.9** ◦	77.7 ± 2.2 ◦
AVG	58.09	77.80	81.81	**86.18**	78.21

All experiments were run as 10-times 10-fold cross validation and the corrected resampled t-test was used to check for statistical significance. The features in ImageNet data were normalized prior to classification. No normalization was performed on the Gaussian mixtures, as it was noticed that it actually harms the classification performance. For example, the average kNN accuracy drops from 59.2% to 41.78% when the Euclidean distance is applied to the normalized feature vectors.

The classification accuracy under the primary metrics (L_1,L_2) is given in Table 3. These results were already discussed from the perspective of classification in presence of class imbalance [**?**], so we will merely use them here as a baseline for comparisons with the classifier performance on the secondary metrics. Both the synthetic and the image data exhibit high hubness, so it is no surprise that the hubness-aware classification methods clearly outperform the basic kNN. In ImageNet data, all hubness-aware algorithms perform similarly, but NHBNN achieves the best result in the synthetic experiments.

Table 4. Experiments with $simhub_{50}$ and $simcos_{50}$ on ImageNet data. Classification accuracy is given for kNN, hw-kNN, h-FNN, NHBNN and HIKNN. All displayed experiments were performed for $k = 5$. The comparisons are done pairwise between the $simhub_{50}$ and $simcos_{50}$ for each classifier, so that the higher value is in bold and •/○ denotes statistically significant worse/better performance of $simhub_{50}$ compared to $simcos_{50}$ ($p < 0.05$)

(a) Distance: $simcos_{50}$

Data set	kNN	hw-kNN	h-FNN	NHBNN	HIKNN
iNet3	76.9 ± 1.8	81.2 ± 1.8	83.6 ± 1.6	83.1 ± 1.4	83.6 ± 1.5
iNet4	59.2 ± 1.4	63.4 ± 1.4	65.6 ± 1.4	65.1 ± 1.3	65.5 ± 1.3
iNet5	56.1 ± 1.4	61.8 ± 1.4	63.9 ± 1.3	63.0 ± 1.2	64.3 ± 1.3
iNet6	61.2 ± 1.3	68.1 ± 1.3	70.0 ± 1.3	69.4 ± 1.2	70.2 ± 1.3
iNet7	47.6 ± 1.0	56.6 ± 1.1	60.1 ± 1.1	59.4 ± 1.0	59.9 ± 0.9
iNet3Imb	86.5 ± 1.8	89.2 ± 1.7	89.8 ± 1.7	86.7 ± 1.8	89.8 ± 1.6
iNet4Imb	67.8 ± 1.6	70.3 ± 1.5	70.8 ± 1.7	68.3 ± 1.6	71.2 ± 1.6
iNet5Imb	64.8 ± 1.7	67.4 ± 1.5	68.6 ± 1.6	63.3 ± 1.7	69.0 ± 1.5
iNet6Imb	62.3 ± 1.6	69.8 ± 1.5	71.7 ± 1.8	68.9 ± 1.6	71.9 ± 1.5
iNet7Imb	56.7 ± 1.9	62.7 ± 2.0	64.8 ± 1.8	61.9 ± 1.9	65.0 ± 2.2
AVG	63.91	69.05	70.89	68.91	71.04

(b) Distance: $simhub_{50}$

Data set	kNN	hw-kNN	h-FNN	NHBNN	HIKNN
iNet3	**83.3 ± 1.7** ○	**84.7 ± 1.7** ○	**84.8 ± 1.6**	**84.7 ± 1.4**	**84.8 ± 1.5**
iNet4	**62.2 ± 1.5** ○	**64.0 ± 4.4**	**66.0 ± 1.4**	**65.9 ± 1.3**	**65.7 ± 1.4**
iNet5	**63.0 ± 1.2** ○	**66.4 ± 1.3** ○	**67.5 ± 1.3** ○	**66.7 ± 1.3** ○	**67.6 ± 1.3** ○
iNet6	**66.6 ± 1.5** ○	**69.7 ± 1.3**	**70.5 ± 1.3**	**70.4 ± 1.4**	**70.5 ± 1.3**
iNet7	**56.6 ± 1.1** ○	**60.9 ± 4.3**	**62.9 ± 1.1** ○	**62.5 ± 1.0** ○	**63.0 ± 1.1** ○
iNet3Imb	**88.9 ± 1.6** ○	**89.8 ± 1.6**	**90.1 ± 1.7**	**88.1 ± 1.8**	**89.9 ± 1.5**
iNet4Imb	**69.7 ± 1.7** ○	**71.2 ± 1.7**	**71.5 ± 1.6**	**69.7 ± 1.6**	**71.6 ± 1.7**
iNet5Imb	**67.3 ± 1.7** ○	**69.7 ± 1.6** ○	**70.4 ± 1.5**	**66.4 ± 1.7** ○	**70.5 ± 1.6**
iNet6Imb	**68.0 ± 1.7** ○	**71.9 ± 1.7**	**72.8 ± 1.8**	**70.6 ± 1.7**	**73.0 ± 1.8**
iNet7Imb	**62.5 ± 2.0** ○	**65.1 ± 1.9** ○	**65.8 ± 1.8**	**63.9 ± 2.1**	**65.8 ± 1.9**
AVG	**68.81**	**71.34**	**72.23**	**70.89**	**72.24**

Classification performance on the image datasets when using the secondary shared-neighbor similarities is given in Table 4. The use of $simcos_{50}$ increases the average kNN accuracy by about 5% when compared to the L_1 distance case. However, the proposed $simhub_{50}$ similarity performs even better and further improves the observed accuracy by another 5%. This is consistent with the observed difference in induced bad occurrence percentages which was shown in Figure 4. Both secondary measures improve not only the basic kNN method, but all the examined hubness-aware approaches as well. The hubness-aware $simhub_{50}$ is clearly to be preferred, since it leads to equal or higher accuracies for all the algorithms on all the datasets.

In both secondary metric spaces, the hubness-aware methods still perform favorably when compared to kNN. On the other hand, when the kNN is coupled with $simhub_{50}$, it performs better than some of the hubness-aware approaches in the primary metric

Table 5. Experiments with $simhub_{50}$ and $simcos_{50}$ on Gaussian mixture data. Classification accuracy is given for kNN, hw-kNN, h-FNN, NHBNN and HIKNN. All displayed experiments were performed for $k = 10$. The comparisons are done pairwise between the $simhub_{50}$ and $simcos_{50}$ for each classifier, so that the higher value is in bold and •/◦ denotes statistically significant worse/better performance of $simhub_{50}$ compared to $simcos_{50}$ ($p < 0.05$)

(a) Distance: $simcos_{50}$

Data set	kNN	hw-kNN	h-FNN	NHBNN	HIKNN
DS_1	64.7 ± 3.1	76.0 ± 3.4	73.7 ± 2.7	76.2 ± 2.4	73.9 ± 2.6
DS_2	69.6 ± 2.6	82.7 ± 2.6	79.7 ± 2.2	80.5 ± 2.5	79.4 ± 2.2
DS_3	81.4 ± 2.1	88.5 ± 1.8	89.1 ± 1.7	88.4 ± 1.8	88.7 ± 1.6
DS_4	72.5 ± 2.3	76.8 ± 2.4	77.9 ± 2.3	79.1 ± 2.1	78.3 ± 2.1
DS_5	77.3 ± 2.2	85.0 ± 1.9	83.4 ± 2.0	83.6 ± 2.1	83.2 ± 2.0
DS_6	76.5 ± 2.6	83.7 ± 2.3	82.2 ± 2.3	83.2 ± 2.4	82.6 ± 2.4
DS_7	81.4 ± 2.2	88.1 ± 2.1	86.2 ± 1.9	87.1 ± 2.0	86.4 ± 1.9
DS_8	82.6 ± 1.9	87.7 ± 1.7	86.9 ± 1.7	86.7 ± 1.7	86.7 ± 1.7
DS_9	81.1 ± 2.3	85.7 ± 1.9	85.9 ± 2.0	86.5 ± 2.0	86.2 ± 2.1
DS_{10}	78.1 ± 2.2	84.3 ± 2.0	86.2 ± 1.9	84.2 ± 1.8	83.6 ± 1.8
AVG	76.25	83.85	83.12	83.55	82.90

(b) Distance: $simhub_{50}$

Data set	kNN	hw-kNN	h-FNN	NHBNN	HIKNN
DS_1	**82.8 ± 2.4** ◦	**83.7 ± 2.5** ◦	**83.6 ± 2.4** ◦	**85.1 ± 2.2** ◦	**83.6 ± 2.4** ◦
DS_2	**84.5 ± 1.7** ◦	**86.5 ± 1.6** ◦	**86.8 ± 1.8** ◦	**87.9 ± 1.6** ◦	**85.8 ± 1.7** ◦
DS_3	**90.0 ± 1.6** ◦	**90.4 ± 1.6**	**91.3 ± 1.5**	**92.9 ± 1.3** ◦	**90.3 ± 1.5**
DS_4	**82.5 ± 2.3** ◦	**84.9 ± 1.7** ◦	**84.5 ± 1.8** ◦	**85.2 ± 1.7** ◦	**83.8 ± 1.9** ◦
DS_5	**85.8 ± 1.9** ◦	**87.3 ± 1.9**	**87.9 ± 1.7** ◦	**88.8 ± 1.6** ◦	**86.8 ± 1.9** ◦
DS_6	**88.4 ± 1.8** ◦	**88.5 ± 1.9** ◦	**89.1 ± 1.8** ◦	**91.4 ± 1.6** ◦	**88.8 ± 1.8** ◦
DS_7	**88.1 ± 1.8** ◦	**89.8 ± 1.6**	**90.2 ± 1.6** ◦	**92.1 ± 1.3** ◦	**88.9 ± 1.8**
DS_8	**88.3 ± 1.7** ◦	**88.7 ± 1.6**	**89.5 ± 1.6** ◦	**90.5 ± 1.5** ◦	**88.6 ± 1.7** ◦
DS_9	**85.8 ± 1.9** ◦	**88.4 ± 1.7** ◦	**88.6 ± 1.7** ◦	**90.3 ± 1.5** ◦	**87.7 ± 1.7**
DS_{10}	**86.8 ± 1.6** ◦	**89.1 ± 1.4** ◦	**89.6 ± 1.5** ◦	**90.9 ± 1.4** ◦	**88.3 ± 1.6** ◦
AVG	**86.30**	**87.73**	**88.11**	**89.51**	**87.26**

space. Nevertheless, the best results are obtained by combining the hubness-aware metric learning with the hubness-aware classification.

The results on the synthetic data (Table 5) are even more convincing. The standard $simcos_{50}$ raises the average kNN classification accuracy from 59.2% to 76.25%. Using the $simhub_{50}$ similarity gives 86.3% instead, which is a substantial further increase. As in the ImageNet data, the hubness-aware methods outperform the basic kNN in both secondary metric spaces and $simhub_{50}$ outperforms $simcos_{50}$ on every algorithm and for every dataset. The major difference is that here we see that using the $simcos_{50}$ similarity actually reduced the accuracy of NHBNN, which was the single best approach in the primary metric space. A decrease was observed on each examined synthetic dataset. Furthermore, the best obtained average result when using the $simcos_{50}$ measure equals to 83.85% (by hw-kNN, Table 5), which is still less than the best result obtained in

the primary L_2 metric space (86.18%, shown in Table 3). This shows that the use of $simcos_{50}$ is not always beneficial to hubness-aware kNN classification.

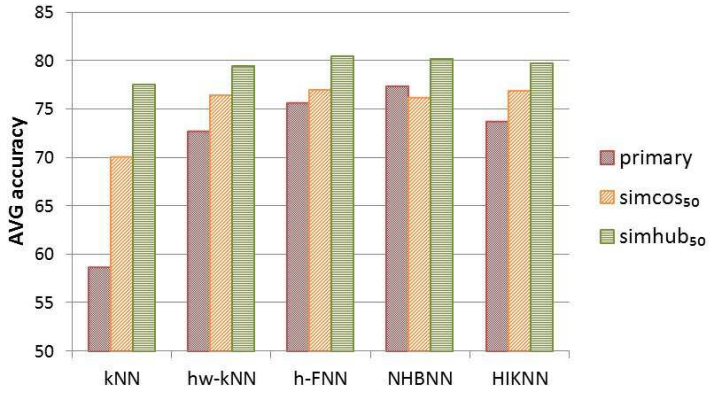


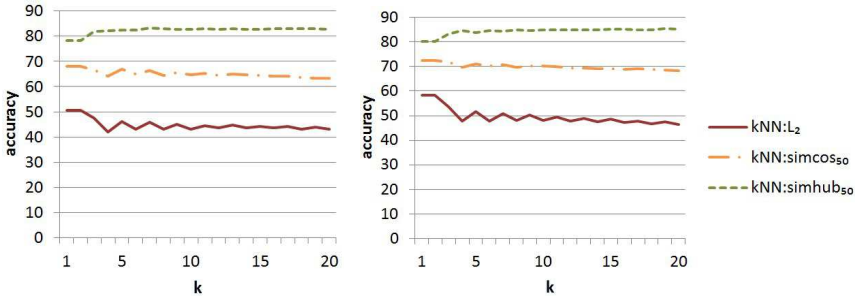
Fig. 8. The average accuracy for each algorithm and similarity measure, when taken over all the analyzed datasets (both ImageNet and the Gaussian mixtures). The increase in performance when using the shared neighbor similarities is most pronounced in kNN, which was to be expected, as the hubness-aware methods are less affected by the dimensionality curse and the hubness phenomenon. The proposed $simhub_{50}$ similarity measure leads to better accuracy in each examined algorithm.

4.4 The influence of neighborhood size

All the previously discussed experiments depended on two neighborhood size parameters (k,s). The choice of s affects the overall quality of the induced secondary kNN graph and the choice of k affects the algorithm performance in the secondary metric space. This is why it is very important to test the shared neighbor similarities over a range of different parameter values, in order to determine if the previously discussed results are relevant and not merely an artifact of a particular (k,s) choice. Figure 9 and Figure 10 show that the kNN and h-FNN classification performance on DS_1 and DS_2 is not greatly affected by a change in k. The same holds on other datasets as well.

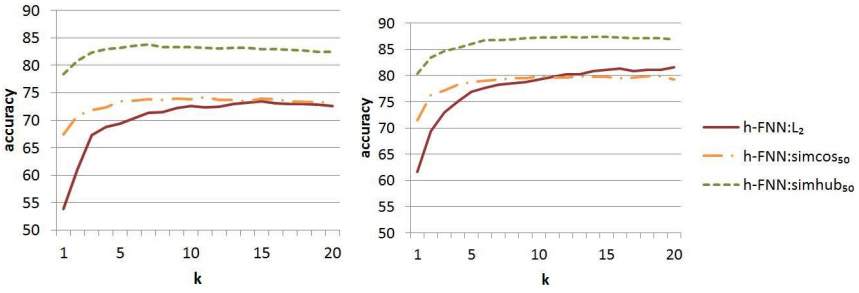
Figure 10 shows a peculiar trend, especially when compared to Figure 9. The secondary $simcos_{50}$ similarity reduces the overall bad hubness in the data, which improves the classification accuracy of kNN. On the other hand, there is a very small improvement in h-FNN and the other hubness-aware methods for $k = 10$ and it seems that even this is lost as the k is further increased. As all algorithms are operating in the same metric space, we would expect the decrease in bad hubness to affect them in similar ways and yet this is not the case when using $simcos_{50}$. This result suggests that there has to be another, more subtle difference between $simcos_{50}$ and $simhub_{50}$.

It turns out that the kNN graphs induced by $simhub_{50}$ have a significantly lower reverse neighbor set entropy, as shown in Figure 11. The reverse neighbor set entropy



(a) DS_1 (b) DS_2

Fig. 9. kNN accuracy over a range of k-neighbor set sizes. The hubness-aware $simhub_{50}$ similarity leads to better results in all cases.



(a) DS_1 (b) DS_2

Fig. 10. h-FNN accuracy over a range of k-neighbor set sizes.

is defined as $H(R_k(x)) = \sum_{c \in C} \frac{N_{k,c}(x)}{N_k(x)} \cdot \log \frac{N_k(x)}{N_{k,c}(x)}$. Anti-hubs with no previous occurrences are assigned a 0 reverse neighbor set entropy by default. The observed difference between the entropies induced by $simcos_{50}$ and $simhub_{50}$ increases with k. In other words, $simhub_{50}$ increases the average purity of neighbor occurrence profiles, which increases the quality and the reliability of occurrence models inferred by the hubness-aware classification methods. This is precisely the reason why the $simhub_{50}$ measure turns out to be more useful than $simcos_{50}$ when used in conjunction with the hubness-aware classifiers. Even though it reduces the overall bad occurrence frequency, $simcos_{50}$ reduces the purity of the secondary neighbor occurrence profiles, especially when considering larger neighborhoods. These two factors cancel each other out, so in the end no significant change in the hubness-aware classification performance remains.

The other neighborhood parameter, s, which is used to determine the size of the neighbor set from which the shared neighbor counts will be taken, is directly involved in the quality of the resulting secondary metric spaces. The use of relatively large s values was advocated for $simcos_s$ [Houle et al., 2010], as it was argued that it leads to a better similarity score. The proper s-size was said to be of the same order as the cluster size. In our synthetic Gaussian mixtures, that would amount to anywhere be-

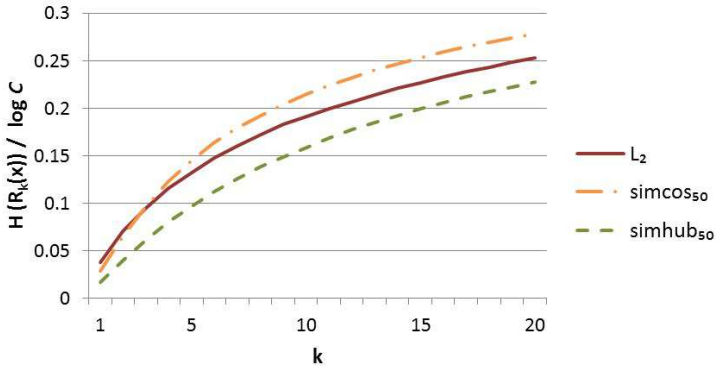
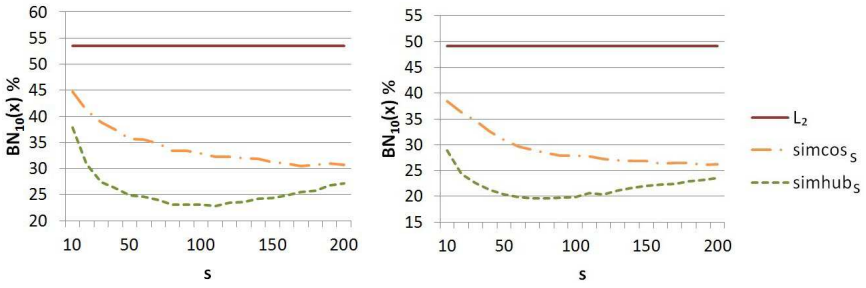


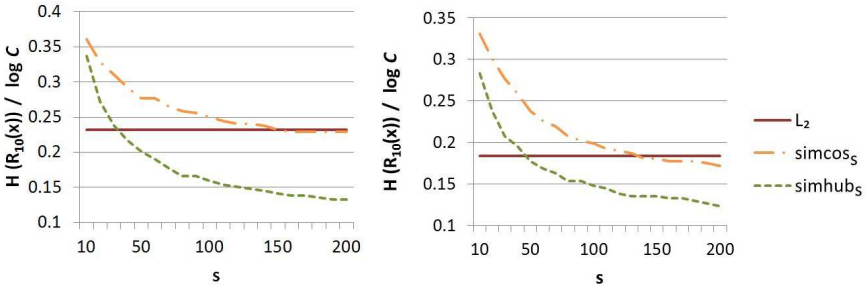
Fig. 11. The normalized reverse neighbor set entropies over a range of neighborhood sizes (k) for L_2, $simcos_{50}$ and $simhub_{50}$, averaged over all the synthetic datasets (DS_1 - DS_{10}). The hubness-aware $simhub_{50}$ increases the purity of reverse neighbor sets, while $simcos_{50}$ decreases it.

tween 50 and 200, depending on the dataset. Indeed, in DS_1 and DS_2 the optimum for $simcos_s$ in terms of bad occurrence frequencies is reached around $s = 150$, as shown in Figure 12. The hubness-aware $simhubs_s$ seems to behave differently, as it reaches its optimum for s values between 50 and 100 in these two datasets. After reaching the optimum, the performance of $simhub_s$ slowly deteriorates if s is further increased. Nevertheless, its bad hubness optimum seems to be well below the $simcos_s$ optimum. Also, for every $s \in [10, 200]$, $BN_{10}^{simhub_s} < BN_{10}^{simcos_s}$ in all the examined cases. It is actually beneficial to reach the optimum for lower s values, if possible, since it entails less computations and a shorter execution time.



(a) DS_1 (b) DS_2

Fig. 12. Bad occurrence frequencies for $k = 10$ in the secondary metric space as the s parameter is varied in $simcos_s$ and $simhub_s$ similarity measures.



(a) DS_1 (b) DS_2

Fig. 13. Normalized reverse neighbor set entropy for $k = 10$ in the secondary metric space as the s parameter is varied in $simcos_s$ and $simhub_s$ similarity measures.

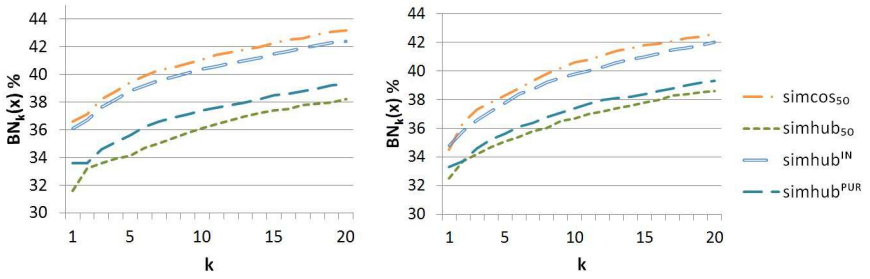
The trends involving the reverse neighbor set entropy are somewhat different. Unlike bad hubness, $H(R_{10}(x))$ monotonously decreases both for $simcos_s$ and $simhub_s$. This is shown in Figure 13, for DS_1 and DS_2. The difference between the two measures seems to be constant, regardless of the choice of s-value. This reaffirms the previously stated observation that $simhub_s$ seems to generate metric spaces where the hubness-aware occurrence models yield greater improvements. Very small s-neighborhoods are not well suited for this task, as the improvement in $H(R_{10}(x))$ over L_2 is achieved by $simhub_s$ only for $s \geq 50$. On the other hand, $simcos_s$ requires at least $s = 150$ to produce equally pure neighbor occurrence profiles as the primary metric.

We can conclude that the proposed $simhub_s$ similarity measure outperforms $simcos_s$ not only for $s = 50$ as confirmed above, but also over the entire range of different s values. Additionally, $simhub_s$ seems to reach its optimum sooner and it seems to be somewhere in the range $s \in [50, 100]$ on the synthetic datasets that we have examined.

4.5 Individual contributions of the two hubness-aware weighting terms

The hubness-aware $simhub_s$ similarity measure is based on the occurrence weighting which incorporates both the unsupervised hubness-aware component ($simhub_s^{\mathrm{IN}}$) and the supervised occurrence profile homogeneity term ($simhub_s^{\mathrm{PUR}}$). Here we will analyze how each of these individual weights affects the properties of the final $simhub_s$ similarity score.

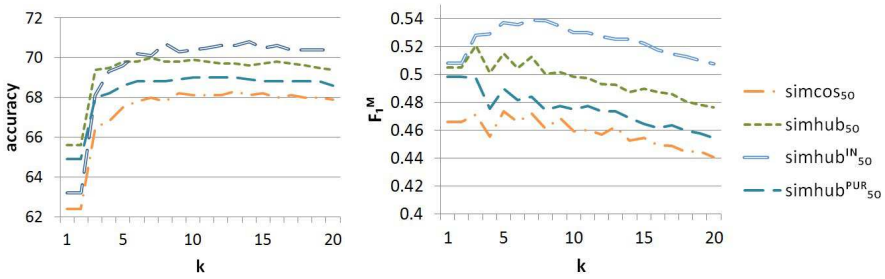
Since bad hubness has been a focal point of the previous discussion, it is important to see how each of these weighting terms helps in reducing the overall bad hubness in the data. Figure 14 shows the reduction rates on two representative ImageNet datasets, iNet5Imb and iNet6Imb. Naturally, as $simhub_s^{\mathrm{IN}}$ is an unsupervised weighting term and $simhub_s^{\mathrm{PUR}}$ a supervised one, $simhub_s^{\mathrm{PUR}}$ induces less bad hubness in the secondary metric space. Nevertheless, as Figure 14 suggests, the unsupervised term also slightly decreases the overall bad hubness. More importantly, it contributes to the overall bad hubness reduction in the final $simhub_s$ measure, as we see that the $simhub_s$ similarity induces less bad hubness than $simhub_s^{\mathrm{PUR}}$ on these image datasets.



(a) iNet5Imb (b) iNet6Imb

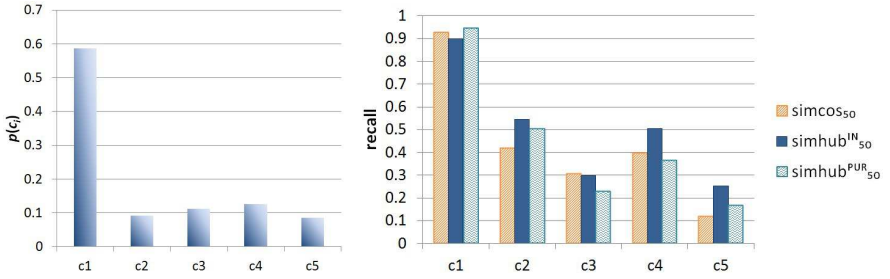
Fig. 14. The induced bad occurrence frequencies in two ImageNet datasets, given over a range of neighborhood sizes for $simcos_{50}$, $simhub_{50}$, $simhub_{50}^{\mathrm{IN}}$ and $simhub_{50}^{\mathrm{PUR}}$.

Figure 14 shows that both hubness-aware terms are relevant in reducing the overall bad hubness of the data, but it also wrongly suggests that $simhub_{s}^{\mathrm{IN}}$ a minor role in the final similarity measure. Even though the bad hubness is a good indicator of the difficulty of the data, it needs not be very strongly correlated with the actual kNN classification performance for $k > 1$. Indeed, as shown in Figure 15, $simhub_{50}^{\mathrm{IN}}$ is the single best similarity measure on the iNet5Imb dataset when $k > 3$, in terms of both the accuracy and the macro-averaged F_1 score. The difference in F_1^M is more pronounced than in the overall accuracy, which implies that $simhub_{50}^{\mathrm{IN}}$ better improves the minority class recall under the class imbalance in the iNet5Imb data. This makes sense, as $simhub_{50}^{\mathrm{IN}}$ gives preference to those neighbors which are judged to be more local to the points of interest. The observation is confirmed in Figure 16, where the recall for each class in the iNet5Imb dataset is shown. Similar trends can also be seen in the other examined imbalanced ImageNet datasets. On the other hand, both weighting terms perform more or less equally on the examined Gaussian mixtures, which is not surprising, as this data is not so highly imbalanced.



(a) accuracy (b) F_1^M

Fig. 15. The accuracy and the macro-averaged F_1 score on INet5Imb for kNN when using some of the different secondary similarities: $simcos_{50}$, $simhub_{50}$, $simhub_{50}^{\mathrm{IN}}$ and $simhub_{50}^{\mathrm{PUR}}$.



(a) iNet5Imb class distribution (b) iNet5Imb recall for each class

Fig. 16. The class-specific recall given for $simcos_{50}$, $simhub_{50}^{\mathrm{IN}}$ and $simhub_{50}^{\mathrm{PUR}}$ on the iNet5Imb dataset for $k = 5$. The unsupervised hubness-aware term $simhub_{50}^{\mathrm{IN}}$ outperforms the supervised $simhub_{50}^{\mathrm{PUR}}$ on all the minority classes in the data. The recall of $simhub_{50}^{\mathrm{PUR}}$ is higher only for the majority class.

Whether it turns out that the stated conclusions hold in general or not, it is already clear that $simhub_s^{\mathrm{IN}}$ and $simhub_s^{\mathrm{PUR}}$ affect the final $simhub_s$ similarity measure in different ways. Therefore, it makes sense to consider a parametrized extension of the $simhub_s$ weighting by introducing regulating exponents to the individual hubness-aware terms.

$$simhub_s^{\alpha,\beta}(x_i, x_j) = \frac{\sum_{x \in D_s(x_i) \cup D_s(x_j)} I_n(x)^{\alpha} \cdot (\max H_s - H(R_s(x)))^{\beta}}{s \cdot \max {H_s}^{\beta} \cdot \max {I_n}^{\alpha}} \tag{8}$$

However, it remains rather unclear how one should go about determining the optimal (α, β) combination for a given dataset without over-fitting on the training split. The parameter values could also be derived from the misclassification cost matrix in unbalanced classification scenarios. A thorough analysis of this idea is beyond the scope of this paper, but it is something that will definitely be carefully investigated in the future work.

4.6 Handling of the difficult points

Some points are more difficult to properly classify than others and each individual dataset is composed of a variety of different point types with respect to the difficulty they pose for certain classification algorithms. A point characterization scheme based on the nearest neighbor interpretation of classification difficulty has recently been proposed for determining types of minority class points in imbalanced data [Napierala and Stefanowski, 2012]. As the method is limited to imbalanced data, it can be used to characterize points in any class and on any dataset. It is the most natural approach to adopt in our analysis, as the point difficulty is expressed in terms of the number of label mismatches among its 5-NN set. Points with at most one mismatch are termed *safe*, points with 2-3 mismatches are referred to as being *borderline examples*, points with 4 mismatches are considered *rare* among their class and points with all neighbors from different classes are said to be *outliers*.

As the SNN similarity measures induce a change in the kNN structure of the data, we can expect that a change in metric might lead to a change in the overall point type distribution. Reducing the overall difficulty of points can be directly correlated with the improvement in the kNN classification performance. This is precisely what happens when the SNN measures are used, as shown in Figure 17 for the synthetic datasets. Both the standard $simcos_{50}$ and the proposed $simhub_{50}$, $simhub_{50}^{\mathrm{IN}}$ and $simhub_{50}^{\mathrm{PUR}}$ significantly increase the number of safe points when compared to the primary L_2 metric. The hubness-aware shared neighbor similarities improve the point difficulty distribution more than $simcos_{50}$, which explains the classification accuracy increase discussed in Section 4.3.

The two hubness-aware weighting terms lead to an approximately equal classification accuracy on the examined Gaussian mixtures, so it is somewhat surprising that they induce different distributions of point difficulty. The purity term, $simhub_{50}^{\mathrm{PUR}}$, is better at increasing the number of safe points than the occurrence self-information term, $simhub_{50}^{\mathrm{IN}}$. This is compensated by the fact that the difference in the number of borderline points is in favor of $simhub_{50}^{\mathrm{IN}}$ by a slightly larger margin. As borderline points are correctly classified approximately 50% of the time, the two distributions exhibit similar overall difficulties for the kNN classification methods.

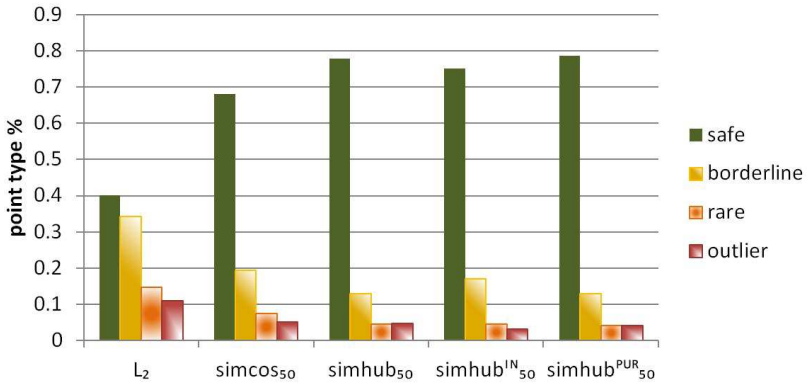


Fig. 17. Distribution of point types on synthetic data under several employed metrics. The hubness-aware secondary similarity measures significantly increase the proportion of safe points which leads to an increase in kNN classification performance.

The difference between the examined similarities/metrics is present in each examined dataset. The proportion of safe points is shown in Figure 18 for each of the Gaussian mixtures. ImageNet data exhibit the same properties. The increase in the proportion of safe points is yet another desirable property of the proposed hubness-aware SNN measures.

