SEMI-AUTOMATIC ONTOLOGY CONSTRUCTION

Blaž Fortuna

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, July 2011

Supervisor:

Prof. Dr. Dunja Mladenić, Jožef Stefan International Postgraduate School, Jamova 39, SI-1000 Ljubljana, Slovenia

Evaluation Board:

Prof. Dr. Marko Bohanec, Jožef Stefan International Postgraduate School, Jamova 39, SI-1000 Ljubljana, Slovenia

Doc. Dr. Irena Nančovska Šerbec, Pedagoška fakulteta, Univerza v Ljubljani, Kardeljeva ploščad 16, SI-1000 Ljubljana, Slovenia

Dr. John Davies, British Telecommunications Plc., Adastral Park, Ipswich, United Kingdom

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Blaž Fortuna

SEMI-AUTOMATIC ONTOLOGY CONSTRUCTION

Doctoral Dissertation

POLAVTOMATSKA GRADNJA ONTOLOGIJ

Doktorska disertacija

Supervisor: Prof. Dr. Dunja Mladenić

Ljubljana, Slovenia, September 2011

Index

Abstract

This thesis addresses the task of formalizing and implementing the process of semi-automatic ontology construction.

We propose a theoretical framework for formalizing the ontology construction process. The process is described as a sequence of operators applied to the ontology. Several types of common operators are identified and each type is abstracted so it can be discovered by a combination of machine learning algorithms and user interactions. The proposed ontology learning framework is generic and can handle various domains. The requirement is, that domain data can be provided in a format supported by the learning algorithms.

Operators defined as part of the ontology construction process are implemented using several machine learning algorithms. Clustering, active learning and large-scale classifications are used to learn operators for adding concepts and relations. A novel visualization approach for visualizing instances, concepts and ontologies is developed, using a combination of dimensionality reduction techniques. The ability to incorporate additional background data is implemented using a novel feature weighting schema, and the addition of new instances to the ontology is translated to a standard classification task.

We also developed a system, which implements the framework, together with the proposed machine learning algorithms. The system takes domain data on the input, and guides the user through the process of constructing the ontology for the given domain. The developed system was applied in several use-cases, where domain data was provided as a text corpus or a social network, to showcase the capabilities.

The system was also evaluated in two user studies, to evaluate the user interface and to compare developed ontologies against manually constructed ones. The results of the users studies show, that the system is user friendly enough to be used by domain experts. The users can construct ontologies that are comparable to manually constructed ontology and can do so in a shorter amount of time.

Povzetek

Disertacija obravnava formalizacijo in implementacijo procesa za polavtomatsko gradnjo ontologij.

Predlagamo teoretični okvir za formalizacijo procesa polavtomatske gradnje ontologij. Proces je predstavljen kot zaporedje operatorjev, uporabljenih na ontologijo. Identificirani so pogosti tipi operatorjev, pri čemer je vsak od njih operator posplošen do mere, potrebne za odkritje operatorja s kombinacijo metod strojnega učenja ter interakcije z uporabnikom. Predlagan teoretski okvir za učenje ontologij je dovolj splošen, da lahko obdela vrsto domen. Edini predpogoj je obstoj domenskih podatkov v obliki, primerni za metode strojnega učenja.

Operatorji, ki so identificirani kot glavni gradniki procesa gradnje ontologije, so implementirani z uporabo naslednjih metod strojnega učenja. Algoritmi za razvrščanje v skupine (*ang. Clustering*), vodeno učenje (*ang. Active Learning*) in klasifikacija so uporabljeni za učenje operaterjev za gradnjo konceptov in relacij. Predstavljene so nove metode za vizualizacijo instanc, konceptov in ontologij, ki temeljijo na kombinaciji metod za zmanjševanje dimenzij (*ang. Dimensionality reduction*). Možnost vključevanja dodatnih domenskih podatkov je omogočena preko nove metode za uteževanje značilk. Dodajanje novih instanc v razvito ontologijo je prevedeno na večrazredni klasifikacijski problem.

Del disertacije je tudi sistem, ki implementira predlagani teoretični okvir, skupaj s predlaganimi metodami strojnega učenja. Sistem na podlagi domenskih podatkov in metod strojnega učenja vodi uporabnika skozi proces gradnje ontologije za dano domeno. Delovanje sistema je prikazano na več praktičnih primerih, kjer so bili domenski podatki podani v obliki korpusa dokumentov oziroma socialnega omrežja.

Razviti sistem je bil ovrednoten v dveh uporabniških študijah. Njun namen je bil ovrednotiti uporabniški vmesnik in primerjati ontologije, zgrajene v okviru razvitega sistema, z ročno grajenimi ontologijami. Rezultati uporabniških študij kažejo, da je razviti sistem primeren za uporabo s strani domenskih strokovnjakov. Razvite ontologije so primerljive z ročno grajenimi, pri čemer je čas gradnje pomembno krajši.

Abbreviations

NLP	=	Natural Language Processing
IE	=	Information Extraction
RDF	=	Resource Description Framework
DF	=	Document Frequency
IDF	=	Inverse Document Frequency
TFIDF	=	Term Frequency, Inverse Document Frequency
LSI	=	Latent Semantic Indexing
LDA	=	Latent Dirichlet Allocation
kNN	=	k Nearest Neighbour
LOD	=	Linked Open Data
ODP	=	Open Directory Project
KB	=	Knowledge Base
US	=	Uncertainty Sampling
SVM	=	Support Vector Machine
URI	=	Uniform Resource Identifier
SVD	=	Singular Value Decomposition
MDS	=	Multidimensional Scaling
CG	=	Conjugate Gradient
UI	=	User Interface

1 Introduction

The ontology is a fundamental data object for organizing knowledge in many areas, from philosophy to knowledge management and artificial intelligence. In [1] the ontology is defined as “formal, explicit specification of a shared conceptualization”.

Ontology construction can be a time consuming task typically done by ontology engineers. The complexity of the task increases with the size of the domain. Various methodologies, tools and systems were developed to help the ontology engineers during the construction process. In recent years, several approaches for automatic extraction of ontologies from text were developed, aiming at replacing or augmenting the role of ontology engineers. However, outputs of such approaches typically require additional work, due to errors introduced by the automatic systems.

The purpose of this dissertation is *to develop, implement and evaluate an ontology learning framework for semi-automatic ontology construction using machine learning techniques*. The role of the ontology learning framework is to abstract the ontology construction process in a way, which enables the application of machine learning techniques to various stages of ontology construction. The framework combines machine learning algorithm with user interface to form an integrated semi-automatic approach which can reduce the necessary time and the complexity of developing domain ontologies. As such, it can bridge the gap between complex ontology editing tools, which require substantial knowhow about ontology engineering, and the domain experts, which are actually constructing the ontology.

The framework implementation resulted in an interactive system that guides and helps the user during the ontology construction process. The system can suggest concepts and relations, propose names for the concepts, automatically assign instances to the concepts, visualizes instances within a concept, and provide a good overview of the ontology to the user through several types of visualizations. The user maintains the control throughout the process by managing all edits and has the ability to modify any properties of the ontology by either accepting or rejecting the system’s suggestions, manually adjusting the suggestions, and parameters of the suggestion algorithms. This provides necessary trust and flexibility for the user in a way that is not possible with the fully automated solutions.

The framework provides the ability to incorporate domain data to support the ontology construction process. Domain data is provided by the user at the beginning of the ontology construction process and should reflect the structure of the domain for which the user is building the ontology. An example of domain data is a document corpus where

ontological instances are either the documents themselves or name-entities occurring in the documents. The framework is not limited only to text and supports other types of input data, such as social networks or images, extending the domain of potential applications.

1.1 Aims and Hypothesis

The aim of the dissertation is *to develop, implement and evaluate a framework for semi-automatic ontology construction using machine learning techniques*. The overall goal is addressed through the following contributions:

A. *The definition of the theoretical framework for ontology learning.* The proposed framework supports interactive semi-automatic ontology construction, and provides the mechanisms for incorporating domain data and machine learning algorithms. This task is addressed in Chapter 3.
B. *Population of the ontology learning framework with machine learning techniques and feature representations.* This task is addressed in Chapter 4.
C. Implementation of the framework in a form of interactive user application, built from scalable components. This task is addressed in Chapter 5.
D. Demonstrate and evaluate the benefits of the system and discuss several applications. This task is addressed in Chapter 6.

The dissertation addresses two hypotheses that are developed and experimentally tested:

A. *The integration of machine learning techniques in the semi-automatic ontology construction enables domain experts with little ontology engineering experience to reliably create domain ontologies.* This hypothesis is tested in a set of use-cases, dedicated user studies measuring the ability of target users to operate the developed system (Section 6.4), and evaluation of ontologies produced by the target users (Section 6.3).
B. *Proposed semi-automatic ontology learning framework can be implemented in a scalable way.* This hypothesis is tested on component basis, by evaluating the time complexity of the core machine learning approaches (Chapter 4). Furthermore, the architecture of the implemented system was tested in large-scale scenarios, where it had to deal with millions of instances (Chapter 5).

1.2 Organization

The thesis starts with a short overview of the area, positioning of the thesis, the main goals and the hypothesis of the thesis. Chapter 2 presents an overview of the related work in the area of ontology construction, with respect to manual, automatic and semi-automatic approaches.

Chapter 3 describes the ontology learning framework, one of the main contributions of the thesis. The Chapter starts by giving a definition of ontology and introducing ontology construction operators. Operators are the primary modifiers of ontology, and correspond to various operations that the ontology engineer can perform on an ontology, such as adding concepts or relations. This abstraction of ontology learning process is completed with introduction of instance and co-occurrence profiles, which provide the ability to represent domain data, required by machine learning approaches, within the framework.

Chapter 4 populates the ontology learning framework with machine learning techniques that can be used to implement operators defined inside the ontology learning framework. The Chapter starts by describing feature representations for text documents and social networks. This is followed with descriptions of machine learning techniques, which can be used to learn concept and relation construction operators, derive concept and relation names, visualize concepts or ontologies, and to populate developed ontologies with new instances.

Chapter 5 describes the system, which was developed as implementation of the proposed ontology learning framework. The architecture of the system is described in more details, and description of how particular infrastructural components were tested in large-scale scenarios is provided.

Chapter 6 presents use-cases and user studies used to test and evaluate the developed framework and system. The two use-cases demonstrate how the system can be applied in two different domains: financial domain using text corpus, and modelling organization structure using social network. The first user study compares the ontologies developed with the system to manually constructed ontologies from the same domain. The second user study evaluates the user interaction aspect of the developed system.

Chapter 7 gives conclusion remarks and directions for future research.

2 Related Work

Ontology construction can be seen as a complex task of capturing knowledge in ontological structure. There are a number of manual approaches to ontology construction and several attempts to involve automatic methods. In this Chapter we will mention only the more general approaches, which relate to the dissertation as a whole. Work related to particular elements of the proposed framework and system will be presented within the specific Sections.

Manual extraction of common sense knowledge from different sources was set up in the Cyc project [2] and has been later extended with some elements of automatic methods including automated ontology population with named entities [78], semi-automatic ontology extension based on the user-interactive dialogue system for knowledge acquisition [3], augmenting Cyc ontology using pattern matching and link analysis [4]. Several methodologies have been proposed for manual ontology construction involving different steps. For instance, the methodology proposed in [5] involves the following stages: identifying the purpose of the ontology (why to build it, how will it be used, the range of the users), building the ontology, evaluation and documentation. Building of the ontology is further divided into three steps. The first is ontology capture, where key concepts and relationships are identified, a precise textual definition of them is written, terms to be used to refer to the concepts and relations are identified, the involved actors agree on the definitions and terms. The second step involves coding of the ontology to represent the defined conceptualization in some formal language (committing to some meta-ontology, choosing a representation language and coding). The third step involves possible integration with existing ontologies. An overview of methodologies for building ontologies is provided in [6], where several methodologies, including the above described one, are presented and analysed against the IEEE Standard for Developing Software Life Cycle Processes, thus viewing ontologies as parts of some software product.