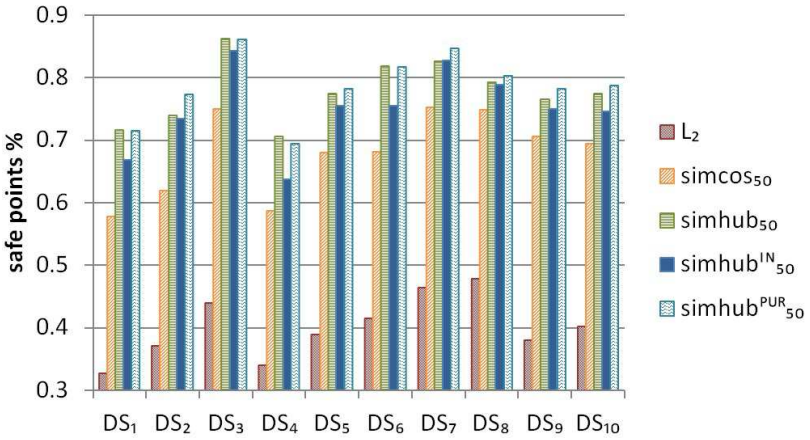


Fig. 18. The percentage of safe points on each of the examined Gaussian mixtures. The proposed $simhub_{50}$ measure induces a larger proportion of safe points in each dataset, when compared to the standard $simcos_{50}$.

4.7 Reducing the error propagation

Data processing and preparation sometimes introduces additional errors in the feature values and these errors can more easily propagate and negatively affect the learning process under the assumption of hubness. We will briefly discuss three such datasets (iNet3Err:100, iNet3Err:150, iNet3Err:1000) described in [Tomašev et al., 2011a]. The three datasets contain the 100, 150 and 1000-dimensional quantized representations, respectively. While the system was extracting the Haar feature representations for the dataset, some I/O errors occurred which left a few images as zero vectors, without having been assigned a proper representation. Surely, this particular error type can easily be prevented by proper error-checking within the system, but we will nevertheless use it as an illustrative example for a more general case of data being compromised by faulty examples. In a practical application, obvious errors such as these would either be removed or their representations recalculated. In general, the errors in the data are not always so easy to detect and correct. This is why the subsequent data analysis ought to be somewhat robust to errors and noise.

Even though errors in the data are certainly undesirable, a few zero vectors among 2731, which is the size of iNet3 data, should not affect the overall classifier performance too much, as long as the classifier has good generalization capabilities. The kNN classifier, however, suffers from a high specificity bias and this is further emphasized by the curse of dimensionality under the assumption of hubness. Namely, the employed metric (L_1) induced an unusually high-hubness of zero-vectors. It can easily be shown that the expected L_1 dissimilarity between any two quantized image representations increases with increasing dimensionality. On the other hand, the distance to the zero-vector remains constant for each image. Eventually, when $d = 1000$ the few zero-vectors in the data infiltrated and dominated all the k-neighbor sets and caused the 5-NN to perform

worse than zero rule, as they were, incidentally, of the minority class. The increasing bad hubness of the top 5 bad hubs is shown in Figure 19.

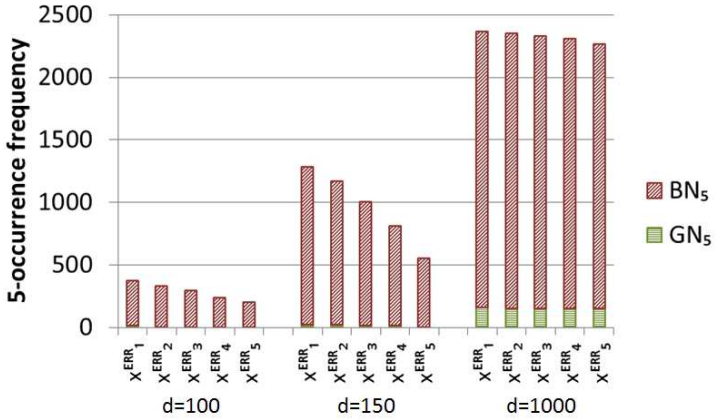


Fig. 19. The increasing bad hubness of the top 5 erroneous bad hubs in the quantized iNet3 Haar feature representations. All the bad hubs were in fact zero-vectors generated by a faulty feature extraction system and all of them were of the minority class. These zero vectors became dominant bad hubs as the dimensionality of the data representation was increased. Such a pathological case clearly illustrates how even a few noisy examples are enough to compromise all k-nearest neighbor inference in high-hubness data.

Such pathological cases are rare, but clearly indicate the dangers in disregarding the skewness of the underlying occurrence distribution. As this example is quite extreme, it is a good test case to examine the robustness of the secondary similarity measures to such a high violation of semantics in the k-nearest neighbor graph. The comparisons were performed as 10-times 10-fold cross validation and the results for kNN are summarized in Figure 20. The neighborhood size $k = 5$ was used.

For the 1000-dimensional faulty representation, the secondary $simcos_{200}$ and $simhub_{200}$ similarities improved the overall kNN accuracy from 20% to 94%, which is undeniably impressive. Both the $simcos_s$ and $simhub_s$ reached their optimum for $s = 200$, but for $s \in [50, 200]$ the hubness-aware similarity measure outperformed its counterpart, as it converges to the correct kNN graph configuration faster than $simcos_s$, which was previously discussed in Section 4.4. This is shown in Figure 20 for $s = 50$.

What this example shows is that the hubness-aware shared neighbor distances are able to significantly reduce the impact of errors on high-hubness data classification. Such robustness is of high importance, as real world data is often inaccurate and noisy. This particular example might have been extreme, but such extreme cases are likely to occur whenever errors end up being hubs in the data, which depends on the choice of feature representation and the primary metric.

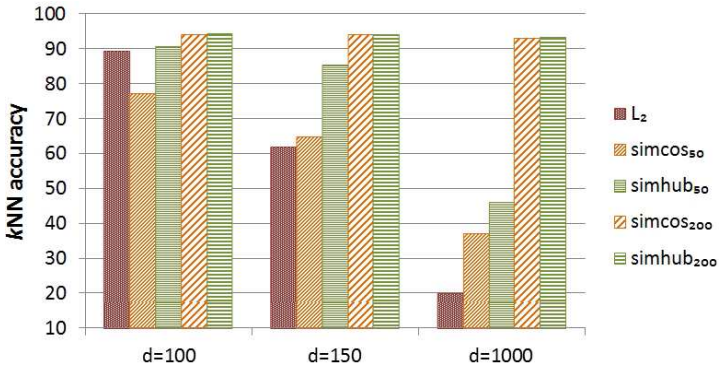


Fig. 20. The kNN accuracy on high-hubness erroneous image data under L_1, $simcos_{50}$, $simhub_{50}$, $simcos_{200}$, $simhub_{200}$. The secondary similarity measures reduce the impact of faulty and inaccurate examples.

4.8 Class separation

Ensuring a good separation between classes is what a good metric should ideally be able to achieve. This is not always possible, as the cluster assumption is sometimes severely violated. Even so, we would expect the examples from the same class to be, on average, closer to each other than the pairs of examples taken from different classes. Increasing the contrast between the average intra-class and inter-class distance is one way to make the classification task somewhat easier. The improvement is not, however, guaranteed, especially when the kNN methods are used. Unless the kNN structure changes in such a way that the ensuing distribution of point difficulty becomes favorable, the contrast is of secondary importance.

The proposed $simhub_s^{\mathrm{PUR}}$ measure was designed in such a way that the neighbors with higher occurrence profile purity are valued more, as they usually contribute more to the intra-class similarities. However, note that this is only guaranteed in binary classification. If there are only two classes in the data, $H(R_s(x_1)) < H(R_s(x_2))$ directly follows from the fact that x_1 has a higher relative contribution to the contrast than x_2.

There is also a downside to using the occurrence entropies for determining neighbor occurrence weights. The entropies measure the relative purity which reflects the relative positive contribution of the neighbor point. However, if we are interested specifically in increasing the contrast, we are interested in rewarding the *absolute* positive contributions, not the relative ones. In other words, even if two points x_1 and x_2 have the same reverse neighbor set purity, x_1 has a higher contribution to the overall similarity if $N_s(x_1) > N_s(x_2)$. Within the $simhub_s$ measure, this problem is even more pronounced because $N_s(x_1) > N_s(x_2) \Rightarrow I_n(x_1) < I_n(x_2)$.

This is very interesting, as we have seen in Section 4.5 that reducing the weight of hubs by $simhub_s^{\mathrm{IN}}$ is highly beneficial. It increases the reverse neighbor set purity, reduces bad hubness and improves the kNN classification as much as $simhub_s^{\mathrm{PUR}}$. However, it seems that it actually reduces the contrast between the intra-class and inter-class similarities, especially when used in conjunction with $simhub_s^{\mathrm{PUR}}$.

In multi-class data, things get even more complicated. Each neighbor point x_i contributes to $\binom{N_s(x_i)}{2} = GS(x_i) + BS(x_i)$ shared neighbor similarity scores, where $GS(x_i)$ and $BS(x_i)$ represent the number of intra-class and inter-class similarities, respectively. Denote by $CS(x_i) = GS(x_i) - BS(x_i)$ the contribution of each x_i to the total difference between the two similarity sums.

$$\begin{aligned} GS(x_i) &= \sum_{c \in C} \binom{N_{s,c}(x_i)}{2} \\ BS(x_i) &= \sum_{c_1, c_2 \in C, c_1 \neq c_2} N_{s,c_1}(x_i) \cdot N_{s,c_2}(x_i) \end{aligned} \tag{9}$$

The occurrence purity $OP(x_i) = \max H_s - H(R_s(x_i))$ is tightly correlated with $CS(x_i)$. Nevertheless, in non-binary classification, some occurrence profiles exist such that $OP(x_i) < OP(x_j)$, but $CS(x_i) > CS(x_j)$ or vice versa. Consider the following 4-class example:

$$\begin{aligned} &C = 4, \qquad \max H_s = \log 4 = 2 \\ &N_s(x_i) = N_s(x_j) = 100 \end{aligned}$$

$$\begin{aligned} &N_{s,1}(x_i) = 5, \quad N_{s,2}(x_i) = 15, \quad N_{s,3}(x_i) = 25, \quad N_{s,4}(x_i) = 55 \\ &N_{s,1}(x_j) = 6, \quad N_{s,2}(x_j) = 10, \quad N_{s,3}(x_j) = 34, \quad N_{s,4}(x_j) = 50 \end{aligned}$$

$$\begin{aligned} CS(x_i) &= GS(x_i) - BS(x_i) = 1900 - 3050 = -1150 \\ CS(x_j) &= GS(x_j) - BS(x_j) = 1846 - 3104 = -1258 \end{aligned} \tag{10}$$

$$\begin{aligned} OP(x_i) &= 2 - H(R_s(x_i)) \approx 2 - 1.6010 = 0.3990 \\ OP(x_j) &= 2 - H(R_s(x_j)) \approx 2 - 1.5989 = 0.4011 \end{aligned}$$

$$OP(x_i) < OP(x_j) \quad \wedge \quad CS(x_i) > CS(x_j)$$

This example shows that the reverse neighbor set purity is not monotonous with respect to the difference between the intra-class and inter-class similarity contributions of a neighbor point.

Note, however, that maximizing the sum total of $CS_D = \sum_{x \in D} CS(x)$ is not equivalent to maximizing the contrast between the inter- and intra-class distances, as that quantity requires normalization. Let $C_D = \frac{\mathrm{AVG}_{y_i \neq y_j}(dist(x_i,x_j)) - \mathrm{AVG}_{y_i = y_j}(dist(x_i,x_j))}{\max_{x_i,x_j \in D} dist(x_i,x_j) - \min_{x_i,x_j \in D} dist(x_i,x_j)}$ quantify the contrast. The denominator is necessary, as it would otherwise be possible to increase the contrast arbitrarily simply by scaling up all the distances. In practice, this means that the contrast also depends on the maximum/minimum pairwise distances on the data - and these quantities also change while we are changing the instance weights when trying to increase CS_D. Nevertheless, increasing CS_D seems like a sensible approach to improving class separation, slightly more natural than increasing the overall

purity $OP_D = \sum_{x \in D} OP(x)$. To see if this is really the case, we defined two additional hubness-aware similarity measures.

$$simhub_s^{01} = \frac{\sum_{x \in D_s(x_i) \cup D_s(x_j)} \mathbb{1}_{\{\bar{x}: CS(\bar{x}) > 0\}}(x)}{s} \tag{11}$$

$$simhub_s^{\mathrm{REL}} = \frac{\sum_{x \in D_s(x_i) \cup D_s(x_j)} (CS(x) - \min_{\bar{x} \in D}(CS(\bar{x})))}{s \cdot (\max_{\bar{x} \in D} CS(\bar{x}) - \min_{\bar{x} \in D} CS(\bar{x}))} \tag{12}$$

If we limit the weight of each shared neighbor point to the interval $w(x) \in [0, 1]$, it is not difficult to see that the CS_D is trivially maximized if and only if $w(x) = 1$ when $CS(x) > 0$ and $w(x) = 0$ when $CS(x) \leq 0$. This weighting in embodied in $simhub_s^{01}$, defined in Equation 11 above. Even though the total difference between the contributions to inter- and intra-class distances is maximized, it is clear that this measure has some very undesirable properties. First of all, it is not impossible to construct a dataset with a severe cluster assumption violation where $\forall x \in D : CS(x) \leq 0$. All the $simhub_s^{01}$ similarities would then equal zero and this is certainly not what we want. In less extreme, real world data, this measure could similarly annul some of the pairwise similarities when all the shared neighbors have $CS(x) \leq 0$. What this example clearly shows is that even though we would like to increase CS_D and improve the contrast, not only does the global optimum for CS_D not guarantee the best class separation, it also involves having a similarity measure which has many practical weaknesses.

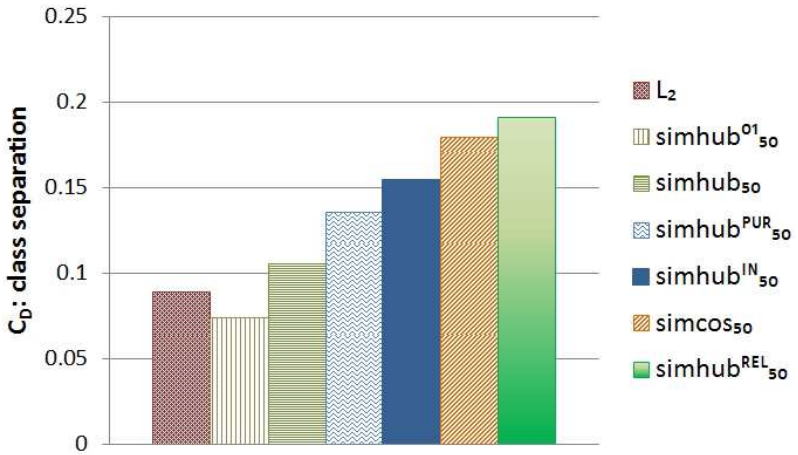


Fig. 21. The average class separation induced by different metrics on the Gaussian mixtures (DS_1-DS_{10}). Even though the $simcos_{50}$ measure has been shown to be inferior in kNN classification, it achieves better class separation than the previously considered $simhub_{50}$, $simhub_{50}^{\mathrm{IN}}$ and $simhub_{50}^{\mathrm{PUR}}$ similarities. On the other hand, the newly proposed $simhub_{50}^{\mathrm{REL}}$ measure gives the best separation between the classes.

The $simhub_s^{\mathrm{REL}}$ similarity score is a far less radical approach than $simhub_s^{01}$. The neighbor occurrence weights are proportional to the normalized neighbor contributions $CS(x)$ to the CS_D total. Even though this measure is in a sense similar to $simhub_s^{\mathrm{PUR}}$, there are no more problems with monotonicity of $w(x)$ with respect to $CS(x)$. This ought to help improve the class separation. Also, $w(x) \geq 0$ for points with $CS(x) < 0$, so there is no risk of having many zero similarities, as was the case with $simhub_s^{01}$.

Figure 21 shows the class separation induced by each of the mentioned similarity measures, on the Gaussian mixture datasets. The standard $simcos_{50}$ measure achieves better class separation than the previously considered hubness-aware SNN measures: $simhub_{50}$, $simhub_{50}^{\mathrm{IN}}$ and $simhub_{50}^{\mathrm{PUR}}$. This is somewhat surprising, given that it was shown to be clearly inferior in terms of kNN classification accuracy, bad hubness, as well as the inverse neighbor set purity. However, this is 10-class data and, as was explained above, there is no guarantee that any of the three hubness-aware measures would improve the separation, as defined by C_D. On the other hand, the newly proposed $simhub_{50}^{\mathrm{REL}}$ measure does manage to increase the separation, unlike the initial choice $simhub_{50}^{01}$, which fails for reasons already discussed.

The difference between the $simcos_{50}$ and $simhub_{50}^{\mathrm{REL}}$ is present in all datasets. The comparisons were also performed in terms of the widely used Silhouette coefficient [Tan et al., 2005], which is shown in Figure 22. The Silhouette coefficient is used for evaluating cluster configurations. If we observe each class as a cluster, a higher Silhouette score means that the classes in the data conform better to the cluster assumption. If the index value is low, it means that the classes are not really compact and either overlap or are composed of several small clusters, scattered around a larger volume of space. The Silhouette values for the considered overlapping Gaussian mixtures are still rather low, but the original ones (in the L_2 metric) were even negative in some datasets, meaning that the points from some different class are on average closer than the points from the same class. So, both $simcos_{50}$ and $simhub_{50}^{\mathrm{REL}}$ improve the cluster structure of the data, but the $simhub_{50}^{\mathrm{REL}}$ does it better.

Regardless of the fact that it improves class separation, $simhub_s^{\mathrm{REL}}$ turns out to be not nearly as good as $simhub_s$ when it comes to reducing bad hubness and improving the classification performance. This is why we would not recommend it for kNN classification purposes. Regardless, as it raises the Silhouette coefficient, $simhub_s^{\mathrm{REL}}$ could be used in some clustering applications. Admittedly, it is a supervised measure (it requires the data points to have labels), but these labels could either be deduced by an initial clustering run or already present in the data. Namely, a considerable amount of research was done in the field of semi-supervised clustering [Bilenko et al., 2004], where some labeled/known examples are used to help improve the clustering process. This was done either by introducing constraints [Lu, 2007] or precisely by some forms of metric learning [Kumar et al., 2005][Bilenko et al., 2004].

To conclude, we can say that not increasing the class separation as much as $simcos_s$ is the only apparent downside of using $simhub_s$, but one which can be tolerated, as we have seen that the proposed hubness-aware shared-neighbor similarity measure helps where it matters the most - in improving the classifier performance and reducing bad hubness, which is a very important aspect of the curse of dimensionality. Nevertheless, $simhub_s$ still significantly improves the class separation when compared to the primary

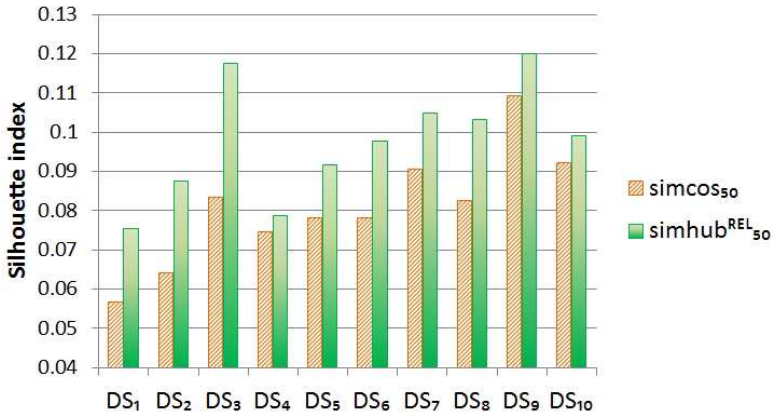


Fig. 22. The comparison in terms of the Silhouette index on the Gaussian mixtures (DS_1-DS_{10}) between $simcos_{50}$ and $simhub_{50}^{\mathrm{REL}}$. The newly proposed hubness-aware SNN measure makes the class-clusters more compact in all considered datasets.

metric, and if the class separation and the cluster structure of the data is of highest importance in a given application, $simhub_s^{\mathrm{REL}}$ is still preferable to the standard $simcos_s$.

4.9 Low-dimensional data

Low dimensional data does not exhibit hubness and is usually easier to handle as it doesn't suffer from the curse of dimensionality. We have analyzed 10 such low-dimensional datasets. The detailed data description was given in Table 2. Some datasets even exhibited negative skewness of the neighbor occurrence distribution, which might even be interpreted as *anti-hubness*, an opposite of what we have been analyzing up until now.

We have compared the $simcos_{50}$ and $simhub_{50}$ with the primary Euclidean distance on this data, by observing the kNN accuracy in 10-times 10-fold cross-validation for $k = 5$. All features were standardized by subtracting the mean and dividing by standard deviation prior to classification. The results are shown in Figure 23.

Apparently, both shared neighbor similarities seem to be somewhat inadequate in this case. They offer no significant improvements over the primary metric, sometimes being slightly better, sometimes slightly worse. The average accuracy over the ten considered low-dimensional datasets is 85.17 for L_2, 84.55 for $simcos_{50}$ and 85.1 for $simhub_{50}$.

This comparison shows that the shared neighbor similarities ought to be used primarily when the data is high dimensional and exhibits noticeable hubness. In low dimensional data, other approaches might be preferable.

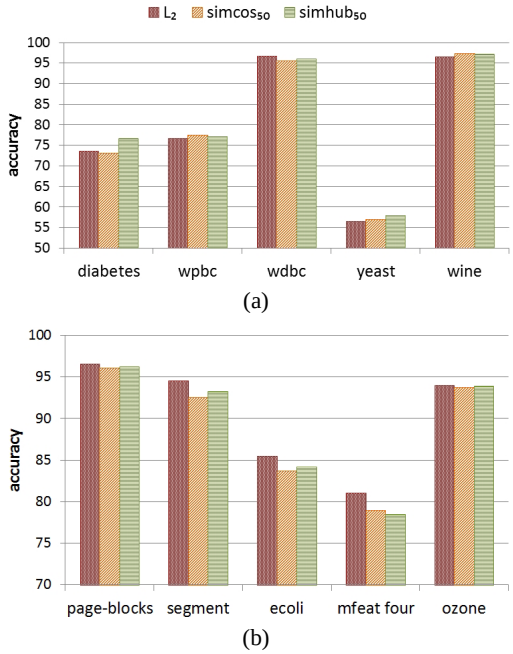


Fig. 23. The accuracy of the k-nearest neighbor classifier on low dimensional data under different distance measures. As there is no hubness in this data, there are no visible improvements.

5 Conclusions and Future Work

In this paper we proposed a new secondary shared-neighbor similarity measure $simhub_s$, in order to improve the k-nearest neighbor classification in high-dimensional data. Unlike the previously used $simcos_s$ score, $simhub_s$ takes hubness into account, which is important as hubness is a known aspect of the curse of dimensionality which can have severe negative effects on all nearest-neighbor methods. Nevertheless, it has only recently come into focus and this is the first attempt at incorporating hubness information into some form of metric learning.

An experimental evaluation was performed both on synthetic high-dimensional overlapping Gaussian mixtures and quantized SIFT representations of multi-class image data. The experiments have verified our hypothesis by showing that the proposed $simhub_s$ similarity measure clearly and significantly outperforms $simcos_s$ in terms of the associated classification performance. This improvement can be attributed to a reduce in the bad hubness of the data and the increased purity of the neighbor occurrence profiles. The kNN graphs induced by the $simhub_s$ measure are less correlated to the primary metric kNN structure, which shows that the hubness-aware measure changes the kNN structure much more radically than $simcos_s$.

As $simhub_s$ was defined in a hybrid way, by exploiting both the supervised and the unsupervised hubness information, we have thoroughly analyzed the influence of both

constituents ($simhub_s^{\mathrm{PUR}}$ and $simhub_s^{\mathrm{IN}}$, respectively) on the final similarity score. It was shown that both factors decrease the bad hubness of the data and that they do it best when combined, as in $simhub_s$. On the other hand, $simhub_s^{\mathrm{IN}}$ seems to be somewhat better in dealing with imbalanced datasets.

All secondary metrics change the overall distribution of point types in the data. The hubness-aware measures excel in increasing the proportion of *safe* points, which are the ones that are least likely to be misclassified in k-nearest neighbor classification. This is closely linked to the improved classifier performance.

The only notable downside to the $simhub_s$ measure is that it does not increase the class separation as much as the standard $simcos_s$. This has been thoroughly discussed in Section 4.8, where we have tried to overcome this difficulty by proposing an additional two hubness-aware SNN measures: $simhub_s^{01}$ and $simhub_s^{\mathrm{REL}}$. The experiments have shown that $simhub_s^{\mathrm{REL}}$ does indeed improve the class separation better than both $simcos_s$ and $simhub_s$. The proposed $simhub_s$ is still to be preferred for classification purposes, but $simhub_s^{\mathrm{REL}}$ might be used in some other applications, as for instance the semi-supervised clustering.

In our future work we would like to compare the outlined approaches to other forms of metric learning, both theoretically under the assumption of hubness, as well is various practical applications. As for the possible extensions, it would be interesting to include position-based weighting, as was done before in some shared nearest neighbor clustering algorithms. In this paper we focused mostly on the supervised case, but we intend also to explore in detail the use of hubness-aware SNN similarity measures in unsupervised data mining tasks.

Acknowledgments

This work was supported by the Slovenian Research Agency, the IST Programme of the EC under PASCAL2 (IST-NoE-216886).

Bibliography

Aggarwal, C. C.; Hinneburg, A.; Keim, D. A. On the surprising behavior of distance metrics in high dimensional spaces. In: *Proc. 8th Int. Conf. on Database Theory (ICDT)*. 420–434 (2001).

Ambert, K. H.; Cohen, A. M. k-information gain scaled nearest neighbors: A novel approach to classifying protein-protein interaction-related documents. *IEEE/ACM Trans. Comput. Biol. Bioinformatics* **9**, 305–310 (2012).

Aucouturier, J. Ten experiments on the modelling of polyphonic timbre. *Technical Report*, Docteral dissertation, University of Paris 6 (2006).

Aucouturier, J.; Pachet, F. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences* **1** (2004).

Ayad, H.; Kamel, M. Finding natural clusters using multi-clusterer combiner based on shared nearest neighbors. In: *Multiple Classifier Systems*. **2709**, 159–159, 159–159, 159–159 (Springer Berlin / Heidelberg, 2003).

Bennett, K. P.; Fayyad, U.; Geiger, D. Density-based indexing for approximate nearest-neighbor queries. In: *ACM SIGKDD Conference Proceedings*. 233–243 (ACM Press, 1999).

Bilenko, M.; Basu, S.; Mooney, R. J. Integrating constraints and metric learning in semi-supervised clustering. In: *Proceedings of the twenty-first international conference on Machine learning*. 11–, ICML '04 (ACM, New York, NY, USA, 2004).

Buza, K.; Nanopoulos, A.; Schmidt-Thieme, L. Insight: efficient and effective instance selection for time-series classification. In: *Proceedings of the 15th Pacific-Asia conference on Advances in knowledge discovery and data mining - Volume Part II*. 149–160, PAKDD'11 (Springer-Verlag, 2011).

Chaovalitwongse, W. A.; Fan, Y.-J.; Sachdeo, R. C. On the time series k-nearest neighbor classification of abnormal brain activity. *IEEE Transactions on Systems, Man, and Cybernetics, Part A* 1005–1016 (2007).

Chen, J.; ren Fang, H.; Saad, Y. Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research* **10**, 1989–2012 (2009).

C.J.Stone. Consistent nonparametric regression. *Annals of Statistics* **5**, 595–645 (1977).

Devroye, L. On the inequality of cover and hart. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **3**, 75–78 (1981).

Durrant, R. J.; Kabán, A. When is 'nearest neighbour' meaningful: A converse theorem and implications. *Journal of Complexity* **25**, 385–397 (2009).

Ertz, L.; Steinbach, M.; Kumar, V. Finding topics in collections of documents: A shared nearest neighbor approach. In: *In Proceedings of Text Mine 01, First SIAM International Conference on Data Mining* (2001).

Fix, E.; Hodges, J. Discriminatory analysis, nonparametric discrimination: consistency properties. *Technical Report*, USAF School of Aviation Medicine, Randolph Field (1951).

Flexer, A.; Gasser, M.; Schnitzer, D. Limitations of interactive music recommendation based on audio content. In: *Proceedings of the 5th Audio Mostly Conference: A*

Conference on Interaction with Sound. 13:1–13:7, AM '10 (ACM, New York, NY, USA, 2010).

Flexer A., S. J., Schnitzer D. Putting the user in the center of music information retrieval. In: *Proceedings of the 13th International Society for Music Information Retrieval Conference*. ISMIR'12 (2012).

François, D.; Wertz, V.; Verleysen, M. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering* **19**, 873–886 (2007).

Gasser M., S. D., Flexer A. Hubs and orphans - an explorative approach. In: *Proceedings of the 7th Sound and Music Computing Conference*. SMC'10 (2010).

Hodge, V. J.; Austin, J. A binary neural k-nearest neighbour technique. *Knowledge and Information Systems (KAIS)* **8**, 276–291 (2005).

Holte, R. C.; Acker, L. E.; Porter, B. W. Concept learning and the problem of small disjuncts. In: *Proc. 11th Int. Conf. AI - Volume 1*. 813–818 (Morgan Kaufmann Publishers Inc., 1989).

Houle, M. E.; Kriegel, H.-P.; Kröger, P.; Schubert, E.; Zimek, A. Can shared-neighbor distances defeat the curse of dimensionality? In: *Proc. of the 22nd int. conf. on Scientific and statistical database management*. 482–500, SSDBM'10 (Springer-Verlag, 2010).

Jarvis, R. A.; Patrick, E. A. Clustering using a similarity measure based on shared near neighbors. *IEEE Trans. Comput.* **22**, 1025–1034 (1973).

Jensen, R.; Cornelis, C. A new approach to fuzzy-rough nearest neighbour classification. In: *Proceedings of the 6th International Conference on Rough Sets and Current Trends in Computing*. 310–319, RSCTC '08 (Springer-Verlag, Berlin, Heidelberg, 2008).

Keller, J. E.; Gray, M. R.; Givens, J. A. A fuzzy k-nearest-neighbor algorithm. In: *IEEE Transactions on Systems, Man and Cybernetics*. 580–585 (1985).

Kumar, N.; Kummamuru, K.; Paranjpe, D. Semi-supervised clustering with metric learning using relative comparisons. In: *Proceedings of the Fifth IEEE International Conference on Data Mining*. 693–696, ICDM '05 (IEEE Computer Society, Washington, DC, USA, 2005).

L. Devroye, A. K., L. Gyorfi; Lugosi, G. On the strong universal consistency of nearest neighbor regression function estimates. *Annals of Statistics* **22**, 1371–1385 (1994).

Li, Y.; Zhang, X. Improving k-nearest neighbor with exemplar generalization for imbalanced classification. In: *Advances in Knowledge Discovery and Data Mining*. **6635**, 321–332, 321–332, 321–332 (Springer, 2011).

Lienhart, R.; Maydt, J. An extended set of haar-like features for rapid object detection. In: *IEEE ICIP 2002*. 900–903 (2002).

Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* **60**, 91 (2004).

Lu, Z. Semi-supervised clustering with pairwise constraints: A discriminative approach. *Journal of Machine Learning Research - Proceedings Track* 299–306 (2007).

Moëllic, P.-A.; Haugeard, J.-E.; Pitel, G. Image clustering based on a shared nearest neighbors approach for tagged collections. In: *Proceedings of the 2008 international conference on Content-based image and video retrieval*. 269–278, CIVR '08 (ACM, New York, NY, USA, 2008).

Napierala, K.; Stefanowski, J. Identification of different types of minority class examples in imbalanced data. In: Corchado, E.; Snasel, V.; Abraham, A.; Wozniak, M.; Graa, M.; Cho, S.-B. (eds.) *Hybrid Artificial Intelligent Systems*. **7209**, 139–150, 139–150, 139–150 (Springer Berlin / Heidelberg, 2012).

Ougiaroglou, S.; Nanopoulos, A.; Papadopoulos, A. N.; Manolopoulos, Y.; Welzer-druzovec, T. Adaptive k-nearest neighbor classification based on a dynamic number of nearest neighbors. In: *Proceedings of ADBIS Conference*. ADBIS 2007 (2007).

Patidar, A. K.; Agrawal, J.; Mishra, N. Analysis of different similarity measure functions and their impacts on shared nearest neighbor clustering approach. *International Journal of Computer Applications* **40**, 1–5 (2012).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In: *Proc. 26th Int. Conf. on Machine Learning (ICML)*. 865–872 (2009).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* **11**, 2487–2531 (2010a).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. On the existence of obstinate results in vector space models. In: *Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*. 186–193 (2010b).

Schedl M., F. A. A mirex meta-analysis of hubness in audio music similarity. In: *Proceedings of the 13th International Society for Music Information Retrieval Conference*. ISMIR'12 (2012).

Schnitzer, D.; Flexer, A.; Schedl, M.; Widmer, G. Using mutual proximity to improve content-based audio similarity. In: *ISMIR'11*. 79–84 (2011).

Scott, D.; Thompson, J. Probability density estimation in higher dimensions. In: *Proceedings of the Fifteenth Symposium on the Interface*. 173–179 (1983).

Song, Y.; Huang, J.; Zhou, D.; Zha, H.; Giles, C. L. Iknn: Informative k-nearest neighbor pattern classification. In: *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*. 248–264, PKDD 2007 (Springer-Verlag, Berlin, Heidelberg, 2007).

Tan, P.-N.; Steinbach, M.; Kumar, V. *Introduction to Data Mining* (Addison Wesley, 2005).

Tan, S. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Syst. Appl.* **28**, 667–671 (2005).

T.M.Cover; P.E.Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* **IT-13**, 21–27 (1967).

Tomašev, N.; Brehar, R.; Mladenić, D.; Nedevschi, S. The influence of hubness on nearest-neighbor methods in object recognition. In: *Proceedings of the 7th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. 367–374 (2011a).

Tomašev, N.; Mladenić, D. Exploring the hubness-related properties of oceanographic sensor data. In: *Proceedings of the SiKDD conference* (2011).

Tomašev, N.; Mladenić, D. The influence of weighting the k-occurrences on hubness-aware classification methods. In: *Proceedings of the SiKDD conference* (Institut "Jozef Stefan", Ljubljana, 2011a).

Tomašev, N.; Mladenić, D. Nearest neighbor voting in high-dimensional data: Learning from past occurrences. In: *ICDM PhD Forum* (2011b).

Tomašev, N.; Mladenić, D. Hubness-aware shared neighbor distances for high-dimensional k-nearest neighbor classification. In: *Proceedings of the 7th International Conference on Hybrid Artificial Intelligence Systems*. HAIS '12 (2012a).

Tomašev, N.; Mladenić, D. Nearest neighbor voting in high dimensional data: Learning from past occurrences. *Computer Science and Information Systems* **9**, 691–712 (2012b).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. In: *Proc. MLDM* (2011b).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. A probabilistic approach to nearest neighbor classification: Naive hubness bayesian k-nearest neighbor. In: *Proceeding of the CIKM conference* (2011c).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. In: *PAKDD (1)'11*. 183–195 (2011d).

Triguero, I.; García, S.; Herrera, F. Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification. *Pattern Recogn.* **44**, 901–916 (2011).

van den Bosch, A.; Weijters, T.; Herik, H. J. V. D.; Daelemans, W. *When small disjuncts abound, try lazy learning: A case study* (1997).

Van Hulse, J.; Khoshgoftaar, T. Knowledge discovery from imbalanced and noisy data. *Data and Knowledge Engineering* **68**, 1513–1542 (2009).

Wang, S.; Li, X.; Xia, J.-F.; Zhang, X.-P. Weighted neighborhood classifier for the classification of imbalanced tumor dataset. *Journal of Circuits, Systems, and Computers* 259–273 (2010).

Xing, Z.; Pei, J.; Yu, P. S. Early prediction on time series: a nearest neighbor approach. In: *Proceedings of the 21st international jont conference on Artifical intelligence*. 1297–1302, IJCAI'09 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2009).

Yin, J.; Fan, X.; Chen, Y.; Ren, J. High-dimensional shared nearest neighbor clustering algorithm. In: *Fuzzy Systems and Knowledge Discovery*. **3614**, 484–484, 484–484, 484–484 (Springer Berlin / Heidelberg, 2005).

Zhang, H.; Berg, A. C.; Maire, M.; Malik, J. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*. 2126–2136, CVPR '06 (IEEE Computer Society, Washington, DC, USA, 2006).

Zhang, Z.; Zhang, R. *Multimedia Data Mining: a Systematic Introduction to Concepts and Theory* (Chapman and Hall, 2008).

Zheng, L.-Z.; Huang, D.-C. Outlier detection and semi-supervised clustering algorithm based on shared nearest neighbors. *Computer Systems and Applications* **29**, 117–121 (2012).

3 Practical Applications of the Hubness-aware Methods

Many frequently encountered types of data exhibit non-negligible hubness. Therefore, the hubness phenomenon is potentially of high interest in various practical applications of machine learning and data mining. Unlike the previous text, this chapter presents a data-driven perspective on the role of hubs in high-dimensional data analysis on four rather different data types: images, sensor data, documents and bug duplicate reports.

We will begin by considering hubs in image data (Section 3.1). We will look at how different feature representations affect the overall neighbor occurrence distribution and proceed by showing how these insights might help in practical image retrieval and object recognition systems.

Vast quantities of data are being collected by sensors nowadays and are being aggregated and assessed by various automatic or semi-automatic analytical systems. We will show that hubs may also arise in this type of data and propose a simple semi-automatic anomaly detection sub-system for oceanographic sensor data in Section 3.2.

Textual data is available in many different languages and cross-lingual information retrieval allows for a unified approach to querying multi-lingual document collections. In Section 3.3, we will examine the correlation between the neighbor occurrence distribution in different languages, as well as the consequences of using the canonical correlation analysis (CCA) for projecting data onto a common semantic space. We will propose a simple weighting scheme that improves the performance of cross-lingual document retrieval systems that are based on CCA.

Bug resolution is an important task in software engineering. Prior to assigning a bug report to a responsible engineer, one needs to determine if the same issue had already been reported in the past. Given that there is a large number of issues being reported and that the users phrase their complaints and/or observations in very different ways, bug report duplicate detection is a difficult problem that has recently received some attention. In Section 3.4, we will show that this data exhibits substantial hubness and that the temporal hubness information can be used for secondary re-ranking of the neighbor sets, which results in the actual duplicates being ranked higher and becoming easier to notice by the users of the system.

3.1 Image Data

Object detection and recognition in images has many practical applications. It is one of the essential tasks in face recognition, optical character recognition, robotics, automated vehicle control, object tracking and content-based image indexing.

3.1.1 Feature Representations

Raw images first need to be assigned a proper feature representation. Different feature types might be appropriate for different analytical tasks. This usually involves extracting infor-

mation from color, texture, edges, or any property which we feel might be worth capturing. All such features can be divided in two groups - local and global image features. Global image features are used to summarize the average properties of an observed segment, while the local features represent information extracted from neighborhoods of some sampling points [Zhang and Zhang, 2008]. These points may simply define a grid over an image, but local features can also be computed at some *informative* points, according to some optimization criterion, as is often the case with extracting SIFT features [Lowe, 1999]. There are many different types of local image features and we have decided to limit our discussion to SIFT, Histogram of oriented features and Haar features, for practical reasons.

Our hypothesis was that different image feature representations may lead to different k-nearest neighbor topologies, so that the image dataset exhibits different degrees of hubness when different types of features are used. To test our hypothesis, we have examined several different types of local features: SIFT, Histogram of oriented gradient features and Haar filters. In some cases, we have also used the color information in form of global color histograms.

The experiments have been performed on both the quantized and the non-quantized feature representations, for each examined feature type. The process of forming a quantized feature representation is given in more detail in Section 3.1.1.4.

3.1.1.1 Haar Features

Haar filters, derived from Haar wavelets, have been introduced by [Papageorgiou et al., 1998] that used the wavelet representation for recognition of two classes of objects: faces and persons. The representation has been exploited and extended by [Jones and Viola, 2001] and [Lienhart and Maydt, 2002] that completed the set of features and improved the detection algorithm.

Haar filters operate on gray level images and their value is represented by the difference of sums computed over rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent. Two types of features have been used: two-rectangular and three-rectangular as shown in Figure 37:

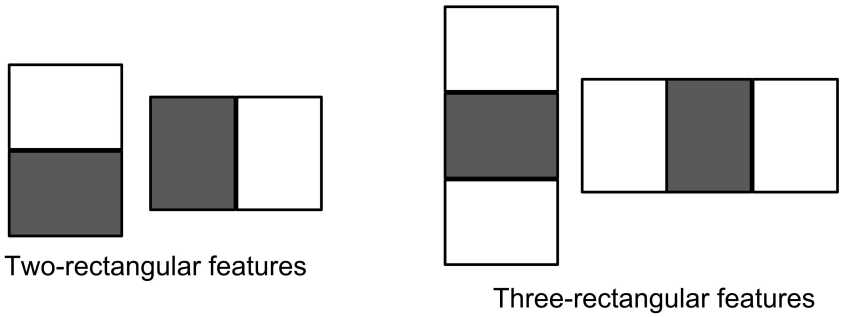


Figure 37: Example of Haar filters

The value of a two-rectangular Haar feature is the difference between the sums of the pixels within two rectangular regions (the white and black regions in Figure 37). The regions have the same size and shape and are horizontally or vertically adjacent. A three-rectangular feature computes the sum within two outside rectangles subtracted from the sum in center rectangle. The integral image representation has been used for a fast computation of feature values [Zhang and Zhang, 2008].

3.1.1.2 Histogram of Oriented Gradient Features

The histograms of oriented gradients (HoG) were used successfully for pedestrian classification in [Triggs and Dalal, 2005] and extended to generic object recognition [Felzenszwalb et

al., 2010].

The process of constructing the histograms of oriented gradients comprises the following steps:

- gradient computation
- spatial/orientation binning
- normalization and descriptor blocks

Figure 38 represents the above mentioned steps. The same feature parameters were used as

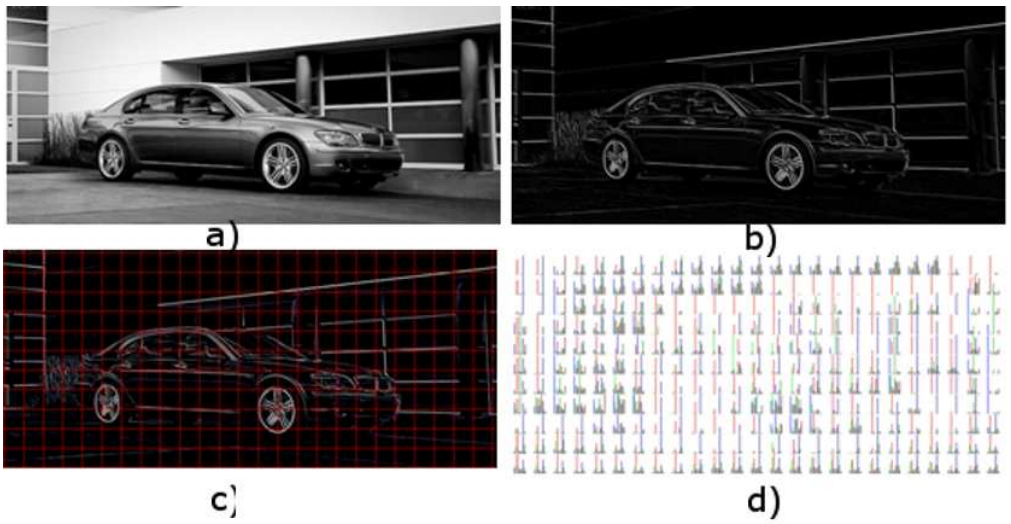


Figure 38: Basic steps in Hog Extraction a) original image; b) gradient computation c) spatial orientation binning and histogram computation within each cell; d) normalization and descriptor values

in [Triggs and Dalal, 2005].

3.1.1.3 SIFT Features

The scale invariant feature transform (SIFT) approach for object recognition or scene matching has been introduced by [Lowe, 1999] and extended by [Lowe, 2004] that applied a fast nearest-neighbor algorithm to invariant keypoints and obtained a robust object recognition scheme. It is one of the most widely used approaches in practice. The main steps of the algorithm comprise:

- Difference of Gaussians used to find candidate interest points that have the property of being invariant to scale and orientation.
- Stability measure of candidate interest point and selection of keypoint locations.
- Rotation invariant representation by representing each keypoint descriptor relative to a consistent orientation.
- Final keypoint representation, considering shape distortion and change in illumination invariance.

3.1.1.4 Quantized Feature Representations

Given a set of extracted local features, one needs a way of measuring similarity between pairs of images, as this is essential in many practical applications. One conceivable way of

doing this would be to calculate the maximum, minimum or average distance between pairs of individual local features from each image. If we're dealing with large images with many features, this would be unfeasible and not a good measure of similarity. This is why the quantized representations are most often used in practice [Zhang and Zhang, 2008].

The quantized feature representation for images is in principle similar to what the *bag of words* representation means in document analysis [Han, 2005]. Not surprisingly, local image features are sometimes referred to as *visual words*.

The process of forming the quantized representation is roughly as follows:

- Form a stratified sample of local features from the image collection.
- Perform clustering on the feature sample.
- Select the centroid of each feature cluster and add it to the *codebook* of representative features.
- For each image in the collection and for each local feature, assign the feature to the closest representative feature from the codebook, i.e. to the closest cluster of visual words.
- Count the codebook assignments for each image and form a histogram of codebook frequencies
- Normalize the codebook histogram, if necessary. TF/IDF is an option, similar to handling textual data.

All quantized representation for a given codebook are of the same length and can thus be easily compared. Several types of metrics can be used, including the Manhattan metric, the Euclidean distance, fractional distances, cosine or even some form of Kullback-Leibler divergence, if the histograms are properly normalized first to form probability distributions.

The quantized representations for different feature types can be combined, as well as extended by color histograms or correlograms. Dimensionality reduction via PCA/SVD [Jolliffe, 2002] or random projections [Bingham and Mannila, 2001] is also possible.

The dimensionality of the codebook representation is a parameter which is sometimes difficult to guess in advance. For a given type of features and a given image collection, the best would be to run a small set of preliminary experiments in order to detect which codebook size works best in the desired context.

3.1.2 Object Recognition in Images

The k-nearest neighbor approach is not uncommon in image retrieval systems, as one of the most typical tasks is usually image search, which returns the "top k" most similar images from the database. Regardless of the algorithm that produces the ranking, this can be viewed as the k-nearest neighbor search.

In object detection and recognition, nearest neighbor methods have recently been shown to be competitive with other state-of-the-art approaches, like support vector machines [Boiman et al., 2008][Wang et al., 2010a][Gu et al., 2011].

We have tested the proposed hubness-aware k-nearest neighbor classification methods and compared them to the baseline kNN, on several different datasets for different types of quantized and non-quantized feature representations.

The overview of datasets and their properties is given in Table 20. The datasets have been split into several groups. Most notably, datasets 1-12 represent binary classification problems, while datasets 13-27 contain multiple categories. Some of the binary datasets are balanced, so that each class comprises 50% of the data. The imbalanced binary datasets are given with the suffix *-imb*. All the given multi-class datasets are unbalanced. Some

Table 20: Summary of data sets. NN-related parameters are given for $k = 5$ and $k = 10$. S_{N_k} denotes the skewness of $N_k(x)$. Mean entropy of neighbor sets is given by H_{N_k} and mean entropy of inverse neighbor sets by H_{IN_k}.$\max N_k$ denotes the highest hubness achieved by a single data point on the data set, which is in fact the maximal node degree in the kNN graph obtained from the data.

Data set	features	size	d	C	S_{N_5}	$S_{N_{10}}$	$\max N_5$	$\max N_{10}$	H_{N_5}	$H_{N_{10}}$	H_{IN_5}	$H_{IN_{10}}$
ds1: nicta	HoG	15891	2268	2	21.68	15.45	844	1215	0.13	0.17	0.07	0.10
ds2: nicta 16x40	HoG	8000	972	2	3.92	3.42	106	160	0.13	0.17	0.11	0.18
ds3: nicta-imb 64x80	HoG	9000	2268	2	10.72	8.30	440	628	0.05	0.06	0.05	0.08
ds4: nicta-imb-small 64x80	HoG	500	2268	2	3.84	3.66	69	122	0.12	0.16	0.13	0.20
ds5: daimler	HoG	9804	1440	2	4.87	4.55	88	166	0.07	0.12	0.09	0.16
ds6: nicta 16x40	Haar	8000	11920	2	1.64	1.60	38	67	0.24	0.29	0.20	0.29
ds7: nicta-imb 64x80	Haar	9000	1881	2	3.05	2.89	95	183	0.04	0.03	0.11	0.17
ds8: nicta-imb-small 64x80	Haar	500	1881	2	2.99	2.54	59	94	0.03	0.03	0.18	0.24
ds9: daimler	Haar	9804	12555	2	2.14	2.21	44	89	0.07	0.12	0.10	0.16
ds10: nicta 64x80	SIFT	8000	4480	2	10.96	8.94	336	512	0.12	0.16	0.09	0.14
ds11: nicta-imb 64x80	SIFT	9000	4480	2	10.72	8.45	376	577	0.05	0.07	0.05	0.09
ds12: nicta-imb-small 64x80	SIFT	500	4480	2	3.89	3.89	71	114	0.14	0.19	0.13	0.22
ds13: caltech-6	Haar-bow	280	650	6	2.79	2.15	40	61	0.07	0.17	0.03	0.07
ds14: ImgNet-s3	Haar-bow	2731	100	3	2.10	1.83	50	72	0.01	0.01	0.01	0.01
ds15: ImgNet-s4	Haar-bow	6054	100	4	1.94	1.75	45	70	0.01	0.01	0.01	0.01
ds16: ImgNet-s5	Haar-bow	6555	100	5	2.06	1.94	45	78	0.01	0.01	0.01	0.01
ds17: ImgNet-s6	Haar-bow	6010	100	6	1.79	1.58	51	79	0.01	0.01	0.1	0.1
ds18: ImgNet-s7	Haar-bow	10544	100	7	2.29	2.03	60	108	0.01	0.01	0.01	0.01
ds19: ImgNet-s3Er	Haar-bow	2731	100	3	20.56	15.67	375	560	0.05	0.20	0.01	0.01
ds20: ImgNet-s3Er	Haar-bow	2731	150	3	25.1	17.83	1280	1683	0.22	0.61	0.01	0.01
ds21: ImgNet-s3Er	Haar-bow	2731	1000	3	23.3	15.67	2363	2426	0.03	0.87	0.01	0.01
ds22: caltech-6	SIFT-bow	280	30	6	7.25	6.20	90	166	0.74	1.12	0.45	0.76
ds23: ImgNet-s3	SIFT-bow+cH	2731	416	3	8.38	6.19	213	294	0.40	0.50	0.23	0.34
ds24: ImgNet-s4	SIFT-bow+cH	6054	416	4	7.69	6.32	204	311	0.77	0.95	0.45	0.65
ds25: ImgNet-s5	SIFT-bow+cH	6555	416	5	14.72	11.88	469	691	0.85	1.05	0.42	0.63
ds26: ImgNet-s6	SIFT-bow+cH	6010	416	6	8.42	6.23	275	384	0.87	1.10	0.48	0.70
ds27: ImgNet-s7	SIFT-bow+cH	10544	416	7	7.65	6.72	268	450	0.86	1.12	0.54	0.80

of the datasets refer to the same image collections, but they represent different feature representations. This is the case with {ds4, ds8, ds12}, {ds3, ds7, ds11}, {ds5, ds9}, {ds2, ds6}, {ds13, ds22}, {ds14, ds19, ds20, ds21, ds23}, {ds15, ds24}, {ds16, ds25}, {ds17, ds26}, {ds18, ds27}.

Several different feature representations were used. Datasets 1-5 are HoG feature representations, datasets 6-9 Haar feature representations, datasets 10-12 are SIFT feature representations. The binary data was represented in a non-quantized way. Datasets 13-27 are quantized representations, 13-21 of Haar features and 22-27 of SIFT features. Datasets 23-27 are hybrid feature representations, where the first 400 dimensions refer to the visual word frequencies and the last 16 features denote a color histogram, calculated globally for each image. All distances between images were calculated by the Manhattan distance (the sum of absolute differences). Also, all features were normalized to the $[0,1]$ range.

The distribution of k-occurrences seems to exhibit high skewness in all cases. Therefore, the hubness phenomenon is present in image data, regardless of the choice of feature representation. As images are inherently high-dimensional, this is not surprising. Specifically, on ds1 the main hub occurs in 10-neighbor sets 1215 times, which greatly exceeds the expected value of 10. Whether such a hub acts as a good hub or a bad hub depends on the context. Both scenarios are possible. Such bad hubs can easily compromise the k-nearest neighbor classification.

If we compare the Haar feature representations ds6-9 with their respective HoG or SIFT representations, we see that the hubness is much less pronounced for Haar representations in

all cases. This suggests that Haar-based image representations might be in general less susceptible to the $N_k(x)$ skew. The comparison between Haar and Hog k-occurrence distribution skewness is given in Figure 39.

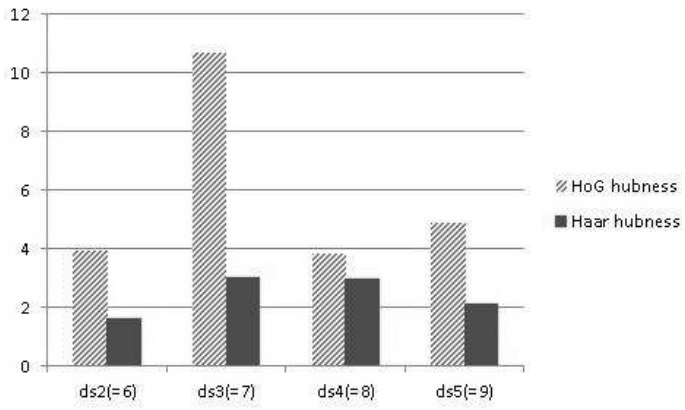


Figure 39: Comparison between the N_5 distribution skewness between HoG and Haar feature representations on several datasets.

The direct and reverse neighbor set entropies (H_{N_k} and H_{IN_k}), as given in Table 20, reflect the uncertainty when basing the classification on the direct or reverse neighbor sets. The hubness-aware classification methods are based on the latter. Figure 40 shows that the direct 5-neighbor sets are much less homogenous than the reverse neighbor sets, for the 6 SIFT quantized feature representations. This strongly implies that taking hubness into account might lead to better results. The experimental evaluation of the hubness-aware classifiers on these datasets confirms these initial observations.

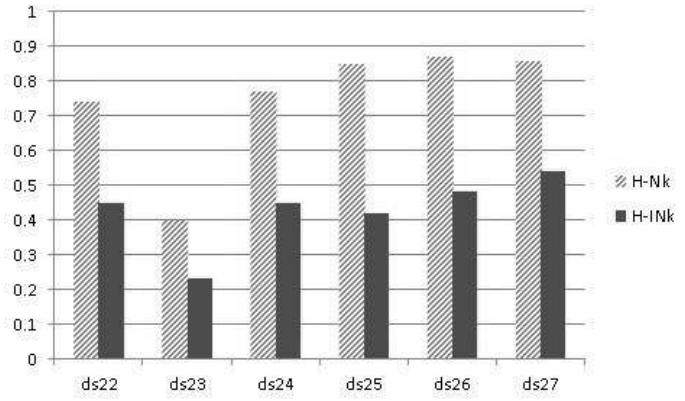


Figure 40: Comparison between H_{N_5} and H_{IN_5} on several quantized multi-class datasets.

kNN, hw-kNN, h-FNN and HIKNN were evaluated on all 27 image datasets/representations via 10-times 10-fold cross-validation for a fixed neighborhood size of $k = 5$. Using an odd number made ties impossible in the binary classification case. Corrected re-sampled t-test was used to check for statistical significance. The results are given in Table 21. The threshold value in h-FNN was set to zero, so no global or local estimate was used, the fuzziness was obtained in all cases directly from previous k-occurrences. The Manhattan distance was used to calculate all image-to-image distances, after previous feature normalization.

Using the hubness based methods often improves the classification results, in cases where such an improvement is possible. Such improvements are more often present in those representations where the entropy of the direct k-neighbor sets dominates the entropy of the reverse neighbor sets, as previously discussed.

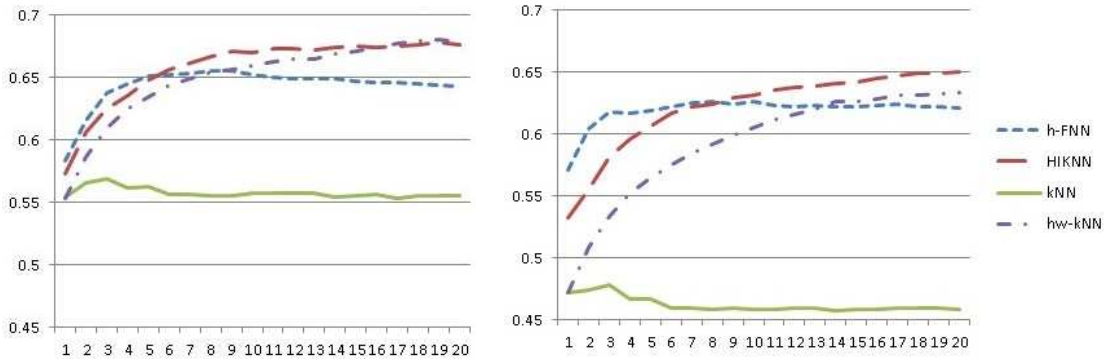
Some improvement has been detected even in the 'simple' binary cases, as can be seen in

Table 21: Classification accuracy of kNN, hubness-weighted kNN (hw-kNN), hubness-based fuzzy nearest neighbor (h-FNN) and hubness information k-nearest neighbor (HIKNN). The symbols •/◦ denote statistically significant better/worse performance ($p < 0.01$) compared to kNN.

Data set	kNN			hw-kNN				h-FNN				HIKNN			
ds1	86.1	±	0.7	89.0	±	0.8	•	94.7	±	0.6	•	93.1	±	0.6	•
ds2	94.7	±	0.7	95.9	±	0.5	•	95.6	±	0.6	•	95.8	±	0.7	•
ds3	91.7	±	0.8	92.5	±	0.6		92.6	±	0.7		92.1	±	0.8	
ds4	91.5	±	0.4	93.5	±	0.3	•	92.6	±	3.6		92.9	±	3.9	
ds5	98.2	±	0.3	98.1	±	0.4		97.8	±	0.4		97.8	±	0.4	
ds6	84.9	±	1.2	88.3	±	1.2	•	87.3	±	1.2	•	86.9	±	1.1	•
ds7	91.7	±	0.8	92.5	±	0.7		92.6	±	0.7		92.1	±	0.9	
ds8	81.3	±	5.4	80.8	±	5.0		80.6	±	4.8		81.8	±	5.1	
ds9	97.4	±	0.5	97.0	±	0.6		97.5	±	0.5		97.5	±	0.5	
ds10	94.2	±	0.8	96.2	±	0.8	•	95.8	±	0.8	•	95.6	±	0.6	•
ds11	97.2	±	0.5	98.1	±	0.4	•	97.6	±	0.5		97.4	±	0.5	
ds12	91.9	±	0.4	94.6	±	0.4	•	94.1	±	0.3	•	94.6	±	0.3	•
ds13	95.7	±	3.8	97.6	±	2.9		95.7	±	3.6		96.8	±	4.7	
ds14	99.7	±	0.01	99.7	±	0.01		99.7	±	0.01		99.7	±	0.01	
ds15	99.9	±	0.01	99.9	±	0.01		99.9	±	0.01		99.9	±	0.01	
ds16	99.7	±	0.01	99.9	±	0.01		99.9	±	0.01		99.9	±	0.01	
ds17	99.6	±	0.01	99.7	±	0.01		99.7	±	0.01		99.7	±	0.01	
ds18	99.8	±	0.01	99.9	±	0.01		99.9	±	0.01		99.9	±	0.01	
ds19	92.4	±	0.02	93.6	±	0.01		97.5	±	0.01	•	97.6	±	0.01	•
ds20	80.0	±	0.02	88.7	±	0.02	•	94.6	±	0.01	•	94.8	±	0.01	•
ds21	21.2	±	0.02	27.1	±	0.11		59.5	±	0.03	•	59.6	±	0.03	•
ds22	73.0	±	8.8	82.1	±	6.8	•	82.7	±	6.6	•	85.0	±	6.2	•
ds23	72.0	±	2.7	80.8	±	2.3	•	82.4	±	2.2	•	82.2	±	2.0	•
ds24	56.2	±	2.0	63.3	±	1.9	•	65.2	±	1.7	•	64.7	±	1.9	•
ds25	46.6	±	2.0	56.3	±	1.7	•	61.9	±	1.7	•	60.8	±	1.9	•
ds26	60.1	±	2.2	68.1	±	1.6	•	69.3	±	1.7	•	69.9	±	1.9	•
ds27	43.4	±	1.7	55.1	±	1.5	•	59.2	±	1.5	•	56.9	±	1.6	•
AVG	83.34			86.23				88.36				88.34			

the first half of the table. The best results there were obtained by using the simplest hubness-aware approach, hw-kNN. This is not surprising, as the other hubness-aware classifiers that are based on modeling the class-specific occurrence profiles offer more in multi-class problems. However, the most tangible improvement in the binary case was seen on ds1, where the 8.6% improvement was achieved by h-FNN. This is interesting, since this particular dataset exhibits the highes skewness in the k-occurrence distribution, as seen in Table 20.

Even though all tests were performed for $k = 5$, the results are by no means an artefact of the selected neighborhood size. For two multi-class datasets we have performed 10-times 10-fold cross-validation for an entire range of neighborhood sizes $k \in \{1, 2, .., 20\}$, as shown in Figure 41. All three hubness-based algorithms remain dominant when compared to the basic kNN throughout the k-range in both datasets. It seems that on these two datasets, HIKNN algorithm performs best. For the lower range of k-values, h-FNN achieves slightly higher accuracies, but this trend is completely reversed for $k > 5$ and the best overall accuracies on both datasets are reached for higher k-values. Similarly, hw-kNN trails behind h-FNN and HIKNN in small-to-medium neighborhood sizes, but it slowly improves and becomes quite competitive for larger k-values.



(a) ds24: ImgNet-s4 (b) ds25: ImgNet-s5

Figure 41: Accuracy of h-FNN, HIKNN, kNN and hw-kNN for different k-values, obtained by 10-times 10-fold cross-validation.

Representations ds19-21 are all quantized Haar representations of ImgNet-s3 data. This is where the most convincing improvement was observed. A closer look at the data reveals that major bad hubs appeared as a result of a preprocessing error that occurred during the feature extraction. Under the chosen metric and normalization, zero vectors tend to become on average closer to all other points as the dimensionality is increased, which results in them becoming major hubs. The images for which an I/O exception occurred were unintentionally left with empty representations, i.e. they were zero vectors and have hence emerged as hubs. This leads to a dramatic decrease in accuracy of the k-nearest neighbor classifier. For a 1000-dimensional representation, kNN was worse than Zero-rule, which would assign each instance to the majority class. Figure 42 shows the major bad hubs in the data.

This is quite a remarkable thing. Only five erroneous instances were enough to render 5-NN classifier totally useless in a very high dimensional setting - due to their high hubness. In a 100-dimensional case, kNN was still at 93.6% accuracy, but already slightly affected, as can be seen by comparing it to h-FNN and HIKNN. Also, this scenario marks a principal difference between hw-kNN and the other two hubness-based classifiers. hw-kNN only weights down the bad hubs, but they still vote by their label. In cases where an entire k-neighborhood is compromised, it is powerless. h-FNN and HIKNN are therefore, more robust.

All this might seem somewhat irrelevant, since we are basically discussing erroneous data, if only very slightly erroneous. However, it is well known that most data in real world

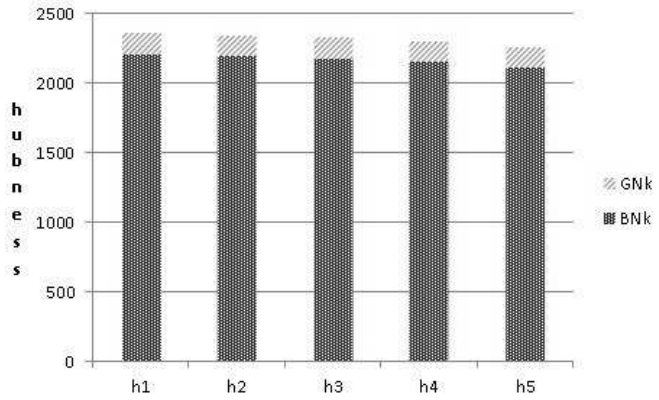


Figure 42: The five most pronounced hubs in ds21 representation of ImgNet-s3 data. Bad hubness (BN_k) completely dominates good hubness (GN_k).

applications contain much higher noise levels and inaccuracies. The erroneous instances do not need to be zero-arrays as in this case. All it takes for a problem to appear is that for a chosen representation and distance metric such instances exhibit a high enough hubness. Unless the representation is very high dimensional with a high enough hubness, this problem may even pass unnoticed, as would have been the case for ds-19, if we had not also generated ds-20 and ds-21 representations where the error became apparent.

Several important points stem from this discussion. The phenomenon of hubness is by itself not necessarily detrimental, as witnessed on several datasets where kNN reaches a very high accuracy, close to 100%. Yet, a combination of class overlap, imbalance and high hubness is in fact quite difficult to handle, as some very bad hubs may emerge and negatively affect the k-nearest neighbor classification.

3.1.3 Visualizing the k-Nearest Neighbor Topology

In practical applications, machine learning practitioners need to chose from among a great variety of feature representations, metrics, data preprocessing techniques, normalizations and learning/ranking/retrieval algorithms and models. This choice is non-trivial and it is not always clear exactly how to proceed. Some methods might be considered state-of-the-art for certain types of tasks, but the performance is usually data dependent and even good/robust algorithms perform worse than expected in some specific cases. The approach usually consists of trial and error, where different techniques are implemented and tested on the actual data that the system will have to handle.

Evaluating the system performance based on a few aggregate measures does not reveal much. In case of classification this might be the accuracy or the F-score, in case of clustering the Silhouette coefficient, etc. These numbers may be used to quickly compare different algorithms and see which one is better. However, they are not well suited for a detailed analysis of a single algorithm's performance. Knowing in which cases an algorithm performs well and in which cases it fails is often essential. Such detailed characterization might allow us either to modify and improve the existing algorithms or to carefully combine them and use different approaches for different types of examples.