Methodologies for semi-automatic ontology construction have some specifics compared to manual ontology construction. For instance, the methodology proposed in [7] consists of the following interrelated phases: domain understanding (what is the area we are dealing with?), data understanding (what is the available data and its relation to semi-automatic ontology construction?), task definition (based on the available data and its properties, define task(s) to be addressed), ontology learning (semi-automated process), ontology evaluation (estimate quality of the solutions to the addressed task(s)), and refinement with human in the loop (perform any transformation needed to improve

the ontology and return to any of the previous steps, as desired).

Most ontology construction systems rely on textual data. In [8] the authors present a framework and system Text-To-Onto for extraction of ontologies from text. The paper proposes an approach for extraction of entities and relations from text, and places several Natural Language Processing (NLP) and Information Extraction (IE) approaches within a coherent framework. The approach was further extended in [9], with introduction of Probabilistic Ontology Model. In [10] and [13] the authors proposed a system for ontology learning, which first extracts domain terminology from some domain texts, and then tries to interpret the semantics of the extracted terms and their relations.

A large body of work related to ontology construction tries to tackle specific tasks, such as concept and relation extraction, extending existing ontologies, and ontology population [11]. In [16] the authors use web documents to automatically extend given existing ontology with concepts and relations. Work was also done on learning taxonomic, (e.g. “sub-concept”) [17], and non-taxonomic (e.g. “has-part”) [8] relations in an existing ontology. Clustering of parsed sentences obtained from a document collection was used for learning concepts in [18], [19] and [77] propose a methodology for semi-automatic ontology extension by glossary terms based on text mining methods and considering ontology content, structure and co-occurrence information. The methodology is successfully applied on extending Cyc ontology. Ontology population overlaps with the area of Information Extraction. Read the Web [14] project from CMU uses bootstrapping to populate a given ontology. The system starts with a small set of instances for each concept, and uses web crawl data to literately learn models for each concept and apply the models to discover new instances. Ontological constraints provide require supervision for the bootstrapping iterations. Alternatively, [15] combines Information Extraction with sentence parsing to extract a large graph of connected concepts and relations from the open web.

Not all work is focused on extraction of ontologies from text. In [12] the authors use Folksonomies as a source of terminology and try to infer ontological structures from them using co-occurrence statistics. Ontology construction from a collection of news stories [20] represent news as a collection of graphs with named entities in nodes and relationships between them based on the context of collocations of the named entities.

The work, presented in this dissertation, draws heavily on the data mining and machine learning abstractions and techniques in defining and populating Ontology Learning Framework, presented in Section 3. Such approach differentials the work from other frameworks, such as Text-To-Onto, which rely more on NLP and IE abstractions and techniques. The chosen approach requires clear definition of instances and feature vectors, which are often implicitly assumed other framework, and in turn provides generality with respect to the underlying data (e.g. text [65], social networks [26], images [63]).

3 Ontology Learning Framework

This Chapter introduces the terminology and definitions, which form the ontology learning framework. The Chapter starts by formally defining ontology, the central object of the framework. Ontologies are manipulated with ontology construction operators. For example, an operation can be an addition of a concept or relation. Finally, the ontology construction operators can be learned combining domain data and user interactions. This Chapter also describes a running example used to illustrate different aspects of the ontology construction throughout the dissertation.

3.1 Ontology Definition

In this thesis, an ontology is defined as a tuple over a set of instances I, set of concepts C and a set of binary relations R. Each concept $c \in C$ is assigned a set of instances $\iota_C \subset I$ and each binary relation $r \in R$ is assigned a set of concept pairs $\sigma_R(r) \in C \times C$. The concepts and relations are organized in a hierarchy. The following definitions provide a formal description of an ontology.

Definition 1 Ontology *is a tuple*

$$O = \{I, C, R, \leq_C, \leq_R, \iota_C, \sigma_R\} \in \mathcal{O} \quad (1)$$

where

$I = \{e_1, \dots, e_N\}$	*is a set of instances,*
$C = \{c_r, c_1, \dots, c_M\}$	*is a set of concepts,*
$R = \{r_1, \dots, r_L\}$	*is a set of binary relations,*
$\leq_C$	*is a partial order between concepts (sub-concept),*
$\leq_R$	*is a partial order between relations (sub-relation),*
$\iota_C: C \rightarrow \wp(I)$	*is an assignment operator mapping concepts to instance sets,*
$\sigma_R: R \rightarrow C \times C$	*is an assignment operator mapping relations to concept pairs,*
c_r	*is a designated concept c named* root *for which* $\iota_C(c) = I$.

The above definition of an ontology is a simplified version of the definitions provided in [21] and [22]. We intentionally left out the data-types, attributes and values (T, A, V) to simplify the definitions in the following sections. Also, we only work with the relations between concepts and not on the relations either between instances or between an instance and a concept.

This definition of ontology follows a pragmatic view of ontology in a sense that it is

"learnable" given an initial set of instances with associated features. Such feature representation of instances, concepts and relations is crucial for the application of machine learning algorithms to the ontology learning problem and will be discussed in the following sections.

Definition 2 Minimal ontology O_0 *for a given set of instances* I *is defined as*

$$O_0 = \{I, \{c_r\}, \emptyset, \emptyset, \emptyset, c_r \longmapsto I, \emptyset\}. \quad (2)$$

Definition 3 Concept neighbourhood N *of a concept* c *is defined as*

$$N(c) = \{e \in I : e \in \iota_C(c') \text{ and } c' \text{ is the father of } c\}. \quad (3)$$

Minimal ontology is the smallest valid ontology for a given set of instances. Concept neighbourhood is defined as a set of instances assigned to the sibling concepts of a given concept based on the partial order $\leq_C$.

The ontology definition is closely related to the Resource Description Framework (RDF) schema [23]; Table 1 shows the correspondences between the two.

Table 1: Relations between the presented ontology definition and RDF schema.

Symbol	RDF equivalent
C	Collection of `rdfs:Class`
c	`rdfs:Class`
R	Collection of `rdf:Property`
$\leq_C$	`rdfs:subClassOf`
$\leq_R$	`rdfs:subPropertyOf`
ι_C	`rdf:type`
σ_R	`rdfs:range` and `rdfs:domain`

3.2 Running Example

In this subsection we introduce an example of semi-automatic ontology construction from the financial domain. This example is then used and extended throughout the dissertation to illustrate various concepts or algorithms.

The example is built around a collection of publicly traded corporations (set of instances I), each represented by its ticker symbol. The domain of publicly traded corporations is modelled based on their industry sectors and locations (set of concepts C). The hierarchy between the concepts is given by the partial order $\leq_C$ and the subsets of instances are assigned to the concepts by assignment the operator ι_C. Figure 1 presents an example that has ten instances and nine concepts. For instance, one of the concepts is "Software" and it is a sub-concept of "Technology". Four instances belong to the "Technology" concept.

I = { *AAPL, MSFT, GOOG, NOK, F, VOW, GM, GS, BCS, C*}
C = { "Root", "Manufacturers", "Auto Manufacturers", "Technology", "Software", "Financial", "US", "Europe", "Germany"}

"Manufacturers" $\leq_C$ "Root"
"Auto Manufactures" $\leq_C$ "Manufacturers"
"Software" $\leq_C$ "Technology"
"Germany" $\leq_C$ "Europe"

ι_C : "Auto Manufacturers" $\longmapsto$ { *F, VOW, GM* }
ι_C : "Technology" $\longmapsto$ { *AAPL, MSFT, GOOG, NOK* }
ι_C : "Financial" $\longmapsto$ { *GS, BCS, C* }

Figure 1 Financial domain ontology example.

The running example does not contain relationships and a more elaborate example is used in Section 4.3 to demonstrate learning of relation construction operators. A more thorough development of the running example using the complete framework is presented in Section 6.1.

3.3 Ontology Construction Operator

Ontology construction either starts with a minimal ontology O_0, based on a given set of instances I, or with an existing ontology O_i. In the former case it is called *ontology construction from scratch* and in the latter case it is called *ontology extension.*

The ontology construction can be seen as a sequence of modification of the initial ontology O_i by changing its concepts, relations, hierarchies or assignments. Each such modification in the sequence can be represented as an operator $f: O_i \mapsto O_{i+1}$. Such operator is called an *ontology construction* operator.

Definition 4 Ontology construction *is an operator*

$$f: \mathcal{O} \to \mathcal{O}. \tag{4}$$

The ontology construction process which started with O_0 and resulted in ontology O_n can be written as a composition of ontology construction operators, each representing one operator in the process of ontology construction:

$$O' = (f_n \circ f_{n-1} \circ \ldots \circ f_1)(O_0)\ . \tag{5}$$

Each ontology construction operator can represent one operation, manually defined by the user through using ontology editing tool. Examples of such operations would be the introduction of one or more concepts or relations.

The ontology construction as addressed in this dissertation can be defined as the development of procedures for automatically assembling ontology construction operators

and devising an interface for the user to evaluate and select among the possible operators.

3.3.1 Concept Construction Operator

We define a *concept construction operator* as an ontology construction operator which only adds new concepts to the ontology. This requires adding new concepts to the set of concepts C and updating the assignment function ι_C and the partial order $\leq_C$. For updating the assignment function we need a subset of instances belonging to each new concept and for updating the partial order we need the position of each new concept in the existing concept hierarchy. In a nutshell, to add a new concept, one has to provide a subset of instances that belong to the new concept and its position in the hierarchy.

Formally, the addition of one concept c to the ontology is an operator

$$f_C\colon (I, C, R, \leq_C, \leq_R, \iota_C, \sigma_R) \mapsto (I, C \cup \{c\}, R, \leq_C', \leq_R, \iota_C', \sigma_R)\ . \qquad (6)$$

The Concept construction operators can be restricted to additions of sub-concepts to one existing concept. Such operator is called *a core concept construction operator*. More complex concept addition operations (e.g. an addition of a sub-tree of concepts) are achievable as a composite of several core concept construction operators. An example can be seen in Figure 2.

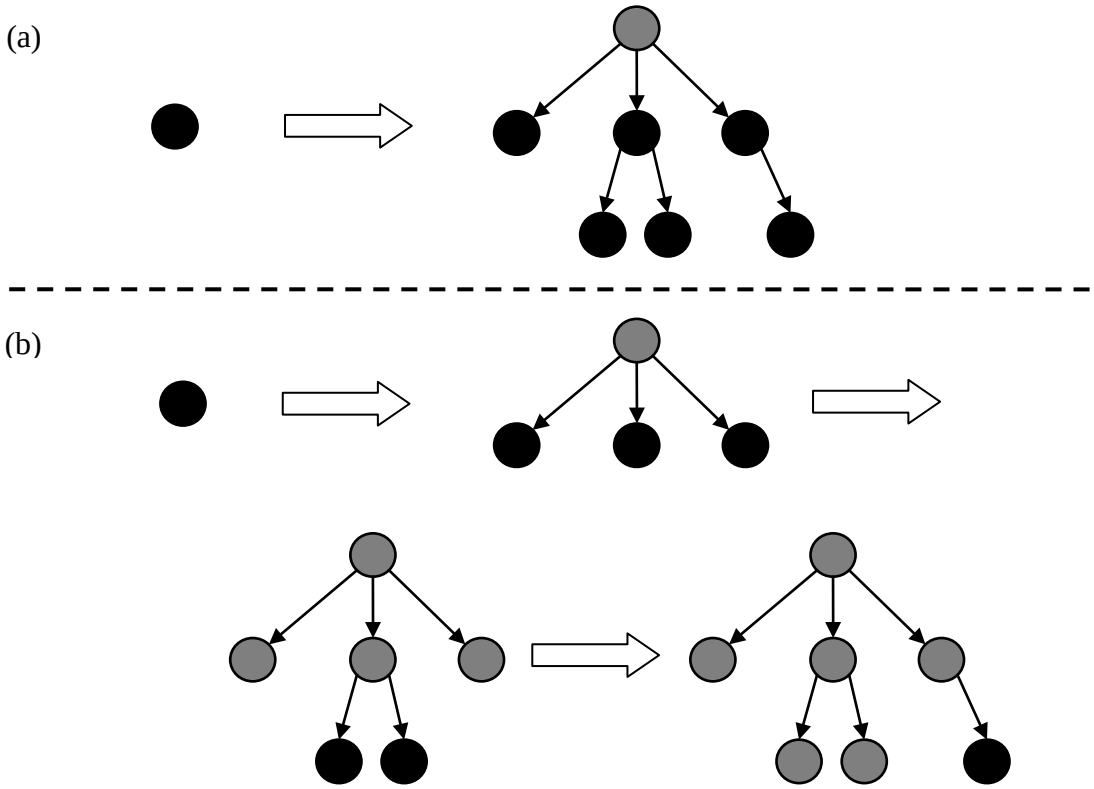


Figure 2: Example of (a) an application of a concept construction operator and (b) an application of an equivalent sequence of core concept construction operators. Black circles represent newly added concepts, and grey circles represent existing concepts already added in one of the previous steps.

3.3.2 Relation Construction Operator

The *relation construction operator* can be defined in a similar way as the concept construction operator. The addition of a new relation results in changes to the set of all relation instances R, the hierarchy between relations $\leq_R$ and the relation assignment function σ_R.

Formally, the addition of one relation *r* to the ontology is a mapping

$$f_R\colon (I, C, R, \leq_C, \leq_R, \iota_C, \sigma_R) \mapsto (I, C, R \cup \{r\}, \leq_C, \leq_R', \iota_C, \sigma_R'). \quad (7)$$

The machine learning algorithms implementing the framework, presented in Section 4.3 cover only the updating of the relation set R and assignment function σ_R.

We define the *core relation construction operator* analogues to the concept construction, by constraining the relation construction operator to addition of relations to a fixed pair of concepts.

3.4 Learning Ontology Construction Operators

An ontology can be seen as a conceptualization or a model of a domain, typically constructed by a domain expert. In order to automatize part of this process using machine learning algorithms, external information about the domain is needed. The information can be given implicitly through the involvement of the domain expert or explicitly, for example, in a form of a document collection describing the domain of interest.

In the ontology learning framework this information is introduced in a two-step process. In the first step, instances are described by some external features. These features serve as input to the ontology learning algorithms, which use them for learning the ontology construction operators. In the second step, the domain expert uses his/her domain knowledge to either apply relevant operators to the ontology or to provide feedback to the ontology learning algorithms to refine the operators.

This section extends the ontology learning framework so it can associate feature sets with instances and use these feature sets to learn ontology operators. The framework will be further extended to include domain expert feedback in the ontology learning and construction process.

We introduce external domain data by defining profiles. Profiles can be seen as feature vectors, which are used by the machine learning and text mining algorithms. The profiles can be associated with instances (instance profile) and pairs of instances (co-occurrence profile). The main role of co-occurrence profiles is to serve as data for extracting feature vectors used by machine learning methods for learning relation construction operators. Note that the co-occurrence profiles are directed. That is, co-occurrence profile $P_{e,f}$ is not necessarily the same as co-occurrence profile $P_{f,e}$.

Definition 5 Instance profile $p_e \in P_I$ *is a feature vector associated to an instance* $e \in I$.

Definition 6 Co-occurrence profile $p_{e,f} \in P_{I \times I}$ *is a feature vector associated to a pair of instances* $(e, f) \in I \times I$.

In the running example introduced in Section 3.2 each instance corresponds to a public company and has a textual description. Two examples of such description are given in Figure 3. In this case an *instance profile* for one company can be a word vector [24] of the description. Details on extracting feature vectors from text are presented in Section 4.1.

APPL[1]	**"**Apple Inc., together with subsidiaries, designs, manufactures, and markets personal computers, mobile communication and media devices, and portable digital music players, as well as sells related software, services, peripherals, networking solutions, and third-party digital content and applications worldwide. The company sells its products worldwide through its online stores, retail stores, direct sales force, third-party wholesalers, resellers, and value-added resellers. In addition, it sells third-party Mac, iPhone, iPad, and iPod compatible products, including application software, printers, storage devices, speakers, headphones, and other accessories and peripherals through its online and retail stores; and digital content and applications through the iTunes Store. The company sells its products to consumer, small and mid-sized business, education, enterprise, government, and creative markets. As of September 25, 2010, it had 317 retail stores, including 233 stores in the United States and 84 stores internationally. The company, formerly known as Apple Computer, Inc., was founded in 1976 and is headquartered in Cupertino, California."
VOW[2]	**"**Volkswagen Aktiengesellschaft manufactures automobiles. The company operates in two divisions, Automotive and Financial Services. The Automotive division engages in the development of vehicles and engines, as well as the production and sale of passenger cars, commercial vehicles, trucks and buses, pick-ups, heavy trucks, and parts. Its product range extends from low-consumption small cars to luxury class vehicles. The Financial Services division offers dealer and customer financing, leasing, banking and insurance, and fleet management services. The company offers its products under Volkswagen Passenger Cars, Audi, SEAT, Bentley, Volkswagen Commercial Vehicles, Scania, Skoda, Bugatti, and Lamborghini brand names, as well as services under Volkswagen Financial Services brand name. Volkswagen Aktiengesellschaft sells its products in various markets in Europe, North America, South America, the Asia-Pacific, and internationally. It has strategic alliances with Higer; Suzuki Motor Corporation; Dr. Ing. h.o. F. Porsohe AG; Daimler AG; Chrysler Group; CHOREN Industries; and IOGEN. The company was founded in 1937 and is headquartered in Wolfsburg, Germany."

Figure 3 Textual descriptions of two public companies. Descriptions of other companies, used in the example can be found in Appendix A.

One could also derive *instance profiles* for the same example from a large corpus of

[1] http://finance.yahoo.com/q/pr?s=AAPL+Profile
[2] http://finance.yahoo.com/q/pr?s=VOW.DE+Profile

text, such as news articles. In this case, each instance (public company) can be described by the set of all the sentences or paragraphs in which it appears. A *co-occurrence profile* of two instances can be a collection of all the sentences or paragraphs in which both instances appear. This scenario inspired the name of "co-occurrence profile". Similar to the previous example, word vector can be used as a feature representation of the descriptions.

There are several important things to note from the above scenarios. First of all, not all instance pairs have co-occurrence profile. For example, two companies that never co-occur in the given corpus have empty co-occurrence profiles. Second of all, not all algorithms require co-occurrence profiles. In our implementation of the ontology learning framework only algorithms for learning relations require such profiles. Finally, the extraction of co-occurrence profiles is possible only for specific cases, one being the above example with companies appearing in news articles.

Instance and co-occurrence profiles are not included in the ontology definition. Profiles are only used to learn ontology construction operators and are as such not a part of the final ontology. Additionally, the proposed approach is based on the feature descriptions of instances, where other feature descriptions (e.g. features describing concepts or relations) are deduced from instance features.

Armed with instance profiles P_I and co-occurrence profiles $P_{I\times I}$ we can extend the framework with procedure for learning ontology construction operators, which can be applied to ontology O_i. The procedure starts by first constructing a list of suggested operators. The list of suggestions is presented to the domain expert, which selects one operator from the list and the selected operator is applied to the ontology O_i. The following definitions present a formulation of these two steps.

Definition 7 The ontology construction operator suggestion *is a mapping*

$$g_A: (O_i, P_I, P_{I\times I}) \mapsto \{f: \mathcal{O} \to \mathcal{O}\}. \tag{8}$$

Definition 8 The ontology construction operator selection *is a mapping*

$$g_M: \{f: \mathcal{O} \to \mathcal{O}\} \mapsto f. \tag{9}$$

The *ontology construction operator suggestion* g_A gets existing ontology, instance and co-occurrence profiles on the input and produces a set of ontology construction operators. This is the *automatic* part of the procedure and can be implemented using machine learning algorithms. The ontology construction operator selection g_M gets a set of operators on the input and returns one of them. This is the *manual* part of the procedure and can be implemented through a user interaction, where the domain expert selects one of the suggestions.

The correct balance between the automatic and manual is an important element which influences the success of such approach. The user might get frustrated by too much work being assumed inside the automatic part with little influence from the user or by passing too many trivial decisions to the users.

Composite $g_M \circ g_A$ gives an operator, which, through a semi-automatic process, derives the next ontology construction operator to be applied to the ontology O_i. This can be formalized as

$$g_M \circ g_A : (O_i, P_I, P_{I\times I}) \mapsto f \text{ , and}$$
$$O_{i+1} = f(O_i) = (g_M \circ g_A)(P_I, P_{I\times I})(O_i) .$$

Definition 9 *Let g_A be an ontology construction operator suggestion and g_M an ontology construction operator selection.* The data-driven semi-automatic ontology construction operator *g is a composite mapping*

$$g_{P_I,P_{I\times I}}(O_i) = (g_M \circ g_A)(P_I, P_{I\times I})(O_i). \quad (10)$$

The data-driven ontology construction operator behaves the same as the ontology construction operator. The difference is twofold. First, the additional dependency to the instance and co-occurrence profiles provides domain data to the suggestion mapping g_A (data-driven). Second, the composition with the selection mapping g_M introduces the user interaction into the process (semi-automatic). We can now rewrite the ontology construction process from (5) as a data-driven semi-automatic ontology construction process

$$O' = g^n_{P_I,P_{I\times I}}(O_0) . \quad (11)$$

We denote the core concept construction operator, which adds sub-concepts to the concept *c*, as an operator $f_C[c]$. In order to break it into a semi-automatic process, we can rewrite core concept construction operator $f_C[c]$ as

$$f_C[c] = g_M^C \circ g_A^C[c](P_I, P_{I\times I}).$$

Here, $g_A^C[c]$ is the ontology construction operator suggestion, returning a set of core concept construction operators, each adding some sub-concepts to the concept c, and g_M^C is the ontology construction operator selection, in which the user manually selects one suggestion, which is then applied to the ontology. Section 4.2 presents several approaches for learning concept construction operators.

The semi-automatic process for learning core relation construction operator $f_R[c_1, c_2]$ is defined as

$$f_R[c_1, c_2] = g_M^R \circ g_A^R[c_1, c_2](P_I, P_{I\times I}).$$

Here $g_A^R[c_1, c_2]$ is the ontology construction operator suggestion, which produces a list of core relation construction operators, each adding some relations between concepts c_1

and c_2, and g_M^R is the ontology construction operator selection, in which the user manually selects one suggestion, which is then applied to the ontology. Section 4.3 presents several approaches for learning relation construction operators.

4 Machine Learning Algorithms for Ontology Learning

This Chapter describes several machine learning approaches for learning ontology construction operators, defined in the previous Chapter. The following subsections first introduce several possible feature representations of instances and co-occurrence profiles. Then unsupervised, supervised and semi-supervised machine learning approaches are used to learn concept construction operators, followed by generalisation of these approaches to learning relation construction operators. Then, several approaches for understanding instances, concepts and ontologies through naming and visualization are introduced. The Chapter finishes by describing approach for introduction of new instances to the developed ontology and incorporating additional background knowledge to the learning ontology construction operators.

4.1 Feature Representation

In order to use machine learning methods we must first derive a feature representation of instances, concepts and relations. This is done through the introduction of instance and co-occurrence profiles, which are defined as feature vectors living in some vector space. The feature representation of concepts and relations is derived from/based on the profiles and is discussed at the end of this section.

The feature representation of profiles depends on the type of data given for the ontology construction: vector space model for text data [24], visual words for image data [25], sparse matrix representation for social networks [26], etc. The ontology learning framework does not prescribe any conditions on the type of feature representations. These constraints are imposed by the actual choice of algorithms for learning the ontology construction operators and the modality of the domain data. For example, the instances of our running example presented in Figure 3 with text are in the remainder of the dissertation represented using word vectors (bag-of-words text representation).

In some cases instances can have multi-modal representation. An example of this would be images with descriptions, where each image can be represented with visual words vectors (extracted from the image) or with bag-of-words vectors (extracted from the textual description). Different feature representations for the same instance can be either combined, using simple concatenation of feature spaces or more advanced methods such as Canonical Correlation Analysis [27], or used separately, depending on the chosen learning algorithm. For example, visual words vectors can be used to derive concept suggestions and the bag-of-words vectors can be used to derive textual summaries (e.g.

top keywords) of the suggested concepts.

4.1.1 Vector Space Model

Most commonly used representation of the documents in text mining is *vector space model* representation, where text documents are presented as vectors of words. The representation was first used in the SMART Information Retrieval System project [24].