In order to explore and analyze the advantages and disadvantages of various approaches for handling image data, we have assembled an application that would help the practitioners in choosing the metrics, methods and feature representations when building larger systems. Additionally, it would allow the researchers deeper insight into the workings of their methods and how it all relates to the phenomenon of hubness. We have named the first version of the software Image Hub Explorer [N. and D., 2013]. It is built on top of the Hub Miner java data mining library (http://ailab.ijs.si/nenad_tomasev/hub-miner-library/) that was

developed during the course of working towards this thesis.

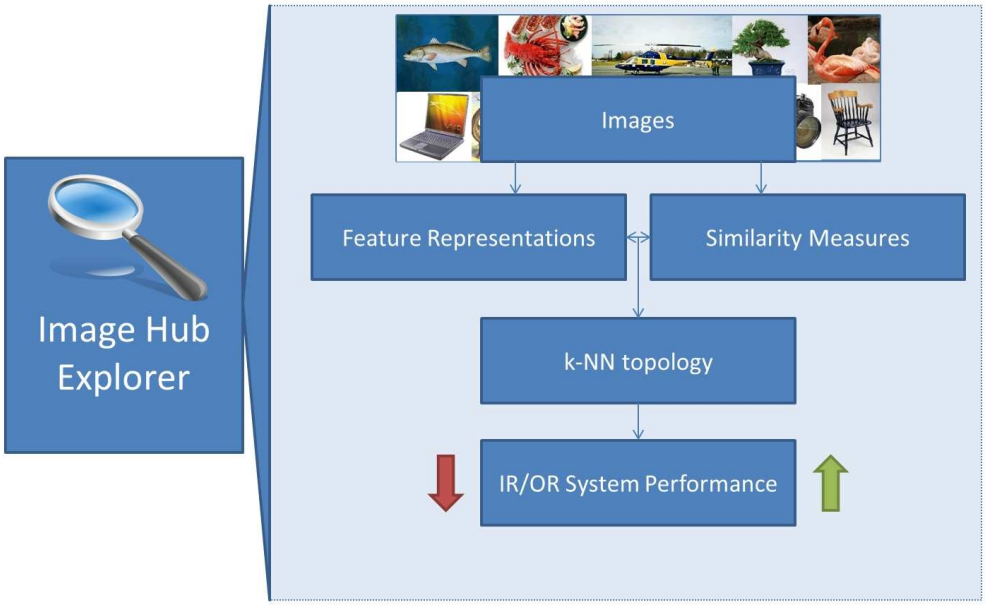


Figure 43: The motivation behind the Image Hub Explorer system.

The examples that we will examine have been generated by analyzing the Leeds Butterfly dataset [Wang et al., 2009] (http://www.comp.leeds.ac.uk/scs6jwks/dataset/leedsbutterfly/).

3.1.3.1 Image Hub Explorer Interfaces

Image Hub Explorer offers several different views of the data, organized in 4 tabs and several drop-down menus. We will go through each of them separately and explain the typical use cases. Several primary and secondary metrics are supported and this will be discussed in more detail after the specification of interfaces and main functionalities.

The primary tabular view is contained in the Data Overview tab. It contains a data visualization component, as well as many data properties that are relevant for the hubness-related analysis. This is shown in Figure 44.

The data is projected onto a 2D plane via multi-dimensional scaling (MDS) [Borg and Groenen, 2005]. It is a dimensionality reduction technique that tries to preserve the ratios between the distances between different data points. In other words, points close in the original feature space tend to be projected in such a way that they remain close in the plane. The MDS panel is interactive and it is possible to select images by clicking on their thumbnails. Each image thumbnail is contained in a slightly larger rectangle that shows the color of the respective image category. This allows for easy visualization of labeled data. As we might be working with some very large datasets, not all the images are shown, but rather a fixed number of hub-images, frequent nearest neighbors.

The background of the MDS panel is colored according to good/bad hubness of the projected points. Naturally, the green color corresponds to good hubness and the red one to bad hubness. The landscape is generated in two steps. The first step is a sort of a Gaussian blur, implemented efficiently, as in [Fortuna et al., 2005]. The panel is split into buckets by a grid and each image is assigned to its bucket according to the (x,y) coordinates obtained by applying MDS. Each pixel is assigned its good hubness weight $w_{G,k}$ and bad hubness weight $w_{B,k}$. Within each bucket B, the weight of each pixel is determined by the following rule: $w_{G,k}(x,y) = \sum_{I_j \in B} GN_k(I_j) \cdot e^{-\sigma((x-x_j)^2+(y-y_j)^2)}$ and $w_{B,k}(x,y) = \sum_{I_j \in B} BN_k(I_j) \cdot e^{-\sigma((x-x_j)^2+(y-y_j)^2)}$. If either of the two weights are non-zero for a given pixel (i.e. the bucket contains some points), the green component of the RGB representation of the color in the pixel is given by

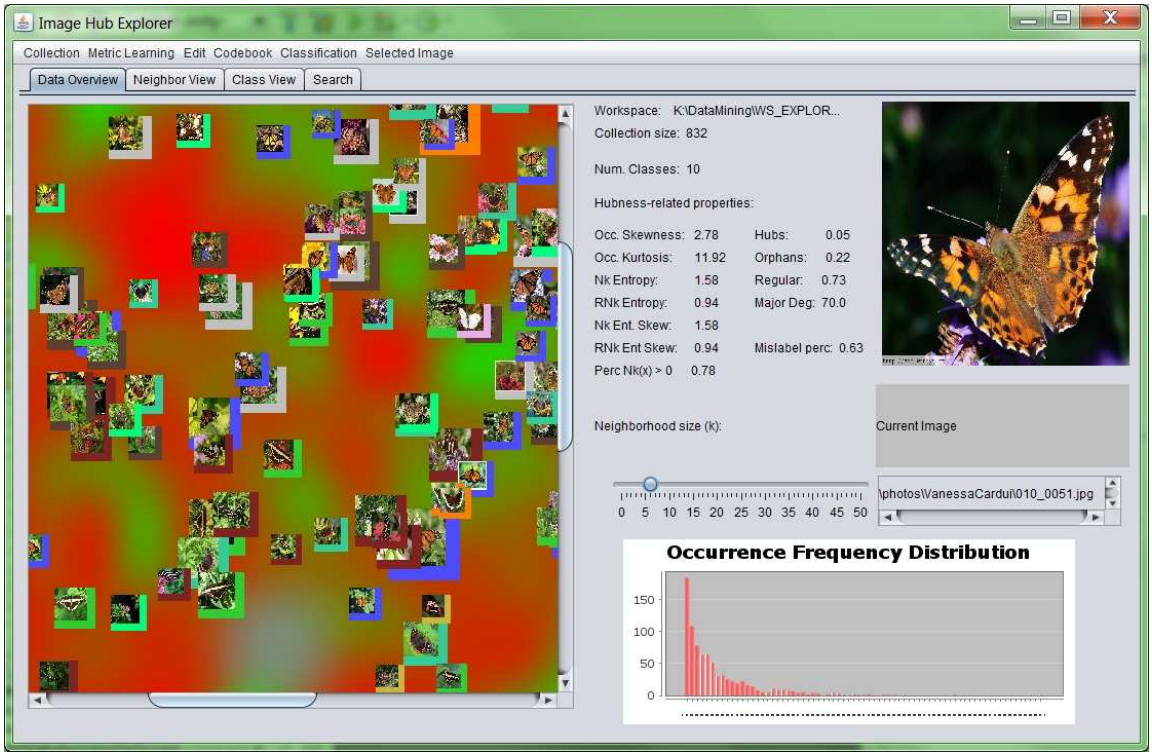


Figure 44: The Data Overview screen of Image Hub Explorer

$g(x,y) = 255 \cdot \frac{w_{G,k}(x,y)}{w_{G,k}(x,y)+w_{B,k}(x,y)}$ and the red component as its complement $r(x,y) = 255 - g(x,y)$. After this initial stage, a two pass box blur is performed in order to further soften the landscape. Box blur sets the color of each pixel in the image to be the color of its neighboring pixels. It is a low pass convolution filter.

One such landscape is generated for each neighborhood size k, as it depends on good and bad hubness that are k-dependent quantities. One of the main features of the application is the slider-selector for neighborhood size, which allows the user to quickly change among different k-values and observe the differences in all quantities and all tabular views of the application.

The quantities that are shown on the Data Overview tab are as follows: data size, the number of classes, neighbor occurrence frequency distribution skewness (hubness), neighbor occurrence frequency distribution kurtosis, entropy of the direct and reverse neighbor sets, skewness of the entropy distribution, percentage of points that occur at least once as neighbors, percentages of hubs, orphans and regular points, degree of the major hub in the data and the percentage of label mismatches in k-neighbor sets (bad hubness). The neighbor occurrence frequency distribution is also given in a separate plot below for easier interpretation.

Whereas the Data Overview tab gives some important properties of the occurrence distribution, the Neighbor View tab allows the user to pinpoint some critical subsets of points that might exhibit interesting behavior. A screenshot is given in Figure 45. All displays in all the tabs and views support image selection, so there are various ways in which a user can select a certain image. Once selected, there is an option to add the image to the selected subgraph that is currently being inspected. Apart from adding images one by one, the interface also supports an option of adding all neighbors of any selected image, as well as all of its reverse nearest neighbors. The graph panel displays the selected image subset and a directed edge is inserted between individual nodes to denote a neighbor relation. The weights on the edges correspond to the distance between the selected points in the selected metric.

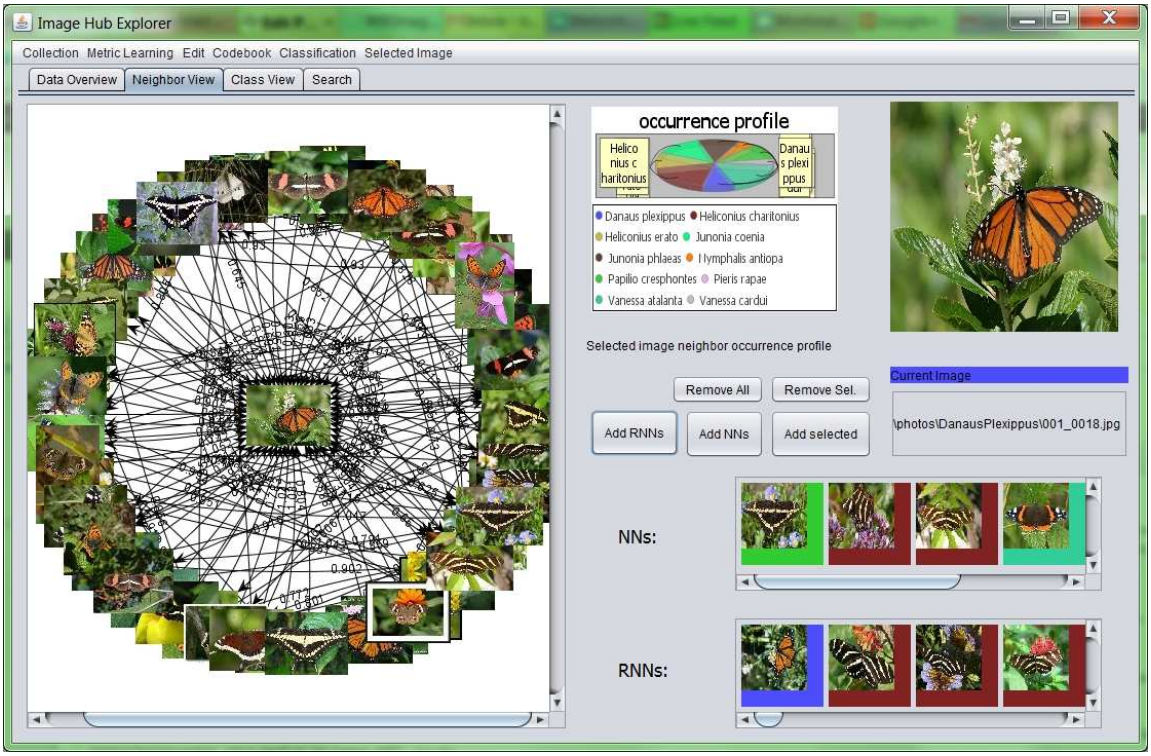


Figure 45: The Neighbor View screen of Image Hub Explorer, showing the selected subgraph of the kNN graph.

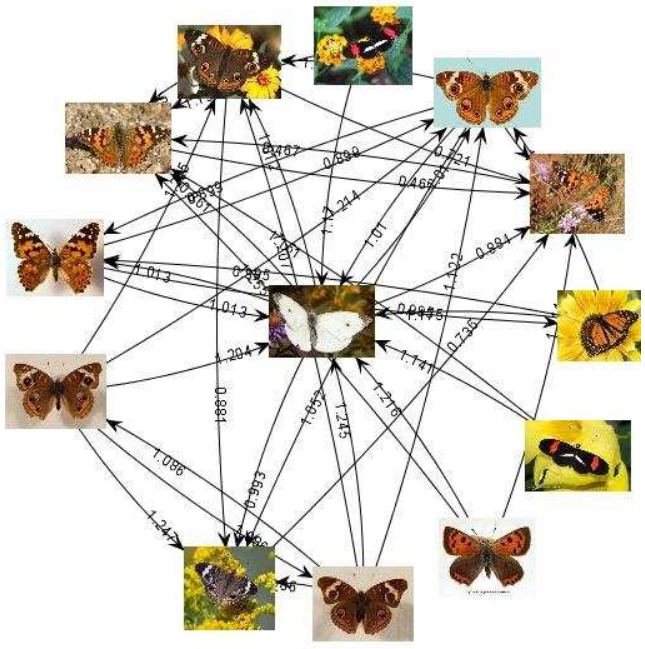


Figure 46: An example of a bad hub shown in the Neighbor View of Image Hub Explorer. We can see that its reverse neighbors originate from different classes.

In the Neighbor View, apart from the graph panel, there is a possibility of examining the neighbor occurrence profile of the selected image. The pie chart showing the class hubness tendencies of the selected neighbor point is shown in the upper part. The list of its nearest neighbors and reverse nearest neighbors is shown below.

Let us say that a user wants to examine a profile of one of the bad hubs in the data. This way, the user can select each reverse neighbor individually and check if there is a label mismatch and if it causes misclassification. The user can also interactively change the neighborhood size and see for which k-values the two points remain connected.

The Class View shown in Figure 47 offers an insight into a hub-structure of each class, as well as the interplay between hubs in different classes, which is summarized in the class-to-class hubness matrix on the lower right side. The main set of panels in this view is contained in a scroll pane and shows an ordered list of major hubs, good hubs and bad hubs for each class separately. As before, they are selectable. Additionaly, there is a point type distribution, where the points are labeled either as safe, borderline, rare or outliers, based on the percentage of label mismatches in the respective k-neighbor sets. The chart in the upper part shows a distribution of classes, which allows us to see if the data is imbalanced. Imbalanced data is known to pose some difficulties for many data mining techniques.

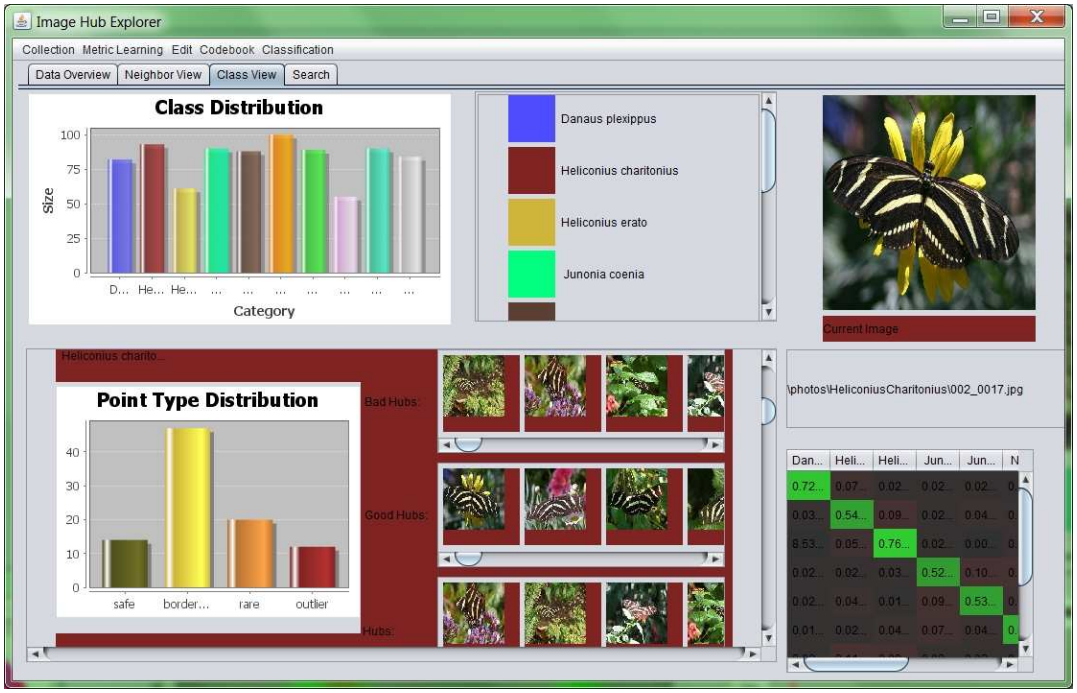


Figure 47: The Class View screen of Image Hub Explorer, which enables the user to examine different properties of data classes and their respective hub structures.

The Query View (Figure 49) is the last view offered by the applet and it deals with potential queries to the image database, i.e. the similarity search. A user can upload an image and the applet will return the set of most similar images, based on the quantized SIFT features extended by binned color histograms. The applet extracts the features of the new image and does the metric comparisons. Apart from the k-neighbor set, a user can also look at how various variants of the k-nearest neighbor algorithms would assign the label, based on the retrieved points. Eight such algorithms are currently supported, some of which are our own and have been recently proposed precisely for dealing with this sort of data. The applet shows the classification confidence of: kNN [Fix and Hodges, 1951], FNN [Keller et al., 1985], NWKNN [Tan, 2005a], AKNN [Wang et al., 2007], hw-kNN [Radovanović et al., 2009], h-FNN [Tomašev et al., 2011b], HIKNN [Tomašev and Mladenić, 2011c][Tomašev and Mladenić, 2012] and NHBNN [Tomašev et al., 2011c].

Apart from classification, a user can also try to invoke hubness-based re-ranking of the neighbor set, performed based on what was learned from the previous occurrences of those neighbor points. In practice, this seems to work quite well. A more detailed analysis is given in Section 3.4.

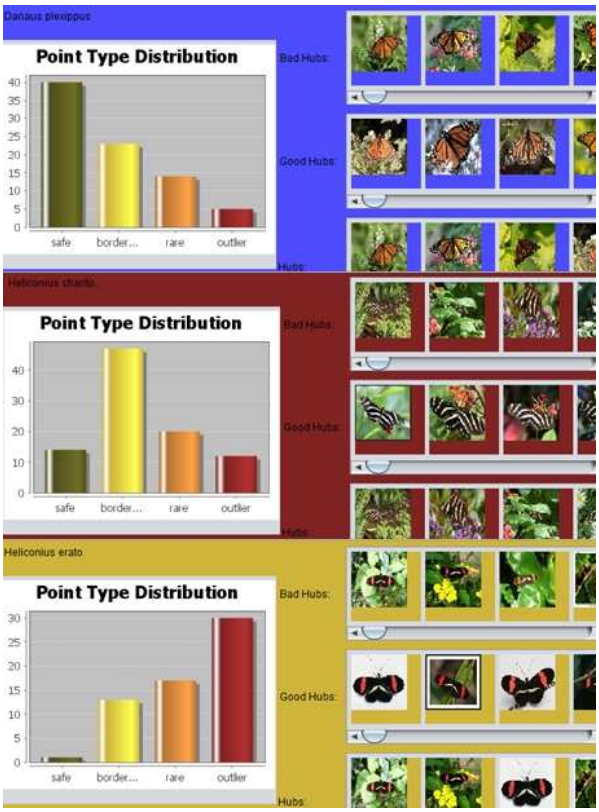


Figure 48: A comparison of three different classes representing butterfly species from the Class View of Image Hub Explorer. We can see a great difference in point type distribution. *Danaus plexippus* seems to be relatively easy to recognize within the observed 10-species dataset. On the other hand, *Heliconius erato* class comprises mostly outliers and rare points, which means that it is much more difficult to handle and should be carefully dealt with within the system.

Some points are bad hubs and in case of images we can visualize what helps an image become a good or a bad hub, feature-wise. Image Hub Explorer includes a feature visualization and assessment component that currently supports the standard SIFT features, that allows the user to visualize the location of good and bad features on each image. Good features are those that occur mostly on images within a single class and therefore help in classification. Bad features are those that occur across different classes and carry little or no discriminative information. The comparison is shown in Figure 51.

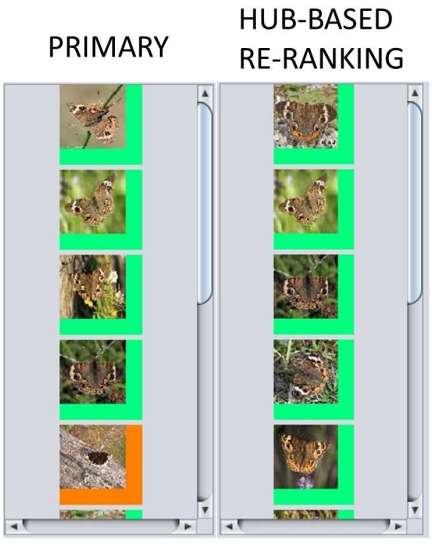


Figure 50: On the left: the original ranking. On the right: the secondary hubness-aware re-ranking.

3.1.3.2 Image Hub Explorer Functionality and Applicability

The architecture of Image Hub Explorer is very flexible. We have seen some examples of its use in analyzing the Leeds Butterfly dataset [Wang et al., 2009] under the Manhattan metric. We have used the Image Hub Explorer to analyze various image collections. The applet can load different sorts of image feature representations and use various primary and secondary metrics. Furthermore, it is not merely re-

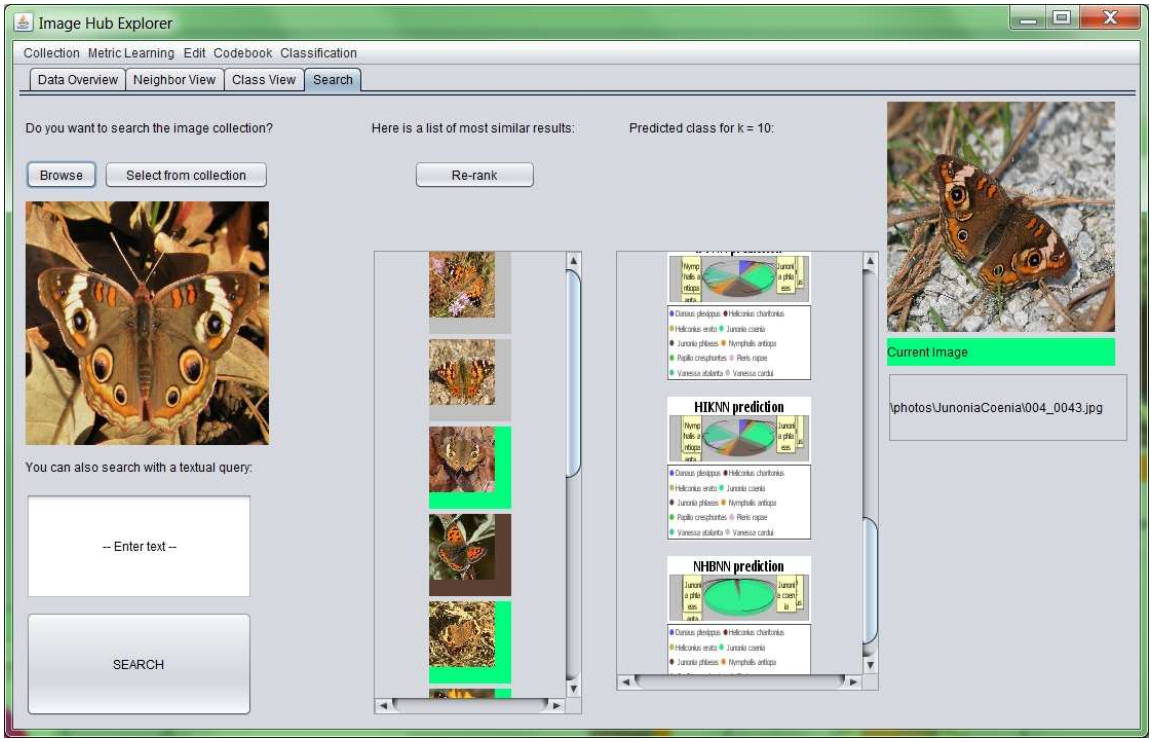


Figure 49: The Query View in Image Hub Explorer, which enables the user to query the image database and label new images by using several kNN classification approaches.

stricted to image data. Any form of data that is aligned with image thumbnails that can be used in visualizations is permitted. In other words, objects that the features are extracted from could also be people or documents or other complex data structures. All that is needed is that each object is assigned an image thumbnail.

The system can calculate the distance matrix in a multi-threaded way from a specified feature representation or it can simply load the distance matrix, in which case the underlying features need not be specified. The distance matrix is persisted to the disk automatically in any case, so that it can be later loaded if the same combination of parameters is invoked at a later time. All files relevant for the analysis are kept in the workspace directory that is selected by the user.

Image Hub Explorer supports several primary and secondary distance measures. The primary metrics include: Manhattan, Euclidean, Cosine and Jaccard. These are all standard distance/similarity measures. The secondary measures of similarity/dissimilarity are more interesting and include $simcos_{50}$ [Houle et al., 2010], $simhub_{50}$ [N. and D., 2012], mutual proximity (MP) [Schnitzer et al., 2011], NICDM [Jegou et al., 2007] and local scaling [Zelnik-manor and Perona, 2004].

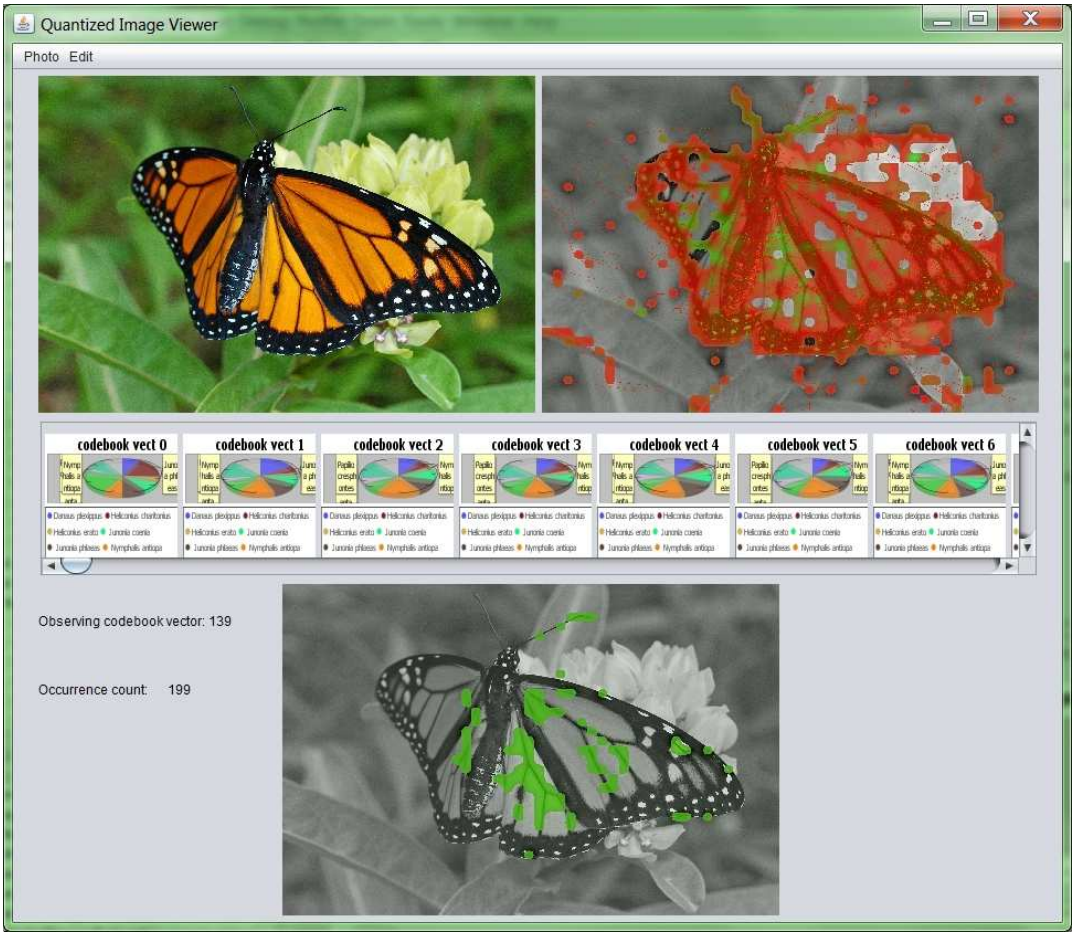


Figure 51: Image Hub Explorer feature assessment tool helps in locating discriminative features and textures, as well as those that do not help in object recognition.

3.2 Hubs in Sensor Data

In this section, we will examine a particular type of sensor data as a test case, the publicly available data that was collected during oceanographic monitoring/survey in 2010 from a series of sensors spread across several bodies of water.

Data mining in sensor data analysis is growing in importance [Ganguly et al., 2008], as the number of sensors and the quantity of data that they output increases. A single sensor is usually a rather simple measurement device that produces a stream of data points, a time series. Several sensors can be grouped at a single node, measuring different quantities. Sensors are used nowadays for many purposes. A typical example would be a sensor that measures power consumption by a household and allows the power distribution company to predict the overall consumption in the network at any point in time. Another typical use case is industrial process control, where sensors are used to detect any anomalies or unexpected patterns that may arise during the production process.

There are many approaches to analyzing sensor data and the applications are domain-specific. Some techniques involve working with streams real-time for continuous monitoring, while other tasks exist that can be processed off-line at request [Ganguly et al., 2008]. Sometimes, there are missing or incorrect values if a sensor hasn't been working properly. Detecting sensor malfunction is very important, as faulty sensors feed the wrong data to the system which may result in incorrect predictive or analytic models. This can be further generalized as *anomaly detection*, detecting unexpected emerging patterns in time series data. An anomaly may or may not be related to some form of malfunction. It may also be caused by correct measurements of an unexpected state of the system that is being monitored by the sensors.

Many approaches exist that allow for anomaly detection in sensor data [Hill et al., 2007][Hill and Minsker, 2010][Yao et al., 2010][Wang et al., 2008b][Siripanadorn et al., 2010]. We will demonstrate that it is possible to exploit hubness information for a simple semi-automatic anomaly detection tool in oceanographic sensor data. These results were published in [Tomašev and Mladenić, 2011a].

In our experiments, we were working with the Integrated Ocean Observing System data (http://www.ioos.gov/). We were analyzing a sample of measurements from many nodes and attached sensors in a period of 20 days in November 2010. Each sensor was monitoring some particular physical property. Eight properties were observed: air temperature, barometric pressure, wind observation, water level observation, water level prediction, salinity, water temperature and conductivity. The data came from sensors distributed across the coastlines of North America, so it was partly about the Pacific, partly about the Atlantic ocean and also partly about the Great Lakes. These three location profiles we used as the labels for the sensors, thereby dividing them into 3 location categories.

Each physical property was analyzed separately. There were some missing values in the data, but not much. Out of the total 4801 time points, usually 50-100 was missing, sometimes none. The values were sampled once every six minutes. This means that there was essentially little difference between neighboring points, so we replaced the missing values by the means of the closest known values.

It is by no coincidence that we have decided to analyze hubness in oceanographic survey data, as hubs are known to emerge in many types of time series data [Radovanović et al., 2010], though they are not always present. Naturally, the degree of occurrence distribution skewness may vary, depending on the processing techniques and the distance measures employed. We have used two distance measures in our primary experiments, the Manhattan distance and the variance of differences between two time series. The obtained results were somewhat similar, as shown in Tables 22 and 23 for two different neighborhood sizes, $k = 3$ and $k = 5$. The same quantities have exhibited occurrence distribution skewness (SN_3,SN_5) in both cases. The label mismatch percentages (BN_3, BN_5) we calculated based on the three

location labels.