A vector space model for a document collection $D = \{d_1, d_2, \dots, d_n\}$ is defined as follows. Let $V = \{w_1, w_2, \dots, w_m\}$ be vocabulary of all the words that occur in D and TF_k^i be the number of occurrences of the word w_k in the document d_i. Document d_i is encoded as a vector $\boldsymbol{x}_i \in \mathbb{R}^m$, with each element x_i^k corresponding to a weight of the word w_k in the document. There are many options on how to choose x_i^k, the two most basic ones being frequency of the word in the corresponding document ($x_i^k = TF_k$) or occurrence of word in the document ($x_i^k = 1_{[TF_k>0]}$). The vectors in vector space model are also known as bag-of-words vectors, since such representation ignores the order of the words in the document.

Note that the dimensionality of the resulting vector space (equals the number of unique words in a corpus) is typically significantly larger compared to the number of unique words that occur in any single document. In such a case, vectors $\boldsymbol{x}_i$ are said to be *sparse*, meaning that only few elements x_i^k are nonzero.

Cosine similarity [28] is a similarity measure commonly used on documents originating from the field of information retrieval that is typically used in text mining. It is defined to be the cosine of the angle between two documents' bag-of-words vectors,

$$sim_{cos}(x_i, x_j) = \sum_{k=1}^{m} x_i^k \cdot x_j^k \Big/ \sqrt{\sum_{k=1}^{m} x_i^k \cdot x_i^k} \sqrt{\sum_{k=1}^{m} x_j^k \cdot x_j^k}.$$

Cosine similarity is reduced to the inner product of vectors x_i and x_j in the case when the vectors are normalized. Note that different similarity measures can be used on different tasks and that for sparse vectors cosine similarity is preferred over Euclidean distance and distances used for string matching [29].

The time complexity of computing the cosine similarity depends on the type of vectors. In the general case, calculation requires iteration over all the elements in vectors x_i and x_j, resulting in average time complexity of $O(m)$. However, in the case of sparse vectors only non-zero elements need to be taken into account, resulting in average time complexity of $O(s)$, where s is the average number of non-zero elements.

The performance of both bag-of-words vectors and cosine similarity largely depends on the choice of values x_i^k. Standard approach for assigning values is to break them down into document level score (e.g. frequency of the word in the document) and corpus level

word weight (e.g. inverse document frequency), and combine the two scores by, for example, multiplication. An example of such approach is TFIDF word weighting schema [24],

$$x_i^k = TF_k \cdot \log \frac{|D|}{|\{d_j : w_k \in d_j\}|}.$$

Influence of word weighting schema can be particularly important for unsupervised methods. More details on this will be presented in Section 4.6, together with an approach for domain specific selection of word weights.

There are also several approaches for improving the performance of vector space model in the pre-processing phase. The calculation of weight in TFIDF weighting already controls against frequently occurring words, such as, “and”, “or”, “then”, etc. However, most common words are typically handled by using a predefined list of stop-words: a list of words to be ignored in the text. Additionally, each word is typically normalized to some canonical form by either removing any suffix (process called stemming) or by replacing the suffix to derive the word’s lemma (process called lemmatization). Finally, frequently occurring sequences of words (also known as *n-grams*) are typically treated as one term [30].

In the reminder of the dissertation we will use a standard English stop-word list, which was defined in [24] and all words will be pre-processed using Porter stemmer [31] replacing the original word by its stem.

4.1.1.1 Example

In this section we demonstrate vector space model on the running example, where instances are public companies and each company is described by a paragraph of text, as shown in Figure 3. The example includes 10 instances.

The first step is the construction of the vocabulary by filtering the descriptions with standard stop-word list, normalizing the words with Porter stemmer, detecting frequent n-grams and keeping only the terms which occurred at least some predefined minimum number of times (in our case 3 times). Table 2 shows the resulting vocabulary, together with document frequency (DF) and inverse document frequencies (IDF). We can see that the term list contains several frequent n-grams, for example “investment banking” and “operating system”, which occurred several times in the text.

Table 2: Vocabulary defining vector space for the instance profile from the running example. Each word is assigned document frequency (DF, a number of documents in the collection containing the word) and inverse document frequency (IDF).

Term	DF	IDF
added	2	0.6990
addition	6	0.2218
advertising	2	0.6990
advisory	3	0.5229
America	3	0.5229
applications	4	0.3979
asset	1	1.0000
automotive	3	0.5229
banking	4	0.3979
based	4	0.3979
brand	6	0.2218
brokerage	3	0.5229
business	6	0.2218
cards	2	0.6990
cars	3	0.5229
client	2	0.6990
commercial	6	0.2218
company	10	0.0000
computers	4	0.3979
consumer	5	0.3010
content	3	0.5229
corporation	6	0.2218
custom	7	0.1549
dealers	4	0.3979
develops	4	0.3979
devices	4	0.3979
digital	3	0.5229
division	2	0.6990
equities	3	0.5229
financial	6	0.2218
financial services	3	0.5229
financing	5	0.3010
fleet	3	0.5229
ford	1	1.0000
founded	10	0.0000
Google	1	1.0000
government	5	0.3010
headquartered	7	0.1549
including	9	0.0458
information	4	0.3979
institutions	3	0.5229
institutions client	2	0.6990
internet	3	0.5229

Term	DF	IDF
investment	3	0.5229
investment banking	3	0.5229
investment managers	2	0.6990
lending	2	0.6990
loans	4	0.3979
managers	6	0.2218
managers services	5	0.3010
manufactures	5	0.3010
markets	7	0.1549
Microsoft	1	1.0000
mobile	4	0.3979
mobile devices	2	0.6990
motor	3	0.5229
networking	3	0.5229
offers	8	0.0969
online	3	0.5229
operating	7	0.1549
operating system	2	0.6990
platform	2	0.6990
products	8	0.0969
real	4	0.3979
related	4	0.3979
retail	7	0.1549
sales	5	0.3010
search	1	1.0000
sectors	1	1.0000
secure	4	0.3979
segment	4	0.3979
segment offers	3	0.5229
sells	5	0.3010
services	9	0.0458
sites	2	0.6990
software	4	0.3979
solutions	5	0.3010
stores	1	1.0000
system	3	0.5229
trucks	3	0.5229
united	3	0.5229
vehicles	3	0.5229
Volkswagen	1	1.0000
web	2	0.6990
windows	1	1.0000
worldwide	5	0.3010

Based on the vocabulary we can construct bag-of-words instance profiles using TFIDF weights. Figure 4 shows two example profiles. The bag-of-words vectors can be used to compute the cosine similarity between the documents. Figure 5 shows the similarity matrix covering all the example instances. The background of the matrix cells corresponds to the similarity score, with darker background for higher scores. The background indicates two clear clusters: financial companies (GS, BCS and C) and car manufacturers (F, VOW and GM). There is also some small signal showing the information technology cluster (APPL, MSFT, GOOG and NOK).

APPL:

added [*TF*=1, *TFIDF*=0.0894], addition [*TF*=1, *TFIDF*=0.0284], applications [*TF*=3, *TFIDF*=0.1527], business [*TF*=1, *TFIDF*=0.0284], company [*TF*=3, *TFIDF*=0.0000], computers [*TF*=2, *TFIDF*=0.1018], consumer[*TF*=1, *TFIDF*=0.0385], content [*TF*=2, *TFIDF*=0.1338], devices [*TF*=2, *TFIDF*=0.1018], digital [*TF*=3, *TFIDF*=0.2006], founded [*TF*=1, *TFIDF*=0.0000], government [*TF*=1, *TFIDF*=0.0385], headquartered [*TF*=1, *TFIDF*=0.0198], including [*TF*=2, *TFIDF*=0.0117], manufactures [*TF*=1, *TFIDF*=0.0385], markets [*TF*=2, *TFIDF*=0.0396], mobile [*TF*=1, *TFIDF*=0.0509], networking [*TF*=1, *TFIDF*=0.0669], online [*TF*=2, *TFIDF*=0.1338], products [*TF*=3, *TFIDF*=0.0372], related [*TF*=1, *TFIDF*=0.0509], retail [*TF*=3, *TFIDF*=0.0594], sales [*TF*=1, *TFIDF*=0.0385], sells [*TF*=4, *TFIDF*=0.1540], services [*TF*=1, *TFIDF*=0.0059], software [*TF*=2, *TFIDF*=0.1018], solutions [*TF*=1, *TFIDF*=0.0385], stores [*TF*=7, *TFIDF*=0.8954], united [*TF*=1, *TFIDF*=0.0669], worldwide [*TF*=2, *TFIDF*=0.0770]

VOW:

America [*TF*=2, *TFIDF*=0.1564], automotive [*TF*=2, *TFIDF*=0.1564], banking [*TF*=1, *TFIDF*=0.0595], brand [*TF*=2, *TFIDF*=0.0663], cars [*TF*=3, *TFIDF*=0.2346], commercial [*TF*=2, *TFIDF*=0.0663], company [*TF*=3, *TFIDF*=0.0000], corporation [*TF*=1, *TFIDF*=0.0332], custom [*TF*=1, *TFIDF*=0.0232], dealers [*TF*=1, *TFIDF*=0.0595], develops [*TF*=1, *TFIDF*=0.0595], division [*TF*=3, *TFIDF*=0.3136], financial [*TF*=3, *TFIDF*=0.0995], financial services [*TF*=3, *TFIDF*=0.2346], financing [*TF*=1, *TFIDF*=0.0450], fleet [*TF*=1, *TFIDF*=0.0782], founded [*TF*=1, *TFIDF*=0.0000], headquartered [*TF*=1, *TFIDF*=0.0232], managers [*TF*=1, *TFIDF*=0.0332], managers services [*TF*=1, *TFIDF*=0.0450], manufactures [*TF*=1, *TFIDF*=0.0450], markets [*TF*=1, *TFIDF*=0.0232], motor [*TF*=1, *TFIDF*=0.0782], offers [*TF*=2, *TFIDF*=0.0290], operating [*TF*=1, *TFIDF*=0.0232], products [*TF*=4, *TFIDF*=0.0580], sales [*TF*=1, *TFIDF*=0.0450], sells [*TF*=1, *TFIDF*=0.0450], services [*TF*=5, *TFIDF*=0.0342], trucks [*TF*=2, *TFIDF*=0.1564], vehicles [*TF*=4, *TFIDF*=0.3128], volkswagen [*TF*=5, *TFIDF*=0.7477]

Figure 4 Two example bag-of-words instance profiles. Each word is assigned a term frequency (number of occurrences in the document) the TFIDF weight (term frequency multiplied with inverse document frequency). Both vectors are normalized.

	APPL	MSFT	GOOG	NOK	F	VOW	GM	GS	BCS	C
APPL	1.000	0.052	0.061	0.185	0.019	0.014	0.073	0.014	0.034	0.024
MSFT	0.052	1.000	0.056	0.174	0.008	0.053	0.008	0.051	0.009	0.024
GOOG	0.061	0.056	1.000	0.124	0.003	0.001	0.010	0.006	0.007	0.014
NOK	0.185	0.174	0.124	1.000	0.049	0.043	0.047	0.126	0.079	0.087
F	0.019	0.008	0.003	0.049	1.000	0.293	0.425	0.036	0.072	0.053
VOW	0.014	0.053	0.001	0.043	0.293	1.000	0.394	0.030	0.080	0.071
GM	0.073	0.008	0.010	0.047	0.425	0.394	1.000	0.054	0.129	0.084
GS	0.014	0.051	0.006	0.126	0.036	0.030	0.054	1.000	0.542	0.428
BCS	0.034	0.009	0.007	0.079	0.072	0.080	0.129	0.542	1.000	0.436
C	0.024	0.024	0.014	0.087	0.053	0.071	0.084	0.428	0.436	1.000

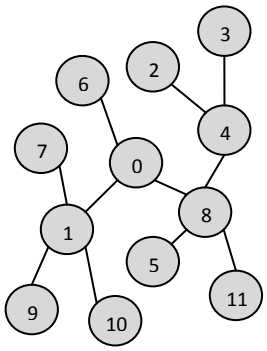
Figure 5 Matrix showing cosine similarities between all 10 example documents from the running example. Background colour corresponds to the similarity (darker means higher similarity).

4.1.2 Social Network Representation

A social network is typically represented as a graph, with nodes corresponding to individuals and edges corresponding to social connections between the individuals, such as, friendship, dislike, communication, etc. The edges can also be weighted depending on the intensity of, for example, communication. The use-case in Section 6.2 deals with such scenario, in which instances in the ontology correspond to individuals in a social network or, formally, nodes in a graph.

We use a representational approach proposed in [32] and [26] to derive instance profiles for individuals. First, a matrix is formed according to the formula proposed in [32], which defines the neighbourhood of a vertex to contain all the vertices at the distance of up to *d* steps from the vertex. Consequently, non-zero components represent the neighbours at step 1, 2, 3 … *d*. The elements of the matrix are computed using the formula $1/2^d$, where *d* is the distance in the number of steps from the vertex. This can be seen as a very rough but computationally efficient approximation of probabilities that a random walker on the graph, starting from the vertex, would reach a neighbouring vertex normalized by the number of neighbours [33]. Figure 6 illustrates the graph transformation on an example graph assuming all the edges in the graph have weight 1.

Instance profiles are defined as rows in the resulting matrix corresponding to the particular individual (node). Similarity between instance profiles is calculated by using cosine similarity, as defined in the vector space model.



	0	1	2	3	4	5	6	7	8	9	10	11
0	1	0.5			0.25	0.25	0.5	0.25	0.5	0.25		0.25
1	0.5	1					0.25	0.5	0.25	0.5	0.5	
2			1	0.25	0.5				0.25			
3			0.25	1	0.5				0.25			
4	0.25		0.5	0.5	1	0.25			0.5			0.25
5	0.25				0.25	1			0.5			0.25
6	0.5	0.25					1		0.25			
7	0.25	0.5						1		0.25		
8	0.5	0.25	0.25	0.25	0.5	0.5	0.25		1			0.5
9	0.25	0.5						0.25		1	0.25	
10		0.25								0.25	1	
11	0.25								0.5			1

Figure 6: Illustration of the graph (left) transformation into a sparse matrix (right) where the rows represent instances (vertices) and columns represent neighbourhood with weights relative to the distance from the vertex in that row. Here we have set the maximal distance to $d = 2$. Notice that the diagonal elements have weight 1 (showing that the each vertex is in its own neighbourhood). The dashed lines point out neighbouring vertices and the corresponding weights for vertex labelled as 2. It has four non-zero elements in its sparse vector representation (1, 0.25, 0.5, 0.25) corresponding to four vertices (labelled in the graph as 2, 3, 4, 8).

4.2 Learning Concept Construction Operators

The ontology learning frameworks, presented in Chapter 3, defined the concept construction operator as the addition of new concepts to the ontology. This was further restricted with core concept construction operator $f_C[c]$, which can add sub-concepts to only one concept *c*. Several such operators can be combined to construct a sub-tree of concepts.

In the previous section we defined feature representation of instance profiles, which will serve as one of the main inputs to concept learning. This section several text mining and machine learning approaches which can be used to implement the core concept learning function $f_C[c]$.

4.2.1 Motivation

Let us assume that we are trying to derive sub-concepts of a concept *c*, with

$$\iota_C(c) = \{e_1, e_2, \ldots, e_n\}.$$

The position of the new sub-concept in the concept hierarchy is given as $c' \leq_C c$ and the definition of a new sub-concept c' requires a subset of instances $\iota_C(c') \in \iota_C(c)$. The set of instances $\iota_C(c')$ can be the result of machine learning tasks using different levels of additional data or user interventions.

There are various reasons why a particular set of instances would form a sub-concept. The assumption throughout this thesis is (1) that the instance profiles contain enough information to reflect the sought for underlying structure of the domain that is being encoded in the ontology and (2) that the sets of instances corresponding to the new sub-concepts can be discovered using machine learning algorithms. Discovered sub-concepts are then presented to the user as suggestions, with the approved ones being added to the ontology.

The case, where the only input is the set of instances $\iota_C(c)$ and their corresponding instances profiles is called *unsupervised learning*. Clustering is a type of unsupervised learning, which produces on the output a set of subsets (clusters) of similar instances, which can be easily transformed into sub-concepts. One example of clustering algorithms is *k*-means [34], presented in more details in Section 4.2.2. Other types of unsupervised learning algorithms include dimensionality reduction (e.g. LSI [35]) or topic modelling (e.g. LDA [36]) techniques. The latter is also well suited for the task of learning concept construction operators and to be used in the ontology learning framework.

Another frequent case is the reuse of concepts from some fixed external vocabulary [37]. One approach is to develop a classifier, which can assign label(s) from the external vocabulary to one instance profile. Sets of instances, assigned to the same label, can be used as sub-concept suggestions. An example of external vocabulary is presented in DMoz Open Directory [38].

The system can also interact with the user during the construction of suggestions. One such approach is active learning [39], where the user comes with the initial idea of what kind of concept to add to the ontology (e.g. “finance”). The system then guides the user through a process of manually preparing a small training set by selecting several instances which belong to the concept. This process is designed in such a way as to minimize the required amount of labelled instances. The assembled training set is then used to train a classifier which can identify other relevant instances, and form a sub-concept.

In the ontology learning framework, a concept is defined by its position in a hierarchy and by a subset of instances. We can use instance profiles to train a classification model for each concept, which can predict the probability of another instance belonging to a particular profile. Such model can be used when adding new instances, not available at the initial stage of ontology engineering, to the ontology.

The classification model can also provide an input to methods such as concept name suggestion, instance outlier detection and visualization of concepts using keyword extraction or topic maps. Examples of using such model will be shown in Sections 4.5 and 4.6.

4.2.2 Clustering

Clustering is a technique for partitioning a set of instances so that each partition (or cluster) contains instances with higher similarity with the instance within the cluster as opposed to the instances from other clusters. In the case of text this can be seen as finding groups of documents which share similar words and, as such, topics. In the case of social network this can be seen as finding communities of individuals which are more tightly connected compared to the whole network.

The system implemented as a part of this dissertation is using *k*-means [34] clustering algorithm. The algorithm has already been successfully used on text documents [40] to

cluster a large document corpus based on the document topic and incorporated in an approach for visualizing a large document collection [41].

4.2.2.1 The *k*-means Algorithm

The *k*-means clustering algorithm is based on an iterative approach, which partitions a set of instances $X = \{x_1, x_2, \ldots, x_n\}$ into k clusters $C = \{c_1, c_2, \ldots, c_k\}$ using the procedure presented in Figure 7. The number of clusters k is fixed, given as a parameter to the algorithm, and each cluster is represented by a *centroid vector*, a mean of all member instances.

Input: set of instances X, similarity measure S, number of clusters k
Output: set of clusters C
Procedure:
  Set initial k centroids
  Repeat
    Assign each instance x to the cluster maximizing $\arg\max_{c_i \in C} S(\mu_i, x)$
    Update the centroids $\mu_i = \frac{1}{|c_i|} \sum_{x \in c_i} x$.
  Until the assignment of instance to clusters has not changed

Figure 7: The *k*-means clustering algorithm.

The algorithm starts by selecting initial centroid vectors, and then iterates between the assignment step, where instances are assigned to the nearest cluster based on a given similarity measure, and the update step, where centroid vectors for all clusters are updated using the new assignments. The algorithm is said to converge once the assignment of instances to the clusters does not change. There are various ways of initializing the algorithm. The simplest approach is to uniformly sample k instances and use them as initial centroids. However, such approach can result in slow convergence or convergence to local optima. Several approaches were suggested in recent years to optimize the initial suggestion. The best known approach is *k*-means++ [42], which still samples initial centroids from the instances, but adjusts the probability of an instance being selected to be proportional to the square of the distance between the instance and the already defined centroids.

4.2.2.2 The complexity of the *k*-means Algorithm

The average time complexity of the *k*-means algorithm depends on the initialization phase, assignment step, update step and number of iterations necessary for convergence. The average time complexity of *k*-means++ initialization and assignment step is $O(k \cdot n \cdot s)$, where s is the average number of non-zero elements in the instance profiles. The average time complexity of the update step is $O(n \cdot s)$ which, together with the

assignment step, results in average time complexity of $O(k \cdot n \cdot s + n \cdot s)$ per iteration. In typical scenario, values for both k and s would be relatively small compared to n, rendering the time complexity of initialization and one iteration practically linear with respect to the number of instances.

The convergence rate is shown to be super-polynomial in worst case, and polynomial with high probability. In practice, the algorithm is known for fast convergence [43].

The memory requirements for the algorithm are, besides the storage of instances profiles, $O(k \cdot m)$ for centroid vectors, where m is the dimensionality of the vector space, and $O(n)$ for assignment of instances to clusters. All together this results in the memory complexity of $O(k \cdot m + n)$.

4.2.2.3 Application to Learning for Concept Construction Operators

Clustering has a natural application to the concept learning as defined within the ontology learning framework in Chapter 3. As shown in Section 3.3.1, a core concept construction operator $f_C[c]$ can be broken down into a semi-automatic process as $g_M^C \circ g_A^C[c](P_I, P_{I\times I})$, where $g_A^C[c]$ is an operator suggestion and g_M^C is an operator selection. Mapping $g_A^C[c]$ can be implemented by first applying clustering to the profiles of instances $\iota_C(c)$, followed by using the instance clusters as sub-concepts of concept c. Mapping g_M^C can be implemented by displaying the discovered clusters and allowing the user to either select the ones, which should be added to the ontology, or revising the initial clustering parameters, such as the number of clusters in case of *k*-means.

We will demonstrate concept learning with clustering by starting with a minimal ontology O_0, given the instance set I of our running example. Minimal ontology, consists only of the root concept c_r and $\iota_C(c_r) = I$. For instance profiles we will use the vector space model representation and cosine similarity as a similarity measure.

Table 3, Table 4 and Table 5 show the results of clustering the instances I for different values of parameter k. In the tables, each cluster is represented by a list of instances, mean similarity of instances towards cluster centroid vector and a set of top keywords. The keywords are extracted as words corresponding to the elements in the centroid vectors with the highest weights. More details on alternative approaches for the keyword extraction are presented in Section 4.4.

Table 3: Results of *k*-means clustering with $k = 2$.

	Instances	Mean Similarity	Keywords
c_1	$\iota_C(c_1) = \{VOW, F, GM, BCS, GS, C\}$	0.581	*vehicles*, *cars*, *investment*, *banking*, *trucks*,
c_3	$\iota_C(c_3) = \{AAPL, MSFT, GOOG, NOK\}$	0.693	*online*, *networking*, *system*, *mobile*

Table 4: Results of *k*-means clustering with $k = 3$.

	Instances	Mean Similarity	Keywords
c_1	$\iota_C(c_1) = \{VOW, F, GM\}$	0.867	*vehicles, cars, trucks, automotive*
c_2	$\iota_C(c_2) = \{BCS, GS, C\}$	0.758	*investment, banking, institutions, loans*
c_3	$\iota_C(c_3) = \{AAPL, MSFT, GOOG, NOK\}$	0.693	*online, networking, system, mobile*

Table 5: Results of *k*-means clustering with $k = 4$.

	Instances	Mean Similarity	Keywords
c_1	$\iota_C(c_1) = \{VOW, F, GM\}$	0.867	*vehicles, cars, trucks, automotive*
c_2	$\iota_C(c_2) = \{BCS, GS, C\}$	0.758	*investment, banking, institutions, loans*
c_3	$\iota_C(c_3) = \{AAPL, GOOG\}$	0.857	*online, content, applications, digital*
c_4	$\iota_C(c_4) = \{MSFT, NOK\}$	0.844	*system, segment, develops, networking*

Looking at the instances from the running example, there are three natural sub-concepts of c_r, namely Finance, Technology and Auto Manufacturers. The results of *k*-means clustering with $k = 3$ directly correspond to these three sub-concepts. In the case of $k = 2$, we can see that finance and auto manufactures were merged, and in the case of $k = 4$, the technology companies are split into two separate clusters.

4.2.3 Reusing Existing Vocabulary

A good practice in ontology construction is to reuse concepts or terminology from existing vocabularies, thesauruses, dictionaries, ontologies, etc. This is also the case, when we want to establish links between a developed ontology and existing external resources, as in the linked data movement [47]. We further refer to external resources as *vocabulary*, since this section focuses only on reusing individual concepts from resources, and not the structure of, for example, existing ontologies.