Table 22: Hubness-related properties of oceanographic sensor data under the Manhattan distance

quantity	size	SN_3	BN_3	SN_5	BN_5
air temperature	211	0.34	4.7%	0.14	6.7%
barometric pressure	214	0.26	3.4%	-0.06	4.2%
wind	205	3.8	23%	3.6	28%
water level obs.	238	0.6	8.1%	0.47	10%
water level pred.	218	0.34	8.7%	-0.03	11%
salinity	18	-0.13	-	-0.67	-
water temperature	183	0.81	22%	0.67	26%
conductivity	18	0	-	-0.73	-

Table 23: Hubness-related properties of oceanographic sensor data when the distance is defined as the variance of point-wise differences

quantity	size	SN_3	BN_3	SN_5	BN_5
air temperature	211	0.60	6%	0.55	7.9%
barometric pressure	214	0.11	3.9%	-0.05	4.3%
wind	205	5.2	20%	4.8	24%
water level obs.	238	0.92	9.5%	0.92	12%
water level pred.	218	0.27	6.6%	-0.03	8.9%
salinity	18	0.79	-	0.68	-
water temperature	183	1.16	26%	1.40	31%
conductivity	18	1.01	-	0.81	-

Tables 22 and 23 show that, even though many measurements of different physical quantities were taken at the same nodes, the nature of the quantity that is measured influences the shape of the time series and its intrinsic dimensionality, which in turn influences the overall hubness. Time series resulting from measurements of different physical quantities exhibit different degrees of hubness, as can be seen from comparing the skewness of air temperature data, barometric pressure data, wind data, water level data and water temperature data. Salinity and conductivity were under-represented in our sample, so we will not discuss them further, as not much can be gained from merely 18 data points (measurement nodes).

The wind measurement data exhibits high hubness, while the air and water temperature data and the water level observations exhibit medium hubness. The neighbor label mismatch percentages are non-negligible in many cases, which means that some sensor reports from one ocean are more similar to observations in another ocean than in its own. Oceans are big bodies of water and this may not sound very surprising, but it will be shown that this might indicate anomalous sensor behavior.

Unlike most other types of data discussed in this thesis, oceanographic sensor data can be visualized on a world map, since the geolocation coordinates are known for each measurement node. This can help in visualizing the spread of hubness across the data space. We have created a java applet which loads the sensor data and generates visualizations of hubness spread in oceanographic data. Some of these visualizations for different physical quantities are shown in Figures 52–56.

Each measurement node is shown on the map as a filled circle. The diameter of the circle is proportional to the hubness of the node, i.e. its number of occurrences as a neighbor to other points. The larger the circle, the more common the neighbor point. This way, a user can easily detect and find hubs in the data. The label mismatch information is encoded by the color of the circle. The color of each node is a linear combination of green and red colors, so that the percentage of red corresponds to the percentage of label mismatches and the

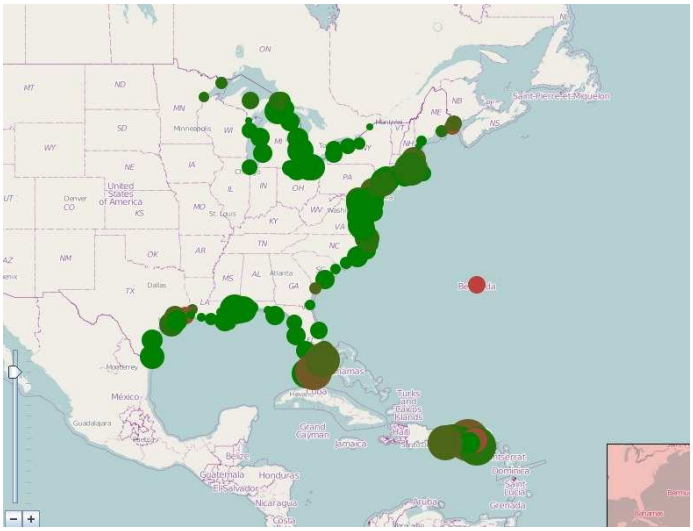


Figure 52: Good/bad and overall hubness of oceanographic sensors shown for air temperature data.

percentage of green corresponds to the number of label agreements in the reverse neighbor sets of the observed node. What we are most interested in is large red circles, i.e. major bad hubs in the data, points that are very frequently neighbors to points from the opposite class.

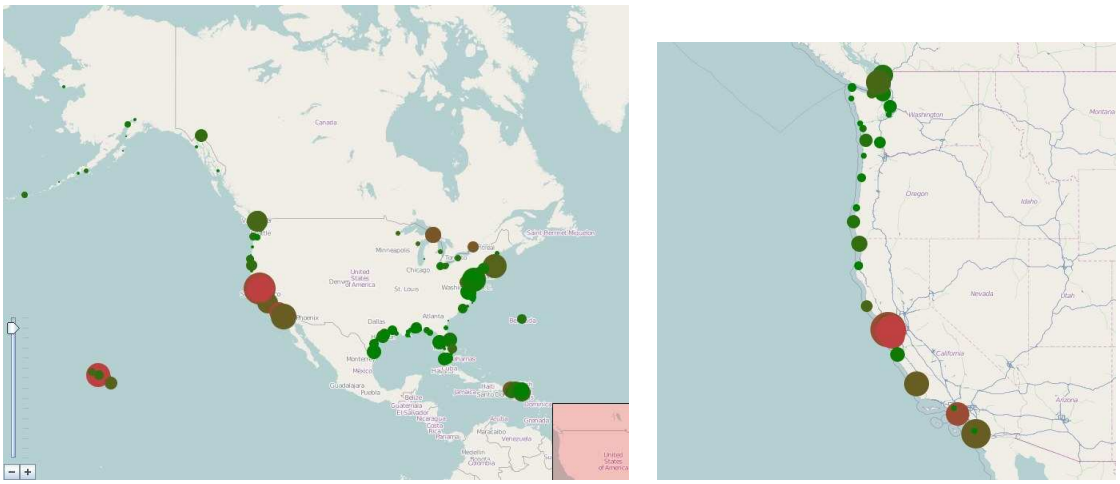
The highest percentage of label mismatches in air temperature data in Figure 52 is achieved by an isolated sensor node in the Bermuda region. As there are no other air temperature sensors nearby, this makes sense. If there were other sensors nearby, we would expect them to report similar measurements and often be each other's neighbors, reducing the bad hubness. So, bad hubness in itself does not suggest anomalous measurements. What does seem potentially suspicious is nodes that are prominent bad hubs, but are located very near to points that exhibit very low bad hubness. Of course, some physical quantities that are being measured vary more than others, but in general - sensor proximity is correlated with measurement similarity.

In Figure 53 that shows hubness of barometric pressure data, such an example can be seen. On the entire coastline, only two nodes exhibit any bad hubness at all and are very prominent bad hubs. Furthermore, they are located next to nodes that have no bad occurrences, that never induce a label mismatch as neighbors. This suggests that something might be wrong with the measurements. Similar cases can be seen in Figures 54–56.

Having bad hubs in close proximity of good hubs and regular nodes is certainly suspicious, but this does not necessarily mean that the sensors are broken. An expert would need to first take a closer look at the data in order to come to that conclusion. What our system allows by means of the shown visualizations is for users to easily detect potential anomalies and faulty sensors. Once such potentially anomalous nodes are pinpointed, checking for actual anomalies in the reported time series shouldn't take too much time.

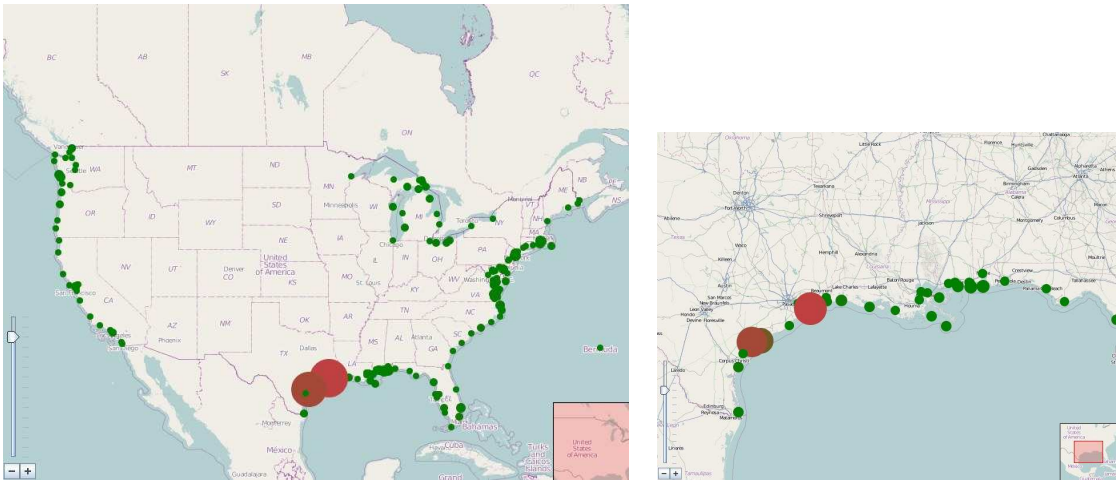
The approach that we have discussed is a very simple method and is by no means meant to be a match for the state-of-the-art anomaly detection systems, especially since it relies on the notion that locations can be clearly separated into disjoint groups, which is not always the case. Generalizations are certainly possible, where instead of label mismatches one might observe the average geolocation distance between a node and its reverse nearest neighbors. In any case, what this system demonstrates is that visualizing and analyzing hubs in sensor data can be beneficial and that it is in principle possible to exploit hubness for semi-automatic anomaly detection.

Figure 53: Good/bad and overall hubness of oceanographic sensors shown for barometric pressure data.



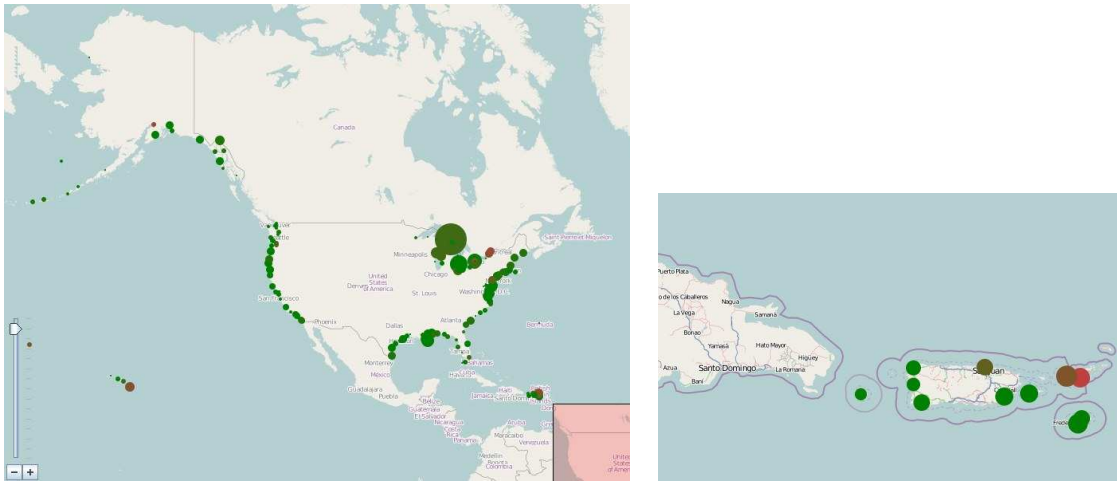
(a) The whole map (b) Zoomed view

Figure 54: Good/bad and overall hubness of oceanographic sensors shown for water temperature data.



(a) The whole map (b) Zoomed view

Figure 55: Good/bad and overall hubness of oceanographic sensors shown for wind data.



(a) The whole map (b) Zoomed view

Figure 56: Good/bad and overall hubness of oceanographic sensors shown for water level data.

3.3 Cross-lingual Document Retrieval

This Section deals with the problem of cross-lingual document retrieval approach extending correlation analysis to be aware of hubness as published in [Tomašev et al., 2013d].

Text mining has always been one of the core data mining tasks, not surprisingly, as we use language to express our understanding of the world around us, encode knowledge and ideas. Analyzing textual data across a variety of sources can lead to some deep and potentially useful insights.

The use of internet has spawned vast amounts of textual data, even more so now with the advent of Web 2.0 and the increased amount of user-generated content. This data, however, is expressed in a multitude of different languages. There is a high demand for effective and efficient cross-language information retrieval tools, as they allow the users to access potentially relevant information that is written in languages they are not familiar with.

Nearest neighbor approaches are common both in text classification [Tan, 2006][Jo, 2008][Trieschnigg et al., 2009] and document retrieval [Chau and Yeh, 2004][Peirsman and Padó, 2010][Lucarella, 1988], which is not surprising given both the simplicity and the effectiveness of most kNN methods. Nearest neighbor methods can be employed both at the document level or at the word level.

Information retrieval in multi-lingual document repositories is of high importance in modern text mining applications. Textual data is known to exhibit hubness [Radovanović et al., 2010b], so it is important to see how this phenomenon relates to textual search and cross-lingual document retrieval.

In order to deepen the understanding of the process, it is important first to compare the k-nearest neighbor topologies across different language representations. An important question that arises is whether hubness is language-dependent. This relates to both the overall skewness of the neighbor occurrence distribution as well as the ratios of relative occurrence frequency between different neighbor points.

3.3.1 Canonical Correlation Analysis

A common approach to analyzing multilingual document collections is to find a common feature representation, so that the documents that are written in different languages can more easily be compared. One way of achieving that is by using the canonical correlation analysis.

Canonical Correlation Analysis (CCA) [Hotelling, 1935] is a dimensionality reduction technique somewhat similar to Principal Component Analysis (PCA) [Pearson, 1901]. It makes an additional assumption that the data comes from two sources or views that share some information, such as a bilingual document corpus [Fortuna et al., 2006] or a collection of images and captions [Hardoon et al., 2004]. Instead of looking for linear combinations of features that maximize the variance (PCA) it looks for a linear combination of feature vectors from the first view and a linear combination for the second view, that are maximally correlated.

Formally, let $S = (x_1, y_1), \ldots, (x_n, y_n)$ be the sample of paired observations where $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}^q$ represent feature vectors from some p and q-dimensional feature spaces. Let $X = [x_1, \ldots, x_n]$ and let $Y = [x_1, \ldots, x_n]$ be the matrices with observation vectors as columns, interpreted as being generated by two random vectors $\mathscr{X}$ and $\mathscr{Y}$. The idea is to find two linear functionals (row vectors) $\alpha \in \mathbb{R}^p$ and $\beta \in \mathbb{R}^q$ so that the random variables $\alpha \cdot \mathscr{X}$ and $\beta \cdot \mathscr{Y}$ are maximally correlated. The α and β map the random vectors to random variables, by computing the weighted sums of vector components. This gives rise to the following optimization problem:

$$\underset{\alpha\in\mathbb{R}^p,\beta\in\mathbb{R}^q}{\text{maximize}} \quad \frac{\alpha C_{XY}\beta'}{\sqrt{\alpha C_{XX}\alpha'}\sqrt{\beta C_{YY}\beta'}}, \tag{9}$$

where C_{XX} and C_{YY} are empirical estimates of the variances of $\mathscr{X}$ and $\mathscr{Y}$ respectively and C_{XY} is an estimate of the covariance matrix. Assuming that the observation vectors are centered, the matrices are computed in the following way: $C_{XX} = \frac{1}{n-1}XX'$, $C_{YY} = \frac{1}{n-1}YY'$ and $C_{XY} = \frac{1}{n-1}XY'$.

This optimization task can be reduced to an eigenvalue problem and includes inverting the variance matrices C_{XX} and C_{YY}. In case of non-invertible matrices, it is possible to use a regularization technique by replacing C_{XX} with $(1-\kappa)C_{XX} + \kappa I$, where $\kappa \in [0,1]$ is the regularization coefficient and I is the identity matrix.

A single canonical variable is usually inadequate in representing the original random vector and typically one looks for k projection pairs $(\alpha_1,\beta_1),\ldots,(\alpha_k,\beta_k)$, so that α_i and β_i are highly correlated and α_i is uncorrelated with α_j for $j \neq i$ and analogously for β.

The problem can be reformulated as a symmetric eigenvalue problem for which efficient solutions exist. If the data is high-dimensional and the feature vectors are sparse, iterative methods can be used, such as the well-known Lanczos algorithm [Cullum and Willoughby, 2002]). If the size of the corpus is not prohibitively large, it is also possible to work with the dual representation and use the "kernel trick" [Jordan and Bach, 2001] to yield a nonlinear version of CCA.

3.3.2 Data

For the experiments, we examined the Acquis aligned corpus data (http://langtech.jrc.it/JRC-Acquis.html), which comprise a set of more than 20000 documents in many different languages. To simplify the initial analysis, we focused on the bi-lingual case and compared the English and French aligned document sets. The documents were labeled and associated with 14 different binary classification problems.

The documents were analyzed in the standard bag-of-words representation after tokenization, lemmatization and stop word removal. Only nouns, verbs, adjectives and adverbs were retained, based on the part-of-speech tags. The inter-document similarity was measured by the cosine similarity measure.

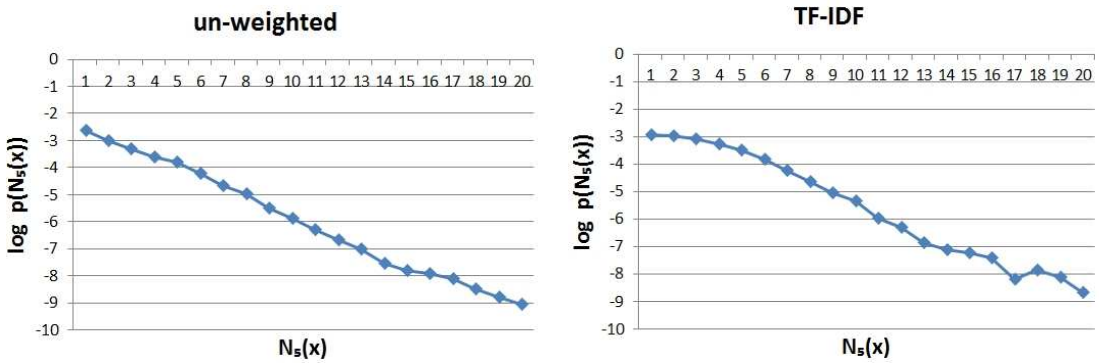
Common semantic representation for the two aligned document sets was obtained by applying CCA. Both English and French documents were then mapped onto the common semantic space (CS:E, CS:F). The used common semantic representation was 300-dimensional, as we wanted to test our assumptions in the context of dimensionality reduction and slight information loss. Longer representations would be preferable in practical applications.

3.3.3 Cross-lingual Hub Structure

The Acquis corpus exhibits high hubness. This is apparent from Figure 57. The data was normalized by applying TF-IDF, which is a standard preprocessing technique. The normalization only slightly reduces the overall hubness.

The common semantic projections exhibit significantly lower hubness than the original feature representations, which already suggests that there might be important differences in the hub structure. The outline of the data is given in Table 24. The two languages exhibit somewhat different levels of hubness.

If the hubness information is to be used in the multi-lingual context, it is necessary to understand how it maps from one language representation to another. Both the quantitative and the qualitative aspects of the mapping need to be considered. The quantitative aspect refers to the the correlation between the total document neighbor occurrence counts and provides the answer to the general question of whether the same documents become hubs



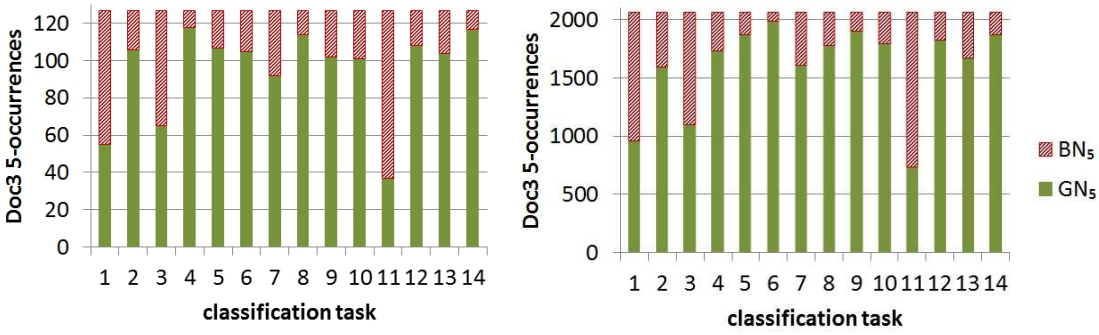
(a) No feature weighting (b) TF-IDF

Figure 57: The logarithmic plots of the 5-occurrence distribution on the set of English Acquis documents with or without performing TF-IDF feature weighting. The straight line in the un-weighted case shows an exponential law in the decrease of the probability of achieving a certain number of neighbor occurrences. Therefore, frequent neighbors are rare and most documents are anti-hubs. Note that $N_5(x)$ is sometimes much more than 20, both charts are cut-off there for clarity. Performing TF-IDF somewhat reduces the overall hubness, even though it still remains high.

Table 24: Overview of the k-occurrence skewness (S_{N_k}) for all four document corpus representations. To further illustrate the severity of the situation, the degree of the major hub ($\max N_k$) is also given. Both quantities are shown for $k = 1$ and $k = 5$.

Data set	size	d	S_{N_1}	$\max N_1$	S_{N_5}	$\max N_5$
ENG	23412	254963	16.13	95	19.45	432
FRA	23412	212955	80.98	868	54.22	3199
CS:E	23412	300	5.20	38	1.99	71
CS:F	23412	300	4.90	38	1.99	62

in different languages. The qualitative aspect is concerned with characterizing the type of influence expressed by the hubs in correlating the good and bad hubness (label mismatch percentages) in both languages.



(a) English version of the text (b) French version of the text

Figure 58: Comparing the 5-occurrences of one randomly chosen document (Doc-3) across various classification tasks (label arrays) in English and French language representations. The hubness of Doc-3 differs greatly, but the type of its influence (good/bad hubness ratio) seems to be preserved.

Let us consider one randomly chosen hub document from the corpus. Figure 58 shows its occurrence profiles in both English and French over all 14 binary classification problems. The good/bad occurrence distributions for this particular document appear to be quite similar in both languages, even though the total hubness greatly differs. From this we can conclude that, even though the overall occurrence frequency depends on the language, the

semantic nature of the document determines the type of influence it will exhibit if and when it becomes a hub. On the other hand, this particular document is an anti-hub in both projections onto the common semantic space, i.e. it never occurs as a neighbor there. This illustrates how the CCA mapping changes the nature of the k-nearest neighbor structure, which is what Table 24 also confirms.

The observations from examining the influence profiles of a single document are easily generalized by considering the average Pearson correlation between bad hubness ratios over the 14 binary label assignments, as shown in Table 25. There is a quite strong positive correlation between document influence profiles in all considered representations and it is strongest between the projections onto the common semantic space, which was to be expected. As for the total number of neighbor occurrences, the Pearson product-moment gives positive correlation between the hubness of English and French texts, as well as between the projected representations. In all other cases there is no linear correlation. We measured the non-linear correlation by using the Spearman correlation coefficient. It seems that there is some positive non-linear correlation between hubness in all the representations.

Table 25: Correlations of document hubness and bad hubness between different language representations: English, French, and their projections onto the common semantic space.

(a) Pearson correlation between bad hubness ratios of documents ($BN_k(x)/N_k(x)$)

ENG	FRA	CS:E	CS:F	
	0.68	0.61	0.58	ENG
		0.56	0.58	FRA
			0.76	CS:E
				CS:F

(b) Pearson correlation between total hubness (occurrence frequencies)

ENG	FRA	CS:E	CS:F	
	0.47	0.08	0.06	ENG
		0.01	0.01	FRA
			0.64	CS:E
				CS:F

(c) Spearman correlation between total hubness (occurrence frequencies)

ENG	FRA	CS:E	CS:F	
	0.67	0.29	0.25	ENG
		0.25	0.29	FRA
			0.70	CS:E
				CS:F

The results of correlation comparisons can be summarized as follows: frequent neighbor documents among English texts are usually also frequent neighbors among the French texts and the nature of their influence is very similar. Good/bad neighbor documents in English texts are expected to be good/bad neighbor documents in French texts and vice-versa. We will exploit this apparent regularity for improving the neighbor structure of the common semantic space.

3.3.4 Hubness-aware CCA extension

In the canonical correlation analysis, all examples contribute equally to the process of building a common semantic space. However, due to hubness, it is not clear whether all documents are to be considered equally relevant or equally reliable. Documents that become bad hubs exhibit a highly negative influence. Furthermore, as shown in Figure 58, a single hub-document can act both as a bad hub and as a good hub at the same time, depending on the specific classification task at hand. Therefore, instance selection doesn't seem to be a good approach, as we cannot both accept and reject an example simultaneously.

Introducing instance weights into the CCA procedure might help to control the influence of hubs on forming the common semantic representation in hope that this would in turn improve the cross-lingual retrieval and classification performance in the common semantic space.

The weights introduce a bias in finding the canonical vectors: the search for canonical vectors is focused on the spaces spanned by the instances with high weights.

Given a document sample S, let $u_1, \ldots, u_n$ be the positive weights for the examples $x_i \in X$ and $v_1, \ldots, v_n$ be the positive weights for the examples $y_i \in Y$. The modified covariance and variance matrices can be computed as follows:

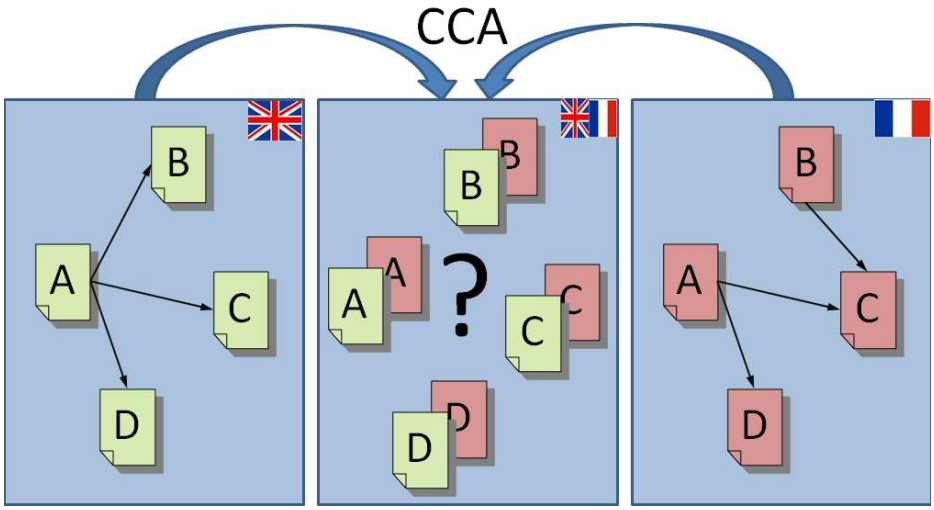


Figure 59: The CCA procedure maps the documents written in different languages onto the common semantic space. According to the analysis given in Table 25, this changes the kNN structure significantly, which has consequences for the subsequent document retrieval and/or classification. By introducing instance weights we can influence the mapping so that we preserve certain aspects of the original hub-structure and reject the unwanted parts of it.

$$\tilde{C}_{XX} := \frac{1}{n-1}\sum_{i=1}^{n} u_i^2 x_i x_i', \qquad \tilde{C}_{YY} := \frac{1}{n-1}\sum_{i=1}^{n} v_i^2 y_i y_i'$$
$$\tilde{C}_{XY} := \frac{1}{n-1}\sum_{i=1}^{n} u_i v_i x_i y_i' \tag{10}$$

These matrices are input for the standard CCA optimization problem. By modifying them, it is possible to directly influence the outcome of the process. The weighting approach is equivalent to performing over-sampling of the instances based on their specified weights and then computing the covariances and variances.

Let $h(x_i,k)$ and $h_B(x_i,k)$ be the standardized hubness and standardized bad hubness scores respectively, i.e. $h(x_i,k) = \frac{N_k(x_i)-\mu_{N_k(x_i)}}{\sigma_{N_k(x_i)}}$ and $h_B(x_i,k) = \frac{BN_k(x_i)-\mu_{BN_k(x_i)}}{\sigma_{BN_k(x_i)}}$. A high standardized hubness score means that the document is very influential and relevant for classification and retrieval, while a high bad hubness score indicates that the document is unreliable.

Many different weighting schemes are possible. Two approaches stand out as most intuitive. The first approach would be to increase the influence of relevant points (*hubs*) in the CCA weighting. The second meaningful approach is to reduce the influence of unreliable points (*bad hubs*). Additionally, the opposite of what is proposed will also be considered for comparisons, i.e. reducing the influence of hubs and increasing the influence of bad hubs. Therefore, the considered weighting schemes are given as follows: un-weighted, $v_i := 1$, emphasized hubs, $v_i := e^{h(x_i,k)}$, de-emphasized hubs, $v_i := e^{-h(x_i,k)}$, emphasized bad hubs, $v_i := e^{h_B(x_i,k)}$, and de-emphasized bad hubs, $v_i := e^{-h_B(x_i,k)}$.

In the experimental protocol, two disjoint subsets of the aligned corpus were randomly selected: 2000 documents were used for training ad 1000 for testing. For each of the 14 binary classification problems five common semantic spaces were computed with CCA on the training set: the non-weighted variant (CS:N), emphasized hubs (CS:H), de-emphasized hubs (CS:h), emphasized bad hubs (CS:B) and de-emphasized bad hubs (CS:b). The training and test documents in both languages were then projected onto the common semantic space. In each case, the quality of the common semantic space was evaluated by measuring the performance of both classification and document retrieval. The whole procedure was repeated 10 times, hence yielding the repeated random sub-sampling validation. The average

performance was measured and its standard deviation calculated.

Many of the binary label distributions were highly imbalanced. This is why the classification performance was measured by considering the Matthews Correlation Coefficient (MCC) [Powers, 2011]. It measures the correlation between the observed and the predicted class assignments, as shown in Equation 11, where TP, FP, TN, FN denotes the number of true positives, false positives, true negatives and false negatives, respectively.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP-FP)(TP+FN)(TN+FN)(TN+FP)}} \tag{11}$$

Comparing the classification performance on the original (non-projected) documents with the performance on the common semantic space usually reveals a clear degradation in performance, unless the dimensionality of the projected space is high enough to capture all the relevant discriminative information.

The overview of the classification experiments is given in Table 26. We only report the result on the English texts and projections, as they are basically the same in the French part of the corpus. The kNN classifier with $k = 5$ was used. It is immediately apparent that the weights which emphasize document hubness (CS:H) achieve the best results among the common semantic document representations. Reducing the influence of bad hubs (CS:b) is in itself not enough to positively affect the classification performance. This might be because many hubs reside in borderline regions, so they might carry some relevant disambiguating feature information. It seems that emphasizing the relevance by increasing the preference for all hub-documents gives the best classification results.

Table 26: The Matthews correlation coefficient (MCC) values achieved on different projected representations. The symbols •/◦ denote statistically significant worse/better performance ($p < 0.01$) compared to the non-weighted projected representation (CS:N).

Label	Original	CS:N	CS:H	CS:h	CS:B	CS:b
lab1	73.0 ± 3.3	34.2 ± 4.6	**69.2 ± 2.8** ◦	66.0 ± 3.3 ◦	52.8 ± 4.8 ◦	46.6 ± 10.2 ◦
lab2	69.2 ± 3.0	52.3 ± 4.4	**65.1 ± 3.9** ◦	38.3 ± 3.8 •	45.8 ± 7.0	35.7 ± 8.6 •
lab3	50.2 ± 3.3	27.6 ± 3.8	44.1 ± 3.0 ◦	42.2 ± 5.0 ◦	**44.8 ± 3.6** ◦	33.7 ± 3.0 ◦
lab4	32.2 ± 4.4	18.8 ± 6.4	**28.1 ± 2.8** ◦	21.1 ± 3.9	20.6 ± 3.7	20.3 ± 6.5
lab5	28.9 ± 12.4	16.8 ± 12.9	17.7 ± 11.7	**21.9 ± 14.4**	10.2 ± 5.5	15.7 ± 6.0
lab6	38.1 ± 6.2	31.2 ± 6.0	29.3 ± 8.2	**33.6 ± 5.4**	23.5 ± 5.8 •	26.2 ± 6.6
lab7	54.5 ± 3.2	38.9 ± 4.0	**48.4 ± 4.2** ◦	45.7 ± 3.0 ◦	42.3 ± 6.3	36.5 ± 6.8
lab8	44.6 ± 6.3	31.5 ± 6.9	**40.4 ± 6.4** ◦	33.5 ± 5.7	23.0 ± 5.0 •	19.6 ± 8.7 •
lab9	76.2 ± 3.4	32.0 ± 5.4	**74.4 ± 3.4** ◦	61.8 ± 3.7 ◦	45.7 ± 5.2 ◦	37.7 ± 7.6
lab10	41.4 ± 4.2	26.1 ± 3.8	34.0 ± 3.8 ◦	31.6 ± 5.5	**34.4 ± 4.6** ◦	26.6 ± 5.2
lab11	53.5 ± 2.5	27.9 ± 2.8	**48.6 ± 4.0** ◦	42.0 ± 3.5 ◦	44.9 ± 3.8 ◦	33.7 ± 3.8 ◦
lab12	39.2 ± 4.0	31.5 ± 3.4	35.4 ± 5.9	**35.6 ± 6.6**	22.8 ± 4.9 •	20.3 ± 5.7 •
lab13	45.4 ± 3.4	29.9 ± 5.2	**38.5 ± 6.0** ◦	37.1 ± 4.6 ◦	32.6 ± 5.4	28.0 ± 4.9
lab14	49.9 ± 4.5	35.4 ± 7.1	**44.8 ± 7.6**	44.1 ± 7.4	22.4 ± 5.9 •	23.4 ± 11.7
AVG	49.7	31.0	**44.1**	39.6	33.3	28.9

In evaluating the document retrieval performance, the k-neighbor set purity was chosen as the most relevant metric. The inverse mate rank is certainly also important, but the label matches are able to capture a certain level of semantic similarity among the fetched results. A higher purity among the neighbor sets ensures that, for instance, if your query is about the civil war, you will not get results about gardening, regardless of whether the aligned mate was retrieved or not. This is certainly quite useful. The comparisons are given in Table 27.