The approach to linking instances in the ontology learning framework to external resources is based on classifiers, which can place a given instance or co-occurrence profile into one or more concepts from the external vocabulary. The reminder of this section presents a library of external resources, approach for linking the instances with labels using vocabulary grounding and the implementation of the core concept an operator using grounded vocabularies.

4.2.3.1 Vocabulary Grounding

Vocabulary grounding, as used in this section, corresponds to developing a classification which can classify a new instance into one or more concepts from the vocabulary.

An important constraint for classifiers is the sheer size of the modern vocabularies. For example, the DMoz vocabulary, presented in the following section, contains one million concepts. The scale does not allow for the use of complex state-of-the-art text classification approaches, such as support vector machines [44].

The system implemented as a part of this dissertation is using the *k*-Nearest Neighbour (kNN) based approach proposed in [50] and [51], which was proven to scale well with respect to the number of instances and concepts, while maintaining satisfactory performance. The approach constructs a centroid vector for each concept, taking into account all the instance profiles. This naturally extends to hierarchies, where instances assigned to a particular concept are automatically propagated to the concepts upper in the hierarchy, and as such also contribute to their centroids.

The classification of a new instance is done by (1) calculating the similarity between the instance's profile and each of the centroids, and (2) selecting the concepts corresponding to the top k most similar centroids. The average time complexity for this procedure is $O(N \cdot s \cdot s_c \cdot \log(k))$, where N is the number of concepts in the external vocabulary, s is the average number of non-zero elements in the instance profile and s_c is the average number of non-zero elements in the centroid vectors. The factor $O(\log(k))$ is the cost of maintaining a heap containing k elements with the highest similarity score.

When using the vector space model as the feature representation, a typical instance profile would typically contain up to a hundred different words. We can speed up the classification process by focusing only on the centroids with high enough word overlap with the instance profile. This can be done efficiently by indexing the centroids using inverted index, which is queried with a disjunction of all the words appearing in the instance profile. When using this approach, the factor N in the average time complexity is replaced with the average number of centroids that overlap in words with the instance profile. This can vary significantly between different vocabularies and can be controlled by specifying a minimal size of the word overlap.

Using the inverted index to select candidate centroids typically results in decreased performance. As an experiment, the proposed approach was applied to the list of top level DMoz categories, listed in Table 6. Results are presented in Table 7, showing the recall when looking at top 1, 3, 5 or 10 centroids. It can be seen, that in both cases the chance of the correct category being top ranked is 50%. However, the recall increases slower in the case of inverted index, and tops at around 80%, whereas direct centroid approach tops, as expected, at 100%. The reason for lower increase or recall in the case of inverted index is the approximation step, where the correct category might not be among the retrieved.

It is important to remember that user interaction is a part of the proposed framework.

This requires responsiveness and speed from the implemented system, but also means that the user can compensate for some of the errors, which would be unacceptable in a fully automatic scenario. For the case of classification into external vocabulary this means, that we are not concerned so much about the recall for the case of $k = 1$, and are more focused for the performance when $k = 5$ or $k = 10$.

Table 6: List of top level DMoz categories.

Category	Documents
Arts	87,039
Business	225,352
Computers	95,715
Games	39,727
Health	54,721
Home	23,769
News	7,636
Recreation	95,426
Reference	55,553
Regional	551,550
Science	107,014
Shopping	85,682
Society	188,226
Sports	83,756

Table 7: Recall at 1, 3, 5 and 10 top ranked categories. Performance is shown for centroid approach, and approximate approach using inverted index.

Rank	Centroid	Inverted Index
1	0.53	0.47
3	0.83	0.70
5	0.93	0.76
10	0.99	0.78

4.2.3.2 Examples of Existing Vocabularies

There is a range of available vocabularies, especially with the ever-increasing Linked Open Data (LOD) [47]. In this section, we will present several vocabularies that were used or tested in the presented framework, and will detail for each dataset what was the source of text documents used for grounding.

DMoz or The Open Directory Project (ODP) [38] is the largest multilingual open content directory of World Wide Web links that is constructed and maintained by a community of volunteer editors. The version at the time of writing contained over 1 million concepts and three different relationships. The taxonomic part can be grounded with the content which is available within the downloadable data, mainly short textual descriptions of the manually categorized web sites within each DMoz category.

EuroVoc [45] is a multilingual thesaurus covering the fields in which the European Communities are active. The thesaurus is the main annotation system for indexing the

documents in the documentation systems of the European institutions and of their users. The European Parliament, the Office for Official Publications of the European Communities, the national and regional parliaments in Europe, some national government departments and European organizations are currently using this controlled vocabulary. EuroVoc concepts can be grounded with the documents from Acquis Communitarian, the corpus of European legislation indexed with EuroVoc descriptors.

The Cyc [46] knowledge base (KB) is a formalized representation of a vast quantity of fundamental human knowledge: facts, rules of thumb, and heuristics for reasoning about the objects and events of everyday life. The original form of representation is a formal language CycL. The KB consists of terms which constitute the vocabulary of CycL and assertions which relate those terms. These assertions include both simple ground facts and rules with variables. The KB is written in English, has two types of classes (concepts and lexical nodes), it has 3295 relations, and 464.988 concepts. Since Cyc has only structure (concepts and facts) each Cyc's concept can be grounded by querying Google with lexical representation for that class and selecting top few results.

ASFA (Aquatic Sciences and Fisheries Abstracts) [49] is a thesaurus used for the Aquatic Sciences and Fisheries Information System (ASFIS), an international co-operative information system for the collection and dissemination of information covering the science, technology and management of marine, brackish water, and freshwater environments. It contains approximately 1 million bibliographic references to the world's aquatic science literature accessioned since 1971 (for some journals and/or subject areas the coverage precedes 1971). All references are machine readable. The thesaurus structure included two types of classes (descriptor and non-descriptor), 5 link types, and 9882 classes. ASFA classes can be grounded with text abstracts available within the records of the crawled data (over 360.000 abstracts).

AgroVoc [48] is a multilingual structured thesaurus of all subject fields in Agriculture, Forestry, Fisheries, Food security and related domains (e.g. Sustainable Development, Nutrition, etc). It consists of words or expressions (terms) in different languages and is organized in the thesaurus relationships (e.g. "broader", "narrower", and "related") used to identify or search resources. Its main role is to standardize the indexing process in order to make search simpler and more efficient, and to provide users with the most relevant resources. AgroVoc classes can be grounded with text abstracts from ASFA document corpus which are close to AgroVoc terms.

4.2.3.3 Application to Learning the Concept Construction Operators

As shown in Section 3.3.1 we can decompose the core concept construction operator $f_C[c]$ into a semi-automatic process as $g_M^C \circ g_A^C[c](P_I, P_{I\times I})$. Mapping $g_A^C[c]$ can be implemented with grounded external vocabulary by first classifying each instance profile separately, and then aggregating and sorting the resulting list of external concepts according to the number of classified instances. Another, faster, option is to first calculate

an instance profile centroid vector for concept c, and classify only the centroid. The manual step g_M^C can be implemented, similarly as in the case of clustering, by displaying a list of the top suggested sub-concepts and allowing the user to select which ones should be added to the ontology.

The differences between the two different approaches for implementing f_A^C can be demonstrated on the running example. Figure 8 shows the resulting list of DMoz concepts, when classifying only the instance profile centroid vector. The list corresponds to the initial concept c, containing concepts such as "Business", and some of the sub-concepts. However, the list is also missing any concept related to auto manufacturer instances. We will show in Section 4.4 how such approach can be used to highlight the topics or contents of a particular concept. Figure 9 shows the resulting list for the same set of instances, but applying classification separately to each instance and aggregating the resulting concepts. The list again covers the topic of the initial concept c with concepts such as "Business", which were identified for almost all instances. However, the list also contains concepts related to all three sub-concepts we identified manually: "Computers", "Automotive" and "Financial Services".

```
Business, Financial Services, Regional, Business and Economy, Investing,
United Kingdom, Europe, Software, Investment Services, Oceania, Australia,
Equipment and Software, Banking Services
```

Figure 8: List of top DMoz concepts when classifying the instance profile centroid from the running example.

```
Business (8), Regional (8), United Kingdom (8), Business and Economy (7),
Europe (7), Computers (4), Africa (3), Australia (3), Automotive (3),
Financial Services (3), Motoring (3), Oceania (3), Shopping (3), Software
(3), South Africa (3)
```

Figure 9: List of top DMoz concepts when classifying each instance profile from the running example separately and aggregating the results. The number of instances, assigned to a particular concept, is listed in the parenthesis.

4.2.4 Active Learning

Both clustering and reuse of existing vocabulary attempt to discover concepts, with the role of the user being largely reduced to agreeing with suggestions or disregarding them. However, sometimes the user would already have some seed, like an idea (e.g. topic) or an example (e.g. subset of instances), around which a new concept can be built. Active learning approach lets the user to construct new concepts around the seed. The construction revolves around guided interaction with the system, and results in an accurate new concept in significantly less time than a manual approach would require.

Active learning [52] is a generic term describing an interactive learning process. In the usual supervised learning the learning method is presented with a static training set that is

used to construct a model. The active learning paradigm assumes the availability of an unlabelled set, from which the learning method ('student') can select few examples and 'asks' the 'oracle' (e.g. a domain expert, the user) to label them [52]. The new labelled examples are used to update the model, closing the active learning loop (Figure 10).

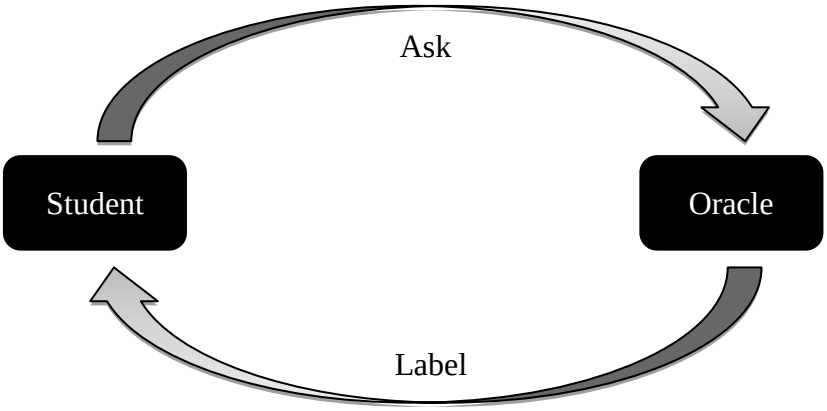


Figure 10: The active learning loop: the student selects examples and asks the oracle to label them. The oracle provides labels back to the student, who uses them to update the model and select the next round of examples to label.

The questions (examples to be labelled) are selected based on their potential to improve the existing model if their labels are known. The goal of the process is to approach the accuracy of a model, trained on completely labelled dataset, but with as few labelled examples as possible. There are different methodologies for selecting examples, uncertainty sampling (US) being one of the most popular [52]. US selects examples, for which the existing model is most unsure on how to label them. In the case when the classifier is SVM [53], this translates to choosing the examples which are closest to the separating hyperplane.

4.2.4.1 Application to Learning the Concept Construction Operators

As shown in Section 3.3.1 we can decompose the core concept construction operator $f_C[c]$ into a semi-automatic process as $g_M^C \circ g_A^C[c](P_I, P_{I\times I})$. This was sufficient for clustering and reusing existing vocabulary, where there was a clear automatic part, followed by the manual part. However, the above formulation needs to be updated due to the loop inherent in the active learning approach.

As a first step, let us formulate the active learning process with only one iteration as a concept construction operator. The input to the automatic step is the concept c, which is being refined, and the seed for active learning (a set of positive examples $I^+ \subseteq \iota_C(c)$ and a set of negative examples $I^- \subseteq \iota_C(c)$). We can thus formulate the automatic step as mapping $g_A^C[c, I^+, I^-]$. The mapping $g_A^C[c, I^+, I^-]$ performs the 'student' part of the active learning loop and selects an example e, which should be labelled by the oracle. The result of this mapping is a set of three core concept construction operators, covering the three potential outcomes of the manual labelling:

- the oracle adds the example e to the positive set I^+,
- the oracle adds the example e to the negative set I^-,
- the oracle does not label the example e.