Once again, the CS:H weighting proves to be the best among the evaluated hubness-aware weighting approaches, as it retains the original purity of labels among the document kNNs. It is significantly better than the un-weighted baseline (CS:N).

Table 27: The average purity of the k-nearest document sets in each representation. The symbols $\bullet/\circ$ denote significantly lower/higher purity ($p < 0.01$) compared to the non-weighted case (CS:N). The best result in each line is in bold.

Label	Original	CS:N	CS:H	CS:h	CS:B	CS:b
lab1	84.5 ± 1.3	80.7 ± 1.6	**84.1 ± 1.1** ∘	83.3 ± 1.5 ∘	83.7 ± 1.5 ∘	81.7 ± 2.1
lab2	90.5 ± 1.2	84.5 ± 3.2	**90.1 ± 1.2** ∘	88.2 ± 2.0 ∘	89.6 ± 1.5 ∘	84.9 ± 3.7
lab3	74.4 ± 0.9	71.3 ± 1.0	74.4 ± 1.0 ∘	73.6 ± 0.9 ∘	**74.6 ± 1.2** ∘	72.6 ± 1.1
lab4	85.8 ± 1.6	84.6 ± 4.4	85.9 ± 1.5	**85.9 ± 1.8**	85.1 ± 1.5	84.1 ± 3.6
lab5	96.0 ± 0.6	95.9 ± 1.3	95.9 ± 0.8	**96.3 ± 0.8**	95.3 ± 1.0	94.5 ± 3.0
lab6	91.7 ± 0.9	90.2 ± 3.4	**91.6 ± 1.1**	91.6 ± 1.5	90.8 ± 1.5	89.5 ± 3.5
lab7	79.7 ± 0.8	78.0 ± 2.2	**79.7 ± 1.0**	79.0 ± 1.6	79.5 ± 0.6	77.8 ± 1.7
lab8	89.1 ± 1.3	87.0 ± 3.4	**89.0 ± 1.2**	88.5 ± 1.6	88.0 ± 1.3	85.6 ± 3.2
lab9	91.8 ± 1.1	84.7 ± 3.1	**92.0 ± 1.1** ∘	89.6 ± 1.5 ∘	90.9 ± 1.3 ∘	83.9 ± 3.1
lab10	84.3 ± 0.7	**84.5 ± 1.4**	84.4 ± 0.6	84.4 ± 0.8	83.7 ± 0.7	83.4 ± 1.6
lab11	77.0 ± 0.9	73.5 ± 1.1	77.1 ± 0.8 ∘	75.5 ± 0.9 ∘	**77.3 ± 0.6** ∘	74.7 ± 1.2
lab12	88.7 ± 1.2	**88.7 ± 3.3**	88.6 ± 1.3	88.7 ± 1.9	87.6 ± 1.5	87.9 ± 3.5
lab13	82.3 ± 1.5	81.9 ± 2.1	**82.4 ± 1.5**	82.2 ± 1.8	82.0 ± 1.4	80.7 ± 2.5
lab14	92.7 ± 0.8	92.1 ± 2.8	92.3 ± 0.7	**92.7 ± 1.2**	91.7 ± 1.3	91.7 ± 3.1
AVG	86.3	84.1	**86.3**	85.7	85.7	83.8

The CS:H weighting produces results most similar to the ones in the original English corpus. Our hypothesis was that it was because this particular document weighting scheme best helps to preserve the kNN structure of the original document set. The relevant correlations were examined and it turns out that this is indeed the case, as shown in Table 28. By preserving the original structure, it compensates for some of the information loss which would have resulted due to the dimensionality reduction during the CCA mapping.

Table 28: The correlations of document hubness between some of the different common semantic representations, as well as the original English documents. CS:H (emphasize hubness when building the rep.) best preserves the original kNN structure, which is why it leads to similar classification performance, despite the dimensionality reduction.

(a) Pearson correlation between total hubness on the **training** set (occurrence frequencies)

ENG	CS:N	CS:H	CS:h	
	0.05	**0.42**	0.02	ENG
		0.03	0.05	CS:N
			0.02	CS:H
				CS:h

(b) Pearson correlation between total hubness on the **test** set (occurrence frequencies)

ENG	CS:N	CS:H	CS:h	
	0.65	**0.88**	0.75	ENG
		0.68	0.93	CS:N
			0.80	CS:H
				CS:h

Even though the hub-structure of the data remains preserved across different languages, it is radically changed by the CCA mapping onto the common semantic space. It seems that the we can slightly compensate for the information loss and kNN topology deformation by introducing the hubness-aware instance weights into the CCA optimization problem and this helps in preserving the original kNN structure of the data during the CCA mapping. These initial results need to be verified on more language pairs and larger document corpora.

3.4 Bug Duplicate Detection

This section deals with the issues of bug duplicate detection and proposes a novel hubness-aware re-ranking approach. The results were reported in [Tomašev et al., 2013a].

In software development, despite the initial testing and the best efforts of the developers,

some minor or major bugs and issues are often detected only after the product/service had been released/deployed. Issue tracking systems are therefore useful tools which allow the users either to report bugs in specific use cases of the software or to suggest new features that would extend the existing functionality of the software. Therefore, bug tracking systems are an integral part of developer infrastructure in most large software engineering companies. Most bugs are reported multiple times by the users and manually checking for similar issues can be time-consuming and costly. Providing some support for handling duplicates is highly desirable. This is often achieved by using semi-automatic bug duplicate detection systems.

There are many existing bug tracking systems. Some of the best known include Bugzilla, JIRA, Launchpad and Redmine[1]. There are several types of information that a user is usually able to report to such systems. Apart from the textual description, a user can leave his/hers contact information, specify the type of the request (eg. bug, suggestion), specify the component where the issue was noticed, the build version, the platform on which the system was running, the priority with which the issue needs to be resolved, etc.

If the issue can be replicated, it is verified by the responsible person and needs to be assigned to the engineer who would be working on the fix. Before the assignment takes place, the system needs to check if the same issue had previously already been reported, as there is no need to forward the issue reports have either already been resolved or are being resolved.

Bug duplicate detection is not an easy task, as different users may formulate their free-text reports in entirely different ways, depending on their background and experience.

As the basic component of bug reports is their textual description and textual data is known to exhibit hubness [Radovanović et al., 2010b], it is to be expected that hubs might play a certain role in similarity based bug duplicate detection systems.

Automatic bug duplicate detection is a rather recent research direction. One of the first information retrieval approaches to duplicate detection was proposed in [Runeson et al., 2007]. Textual clustering was another approach [Hiew, 2006]. A natural language processing based duplicate detection method based on both the textual and the execution similarity was used in [Wang et al., 2008a]. Using n-grams is also possible and somewhat more robust to noise and domain-specific term usage [Sureka and Jalote, 2010]. Classifiers can be trained to classify the reports into duplicates, but they currently do not achieve high precision [Jalbert and Weimer, 2008]. Using advanced techniques for topic modeling also seems promising [Nguyen et al., 2012].

3.4.1 Bug Duplicate Data

The November 2010 image of the KDE bug repository (https://bugs.kde.org/) was used for the experiments and it contained 249.129 reports. A large portion of these reports were marked as duplicates, 47.061. The reports were filed between 21.1.1999 and 2.11.2010. Bugs in KDE repository are related to 479 different products.

The similarity between the reports is obtained by calculating the similarity between the textual bug report descriptions. All the relevant provided information was used, including the subject line, main description and the associated comments.

Standard text processing was performed, including stemming and stop word removal. Porter stemmer [Porter, 1980] was used for word inflection removal and the data was represented by the standard vector space model [Raghavan and Wong, 1986] with TF-IDF weighting [Feldman and Sanger, 2006]. Assuming $V = [v_1, v_2..v_W]$ is the vocabulary used to form the reports, each report is then represented as a $|V|$-dimensional vector of weights. Each weight corresponds to a given word in the vocabulary, its frequency and the inverse document frequency. This is a well-known textual representation.

[1] http://en.wikipedia.org/wiki/Comparison_of_issue-tracking_systems

Let R be the set of all filed issues and let $\vec{r}_i = [r_i^1, r_i^2 \dots r_i^W]$ and $\vec{r}_j = [r_j^1, r_j^2 \dots r_j^W]$ be the vectors representing two such bug reports. The similarity between the two reports is then defined as the cosine similarity between the corresponding vectors [Feldman and Sanger, 2006].

$$sim(\vec{r}_i, \vec{r}_j) = \frac{\vec{r}_i \cdot \vec{r}_j}{\|\vec{r}_i\| \cdot \|\vec{r}_j\|} = \frac{\sum_{p=1}^{W} r_i^p \cdot r_j^p}{\sqrt{(\sum_{p=1}^{W}(r_i^p)^2) \cdot (\sum_{p=1}^{W}(r_j^p)^2)}} \tag{12}$$

3.4.2 Outline of the Duplicate Detection Process

Issue reports arrive at some fixed order of precedence. Upon arrival, each report is compared to the previously archived documents to check for possible duplicates. Therefore, if a bug arrives at time T, it is only compared to the reports that have arrived at some $t < T$. All the experiments are performed by taking this into account.

Time t_i corresponds to the time of receipt of report r_i. Let $T_f = t_N$ be the time of receipt for the last received report. When the newly received report is analyzed, its cosine similarity towards each earlier report is calculated and the list is sorted in a descending order, so that the most similar earlier report is shown first to the system user. This allows the responsible engineers and easy overview of the possible duplicate reports. By inspecting the list, they will mark some of the suggestions as duplicates.

This process is far from perfect, as it involves manual inspection of many free-text issue reports. Sometimes, duplicates do not get noticed. However, the precision of the automatic duplicate detection software is not high enough to be able to replace actual human participation in the process. This is why bug report duplicate detection remains a semi-automatic process for the time being.

3.4.3 The Influence of Hub Reports

The temporal component of the duplicate detection process requires us to introduce temporal hubness as a concept and extend all the relevant quantities by the parameter T.

Denote by $N_k^T(r)$ the number of times that the report r occurred in the top-k similarity queries of the duplicate detection system up until time T. We will refer to $N_k^T(r)$ as the temporal occurrence frequency of r. It is a function that is monotonously non-decreasing with respect to T. The temporal neighbor occurrence skewness at time T is defined as follows:

$$SN_k^T = \frac{\frac{1}{N}\sum_{i=1}^{N}(N_k^T(r_i) - k)^3}{(\frac{1}{N}\sum_{i=1}^{N}(N_k^T(r_i) - k)^2)^{3/2}} \tag{13}$$

Positive (right) skewness means that the distribution tail is longer on the right side and that most values are lower than the mean. Figure 60 shows the skewness of the bug report occurrence frequency distribution of the analyzed data for different values of k. The bug reports exhibit very high hubness.

The consequences of high occurrence frequency skewness can be seen in Figure 61, where the degree of the major hub is shown, i.e. the maximal $N_k^{T_f}(r)$ over the set of all reports. We can see that the major hub report occurs very frequently, much more often than it actually occurs as a genuine duplicate. For $k = 10$, the major hub report occurs 1421 times, out of which it is only an actual duplicate once and not a duplicate 1420 times. Out of those 1420 occurrences, in 373 cases an actual duplicate of the query report existed, but it was not the major hub. As the system can not always pinpoint the exact duplicate as the most similar document, $k > 1$ has to be used in practice. The above given analysis of the nature of hub occurrences suggests that this might significantly increase the number of false positives.

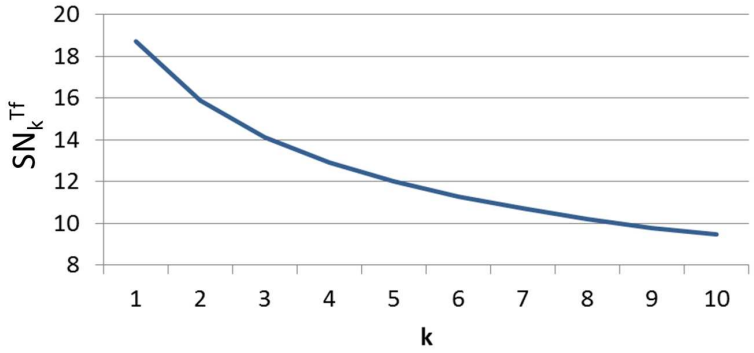


Figure 60: Final skewness of the bug report occurrence frequency distribution, indicating high hubness, as everything above SN_k of 1-2 can be considered quite high).

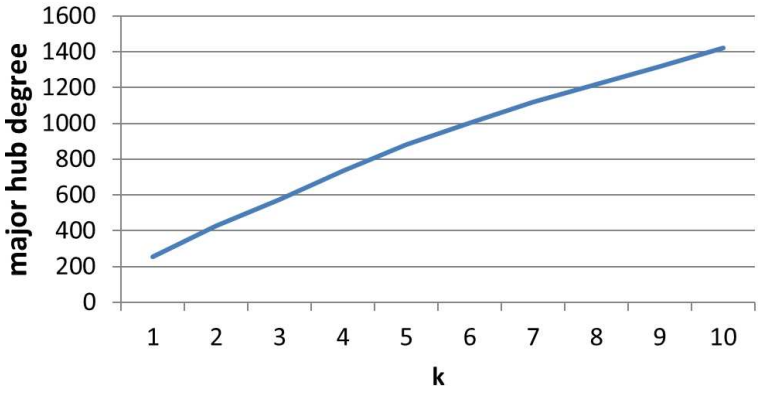


Figure 61: The maximum report occurrence frequency, shown for different values of k.

In the context of bug duplicate detection, a bad k-occurence is defined as each occurrence of report r_i in the query top-k result set of report q such that r_i is not a duplicate of q, if the report q has duplicates in the previously gathered data. Denote by $BN_k^{T_f}(r)$ the temporal bad occurrence frequency. The occurrences that are not bad will be referred to as good occurrences and we will denote them by $GN_k^{T_f}(r)$,so that $N_k^{T_f}(r) = BN_k^{T_f}(r) + GN_k^{T_f}(r)$.

If a report has no previous duplicates, whatever gets retrieved by the system does not affect its performance. Therefore, we are only interested in those queries where actual duplicates exist.

Most occurrences in the systems turn out to be bad occurrences, as shown in Figure 62. This is not surprising, as the problem of bug duplicate detection is a hard one and systems usually achieve low precision.

3.4.4 Secondary Hubness-aware Re-Ranking

In the spirit of learning from past occurrences, potential duplicate occurrence profiles can be exploited in order to improve system effectiveness in detecting actual duplicates.

Denote by q_T the currently received report at time T for which the query is made to the bug duplicate detection system. Let $R_{q,T} = \{r_{i_1}, r_{i_2}, ...r_{i_k}\}$ be the top-k result set returned by the system. The system also provides the primary similarity scores. The secondary hubness-aware re-ranking is performed by re-defining the similarity measure based on the temporal occurrence model. The secondary similarity will be denoted by $sim_k^{H,T}$. After the secondary similarity is calculated, the final ranking is obtained by re-ordering the initial result set based on the new similarity value.

In the temporal occurrence model, the nature of the report occurrence profile might change over time. A good hub might become a bad hub and a false positive as time goes by

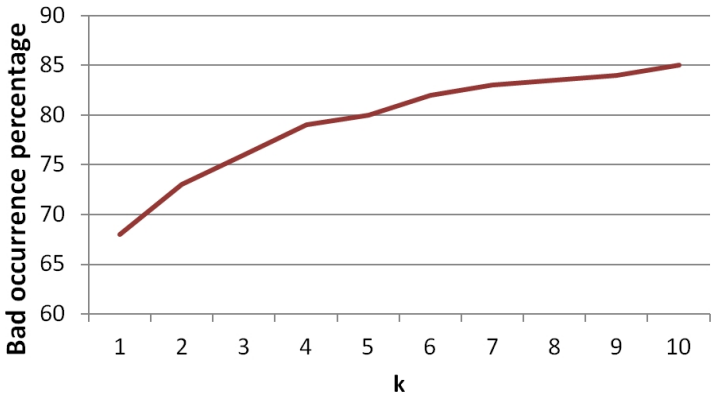


Figure 62: The total cumulative percentage of bad occurrences in the result sets of the initial bug duplicate detection system, up until time T_f.

and the distribution of the data changes, or vice-versa.

Therefore, the total aggregate occurrence count is not a reliable quantity. This is it is beneficial to observe only the occurrence counts within a time-dependent sliding window. Denote by $GN^T_{k,\mu}(r)$ the good truncated occurrence frequencies, that is obtained by counting only among the past μ occurrences of r (or less, if r has occurred less than ĕ times total). The truncated total occurrence count is then $N^T_{k,\mu}(r) = \min(\mu, (N^T_k(r)+1))$, where we have increased the original occurrence count by one to avoid zero divisions. In a sense, each query report is implicitly included in its own result set at position zero.

The secondary $sim^{H,T}_k$ similarity measure is defined as follows:

$$sim^{H,T}_k = \frac{GN^T_{k,\mu}}{N^T_{k,\mu}} \cdot sim(q_T, r) \tag{14}$$

This simple formula increases the similarity between the query report and those retrieved reports that have recently had very few bad occurrences. This increases the confidence that the retrieved report is actually relevant for the query and is more likely to be an actual duplicate, if such exists.

It would probably not be advisable to sort the query set according to the good occurrence percentages alone, as the primary similarity measure is needed to convey the content-wise relevance information.

The effectiveness of the proposed approach was evaluated by considering the changes in two quality indices: the average rank of the first detected duplicate ($rank^{FD}_{AVG}$) and the overall average rank of detected duplicates ($rank^{D}_{AVG}$). A lower rank would indicate and improvement in performance, as it is important to sort the reports in a way that requires least effort from the user. This is why ranking is important. Additionally, re-ranking can also improve the detection rates, if the system operates with two different k-values, one for $sim^{H,T}_k$ calculation and the other for top-k result retrieval.

The difference in the average first and overall KDE duplicate ranks before and after re-ranking by $sim^{H,T}_k$ is shown in Figures 63 and 64, for different values of k. The rank reduction for the first (most similar) duplicate is present for all k-values. The improvement in reducing the overall detected duplicate rank increases with k.

The proposed ranking system is self-adaptive as it learns the secondary similarity measure by observing and estimating the deficiencies of the primary similarity at each time step, thereby making duplicate reports more similar to each other. The effectiveness of this simple modification to the original bug duplicate detection system demonstrates that learning from past occurrences might be promising in temporal query-based systems. Figure 50 shows the use of the same re-ranking idea in querying image databases. Naturally, there

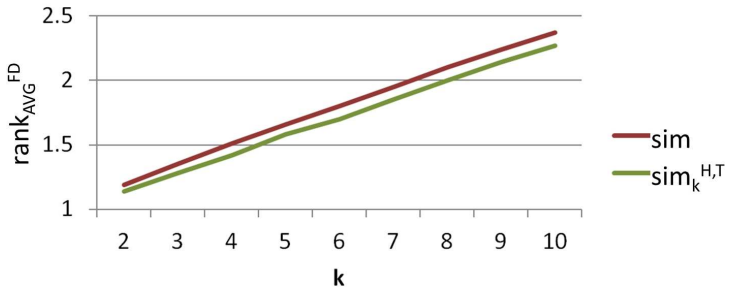


Figure 63: Average rank of the first detected duplicate before and after re-ranking.

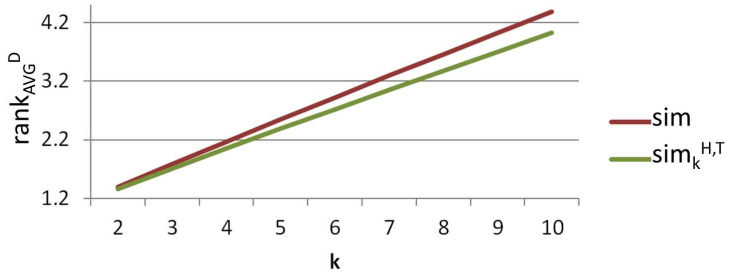


Figure 64: Average rank of the detected duplicates before and after re-ranking.

might be better solutions for exploiting the temporal good/bad hubness information and more complex and elaborate ideas ought to be tested in the future.

4 Conclusions

This thesis examines the phenomenon of emerging hubs in the k-nearest neighbor topologies of intrinsically high-dimensional data, known as *hubness*. It is a recently described aspect of the well-known *curse of dimensionality* that is known to plague many standard data analysis and modeling approaches. The hubness phenomenon has previously mostly been examined from a theoretical standpoint and this thesis presents some of the first steps towards designing robust, hubness-aware machine learning and data mining algorithms that are able to perform well even under the assumption of severe hubness of the data.

We have proposed several novel analytic algorithms, provided their theoretical justification, and evaluated their performance on a wide spectrum of high-dimensional datasets, including images, documents, bug duplicate reports, sensor data and other types of time series, as well as different kinds of synthetic data.

4.1 Scientific Contributions

The main scientific contributions of the thesis are discussed below.

SC 1. Clustering: We have closely examined the role of hubs and hubness in clustering high dimensional data.

- **SC 1.1:** We have shown that hubs can be used as cluster prototypes and that hubness is a good **local centrality measure** in high-dimensional feature spaces, unlike density, which works good only in the low-dimensional case.
- **SC 1.2:** We have closely examined the role of hubs and hubness in **clustering** high dimensional data. We have proposed three novel clustering algorithms that exploit this fact: K-hubs, global hubness-proportional clustering (GHPC) and global hubness-proportional K-means (GHPKM). They were shown to be quite robust in various experimental setups. Improvements in clustering quality have been observed in terms of cluster homogeneity, isolation and the Silhouette index.
- **SC 1.3:** Further analysis has shown the improvements to be related primarily to a better overall clustering of hub points, which are usually difficult to handle [Radovanović, 2011]. The proposed algorithms also manage to find the globally optimal cluster configuration more often, as their stochastic nature allows them to avoid premature convergence to local optima.

SC 2. Classification: In the case of supervised learning, we have shown that it is possible to exploit the class-conditional neighbor **k-occurrence models** to design robust hubness-aware methods for high-dimensional data **classification**. We have proposed three novel classification methods: hubness-based fuzzy k-nearest neighbor (h-FNN), hubness-information k-nearest neighbor (HIKNN) and the naive hubness-Bayesian k-nearest neighbor (NHBNN).

- **SC 2.1:** We have shown that it is possible to use the class-conditional neighbor occurrence information to derive hubness-based fuzzy scores and use them within

the fuzzy k-nearest neighbor classification framework in order to improve the performance of fuzzy kNN classification. The proposed h-FNN algorithm was also shown to improve on the hw-kNN, a previous instance-weighting approach for dealing with hubness in high-dimensional kNN classification.

SC 2.2: The h-FNN model was subsequently extended to include a way of weighting the individual votes based on the neighbor relevance scores that were derived from the neighbor occurrence self-information. The proposed HIKNN algorithm also included ways of combining the label information with the past occurrence information. It has been shown to improve the accuracy of h-FNN on many intrinsically high-dimensional data types.

SC 2.3: As an alternative to the fuzzy kNN classification models, we have proposed a Naive Bayesian way for learning from past neighbor occurrences. All occurrences are treated as random events and the Naive Bayes rule is then applied in order to determine the class affiliation of the query point. Special mechanisms have been introduced in order to properly handle anti-hubs and orphans, as these points have no past occurrence information to learn from. NHBNN was shown to be better than the probabilistic k-nearest neighbor classifier (PNN) on intrinsically high-dimensional data. Additionally, it was shown to be better suited for learning under class imbalance than h-FNN and HIKNN.

The robustness of the proposed methods was also confirmed in presence of high levels of mislabeling and class overlap. They have all been shown to achieve a superior average performance in terms of classification accuracy and F-measure on intrinsically high-dimensional datasets, when compared to the standard k-nearest neighbor classification baselines. The proposed approaches seem to be scalable, as our initial experiments suggest that using the approximate k-nearest neighbor sets to learn the model does not induce a significant difference in classification performance.

SC 3. Class imbalance: This thesis presents the first study that was aimed at better understanding the impact of class imbalance on classification under the assumption of hubness in intrinsically high-dimensional data.

SC 3.1: While examining the link between hubness and **class imbalance**, we have discovered and described a previously unknown high-dimensional phenomenon which we have named the *curse of minority hubs*. In low-dimensional data, majority class usually overwhelms the minority class and the relative difference in density causes the minority class examples to be misclassified. However, we have noticed that as the dimensionality is increased, the minority class points sometimes tend to become large bad hubs and induce significant misclassification of the majority class.

SC 3.2: Additionally, we have demonstrated that our proposed hubness-aware kNN classification methods help in dealing with this issue. As learning under class imbalance is among the most important fields of research in supervised learning, this is an important discovery. Many real-world problems are inherently imbalanced and we hope that these observations might help in designing better and more reliable adaptive systems in those domains.

SC 4. Instance selection: **Instance selection** is often performed as a preprocessing step prior to k-nearest neighbor classification. We have performed a first in-depth study of the influence of hubness on various standard prototype selection strategies.

SC 4.1: We have implemented and compared a series of standard instance selection strategies. The experiments suggest that these selection strategies vary in terms

of hub selection bias, as some tend to select more hubs than others, on average. Selecting a large proportion of hubs can help in maintaining the original structure of influence/relevance in the data.

SC 4.2: Different instance selection strategies induce different degrees of hubness in the prototype neighbor space. This induced hubness sometimes greatly differs from the actual hubness of the data and this influences subsequent classification as it leads to underestimation or overestimation of some hubness-related data properties by the kNN classification models.

SC 4.3: In order to deal with the difficulties outlined in SC 4.2 and SC 4.3, we have proposed a new hubness-aware framework for high-dimensional instance selection and classification. The experiments have shown that the use of the unbiased hubness estimator alongside the hubness-aware classification methods yields the best performance under data reduction.

SC 5. Metric learning: Many standard primary metrics do not perform well in high-dimensional feature spaces, mostly due to the well-known distance concentration phenomenon. Secondary distances are often used in order to deal with this problem. The shared-neighbor similarity/distance framework is frequently used for analysing high-dimensional data. We have extended this framework by including the information about past neighbor occurrences.

SC 5.1: We have proposed a novel hubness-aware **secondary shared-neighbor similarity measure**, $simhub_s$. The experimental evaluation suggests that the new similarity measure might be more appropriate for high-dimensional data analysis than the widely used $simcos_s$ similarity score. Substantial improvements in terms of classification accuracy and F-measure have been observed in most studied experimental setups. We have shown that $simhub_s$ significantly changes the structure of the kNN graph and reduces the label mismatch percentages, as well as the overall hubness of the data.

SC R. Other/Remaining: We have applied our proposed methodologies and approaches to a series of practical problems that incorporate different data domains.

SC R.1: The role of hubness in **object recognition** from images was carefully examined under several different feature representations. We have shown that different local feature types exhibit different degrees of hubness and that this influences the recognition process. In order to help the image mining practitioners in selecting the proper combination of feature representation and metric, we have developed a new visualization tool, **Image Hub Explorer**. It provides support in hub detection, local subgraph visualization, feature assessment, metric learning, querying, labeling and re-ranking.

SC R.2: The analysis of the **oceanographic survey sensor data** has shown that this type of time series data exhibits some hubness and that bad hubs can be used for semi-automatic anomaly detection. The anomalous streams can be detected by observing the discrepancies between spatial proximity and measurement similarity. Bad hubs appear as potential sensor anomalies.

SC R.3: We have explored the correlations between the hub-structure in different languages, by analyzing an aligned document corpus. By focusing on the case of **cross-lingual document retrieval**, we have shown that it is possible to exploit the hubness information to improve the common semantic feature space that the data is projected to. Preserving the original neighbor structure helps us in improving the performance of the cross-lingual document retrieval system.

SC R.4: Analyzing the **temporal aspect** of hubness in the problem of bug duplicate detection helped us to devise a secondary self-adaptive **re-ranking** procedure that improves the overall retrieval performance by improving the visibility of the previously bug duplicates.

The main conclusion of all the experiments and research presented in this thesis would be that hubness as a phenomenon plays a very important role in high-dimensional data analysis, negatively affecting many types of standard instance learning algorithms in many data domains. Yet, we have shown that it is possible to exploit the information contained in the neighbor k-occurrence models and learn from the past occurrences in order to achieve better system performance.

4.2 Future Work

As this thesis has shown, data hubness has an impact on many different types of data mining methods. We have but scratched the surface with the presented analysis and there are certainly many opportunities for further advancing the proposed approaches, as well as finding new practical applications.

The proposed hubness-based clustering methods (K-hubs, GHPC, GHPKM) can only handle hyper-spherical clusters and need to be extended in order to properly handle arbitrarily shaped clusters. We intend to achieve this by considering a potential application of kernel methods, as in kernel K-means. Additionally, we intend to examine the role of hubs in shared neighbor clustering, which has previously been successfully applied to high-dimensional data.

There are many ways in which hubness information can be exploited in classification. The proposed methods (h-FNN, HIKNN, NHBNN) are not the only ideas worth exploring. Furthermore, many other existing kNN classifiers could be extended by applying the same ideas embodied in these hubness-aware approaches. As for the three hubness-aware methods that have been proposed in this work, NHBNN is the one that offers most possibilities for further improvement, as it assumes independence between co-occurring neighbors, an assumption that is clearly violated in practice. Therefore, a more careful Bayesian treatment could potentially yield better results in practice.

Unlike many existing kNN classifiers, the hubness-aware approaches that are based on the neighbor k-occurrence models can be boosted. This follows from the fact that instance weights could be used to obtain the weighted hubness scores. We intend to explore the possibilities for successful boosting of the hubness-aware classifiers in our future work.

Almost all examined instance selection strategies have proved ineffective under the assumption of hubness, so it would be useful to design novel, hubness-aware instance selection schemes. Furthermore, as hubness-aware classifiers depend on the neighbor k-occurrence models, it would be interesting to optimize selection in order to reduce the overall prototype reverse neighbor set entropy, i.e. induce more purity in the prototype occurrence profiles.

Apart from the theoretical research and algorithm design, we intend to continue working on visualizations and various practical applications on multimedia data, in order to increase the robustness and improve the overall performance of various multimedia systems.

5 Acknowledgements

First, I would like to thank my advisor Dunja Mladenić for her advice and support. This thesis would not have been possible without her guidance and supervision. Her suggestions have been of great value for me in my work.

The members of my thesis committee have all offered valuable advice and contributed in the process of formalizing and forming the final manuscript. I thank them for their time and patience.

I've spent most of my second year mulling over possible topics and research directions and it was Miloš Radovanović that got me hooked on the story of hubs and their role in high-dimensional data analysis, which turned out to be the core of my research in the following years. We've had many brainstorming sessions in Novi Sad and Ljubljana during our bilateral project and this is where most of my initial ideas took shape. Coincidentally, it was Miloš that had sparked an interest in me for machine learning and data mining in the first place, while I was still an undergrad at the University of Novi Sad. I therefore must thank him wholeheartedly for all his advice and help during the course of my research.

Completing this thesis would not have been possible without a great deal of enthusiasm. One must keep asking questions and face all problems that arise with a smile. Looking back at the early years, it was my time spent with the crowd from Petnica Science Center that had molded my thoughts and put me on the right track. Those few summers were unforgettable. True enough, my blood no longer runs as hot for biology as it once used to, but my curiosity remains the same.

Doing research in computer science is as much theoretical as it is practical and requires a great deal of skill in translating ideas into code into experiments into results. I would like to thank DMS and Google for offering me internships and an opportunity to learn and grow and apply my ideas to solving real-world practical problems.

I must thank all the professors and teachers that have influenced me by their thinking and helped me to become who I am today. I offer my sincere thanks to Eugene Plančak, Obrad Ostojić, prof. Ratko Tošic, Ana Harpanj, prof. Sinisa Crvenković, prof. Dragan Masulović, prof. Mirjana Ivanović, prof. Marko Nedeljkov, prof. Dragoslav Herceg, prof. Zorana Lužanin, Maja Pech, Boris Šobot and Petar Marković. As they say, we are the sum of all our interactions and experiences.

This work would not have been what it is today if not for my colleagues and co-authors. Thank you for participating in this story and helping in filling its pages along the way: Miloš Radovanović, Dunja Mladenić, Mirjana Ivanović, Blaž Fortuna, Raluca Brehar, Carolina Fortuna, Jan Rupnik and Krisztian Buza.

People might think of science as something contemplative and slow. In fact, it most closely resembles war. War against time, against the deadlines, against unexpected setbacks and 3AM bugs, against one's urge to sleep, against complacency and arrogance. I must thank my old chess coach, Vladimir Deže for infusing me with the fighting spirit that I now possess. Life itself is no more than a big, complex, game of chess.

I wish to thank all my dearest friends in Novi Sad and Ljubljana for filling my life with countless moments of joy and helping me to overcome all the difficulties that I had to face along the way. Thanks to Jelena for her endless optimism, Milena for her endless pessimism,

Saša and Davor for the beers, Ivan S. and Holo-Sanja for the long talks and even longer meals, Ivan D. for his useful tips and his love of spoons, Bruno for his climbing spirit, Veljko and Srdjan for the parties, Timotej for math and video-games, Vladislav for his infinite collection of links, Jon for criticising my grammar, Jovan for his superb geekiness, Marko for his binary interests and the Dejan&co volleyball crew for the dynamism of Wednesday evenings.

Most importantly, I thank my parents and family for supporting me along the way. My special thanks goes to my mother Milana for her encouragement and delicious food and Miloje for helping me deal with stress. I thank my father Vojislav for his pragmatism and I thank my grandmother Danica and grandfather Nenad for being around and for all their endless love. I must also thank Maja Domazetović for setting me off in the right direction and always being there when I needed her.

In the very end, I would like to share with you a short quote taken from 'Kafka on the Shore', a book written by one of my favorite writers – Haruki Murakami: "And once the storm is over, you won't remember how you made it through, how you managed to survive. You won't even be sure, whether the storm is really over. But one thing is certain. When you come out of the storm, you won't be the same person who walked in. That's what this storm's all about.".

6 References

Adamic, L. A.; Huberman, B. A. Zipf's law and the Internet. *Glottometrics* **3**, 143–150 (2002).

Aggarwal, C. C. Re-designing distance functions and distance-based applications for high dimensional data. *ACM Sigmod Record* **30**, 13–18 (2001).

Aggarwal, C. C. On k-anonymity and the curse of dimensionality. In: *Proceedings of the 31st International Conference on Very Large Data Bases*. 901–909 (Springer-Verlag, Berlin, Germany, 2005).

Aggarwal, C. C. On randomization, public information and the curse of dimensionality. In: *IEEE 23rd International Conference on Data Engineering*. 136–145 (IEEE, New York, NY, USA, 2007).

Aggarwal, C. C.; Hinneburg, A.; Keim, D. A. On the surprising behavior of distance metrics in high dimensional spaces. In: *Proceedings of the 8th International Conference on Database Theory (ICDT)*. 420–434 (ACM, New York, NY, USA, 2001).

Albert, R. Scale-free networks in cell biology. *Journal of Cell Science* **118**, 4947–4957 (2005).

Arefin, A.; Riveros, C.; Berretta, R.; Moscato, P. kNN-MST-Agglomerative: A fast and scalable graph-based data clustering approach on gpu. In: *7th International Conference on Computer Science Education (ICCSE)*. 585–590 (ACM, New York, NY, USA, 2012).

Arthur, D.; Vassilvitskii, S. K-means++: The advantages of careful seeding. In: *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 1027–1035 (SIAM, Philadelphia, PA, USA, 2007).

Arya, S. Space-efficient approximate Voronoi diagrams. In: *Proceedings of the 34th Annual ACM Symposium on Theory of Computation*. 721–730 (ACM, New York, NY, USA, 2002).

Aucouturier, J. *Ten experiments on the modelling of polyphonic timbre. Doctoral dissertation* (University of Paris, 2006).

Aucouturier, J.; Pachet, F. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences* **1** (2004).

Axelsen, J. B.; Bernhardsson, S.; Rosvall, M.; Sneppen, K.; Trusina, A. Degree landscapes in scale-free networks. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics* **74**, 036119 (2006).

Barabasi, A.-L. *Linked: How Everything Is Connected to Everything Else and What It Means for Business, Science, and Everyday Life* (Plume Books, New York, NY, USA, 2003).

Barabási, A.-L.; Albert, R. Emergence of Scaling in Random Networks. *Science* **286**, 509–512 (1999).

Barabási, A.-L.; Bonabeau, E. Scale-free networks. *Scientific American* **288**, 50–59 (2003).

Batada, N. N.; Hurst, L. D.; Tyers, M. Evolutionary and physiological importance of hub proteins. *PLoS Computational Biology* **2**, 748–756 (2006).

Batista, G. E. A. P. A.; Prati, R. C.; Monard, M. C. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorer Newsletter* **6**, 20–29 (2004).

Beirlant, J.; Dudewicz, E. J.; Gyorfi, L.; van der Meulen, E. C. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences* **6**, 17–39 (1997).

Bellman, R. E. *Adaptive Control Processes - A Guided Tour* (Princeton University Press, Princeton, New Jersey, U.S.A., 1961).

Bengio, Y.; Delalleau, O.; Le Roux, N. The curse of dimensionality for local kernel machines. *Techn. Rep* **1258** (2005).

Benzi, M.; Estrada, E.; Klymko, C. Ranking hubs and authorities using matrix functions. *Computing Research Repository CoRR* **abs/1201.3120** (2012).

Berenzweig, D. *Anchors and Hubs in Audio-based Music Similarity* (Columbia University, New York, NY, USA, 2007).

Beyer, K.; Goldstein, J.; Ramakrishnan, R.; Shaft, U. When is "nearest neighbor" meaningful? In: *Proceedings of the International Conference on Database Theory*. 217–235 (ACM, New York, NY, USA, 1999).

Biçici, E.; Yuret, D. Locally scaled density based clustering. In: *Proceedings of the 8th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA), Part I*. 739–748 (Springer-Verlag, Berlin, Germany, 2007).

Bingham, E.; Mannila, H. Random projection in dimensionality reduction: Applications to image and text data. In: *Knowledge Discovery and Data Mining*. 245–250 (ACM, New York, NY, USA, 2001).

Block, M.; Bader, M.; Tapia, E.; Ramírez, M.; Gunnarsson, K.; Cuevas, E.; Zaldivar, D.; Rojas, R. Using reinforcement learning in chess engines. *Research in Computing Science* 31–40 (2008).

Boiman, O.; Shechtman, E.; Irani, M. In defense of nearest-neighbor based image classification. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition*. 1–8 (IEEE, New York, NY, USA, 2008).

Borg, I.; Groenen, P. *Modern Multidimensional Scaling: Theory and Applications* (Springer-Verlag, Berlin, Germany, 2005).

Breiman, L. Random forests. *Machine Learning* **45**, 5–32 (2001).

Brin, S.; Page, L. The anatomy of a large-scale hypertextual web search engine. In: *Proceedings of the seventh International Conference on World Wide Web 7*. 107–117 (Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 1998).

Buza, K.; Nanopoulos, A.; Schmidt-Thieme, L. Time-series classification based on individualised error prediction. In: *Proceedings of the 2010 13th IEEE International Conference on Computational Science and Engineering*. 48–54 (IEEE Computer Society, Washington, DC, USA, 2010).

Buza, K.; Nanopoulos, A.; Schmidt-Thieme, L. Insight: efficient and effective instance selection for time-series classification. In: *Proceedings of the 15th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*. 149–160 (Springer-Verlag, Berlin, Heidelberg, 2011).

C. M. Newman, Y. R.; Tversky, A. Nearest neighbors and Voronoi regions in certain point processes. *Advances in Applied Probability* **15**, 726–751 (1983).

Caises, Y.; González, A.; Leyva, E.; Pérez, R. Combining instance selection methods based on data characterization: An approach to increase their effectiveness. *Information Sciences* **181**, 4780–4798 (2011).

Camastra, F.; Vinciarelli, A. Estimating the intrinsic dimension of data with a fractal-based method. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**, 1404–1407 (2002).

Cano, J. R.; Herrera, F.; Lozano, M. Using evolutionary algorithms as instance selection for data reduction in KDD: An experimental study. *IEEE Transactions on Evolutionary Computing* **7**, 561–575 (2003).

Carter, K.; Raich, R.; Hero, A. On local intrinsic dimension estimation and its applications. *IEEE Transactions on Signal Processing* **58**, 650–663 (2010).

Champandard, A. *AI Game Development: Synthetic Creatures With Learning and Reactive Behaviors. New Riders Games Series* (New Riders, San Francisco, CA, USA, 2003).

Chau, R.; Yeh, C.-H. A multilingual text mining approach to web cross-lingual text retrieval. *Knowledge-Based Systems* 219–227 (2004).

Chavez, E.; Navarro, G. Probabilistic proximity search: Fighting the curse of dimensionality in metric spaces. *Information Processing Letters* **85**, 39 – 46 (2003).

Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; Kegelmeyer, W. P. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* **16**, 321–357 (2002).

Chen, C.; Horng, S.-J.; Huang, C.-P. Locality sensitive hashing for sampling-based algorithms in association rule mining. *Expert Systems with Applications* **38**, 12388–12397 (2011).

Chen, J.; ren Fang, H.; Saad, Y. Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research* **10**, 1989–2012 (2009).

Chiang, T.-H.; Lo, H.-Y.; Lin, S.-D. A ranking-based kNN approach for multi-label classification. *Journal of Machine Learning Research: Proceedings Track* **25**, 81–96 (2012).

Chirita, P. A.; Olmedilla, D.; Nejdl, W. Finding related hubs and authorities. In: *Proceedings of the 1st Latin American Web Congress*. 214–215 (IEEE, New York, NY, USA, 2003).

Chou, C.-H.; Kuo, B.-H.; Chang, F. The generalized condensed nearest neighbor rule as a data reduction method. In: *Proceedings of the 18th International Conference on Pattern Recognition - Volume 02*. 556–559 (IEEE Computer Society, Washington, DC, USA, 2006).

Chou, K.-C.; Shen, H.-B. Predicting eukaryotic protein subcellular location by fusing optimized evidence-theoretic k-nearest neighbor classifiers. *Journal of Proteome Research* **5**, 1888–1897 (2006).

Chávez, E.; Navarro, G. A probabilistic spell for the curse of dimensionality. In: *Algorithm Engineering and Experimentation*. 147–160 (Springer, Berlin Heidelberg, Germany, 2001).

Cover, T. M.; Hart, P. E. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* **IT-13**, 21–27 (1967).

Csatári, B.; Prekopcsák, Z. Class-based attribute weighting for time series classification. In: *Proceedings of the 14th International Student Conference on Electrical Engineering* (Prague Technical University, Prague, 2010).

Cullum, J. K.; Willoughby, R. A. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. 1* (Society for Industrial and Applied Mathematics, Philadelphia,USA, 2002).

Dai, B.-R.; Hsu, S.-M. An instance selection algorithm based on reverse nearest neighbor. In: *Proceedings of the 15th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*. 1–12 (Springer-Verlag, Berlin/Heidelberg, 2011).

David, E.; Jon, K. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World* (Cambridge University Press, New York, NY, USA, 2010).

Derrac, J.; García, S.; Herrera, F. A first study on the use of coevolutionary algorithms for instance and feature selection. In: *Proceedings of the 4th International Conference on Hybrid Artificial Intelligence Systems*. 557–564 (Springer-Verlag, Berlin/Heidelberg, 2009).

Devroye, L. On the inequality of cover and hart. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **3**, 75–78 (1981).

Dezso, Z.; Barabási, A.-L. Halting viruses in scale-free networks. *Physical Review E* **65**, 1–4 (2001).

Doddington, G.; Liggett, W.; Martin, A.; Przybocki, M.; Reynolds, D. Sheep, goats, lambs and wolves: A statistical analysis of speaker performance in the nist 1998 speaker recognition evaluation. In: *International Conference on Spoken Language Processing* (ISCA, Grenoble, France, 1998).

Dominik Schnitzer, J. S., Arthur Flexer. The relation of hubs to the Doddington zoo in speaker verification. *Technical Report* (Vienna, Austria, 2012).

Dong, W.; Moses, C.; Li, K. Efficient k-nearest neighbor graph construction for generic similarity measures. In: *Proceedings of the 20th International Conference on World Wide Web*. 577–586 (ACM, New York, NY, USA, 2011).

Durrant, R. J.; Kabán, A. When is 'nearest neighbour' meaningful: A converse theorem and implications. *Journal of Complexity* **25**, 385–397 (2009).

Ekman, D.; Light, S.; Björklund, A.; Elofsson, A. What properties characterize the hub proteins of the protein-protein interaction network of saccharomyces cerevisiae? *Genome Biology* **7**, 1–13 (2006).

Ertekin, S.; Huang, J.; Giles, C. L. Active learning for class imbalance problem. In: *Proceedings of the 30th annual International ACM SIGIR Conference on Research and development in information retrieval*. 823–824 (ACM, New York, NY, USA, 2007a).

Ertekin, S. E.; Huang, J.; Bottou, L.; Giles, C. L. Learning on the border: Active learning in imbalanced data classification. In: *Proceedings of the ACM Conference on Information and Knowledge Management*. 127–136 (ACM, New York, NY, USA, 2007b).

Ertz, L.; Steinbach, M.; Kumar, V. Finding topics in collections of documents: A shared nearest neighbor approach. In: *Proceedings of Text Mine 01, First SIAM International Conference on Data Mining*. 1–8 (SIAM, Philadelphia, PA, USA, 2001).

Evangelista, P. F.; Embrechts, M. J.; Szymanski, B. K. Taming the curse of dimensionality in kernels and novelty detection. In: *Applied Soft Computing Technologies: The Challenge of Complexity*. 425–438 (Springer-Verlag, Berlin, Germany, 2006).

Ezawa, K.; Singh, M.; Norton, S. W. Learning goal oriented bayesian networks for telecommunications risk management. In: *Proceedings of the 13th International Conference on Machine Learning*. 139–147 (Morgan Kaufmann, San Francisco, CA, USA, 1996).

Ezawa, K. J.; Schuermann, T. Fraud/uncollectible debt detection using a bayesian network based learning system: A rare binary outcome with mixed data structures. In: *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. 157–166 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995).

Faivishevsky, L.; Goldberger, J. ICA based on a smooth estimation of the differential entropy. In: *Advances in Neural Information Processing Systems*. 433–440 (MIT Press, Cambridge, MA, 2008).

Farahmand, A. M.; Szepesvári, C. Manifold-adaptive dimension estimation. In: *Proceedings of the 24th International Conference on Machine Learning*. 265–272 (ACM, New York, NY, USA, 2007).

Farkas, I.; Deranyi, I.; Jeong, H.; Naoda, Z.; Oltvai, Z.; Ravasz, E.; Schubert, A.; Barabasi, A.-L.; Vicsek, T. Networks in life: Scaling properties and eigenvalue spectra. *Physica A: Statistical Mechanics and its Applications* **314**, 25–34 (2002).

Feldman, R.; Sanger, J. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data* (Cambridge University Press, USA, 2006).

Felzenszwalb, P.; Girshick, R.; McAllester, D.; Ramanan, D. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**, 1627–1645 (2010).

Fernandez, A.; Garcia, S.; Herrera, F. Addressing the classification with imbalanced data: Open problems and new challenges on class distribution. In: *Proceedings of the HAIS Conference*. 1–10 (Springer-Verlag, Berlin / Heidelberg, Germany, 2011).

Fix, E.; Hodges, J. Discriminatory analysis, nonparametric discrimination: consistency properties. *Technical Report* (Randolph Field, USA, 1951).

Flexer A., S. J., Schnitzer D. Putting the user in the center of music information retrieval. In: *Proceedings of the 13th International Society for Music Information Retrieval Conference* (International Society for Music Information Retrieval, Canada, 2012).

Fortuna, B.; Cristianini, N.; Shawe-Taylor, J. *Kernel methods in bioengineering, communications and image processing*, chapter A Kernel Canonical Correlation Analysis for Learning the Semantics of Text, 263–282 (Idea Group Publishing, Hershey, PA, USA, 2006).

Fortuna, B.; Grobelnik, M.; Mladenić, D. Visualization of text document corpus. *Informatica* **29**, 497–502 (2005).

François, D.; Wertz, V.; Verleysen, M. The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering* **19**, 873–886 (2007).

Franti, P.; Virmajoki, O.; Hautamaki, V. Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**, 1875–1881 (2006).

Fukunaga, K.; Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory* **21**, 32–40 (1975).

Ganguly, A. R.; Gama, J.; Omitaomu, O. A.; Gaber, M. M.; Vatsavai, R. R. *Knowledge Discovery from Sensor Data* (CRC Press, Inc., Boca Raton, FL, USA, 2008).

Garcia, S.; Derrac, J.; Cano, J.; Herrera, F. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**, 417–435 (2012).

Garcia, V.; Mollineda, R. A.; Sanchez, J. S. On the k-NN performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis Applications* **11**, 269–280 (2008).

Gasser M., S. D., Flexer A. Hubs and orphans - an explorative approach. In: *Proceedings of the 7th Sound and Music Computing Conference* (ACM, New York, NY, USA, 2010).

Gemmell, J.; Schimoler, T.; Ramezani, M.; Mobasher, B. Adapting k-Nearest Neighbor for tag recommendation in folksonomies. In: Anand, S. S.; Mobasher, B.; Kobsa, A.; Jannach, D.; Anand, S. S.; Mobasher, B.; Kobsa, A.; Jannach, D. (eds.) *Proceedings of Intelligent Techniques for Web Personalization Workshop*. 1–12 (CEUR-WS, Tilburg, Germany, 2009).

Gjoka, M.; Soldo, F. Exploring collaborative filters: Neighborhood-based approach. In: *CS 273 - Machine Learning* (Department of MSIS, University of California, Irvine, 2008).

Gorisse, D.; Cord, M.; Precioso, F. Locality-sensitive hashing for Chi-2 distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**, 402–409 (2011).

Grčar, M.; Fortuna, B.; Mladenić, D.; Grobelnik, M. kNN versus SVM in the collaborative filtering framework. In: Batagelj, V.; Bock, H.-H.; Ferligoj, A.; Ziberna, A. (eds.) *Data Science and Classification*. 251–260 (Springer, Berlin/Heidelberg, Germany, 2006).

Gu, S.; Zheng, Y.; Tomasi, C. Efficient visual object tracking with online nearest neighbor classifier. In: *Proceedings of the 10th Asian Conference on Computer Vision*. 271–282 (Springer-Verlag, Berlin/Heidelberg, Germany, 2011).

Guan, D.; Yuan, W.; Lee, Y.-K.; Lee, S. Identifying mislabeled training data with the aid of unlabeled data. *Applied Intelligence* **35**, 345–358 (2011).

Guid, M.; Možina, M.; Sadikov, A.; Bratko, I. Deriving concepts and strategies from chess tablebases. In: *Advances in Computer Games*. 195–207 (Springer Verlag, Berlin, 2010).

Gupta, M. D.; Huang, T. S. Regularized maximum likelihood for intrinsic dimension estimation. *Computing Research Repository CoRR* **abs/1203.3483** (2012).

Hader, S.; Hamprecht, F. A. Efficient density clustering using basin spanning trees. In: *Proceedings of the 26th Annual Conference of the German Classification Society*. 39–48 (Springer-Verlag, Berlin, Germany, 2003).

Haghani, P.; Michel, S.; Aberer, K. Distributed similarity search in high dimensions using locality sensitive hashing. In: *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*. 744–755 (ACM, New York, NY, USA, 2009).

Hamed, M.; Serrurier, M.; Durand, N. Possibilistic kNN regression using tolerance intervals. In: *Advances in Computational Intelligence*. 410–419 (Springer, Berlin/Heidelberg, Germany, 2012).

Han, E.-H.; Karypis, G.; Kumar, V. Text categorization using weight adjusted k-nearest neighbor classification. In: *Advances in Knowledge Discovery and Data Mining*. **2035**, 53–65 (Springer-Verlag, Berlin, Germany, 2001).

Han, J. *Data Mining: Concepts and Techniques* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005).

Hand, D. J.; Vinciotti, V. Choosing k for two-class nearest neighbour classifiers with unbalanced classes. *Pattern Recognition Letters* **24**, 1555–1562 (2003).

Hardoon, D. R.; Szedmák, S.; Shawe-Taylor, J. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation* **16**, 2639–2664 (2004).

Hastie, T.; Tibshirani, R. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **18**, 607–616 (1996).

Hayashi, K. A simple extension of boosting for asymmetric mislabeled data. *Statistics and Probability Letters* **82**, 348–356 (2012).

He, H.; Bai, Y.; Garcia, E. A.; Li, S. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In: *Proceedings of the International Joint Conference on Neural Networks*. 1322–1328 (IEEE, New York, NY, USA, 2008).

He, H.; Garcia, E. A. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering* **21**, 1263–1284 (2009).

He, X.; Zhang, J. Why do hubs tend to be essential in protein networks? *PLoS Genetics* **2**, 826–834 (2006).

Hiew, L. *Assisted Detection of Duplicate Bug Reports* (University of British Columbia, Canada, 2006).

Hill, D. J.; Minsker, B. S. Anomaly detection in streaming environmental sensor data: A data-driven modeling approach. *Environmental Modelling and Software* **25**, 1014–1022 (2010).

Hill, D. J.; Minsker, B. S.; Amir, E. Real-time bayesian anomaly detection for environmental sensor data. In: *Proceedings of the 32nd Conference of IAHR* (IAHR, Venice, Italy, 2007).

Hinneburg, A.; Aggarwal, C.; Keim, D. A. What is the nearest neighbor in high dimensional spaces? In: *Proceedings of the 26th International Conference on Very Large Data Bases*. 506–515 (Morgan Kaufmann, New York, NY, USA, 2000).

Hnizdo, V.; Tan, J.; Killian, B. J.; Gilson, M. K. Efficient calculation of configurational entropy from molecular simulations by combining the mutual-information expansion and nearest-neighbor methods. *Journal of Computational Chemistry* **29**, 1605–1614 (2008).

Holmes, C. C.; Adams, N. M. A probabilistic nearest neighbor method for statistical pattern recognition. *Journal of the Royal Statistical Society: Series B* **64**, 295–306 (2002).

Holte, R. C.; Acker, L. E.; Porter, B. W. Concept learning and the problem of small disjuncts. In: *Proceedings of the 11th International Conference on AI - Volume 1*. 813–818 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989).

Hong, X.; Chen, S.; Harris, C. J. A kernel-based two-class classifier for imbalanced data sets. *IEEE Transactions on Neural Networks* **18**, 28–41 (2007).

Hotelling, H. The most predictable criterion. *Journal of Educational Psychology* **26**, 139–142 (1935).

Houle, M. E.; Kriegel, H.-P.; Kröger, P.; Schubert, E.; Zimek, A. Can shared-neighbor distances defeat the curse of dimensionality? In: *Proceedings of the 22nd International Conference on Scientific and Statistical Database Management*. 482–500 (Springer-Verlag, Berline/Heidelberg, Germany, 2010).

Hsieh, T.; Taur, J.; Kung, S. A kNN-scoring based core-growing approach to cluster analysis. *Journal of Signal Processing Systems* **60**, 105–114 (2010).

Hsu, C.-M.; Chen, M.-S. On the design and applicability of distance functions in high-dimensional data space. *IEEE Transactions on Knowledge and Data Engineering* **21**, 523 –536 (2009).

Hu, Z.; Bhatnagar, R. Clustering algorithm based on mutual k-nearest neighbor relationships. *Statistical Analysis and Data Mining* **5**, 100–113 (2012).

Hulse, J. V.; Khoshgoftaar, T. M.; Napolitano, A. Skewed class distributions and mislabeled examples. In: *Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*. 477–482 (IEEE Computer Society, Washington, DC, USA, 2007).

Hwang, S.; Lee, D.-S.; Kahng, B. Origin of the hub spectral dimension in scale-free networks. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics* **87**, 022816 (2013).

Iwerks, G. S.; Samet, H.; Smith, K. Continuous k-nearest neighbor queries for continuously moving points with updates. In: *Proceedings of the 29th International Conference on Very Large Data Bases*. 512–523 (Springer-Verlag, Berlin, Germany, 2003).

Jagadish, H. V.; Ooi, B. C.; Tan, K.-L.; Yu, C.; Zhang, R. idistance: An adaptive b+-tree based indexing method for nearest neighbor search. *ACM Transactions on Database Systems* **30**, 364–397 (2005).

Jalba, A.; Wilkinson, M.; Roerdink, J.; Bayer, M.; Juggins, S. Automatic diatom identification using contour analysis by morphological curvature scale spaces. *Machine Vision and Applications* **16**, 217–228 (2005).

Jalbert, N.; Weimer, W. Automated duplicate detection for bug tracking systems. In: *The 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'08)*. 52–61 (IEEE Computer Science Press, New York, NY, USA, 2008).

Jarvis, R. A.; Patrick, E. A. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computing* **22**, 1025–1034 (1973).

Jayaram, B.; Klawonn, F. Can unbounded distance measures mitigate the curse of dimensionality? *International Journal of Data Mining, Modelling and Management* **4**, 361–383 (2012).

Jegou, H.; Harzallah, H.; Schmid, C. A contextual dissimilarity measure for accurate and efficient image search. In: *Computer Vision and Pattern Recognition*. 1–8 (IEEE, New York, NY, USA, 2007).

Jensen, R.; Cornelis, C. A new approach to fuzzy-rough nearest neighbour classification. In: Chan, C.-C.; Grzymala-Busse, J.; Ziarko, W. (eds.) *Rough Sets and Current Trends in Computing*. **5306**, 310–319 (Springer-Verlag, Berlin, Germany, 2008).

Jia, Y.; Hoberock, J.; Garland, M.; Hart, J. On the visualization of social and other scale-free networks. *IEEE Transactions on Visualization and Computer Graphics* **14**, 1285–1292 (2008).

Jo, T. Inverted index based modified version of kNN for text categorization. *Journal of Information Processing Systems* **4**, 17–26 (2008).

Jolliffe, I. T. *Principal Component Analysis* (Springer-Verlag, Berlin, Germany, 2002).

Jones, M.; Viola, P. Rapid object detection using a boosted cascade of simple features. In: *Conference on Computer Vision and Pattern Recognition*. 511–518 (IEEE, New York, NY, USA, 2001).

Jordan, M. I.; Bach, F. R. Kernel independent component analysis. *Journal of Machine Learning Research* **3**, 1–48 (2001).

Kabán, A. On the distance concentration awareness of certain data reduction techniques. *Pattern Recognition* **44**, 265–277 (2011).

Kabán, A. Non-parametric detection of meaningless distances in high dimensional data. *Statistics and Computing* **22**, 375–385 (2012).

Kalager, M.; Adami, H.; Bretthauer, M.; Tamimi, R. Overdiagnosis of invasive breast cancer due to mammography screening: Results from the norwegian screening program. *Annals of Internal Medicine* **156**, 491–499 (2012).

Katayama, N.; Satoh, S. The sr-tree: an index structure for high-dimensional nearest neighbor queries. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 369–380 (ACM, New York, NY, USA, 1997).

Keller, J. E.; Gray, M. R.; Givens, J. A. A fuzzy k-nearest-neighbor algorithm. *IEEE Transactions on Systems, Man and Cybernetics* **15**, 580–585 (1985).

Keogh, E.; Wei, L.; Xi, X.; Lonardi, S.; Shieh, J.; Sirowy, S. Intelligent icons: Integrating lite-weight data mining and visualization into gui operating systems. In: *Sixth International Conference on Data Mining*. 912–916 (IEEE, New York, NY, USA, 2006).

Kim, H. J.; Kim, I. M.; Lee, Y.; Kahng, B. Scale-free network in stock markets. *Journal of the Korean Physical Society* **40**, 1105–1108 (2002).

Kim, K.-j. Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications* **30**, 519–526 (2006).

Kleinberg, J. M. Authoritative sources in a hyperlinked environment. *Journal of the American Society for Computing Machinery* **46**, 604–632 (1999a).

Kleinberg, J. M. Hubs, authorities, and communities. *ACM Comput. Surv.* **31** (1999b).

Ko, M. H.; West, G.; Venkatesh, S.; Kumar, M. Using dynamic time warping for online temporal fusion in multisensor systems. *Information Fusion: Special Issue on Distributed Sensor Networks* **9**, 370–388 (2008).

Kolahdouzan, M.; Shahabi, C. Voronoi-based k-nearest neighbor search for spatial network databases. In: *Proceedings of the 30th International Conference on Very Large Data Bases*. 840–851 (Springer-Verlag, Berlin, Germany, 2004).

Kramer, O. Unsupervised K-Nearest Neighbor Regression. *Computer Research Repository CoRR* (2011).

Kramer, O. On evolutionary approaches to unsupervised nearest neighbor regression. In: *Applications of Evolutionary Computation*. 346–355 (Springer, Berlin Heidelberg, Germany, 2012).

Kubat, M.; Matwin, S. Addressing the curse of imbalanced training sets: One-sided selection. In: *Proceedings of the Fourteenth International Conference on Machine Learning*. 179–186 (Morgan Kaufmann, San Francisco, California, USA, 1997).

Kulis, B.; Grauman, K. Kernelized locality-sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**, 1092–1104 (2011).

Kullback, S.; Leibler, R. A. On information and sufficiency. *The Annals of Mathematical Statistics* **22**, 79–86 (1951).

L. Devroye, A. K., L. Gyorfi; Lugosi, G. On the strong universal consistency of nearest neighbor regression function estimates. *Annals of Statistics* **22**, 1371–1385 (1994).

Lee, J. A.; Verleysen, M. Shift-invariant similarities circumvent distance concentration in stochastic neighbor embedding and variants. *Procedia CS* **4**, 538–547 (2011).

Levenshtein, V. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. In: *Soviet Physics Doklady*. 707–710, 8 (Springer, Berlin/Heidelberg, Germany, 1966).

Levy, M.; Brennan, M.; Levy, H.; Ríos-rull, V.; Roll, R.; Schwartz, E.; Slemrod, J.; Solomon, S. Market efficiency, the pareto wealth distribution, and the lévy distribution of stock returns. *The Economy as an Evolving Complex System* **3** (2005).

Li, L.; Alderson, D.; Tanaka, R.; Doyle, J. C.; Willinger, W. Towards a theory of scale-free graphs: Definition, properties, and implications (extended version). *Computing Research Repository CoRR* (2005).

Li, X.; Ng, M. K.; Ye, Y. Har: Hub, authority and relevance scores in multi-relational data for query search. In: *Proceedings of the SIAM Data Mining Conference*. 141–152 (SIAM, Philadelphia, PA, USA, 2012).

Li, Y.; Zhang, X. Improving k-nearest neighbor with exemplar generalization for imbalanced classification. In: *Advances in Knowledge Discovery and Data Mining*. 321–332 (Springer-Verlag, Berlin, Germany, 2011).

Liao, Y.; Vemuri, V. Use of k-nearest neighbor classifier for intrusion detection. *Computers and Security* **21**, 439–448 (2002).

Lienhart, R.; Maydt, J. An extended set of haar-like features for rapid object detection. In: *Proceedings of ICIP Conference*. 900–903 (IEEE, New York, NY, USA, 2002).

Liotta, G.; Preparata, F. P.; Tamassia, R. Robust proximity queries in implicit Voronoi diagrams. In: *Proceedings of the 8th Canadian Conference on Computational Geometry* (Brown University, Providence, RI, USA, 1996).

Liu, H. *Instance Selection and Construction for Data Mining* (Springer-Verlag, Berlin, Heidelberg, 2010).

Liu, H.; Motoda, H. On issues of instance selection. *Data Mining and Knowledge Discovery* **6**, 115–130 (2002).

Liu, W.; Chawla, S. Class confidence weighted kNN algorithms for imbalanced data sets. In: *Advances in Knowledge Discovery and Data Mining*. 345–356 (Springer-Verlag, Berlin, Germany, 2011).

Liu, W.; Chawla, S.; Cieslak, D. A.; Chawla, N. V. A robust decision tree algorithm for imbalanced data sets. In: *Proceedings of the SIAM Data Mining Conference*. 766–777 (SIAM, Philadelphia, PA, USA, 2010).

Liu, X.-Y.; Wu, J.; Zhou, Z.-H. Exploratory under-sampling for class-imbalance learning. In: *Proceedings of the Sixth International Conference on Data Mining*. 965–969 (IEEE Computer Society, Washington, DC, USA, 2006).

Liu, Y.; Xu, Z.; Shi, B.; Zhang, B. Time-based k-nearest neighbor collaborative filtering. In: *IEEE 12th International Conference on Computer and Information Technology (CIT)*. 1061–1065 (IEEE, New York, NY, USA, 2012).

Lowe, D. Object recognition from local scale-invariant features. In: *The Proceedings of the Seventh International Conference on Computer Vision*. 1150–1157 (IEEE, New York, NY, USA, 1999).

Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* **60**, 91–110 (2004).

Lucarella, D. A document retrieval system based on nearest neighbour searching. *Journal of Information Science* **14**, 25–33 (1988).

Lucinska, M.; Wierzchon, S. Spectral clustering based on k-nearest neighbor graph. In: *Computer Information Systems and Industrial Management*. 254–265 (Springer-Verlag, Berlin Heidelberg, Germany, 2012).

M., S.; A., F. A mirex meta-analysis of hubness in audio music similarity. In: *Proceedings of the 13th International Society for Music Information Retrieval Conference* (International Society for Music Information Retrieval, Canada, 2012).

Mack, Y.; Rosenblatt, M. Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis* **9**, 1–15 (1979).

Maier, M.; Hein, M.; Luxburg, U. Cluster identification in nearest-neighbor graphs. In: *Proceedings of the 18th International Conference on Algorithmic Learning Theory*. 196–210 (Springer-Verlag, Berlin Heidelberg, Germany, 2007).

Maier, M.; Hein, M.; von Luxburg, U. Optimal construction of -nearest-neighbor graphs for identifying noisy clusters. *Theoretical Computer Science* **410**, 1749–1764 (2009). Algorithmic Learning Theory.