Each outcome defines a core concept construction operator, by (1) training a classifier on the updated training set $I^+ \cup I^-$, (2) applying the classifier to the instances from $\iota_C(c)$, and (3) creating a sub-concept containing all the positively classified instances. The manual step g_M^C corresponds to the user selecting one of the suggested operators. The above semi-automatic process can be easily generalized to more iterations of active learning, by looping over the above two steps.

We can demonstrate the active learning process on the running example. Table 8 shows an example using active learning to learn the concept of "Information Technology Company". The example starts with Nokia and Microsoft as positive and General Motors and Citigroup as negative examples. Based on this the system decides to ask whether Apple should belong to the concept (Step 1 in Table 8). After obtaining a positive answer, the system updates the model and asks for Volkswagen (Step 2 in Table 8). It should be noted, that in this simple scenario, the model derived after the first step already correctly classifies all the instances from the running example.

Another example can be seen in Table 9, where active learning is used to learn the concept of finance companies. An interesting thing to note here is that the initial set of negative examples contains two car manufacturing companies and no information technology ones. As such, the first question produced in the active learning process is to establish the position of information technology companies Apple and Microsoft. This loop is further illustrated with a manually drawn visualization of the process from Table 9 as presented in Figure 11. The visualization shows three clusters ("finance", "car manufacturing", and "information technology") and how, due to no information technology company being in the initial training set, the active learning process first focuses on determining their polarity (positive or negative).

Table 8: Active learning process example learning the concept of "Information Technology Company".

Step	I^+	I^-	Question
1	$\{NOK, MSFT\}$	$\{GM, C\}$	$APPL$
2	$\{NOK, MSFT, APPL\}$	$\{GM, C\}$	VOW
3	$\{NOK, MSFT, APPL\}$	$\{GM, C, VOW\}$	$GOOG$

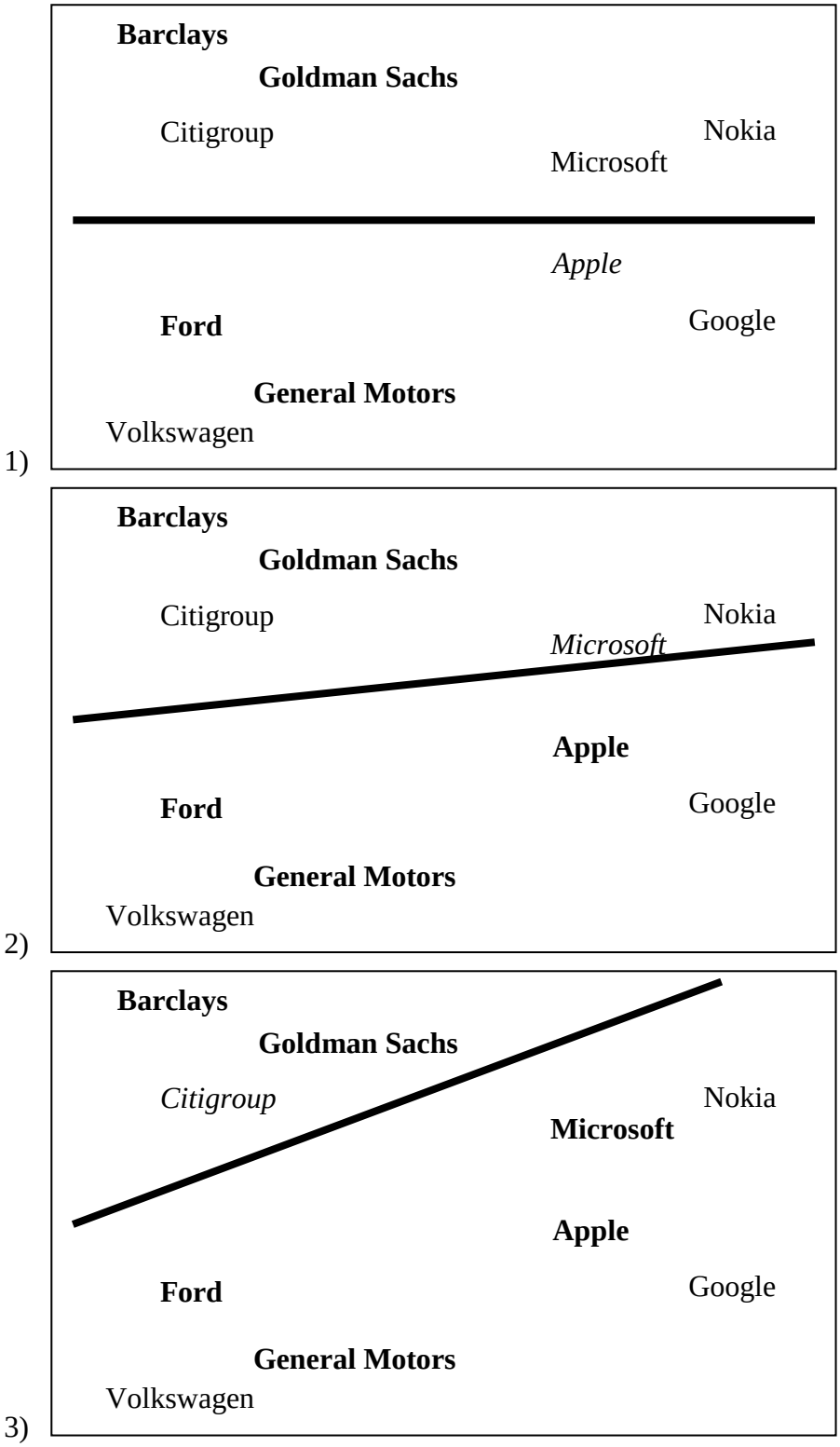


Figure 11: Visualization of the active learning loop from Table 9. The visualization shows, how the financial companies are isolated through sampling of the space and how technology companies, not in the initial training set, are then being selected as most informative for determining the space. Instances from the training set are marked as bold and the instance in question is marked as italic. The classification model is depicted as a separating line (SVM hyperplane), with the instance closest to the line being selected as the question.

Table 9: Active learning process example learning the concept of “Finance company”.

Step	I^+	I^-	Question
1	$\{BCS, GS\}$	$\{F, GM\}$	$APPL$
2	$\{BCS, GS\}$	$\{F, GM, APPL\}$	$MSFT$
3	$\{BCS, GS\}$	$\{F, GM, APPL, MSFT\}$	C

Note that in order to efficiently kick-start the active learning process, an initial seed of positive and negative instances is required. In some cases, this can be relaxed with the introduction of a separate process, which lets the user to select the initial seed. For example, in the case where instance profiles are text documents, this process can be reduced to a simple search and ranking exercise: (1) the user writes a query describing the desired sub-concept, (2) the system selects few most similar and least similar instances from $\iota_C(c)$ and (3) the user manually selects the correct positive and negative instances. This process can generate enough material for the active learning to start.

An illustration of two such queries over instances from our running example is given in Figure 12 and Figure 13. The error in the last example (GS is among least similar instances and APPL is among most similar) shows that there is a need for a manual filtering step performed by the user, to start with a good initial seed.

Query: “`technology, computers, internet`”
Most similar: $\{APPL, MSFT, GOOG\}$
Least similar: $\{GS, BCS, C\}$

Figure 12: Example query over instances from the running example.

Query: “`banking, finance`”
Most similar: $\{C, BCS, APPL\}$
Least similar: $\{VOW, GM, GS\}$

Figure 13: Example query over instances from the running example.

4.3 Learning Relation Construction Operators

The ontology learning framework defines the relation construction operators as addition of new relations to the ontology. By restricting operator to only one pair of concepts, we get the core relation construction operator $f_R[c_1, c_2]$. This section will show how machine learning techniques can be used to learn core relation construction operators.

In [55] co-occurrence profiles were used for the identification of relations between name entities with application to automatic building of ontology from a data stream. As an example, it used a stream of news articles from which the main name-entities were

extracted. Each name-entity was represented by a time dependent profile and each pair of name-entities that co-occurs in one or more news articles was represented by a time dependent co-occurrence profile. In this work, we follow similar approach. We use co-occurrence profiles of instances from different concepts as a set of feature vectors used by machine learning techniques to generate suggestions for possible relations. Also, we focus only on static profiles, with no dependencies on time.

For concept operators, the number of instances was a good indicator of whether the concept should be further split into sub-concepts or not. In the case of relations, there is no direct signal for this within the ontology representation. However, co-occurrence profiles can be used as an indicator of the overlap between concepts, which can be expressed as relation. In any case, machine learning algorithms require some amount of co-occurrence profiles to be present.

The approach for learning relation construction operators is similar to the learning of the concept construction operators presented in the Section 4.2.2. The major difference is the use of co-occurrence profiles as opposed to instance profiles. However, once we arrive to a set of feature vectors, same machine learning approaches can be applied, such as clustering and active learning.

Once a relation is constructed between a pair of concepts, it can be modelled and applied to other concept pairs. Approach from Section 4.2.3 for reusing existing vocabularies can be used here, where already constructed relations serve as existing vocabulary.

4.3.1 Relation Identification

The proposed relation identification method is based on [54]. The method is based on ranking pairs of concepts proportionally to the ratio of co-occurrence profiles with an instance from the first and an instance from the second concept.

We start by defining the set of all co-occurrence profiles connecting concept c_1 with concept c_2 as

$$P_{I\times I}(c_1, c_2) = \{(e_1, e_2) \in P_{I\times I} | e_1 \in \iota_C(c_1), e_2 \in \iota_C(c_2)\},$$

and the set of all co-occurrence profiles, originating from concept c_1 as

$$P_{I\times I}(c_1, \cdot) = \{(e_1, e_2) \in P_{I\times I} | e_1 \in \iota_C(c_1), e_2 \notin \iota_C(c_1)\}.$$

Based on this notation we can denote the ratio of co-occurrence profiles linking c_1 with c_2, with respect to all the co-occurrence profiles originating in concept c_1, as

$$\rho(c_1, c_2) = \frac{|P_{I\times I}(c_1, c_2)|}{|P_{I\times I}(c_1, \cdot)|}.$$

Similarly, $\rho(c_2, c_1)$ denotes the ratio of co-occurrence profiles linking c_2 with c_1. Finally, we denote the ratio of all co-occurrence profiles shared by two concepts as

$$\rho_{all}(c_1, c_2) = \rho_{all}(c_2, c_1) = \frac{|P_{I\times I}(c_1, c_2)| + |P_{I\times I}(c_2, c_1)|}{|P_{I\times I}(c_1, \cdot)| + |P_{I\times I}(c_2, \cdot)|}.$$

There are several ways to rank a pair of concepts c_1 and c_2 using the above scores. The existence of a relation from concept c_1 to concept c_2 can be observed through the score $\rho(c_1, c_2)$. Similarly, a relation from concept c_2 to concept c_1 can be observed through the score $\rho(c_2, c_1)$. Finally, mutual relations can be identified using the score $\rho_{all}(c_1, c_2)$.

The above scores can serve only as an indication for potential relation. After the identification of two potentially related concepts, machine learning approaches can be applied to generate suggestions.

4.3.2 Suggesting Relations

Let c_1 and c_2 be a pair of concepts for which we would like to find relations. We can isolate all the co-occurrence profiles to a set $P_{I\times I}(c_1, c_2)$, by keeping just the profiles connecting instances from these two concepts. These profiles can be used to extract feature vectors and derive relations using the machine learning approaches presented in Section 4.2.

More formally, the goal is to learn the core relation construction operator $f_R[c_1, c_2]$. In Section 3.4 we defined the semi-automatic process for learning core relation construction operators as composite of an automatic and a manual step $g_M^R \circ g_A^R[c_1, c_2]$ applied to instance and co-occurrence profiles $(P_I, P_{I\times I})$. Here, $g_A^R[c_1, c_2]$ is an operator suggestion, providing a list of potential core relation construction operators, and g_M^R is an operator selection, selection one from the provided list.

The operator suggestion $g_A^R[c_1, c_2]$ can be implemented by any of the already mentioned machine learning approaches. For example, clustering can identify groups of co-occurrence profiles, which correspond to a particular relation, external relation vocabularies can be used to identify relations, by classifying co-occurrence profiles, and active learning can be used to train the system to recognize a particular relation, that we know there exists, but would like to have it grounded in co-occurrence profiles.