Makaruk, H. E.; Owczarek, R. Hubs in languages: Scale free networks of synonyms. *Computing Research Repository CoRR* **abs/0802.4112** (2008).

Malek, B.; Orozco, M.; Saddik, A. E. Novel shoulder-surfing resistant haptic-based graphical password. In: *Proceedings of EuroHaptics* (Springer, Berlin, Germany, 2006).

Maloney, L. T. Nearest neighbor analysis of point processes: Simulations and evaluations. *Journal of Mathematical Psychology* **27**, 251–260 (1983).

Marinčič, D.; Kompara, T.; Gams, M. Question classification with active learning. In: *Text, Speech and Dialogue*. 673–680 (Springer-Verlag, Berlin, Germany,).

McCarthy, K.; Zabar, B.; Weiss, G. Does cost-sensitive learning beat sampling for classifying rare classes? In: *Proceedings of the 1st International Workshop on Utility-based Data Mining*. 69–77 (ACM, New York, NY, USA, 2005).

Milgram, S. The small world problem. *Psychology Today* **2**, 60–67 (1967).

Miller, J. C.; Rae, G.; Schaefer, F.; Ward, L. A.; LoFaro, T.; Farahat, A. Modifications of kleinberg's hits algorithm using matrix exponentiation and web log records. In: *Proceedings of the 24th annual International ACM SIGIR Conference on Research and development in information retrieval*. 444–445 (ACM, New York, NY, USA, 2001).

Min, M. R.; Stanley, D. A.; Yuan, Z.; Bonner, A. J.; Zhang, Z. A deep non-linear feature mapping for large-margin kNN classification. In: *ICDM*. 357–366 (IEEE, New York, NY, USA, 2009).

Mitchell, T. M. *Machine Learning* (McGraw-Hill, New York, NY, USA, 1997).

Moëllic, P.-A.; Haugeard, J.-E.; Pitel, G. Image clustering based on a shared nearest neighbors approach for tagged collections. In: *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*. 269–278 (ACM, New York, NY, USA, 2008).

Mörchen, F.; Ultsch, A.; Hoos, O. Discovering interpretable muscle activation patterns with the temporal data mining method. In: *Knowledge Discovery in Databases: PKDD 2004*. 512–514 (Springer, Berlin Heidelberg, Germany, 2004).

Možina, M.; Guid, M.; Sadikov, A.; Groznik, V.; Bratko, I. Goal-oriented conceptualization of procedural knowledge. In: *Intelligent Tutoring Systems*. 286–291 (Springer Verlag, Berlin, 2012).

N., T.; D., M. Hubness-aware shared neighbor distances for high-dimensional k-nearest neighbor classification. In: *Proceedings of the 7th International Conference on Hybrid Artificial Intelligence Systems*. 116–127 (Springer-Verlag, Berlin, Germany, 2012).

N., T.; D., M. Image hub explorer: Evaluating representations and metrics for content-based image retrieval and object recognition. In: *Proceedings of the ECML conference* (Springer-Verlag, Berlin, Germany, 2013).

Nanopoulos, A.; Radovanović, M.; Ivanović, M. How does high dimensionality affect collaborative filtering? In: *Proceedings of the Third ACM Conference on Recommender Systems*. 293–296 (ACM, New York, NY, USA, 2009).

Napierala, K.; Stefanowski, J. Identification of different types of minority class examples in imbalanced data. In: *Hybrid Artificial Intelligent Systems*. 139–150 (Springer-Verlag, Berlin / Heidelberg, Germany, 2012).

Newman, C. M.; Rinott, Y. Nearest neighbors and Voronoi volumes in high-dimensional point processes with various distance functions. *Advances in Applied Probability* **17**, 794–809 (1985).

Nguyen, A. T.; Nguyen, T. T.; Nguyen, T. N.; Lo, D.; Sun, C. Duplicate bug report detection with a combination of information retrieval and topic modeling. In: *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*. 70–79 (ACM, New York, NY, USA, 2012).

Olvera-López, J. A.; Carrasco-Ochoa, J. A.; Martínez-Trinidad, J. F.; Kittler, J. A review of instance selection methods. *Artificial Intelligence Review* **34**, 133–143 (2010).

Ougiaroglou, S.; Nanopoulos, A.; Papadopoulos, A. N.; Manolopoulos, Y.; Welzer-Druzovec, T. Adaptive k-nearest-neighbor classification using a dynamic number of nearest neighbors. In: *Proceedings of the 11th East European Conference on Advances in Databases and Information Systems*. 66–82 (Springer-Verlag, Berlin, 2007).

Ovalle-Perandones, M.; Perianes-Rodriguez, A.; Olmeda-Gomez, C. Hubs and authorities in a Spanish co-authorship network. In: *13th International Conference on Information Visualisation*. 514–518 (IEEE, New York, NY, USA, 2009).

Paik, M.; Yang, Y. Combining nearest neighbor classifiers versus cross-validation selection. *Statistical Applications in Genetics and Molecular Biology* **3** (2004).

Pan, R.; Dolog, P.; Xu, G. kNN-based clustering for improving social recommender systems. In: *Agents and Data Mining Interaction*. 115–125 (Springer, Berlin Heidelberg, Germany, 2013).

Papageorgiou, C.; Oren, M.; Poggio, T. A general framework for object detection. In: *International Conference on Computer Vision* (IEEE, New York, NY, USA, 1998).

Patidar, A. K.; Agrawal, J.; Mishra, N. Analysis of different similarity measure functions and their impacts on shared nearest neighbor clustering approach. *International Journal of Computer Applications* **40**, 1–5 (2012).

Patil, A.; Kinoshita, K.; Nakamura, H. Hub promiscuity in protein-protein interaction networks. *International Journal of Molecular Sciences* **11**, 1930–1943 (2010).

Pauleve, L.; Jegou, H.; Amsaleg, L. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern Recognition Letters* **31**, 1348–1358 (2010).

PE, H. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory* **14**, 515–516 (1968).

Pearson, K. On lines and planes of closest fit to systems of points in space. *Philosophy Magazine* **2**, 559–572 (1901).

Peirsman, Y.; Padó, S. Cross-lingual induction of selectional preferences with bilingual vector spaces. In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 921–929 (MIT Press, Cambridge, MA, USA, 2010).

Peng, J.; Heisterkamp, D. R.; Dai, H. K. Adaptive quasiconformal kernel nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26**, 656–661 (2004).

Penrose, M. *Random Geometric Graphs (Oxford Studies in Probability)* (Oxford University Press, USA, 2003).

Pérez-Rodríguez, J.; De Haro-García, A.; Garcá-Pedrajas, N. Instance selection for class imbalanced problems by means of selecting instances more than once. In: *Proceedings of the 14th International Conference on Advances in Artificial Intelligence: Spanish Association for Artificial Intelligence*. 104–113 (Springer-Verlag, Berlin, Heidelberg, 2011).

Pestov, V. On the geometry of similarity search: Dimensionality curse and concentration of measure. *Information Processing Letters* **73**, 47–51 (2000).

Pettis, K. W.; Bailey, T. A.; Jain, A. K.; Dubes, R. C. An intrinsic dimensionality estimator from near-neighbor information. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **1**, 25–37 (1979).

Póczos, B.; Lörincz, A. Independent subspace analysis using k-nearest neighborhood distances. In: *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and their Applications - Volume Part II*. 163–168 (Springer-Verlag, Berlin, Heidelberg, Germany, 2005).

Pogorelc, B.; Gams, M. Recognition of patterns of health problems and falls in the elderly using data mining. In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. 463–471 (Springer Verlag, Berlin, 2012).

Porter, M. F. *An algorithm for suffix stripping - program* (1980).

Powers, D. M. W. Evaluation: From precision, recall and F-factor to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies* **2**, 37–63 (2011).

Prati, R.; Batista, G.; Monard, M. Class imbalances versus class overlapping: An analysis of a learning system behavior. In: *Proceedings of the Mexican International Conference on Artificial Intelligence*. 312–321 (IEEE, Washington, DC, USA, 2004).

Quinlan, R. *C4.5: Programs for Machine Learning* (Morgan Kaufmann Publishers, San Francisco, CA, USA, 1993).

Radovanović, M. *Representations and Metrics in High-Dimensional Data Mining* (Izdavačka knjižarnica Zorana Stojanovića, Novi Sad, Serbia, 2011).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In: *Proceedings of the 26th International Conference on Machine Learning (ICML)*. 865–872 (Morgan Kaufmann, San Francisco, CA, USA, 2009).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* **11**, 2487–2531 (2010a).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. On the existence of obstinate results in vector space models. In: *Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 186–193 (ACM, New York, NY, USA, 2010b).

Radovanović, M.; Nanopoulos, A.; Ivanović, M. Time-series classification in many intrinsic dimensions. In: *Proceedings of the SIAM Data Mining Conference*. 677–688 (SIAM, Philadelphia, PA, USA, 2010).

Raghavan, V. V.; Wong, S. K. M. A critical analysis of vector space model for information retrieval. *Journal of the American Society for Information Science* **37**, 279–287 (1986).

Rajagopalan, B.; Lall, U. A k-nearest neighbor simulator for daily precipitation and other weather variables. *Water Resources Research* **35**, 3089–3101 (1999).

Rasheed, Z.; Rangwala, H.; Barbara, D. LSH-Div: Species diversity estimation using locality sensitive hashing. In: *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 1–6 (IEEE, New York, NY, USA, 2012).

Rasskin-Gutman, D. *Chess Metaphors: Artificial Intelligence and the Human Mind* (The MIT Press, Cambridge, MA, USA, 2009).

Ratanamahatana, C.; Keogh, E. Everything you know about dynamic time warping is wrong. In: *SIGKDD International Workshop on Mining Temporal and Sequential Data*. 1–11 (ACM, New York, NY, USA, 2004).

Rath, T.; Manmatha, R. Word image matching using dynamic time warping. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 514–521 (IEEE, New York, NY, USA, 2003).

Rish, I. An empirical study of the naive bayes classifier. In: *Proceedings of the IJCAI Workshop on Empirical Methods in Artificial Intelligence* (AAAI Press, Menlo Park, CA, USA, 2001).

Rozza, A.; Lombardi, G.; Ceruti, C.; Casiraghi, E.; Campadelli, P. Novel high intrinsic dimensionality estimators. *Machine Learning* **89**, 37–65 (2012).

Runeson, P.; Alexandersson, M.; Nyholm, O. Detection of duplicate defect reports using natural language processing. In: *Proceedings of the 29th International Conference on Software Engineering*. 499–510 (IEEE Computer Society, Washington, DC, USA, 2007).

Russell, S. J.; Norvig, P. *Artificial Intelligence: A Modern Approach* (Pearson Education, New Jersey, USA, 2003).

Schnitzer, D.; Flexer, A.; Schedl, M.; Widmer, G. Using mutual proximity to improve content-based audio similarity. In: *ISMIR'11*. 79–84 (International Society for Music Information Retrieval, Canada, 2011).

Schölkopf, B.; Smola, A. J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)* (The MIT Press, Cambridge, MA, USA, 2001).

Seidl, T.; Kriegel, H.-P. Optimal multi-step k-nearest neighbor search. In: *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*. 154–165 (ACM, New York, NY, USA, 1998).

Serpen, G.; Pathical, S. Classification in high-dimensional feature spaces: Random subsample ensemble. In: *International Conference on Machine Learning and Applications*. 740–745 (IEEE, New York, NY, USA, 2009).

Shaneck, M.; Kim, Y.; Kumar, V. Privacy preserving nearest neighbor search. In: *Machine Learning in Cyber Trust*. 247–276 (Springer US, New York, NY, USA, 2009).

Shang, W.; Huang, H.; Zhu, H.; Lin, Y.; Qu, Y.; Dong, H. An adaptive fuzzy kNN text classifier. In: *Computational Science ICCS 2006*. 216–223 (Springer-Verlag, Berlin, Germany, 2006).

Shen, H.; Chou, K.-C. Using optimized evidence-theoretic k-nearest neighbor classifier and pseudo-amino acid composition to predict membrane protein types. *Biochemical and Biophysical Research Communications* **334**, 288–292 (2005).

Short, R. D.; Fukunaga, K. The optimal distance measure for nearest neighbor classification. *IEEE Transactions on Information Theory* **27**, 622–627 (1981).

Singh, H.; Misra, N.; Hnizdo, V.; Fedorowicz, A.; Demchuk, E. Nearest neighbor estimates of entropy. *American Journal of Mathematical and Management Sciences* **23**, 301–321 (2003).

Siripanadorn, S.; Hattagam, W.; Teaumroong, N. Anomaly detection using self-organizing map and wavelets in wireless sensor networks. In: *Proceedings of the 10th WSEAS International Conference on Applied Computer Science*. 291–297 (WSEAS, Stevens Point, Wisconsin, USA, 2010).

Song, Y.; Huang, J.; Zhou, D.; Zha, H.; Giles, C. L. Iknn: Informative k-nearest neighbor pattern classification. In: *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*. 248–264 (Springer-Verlag, Berlin, Germany, 2007).

Song, Z.; Roussopoulos, N. K-nearest neighbor search for moving query point. In: *Advances in Spatial and Temporal Databases*. 79–96 (Springer, Berlin Heidelberg, Germany, 2001).

Sricharan, K.; Raich, R.; Hero, A. O. k-nearest neighbor estimation of entropies with confidence. In: *Information Theory Proceedings ISIT*. 1205–1209 (IEEE, New York, NY, USA, 2011).

Stone, C. J. Consistent nonparametric regression. *Annals of Statistics* **5**, 595–645 (1977).

Sun, S.; Wang, Y. K-nearest neighbor clustering algorithm based on kernel methods. In: *Second WRI Global Congress on Intelligent Systems (GCIS)*. 335–338 (IEEE, New York, NY, USA, 2010).

Sureka, A.; Jalote, P. Detecting duplicate bug report using character n-gram-based features. In: *Proceedings of the 2010 Asia Pacific Software Engineering Conference*. 366–374 (IEEE Computer Society, Washington, DC, USA, 2010).

Talwalkar, A.; Kumar, S.; Rowley, H. A. Large-scale manifold learning. In: *Computer Vision and Pattern Recognition (CVPR)*. 1–8 (IEEE, New York, NY, USA, 2008).

Tan, S. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications* **28**, 667–671 (2005a).

Tan, S. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications* **28**, 667–671 (2005b).

Tan, S. An effective refinement strategy for kNN text classifier. *Expert Systems with Applications* **30**, 290–298 (2006).

Tao, Y.; Yi, K.; Sheng, C.; Kalnis, P. Quality and efficiency in high dimensional nearest neighbor search. In: *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*. 563–576 (ACM, New York, NY, USA, 2009).

Tao, Y.; Yi, K.; Sheng, C.; Kalnis, P. Efficient and accurate nearest neighbor and closest pair search in high-dimensional space. *ACM Transactions on Database Systems* **35**, 20:1–20:46 (2010).

Ting, K. M. An instance-weighting method to induce cost-sensitive trees. *IEEE Transactions on Knowledge and Data Engineering* **14**, 659–665 (2002).

Tomašev, N.; Brehar, R.; Mladenić, D.; Nedevschi, S. The influence of hubness on nearest-neighbor methods in object recognition. In: *Proceedings of the 7th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. 367–374 (IEEE, New York, NY, USA, 2011a).

Tomašev, N.; Leban, G.; Mladenić, D. Exploiting hubs for self-adaptive secondary re-ranking in bug report duplicate detection. In: *Proceedings of the Conference on Information Technology Interfaces* (SRCE, Zagreb, Croatia, 2013a).

Tomašev, N.; Mladenić, D. Exploring the hubness-related properties of oceanographic sensor data. In: *Proceedings of the SiKDD Conference*. 149–152 (Institut "Jožef Stefan", Ljubljana, Ljubljana, 2011a).

Tomašev, N.; Mladenić, D. The influence of weighting the k-occurrences on hubness-aware classification methods. In: *Proceedings of the SiKDD Conference*. 153–156 (Institut "Jožef Stefan", Ljubljana, 2011b).

Tomašev, N.; Mladenić, D. Nearest neighbor voting in high-dimensional data: Learning from past occurrences. In: *PhD Forum of the International Conference on Data Mining*. 1215–1218 (IEEE, New York, NY, USA, 2011c).

Tomašev, N.; Mladenić, D. Nearest neighbor voting in high dimensional data: Learning from past occurrences. *Computer Science and Information Systems* **9**, 691–712 (2012).

Tomašev, N.; Mladenić, D. Class imbalance and the curse of minority hubs. *Knowledge-Based Systems* (2013).

Tomašev, N.; Mladenić, D. Hubness-aware shared neighbor distances for high-dimensional k-nearest neighbor classification. *Knowledge and Information Systems* 1–34 (2013).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. In: *Proceedings of the Machine Learning and Data Mining Conference*. 16–30 (Springer-Verlag, Berlin, Germany, 2011b).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. A probabilistic approach to nearest neighbor classification: Naive hubness bayesian k-nearest neighbor. In: *Proceeding of the Conference on Information and Knowledge Management*. 2173–2176 (ACM, New York, NY, USA, 2011c).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. In: *Advances in Knowledge Discovery and Data Mining*. **6634**, 183–195 (Springer-Verlag, Berlin, Germany, 2011d).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. In: *Proceedings of the Pacific-Asian Knowledge Discovery and Data Mining Conference*. 183–195 (Springer-Verlag, Berlin, Germany, 2011e).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. *International Journal of Machine Learning and Cybernetics* (2013b).

Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering* **99**, 1 (2013c).

Tomašev, N.; Rupnik, J.; Mladenić, D. The role of hubs in cross-lingual supervised document retrieval. In: *Proceedings of the Pacific Asian Knowledge Discovery and Data Mining Conference*. 185–196 (Springer-Verlag, Berlin / Heidelberg, Germany, 2013d).

Töscher, A.; Jahrer, M.; Legenstein, R. Improved neighborhood-based algorithms for large-scale recommender systems. In: *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*. 4:1–4:6 (ACM, New York, NY, USA, 2008).

Towfic, F.; VanderPlas, S.; OIiver, C. A.; Couture, O.; Tuggle, C. K.; West Greenlee, H.; Honavar, V. Detection of gene orthology from gene co-expression and protein interaction networks. *BMC Bioinformatics* **11**, 1–10 (2010).

Tran, T. N.; Wehrens, R.; Buydens, L. M. C. kNN density-based clustering for high dimensional multispectral images. In: *Proceedings of the 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas*. 147–151 (IEEE, New York, NY, USA, 2003).

Trieschnigg, D.; Pezik, P.; Lee, V.; Jong, F. D.; Rebholz-schuhmann, D. Mesh up: Effective mesh text classification for improved document retrieval. *Bioinformatics* **25**, 1412–1418 (2009).

Triggs, B.; Dalal, N. Histogram of oriented gradients for human detection. In: *Computer Vision and Pattern Recognition*. 886–893 (IEEE, New York, NY, USA, 2005).

Tversky, A.; Hutchinson, J. W. Nearest neighbor analysis of psychological spaces. *Psychological Review* **93**, 3–22 (1986).

Tversky, A.; Rinott, Y.; Newman, C. M. Nearest neighbor analysis of point processes: Applications to multidimensional scaling. *Journal of Mathematical Psychology* **27**, 235–250 (1983).

Valizadegan, H.; Tan, P.-N. Kernel based detection of mislabeled training examples. In: *Proceedings of the SIAM Data Mining Conference* (SIAM, Philadelphia, PA, USA, 2007).

van den Bosch, A.; Weijters, T.; Herik, H. J. V. D.; Daelemans, W. When small disjuncts abound, try lazy learning: A case study. In: *Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning*. 109–118 (Tilburg University, Tilburg, 1997).

Veness, J.; Silver, D.; Uther, W. T. B.; Blair, A. Bootstrapping from game tree search. In: *NIPS*. 1937–1945 (Curran Associates, Red Hook, USA, 2009).

Verleysen, M.; Francois, D.; Simon, G.; Wertz, V. On the effects of dimensionality on data analysis with neural networks. In: *Proceedings of the 7th International Work-Conference on Artificial and Natural Neural Networks: Part II: Artificial Neural Nets Problem Solving Methods*. 105–112 (Springer-Verlag, Berlin, Heidelberg, 2003).

Vidulin, V.; Gams, M. Impact of high-level knowledge on economic welfare through interactive data mining. *Applied Artificial Intelligence* **25**, 267–291 (2011).

Wang, D.; Zheng, Y.; Cao, J. Parallel construction of approximate kNN graph. In: *Distributed Computing and Applications to Business, Engineering Science (DCABES)*. 22–26 (CPS/IEEE, New York, NY, USA, 2012).

Wang, J.; Markert, K.; Everingham, M. Learning models for object recognition from natural language descriptions. In: *Proceedings of the British Machine Vision Conference* (BMVA Press, London, UK, 2009).

Wang, J.; Neskovic, P.; Cooper, L. N. Neighborhood size selection in the k-nearest-neighbor rule using statistical confidence. *Pattern Recognition* **39**, 417–423 (2006).

Wang, J.; Neskovic, P.; Cooper, L. N. Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recognition Letters* **28**, 207–213 (2007).

Wang, S.; Huang, Q.; Jiang, S.; Tian, Q. Nearest-neighbor classification using unlabeled data for real world image application. In: *Proceedings of the International Conference on Multimedia*. 1151–1154 (ACM, New York, NY, USA, 2010a).

Wang, S.; Li, X.; Xia, J.-F.; Zhang, X.-P. Weighted neighborhood classifier for the classification of imbalanced tumor dataset. *Journal of Circuits, Systems, and Computers* **19**, 259–273 (2010b).

Wang, X.; Zhang, L.; Xie, T.; Anvik, J.; Sun, J. An approach to detecting duplicate bug reports using natural language and execution information. In: *Proceedings of the 30th International Conference on Software Engineering*. 461–470 (ACM, New York, NY, USA, 2008a).

Wang, X. F.; Chen, G. Complex networks: Small-world, scale-free and beyond. *Circuits and Systems Magazine* **3**, 6–20 (2003).

Wang, X. R.; Lizier, J. T.; Obst, O.; Prokopenko, M.; Wang, P. Spatiotemporal anomaly detection in gas monitoring sensor networks. In: *Proceedings of the 5th European Conference on Wireless Sensor Networks*. 90–105 (Springer-Verlag, Berlin, Germany, 2008b).

Weinberger, K. Q.; Blitzer, J.; Saul, L. K. Distance metric learning for large margin nearest neighbor classification. In: *Proceedings of the Conference on Neural Information Processing Systems*. 265–273 (MIT Press, Cambridge, MA, USA, 2005).

Weiss, G. M. Mining with rarity: a unifying framework. *SIGKDD Explorer Newsletter* **6**, 7–19 (2004).

Wilson, D. R. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics* **2**, 408–421 (1972).

Wilson, D. R.; Martinez, T. R. Instance pruning techniques. In: *Proceedings of the Fourteenth International Conference on Machine Learning (ICML)*. 404–411 (Morgan Kaufmann, San Francisco, CA, USA, 1997).

Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques* (Morgan Kaufmann Publishers, San Francisco, CA, USA, 2005a).

Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005b).

Wu, G.; Chang, E. Y. KBA: Kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on Knowledge and Data Engineering* **17**, 786–795 (2005).

Xiaoyuan Su, G. R., Khoshgoftaar T.M. Imputed neighborhood based collaborative filtering. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. **1**, 633–639 (IEEE, New York, NY, USA, 2008).

Yao, Y.; G.Simons. A large-dimensional independent and identically distributed property for nearest neighbor counts in poisson processes. *Annals of Applied Probability* **6**, 561–571 (1996).

Yao, Y.; Sharma, A.; Golubchik, L.; Govindan, R. Online anomaly detection for sensor systems: A simple and efficient approach. *Performance Evaluation* **67**, 1059–1075 (2010).

Yianilos, P. N. Locally lifting the curse of dimensionality for nearest neighbor search. In: *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*. 361–370 (Society for Industrial and Applied Mathematics, Philadelphia,USA, 2000).

Yin, J.; Fan, X.; Chen, Y.; Ren, J. High-dimensional shared nearest neighbor clustering algorithm. In: *Fuzzy Systems and Knowledge Discovery*. 494–502 (Springer-Verlag, Berlin, Germany, 2005).

Younes, Z.; Aballah, F.; Denoeux, T. Multi-label classification algorithm derived from k-nearest neighbor rule with label dependencies. In: *Proceedings of the 16th European Signal Processing Conference* (European Association for Signal Processing, Lausanne, Switzerland, 2008).

Yu, X.; Pu, K.; Koudas, N. Monitoring k-nearest neighbor queries over moving objects. In: *Proceedings. 21st International Conference on Data Engineering*. 631–642 (IEEE, New York, NY, USA, 2005).

Yule, G. U. A Mathematical Theory of Evolution, Based on the Conclusions of Dr. J. C. Willis, F. R. S. *Philosophical Transactions of the Royal Society of London. Series B, Containing Papers of a Biological Character* **213**, 21–87 (1925).

Zelnik-manor, L.; Perona, P. Self-tuning spectral clustering. In: *Advances in Neural Information Processing Systems*. 1601–1608 (MIT Press, Cambridge, MA, USA, 2004).

Zhang, C.; Zhang, X.; Zhang, M. Q.; Li, Y. Neighbor number, valley seeking and clustering. *Pattern Recognition Letters* **28**, 173–180 (2007).

Zhang, H.; Berg, A. C.; Maire, M.; Malik, J. SVM-kNN: Discriminative nearest neighbor classification for visual category recognition. In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2126–2136 (IEEE Computer Society, Washington, DC, USA, 2006).

Zhang, J.; Mani, I. kNN approach to unbalanced data distributions: A case study involving information extraction. In: *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets* (Morgan Kaufmann, San Francisco, CA, USA, 2003).

Zhang, M.; Zhou, Z. Ml-kNN: A lazy learning approach to multi-label learning. *Pattern Recognition* **40**, 2038–2048 (2007).

Zhang, M.-L.; Zhou, Z.-H. A k-nearest neighbor based algorithm for multi-label classification. In: Hu, X.; Liu, Q.; Skowron, A.; Lin, T. Y.; Yager, R. R.; Zhang, B. (eds.) *Granular Computing*. 718–721 (IEEE, New York, NY, USA, 2005).

Zhang, Y.; Adl, K.; Glass, J. Fast spoken query detection using lower-bound dynamic time warping on graphical processing units. In: *Acoustics, Speech and Signal Processing*. 5173–5176 (IEEE Computer Society, Washington, DC, USA, 2012a).

Zhang, Z.; Wang, J.; Zha, H. Adaptive manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **34**, 253 –265 (2012b).

Zhang, Z.; Zhang, R. *Multimedia Data Mining: A Systematic Introduction to Concepts and Theory* (Chapman and Hall, London,UK, 2008).

Zhao, J.; Xu, K. Enhancing the robustness of scale-free networks. *Computing Research Repository* (2009).

Zheng, L.-Z.; Huang, D.-C. Outlier detection and semi-supervised clustering algorithm based on shared nearest neighbors. *Computer Systems and Applications* **29**, 117–121 (2012).

Zhou, Z.-H.; Liu, X.-Y. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering* **18**, 63–77 (2006).

Zhu, X.; Wu, X. Scalable representative instance selection and ranking. In: *Proceedings of the 18th International Conference on Pattern Recognition*. 352–355 (IEEE Computer Society, Washington, DC, USA, 2006).

Index of Figures

Index of Tables

Index of Algorithms Proposed in the Thesis

LKH (Local *K*-Hubs): A hub-based clustering method based on intra-cluster restricted hubness.

LHPC (Local Hubness-Proportional Clustering): A stochastic hubness-proportional clustering method based on intra-cluster restricted hubness.

GKH (Global *K*-Hubs): A hub-based clustering method based on clustering with global hub points.

GHPC (Global Hubness-Proportional Clustering): A stochastic hubness-proportional clustering method based on global point hubness.

GHPKM (Global Hubness-Proportional *K*-Means): A hybrid stochastic clustering approach that uses hubness-proportional simulating annealing in guiding the *K*-means centroid search.

h-FNN, dwh-FNN (Hubness-based Fuzzy *k*-Nearest Neighbor): A hubness-aware classification method that implements hubness based fuzzy measures. Distances weights can be used (dwh-FNN) or left out (h-FNN).

NHBNN (Naive Hubness-Bayesian *k*-Nearest Neighbor): A hubness-aware classification method based on a Naive Bayesian interpretation of neighbor occurrences.

HIKNN (Hubness Information *k*-Nearest Neighbor): A hubness-aware information-theoretic framework for *k*-nearest neighbor classification.

Unbiased Prototype Hubness Estimator: A hubness-aware framework for instance selection and classification in high-dimensional data.

***simhub*$_s$:** Hubness-aware shared neighbor distances for metric learning in high-dimensional data.

Hubness-aware CCA: An instance weighted extension of canonical correlation analysis for constructing hubness-aware common semantic mappings.

Self-Adaptive Hubness-aware Re-ranking: A secondary re-ranking procedure that reduces the query distance towards good hubs and increases the query distance towards bad hubs. It was used in semi-automatic bug duplicate detection.

Appendix

A Personal Bibliography

A.1 List of Publications Related to this Thesis

1.01 Original Scientific Articles

1. Tomašev, N.; Mladenić, D. Class imbalance and the curse of minority hubs. *Knowledge-Based Systems*, (in press, 2013).

2. Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering* **99**, (in press, 2013).

3. Tomašev, N.; Mladenić, D. Hubness-aware shared neighbor distances for high-dimensional k-nearest neighbor classification. *Knowledge and Information Systems*, (in press, 2013).

4. Tomašev, N.; ; Rupnik, J.; Mladenić, D. The role of hubs in cross-lingual supervised document retrieval. In: *Proceedings of the Pacific Asian Knowledge Discovery and Data Mining Conference*. 185–196 (Springer Verlag, Berlin, Germany, 2013).

5. Tomašev, N.; Mladenić, D. Nearest neighbor voting in high dimensional data: Learning from past occurrences. *Computer Science and Information Systems* **9**, 691–712 (2012).

6. Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. *International Journal of Machine Learning and Cybernetics*, (in press, 2013).

7. Tomašev, N.; Mladenić, D. Hubness-aware shared neighbor distances for high-dimensional k-nearest neighbor classification. In: *Proceedings of the 7th International Conference on Hybrid Artificial Intelligence Systems*. 116–127 (Springer Verlag, Berlin, Germany, 2012).

8. Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. The role of hubness in clustering high-dimensional data. In: *Advances in Knowledge Discovery and Data Mining* **6634**, 183–195 (Springer Verlag, Berlin, Germany, 2011).

9. Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. In: *Proceedings of the Machine Learning and Data Mining Conference*. 16–30 (Springer Verlag, Berlin, Germany, 2011).

1.08 Published Scientific Conference Contributions

1. Tomašev, N.; Radovanović, M.; Mladenić, D.; Ivanović, M. A probabilistic approach to nearest neighbor classification: Naive hubness bayesian k-nearest neighbor. In: *Proceedings of the Conference on Information and Knowledge Management.* 2173–2176 (ACM, New York, NY, USA, 2011).

2. Tomašev, N.; Mladenić, D. Nearest neighbor voting in high-dimensional data: Learning from past occurrences. In: *PhD Forum of the International Conference on Data Mining.* 1215–1218 (IEEE, New York, NY, USA, 2011).

3. Tomašev, N.; Brehar, R.; Mladenić, D.; Nedevschi, S. The influence of hubness on nearest-neighbor methods in object recognition. In: *Proceedings of the 7th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP).* 367–374 (IEEE, New York, NY, USA, 2011).

4. Tomašev, N.; Mladenić, D. Exploring the hubness-related properties of oceanographic sensor data. In: *Proceedings of the SiKDD Conference.* 149–152 (Institut "Jožef Stefan", Ljubljana, 2011).

5. Tomašev, N.; Mladenić, D. The influence of weighting the k-occurrences on hubness-aware classification methods. In: *Proceedings of the SiKDD Conference.* 153–156 (Institut "Jožef Stefan", Ljubljana, 2011).

B Author Biography

Nenad Tomašev was born in Novi Sad on March, 19th, 1985.

He studied computer science and mathematics at the Department of Mathematics and Informatics at the Faculty of Natural Sciences in Novi Sad. He obtained his Bachelor of Science degree in Informatics in July 2008 and graduated with honors. The graduation thesis was titled *Building a CoreWar Optimizer* and its topic was the use of stochastic optimization methods in genetic programming for artificial life simulations.

From February to August 2008, Nenad was a software engineering intern in Telvent DMS (currently Schneider Electric DMS NS), where he worked on consumer profiling in power distribution systems.

In October 2008, Nenad enrolled in the *New Media and E-Science* postgraduate program of the Jožef Stefan International Postgraduate School. His mentor was Dunja Mladenić and he participated in the work and projects of the Artificial Intelligence Laboratory at the Jožef Stefan Institute. His research focus was in Machine Learning and Data Mining. In particular, he was analyzing the uneven distribution of influence in high-dimensional data and the emergence of hubs. He developed and evaluated many novel data mining methods for various sorts of data analysis, including clustering, classification, anomaly detection, information retrieval, metric learning and re-ranking.

Apart from research, Nenad has also gained valuable teaching experience, as he has actively participated as a teaching assistant in Petnica Science Center and Višnjan Summer School.