An important assumption in learning the relation construction operator is that a pair of instances has a co-occurrence profile when these two instances are in some way related and the profile carries the information on the type of relation.

Once a new relation is added to the ontology it can be easily reused. We can define a new vocabulary consisting of all constructed relations. Since each relation is grounded in a set of co-occurrence profiles, a classifier can be trained which can identify the relation. These classifiers can be used to identify the relation among other concept pairs, same as in the approach for reusing existing external vocabularies.

4.3.3 Example

The running example does not contain co-occurrence profiles and as such an extended dataset is required to demonstrate the learning of the core relations construction operators. In this example we will use a dataset derived from Reuters RCV1 corpus [56], which contains almost one million news articles produced by Reuters from the summer of 1996 till the summer of 1997.

All articles from the corpus were automatically processed and all the named entities were extracted using Enrycher [57]. In this example, named entities correspond to instances and the concatenation of all the sentences, in which an entity occurs, is taken as its instance profile. Occurrence of other named entities in the profile is removed, to restrict the relational information to co-occurrence profiles. Figure 14 shows an example of one such instance profile, listing sentences in which instance “Agassi” occurred.

- “There will be no repeat of last year's men's final with eighth-ranked Agassi landing in ___'s half of the draw.”
- “Bumping ____ up to the sixth seeding avoided the possibility that he would run into Agassi as early as the quarter-finals, but they could lock horns in the semis.”
- “Olympic champion Agassi meets ___ of Morocco in the first round”
- ...

Figure 14: Example instance profile for instance “Agassi”. Missing entities are marked with underscore.

- “There will be no repeat of last year's men's final with eighth-ranked ___ landing in ___'s half of the draw.”
- “Bumping ____ up to the sixth seeding avoided the possibility that he would run into ___ as early as the quarter-finals, but they could lock horns in the semis.”
- “‘It's just to put ___ not to face (Pete) ___ in the quarters,” Muster charged, suggesting that officials were hoping to arrange a rematch of last year's - final.”
- “The eighth-ranked ____, who had miserable showings at Wimbledon and the French Open this year, could well have run into top-ranked ___ in the quarter-finals had Open officials seeded him eighth.”
- ...

Figure 15: Example co-occurrence profile for instances “Agassi” and “Sampras”. Missing entities are marked with underscore.

The co-occurrence profile for instances e_1 and e_2 is constructed by concatenating all the sentences in which the two instances co-occur. If the two instances never co-occur in the same sentence, then they do not have a co-occurrence profile. All named entities were

removed from the sentences in the same way as was done in the instance profile. Figure 15 shows an example of co-occurrence profile, listing all the sentences in which instance "Agassi" and "Sampras" occurred.

By applying clustering to the 1000 most frequent named entities we arrive to a sample ontology, presented in Figure 16, containing five concepts. Checking the instances assigned to each concept we can see the following: c_1 contains a list of countries, c_2 and c_3 contain lists of politicians, c_4 legal institutions, and c_5 contains a list of public utility companies.

The figure shows top keywords (details in Section 4.4.1) extracted from co-occurrence profiles between some of the concept pairs. From the keywords it can be seen, that c_2 contains ministers ("ministerOf" relation) and c_3 contains presidents ("presidentOf") of countries from c_1. The same relation can also be abstracted to the level "politicianOf" and applied to both concept pairs.

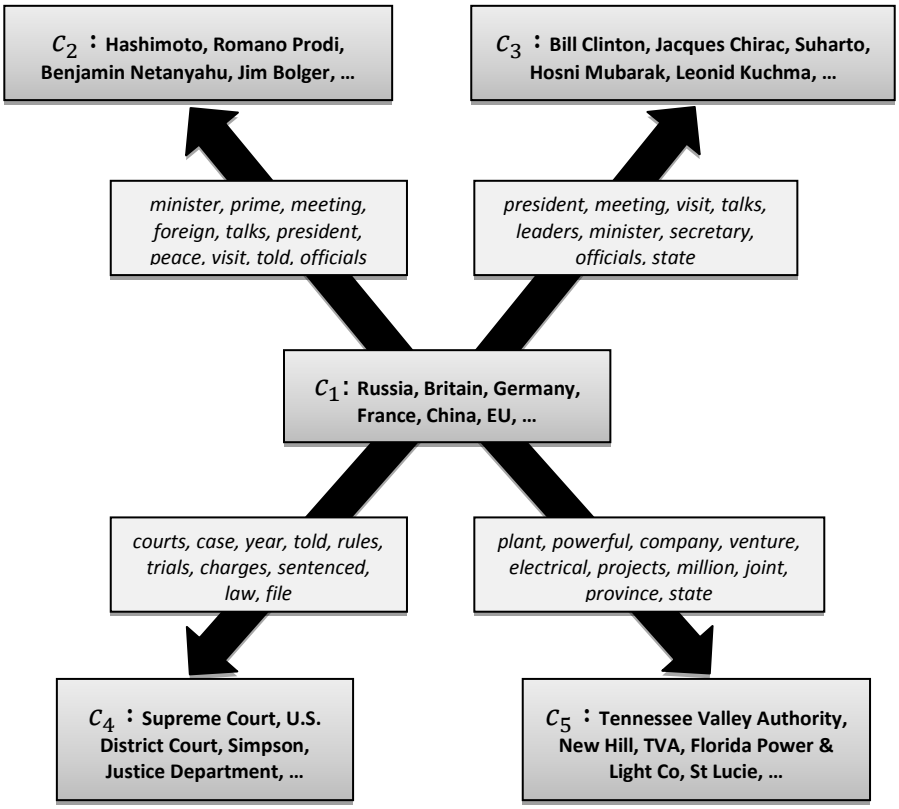


Figure 16: Example ontology with five concepts and top keywords extracted from co-occurrence profiles for four concept pairs.

4.4 Concept and Relation Naming

The names of concepts and relations are not an integral part of ontology learning framework, but constitute an important part when using the ontology. The name of a concept or relation would typically correspond to its meaning but by itself does not

necessarily define the semantics of a concept, but is only one possible materialized representation of it.

For example, in OpenCyc [58] the ontology engineers opted for randomly generated names of concepts. Each concept can be linked with a comment or a human readable label, but their semantics does not depend on it. Figure 17 shows an example of an OpenCyc URI for a concept with label “Game”. Nevertheless, it is a good practice in ontology engineering to use user-friendly names for concepts and relations, same as for naming variables in good software engineering practice.

```
http://sw.opencyc.org/2009/04/07/concept/Mx4rvkS9GZwpEbGdrcN5Y29ycA
```

Figure 17: Example of OpenCyc URI for concept *Game*.

In the ontology learning framework, concepts and relations are grounded using instance and co-occurrence profiles respectively. We can also use these profiles to derive summaries, which can then be used by the user to assign friendly names to the concepts. Profiles can also be used to retrieve suggestions for possible names from some predefined external vocabulary.

4.4.1 Extracting Descriptive Keywords

Keyword extraction tries to describe the concept c with a set of keywords. It requires vector space model to be used as a feature representation. The extraction starts by first computing the centroid vector of concept c, a mean of the concept’s instances profiles. The keywords are selected to be the words with the highest weights in the centroid vector. Same approach can be applied to relations, but using co-occurrence profiles instead of instance profiles. We call keywords extracted using this approach as *descriptive*.

For example, concepts from the running example would be described with the keywords presented in Table 10.

Table 10: Extracted descriptive keywords from the concepts in the running example.

Concept	Extracted Keywords
“Auto Manufactures”	vehicles, cars, automotive, trucks, dealers, financing, fleet, motor, financial services, leasing
“Technology”	online, software, devices, networking, mobile, applications, system, segment, content, develops
“Financial”	banking, investment, loans, institutions, secure, segment, equities, brokerage, managers, advisory

In some cases, the features in the feature representation are not as intuitive as in text

representation using the vector space model. For example, with visual words, displaying the feature does not work from the user's perspective. When the instances themselves are meaningful and easy to understand, the top features (as is the case with keyword extraction) can be replaced with a list of "most representative" examples. In the case of concepts, this can be the instances with instance profiles closest to the centroid vector. In the case of relations, this can be instance pairs with co-occurrence profile closest to the centroid vector of relation.

4.4.2 Extracting Distinctive Keywords

Another approach for keyword extraction works by combining the concept c and its neighbourhood $N(c)$, to derive keywords which are specific or *distinctive* for concept c. The method is based on the feature selection approach presented in [59] which uses the SVM classifier [44] which aims at selecting features which are most important for positive or negative classification.

First, we define the binary classification problem by selecting positive instances as $I^+ = \iota_C(c)$ and negative instances as $I^- = N(c) \setminus \iota_C(c)$. Second, we train the binary SVM classifier on these instances and use the obtained model to select features (words in the case of vector space model) which on average contribute most to the correct positive classification of instances. Details on the SVM feature selection algorithm are presented in Section 4.7.

Table 11 shows the results of applying the presented algorithm to the running example. It can be seen, that there are several minor differences in the selection and ranking of keywords when compared to results in Table 10. The difference between distinctive and descriptive keywords in this example is marginal due to low overlap of keywords between concepts.

Table 11: Extracted distinctive keywords from example concepts.

Concept	Extracted Keywords
"Auto Manufactures"	vehicles, cars, trucks, automotive, financing, dealers, fleet, motor, Volkswagen, financial services
"Technology"	Google, stores, networking, software, devices, mobile, windows, applications, system, Microsoft
"Financial"	banking, investment, loans, institutions, client, equities, secure, brokerage, advisory, cards

To show the potential of the presented approach, we demonstrate it on an extension of the running example – a larger set of instances used in the use-case presented in Section 6.1. Figure 18 shows a section of the constructed ontology centred around financial corporations. The concept of all insurance corporations is further broken down into a

concept of life insurance corporations and others, mainly property and casualty insurance. Table 12 and Table 13 show extracted descriptive and distinctive keywords respectively. There is a significant overlap between the two descriptive keywords lists, making it harder to see the distinguishing features for each of the concepts. On the other hand, there is no overlap between the two lists of distinctive keywords. For example, that word "insurance" appears among descriptive keywords for both concepts, whereas it appears only in one list among distinctive keywords.

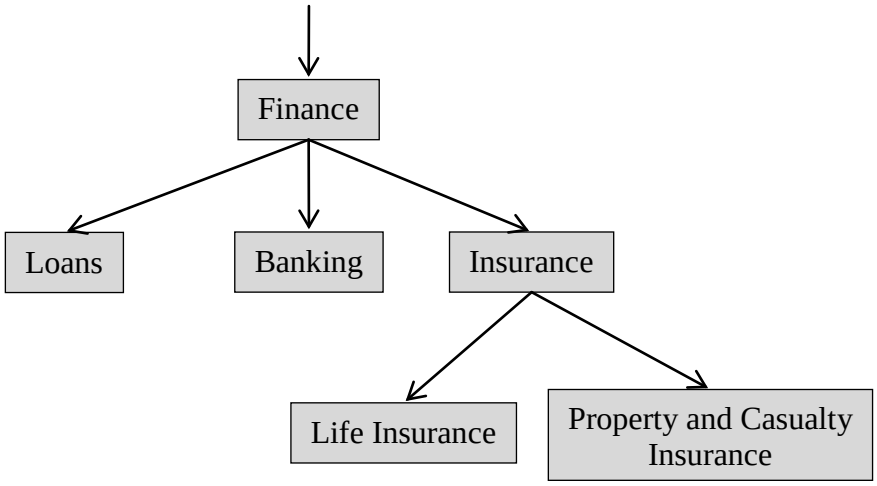


Figure 18: Partial ontology of financial corporations.

Table 12: Descriptive keywords for two insurance concepts.

Concept	Extracted Keywords
"Life Insurance"	life, insurance, life insurance, financial, annuities, insurance company, financial services, investment, individuals, life insurance company
"Property and Casualty Insurance"	insurance, insurance company, casualty, property casualty, reinsurance, casualty insurance, underwriting, property casualty insurance, property, risks

Table 13: Distinctive keywords for two insurance concepts.

Concept	Extracted Keywords
"Life Insurance"	life, life insurance, financial, individuals, annuities, assets management, assets, financial services, investment, management
"Property and Casualty Insurance"	insurance, group, holding, property casualty, casualty, commercialization, casualty insurance, underwriting, property casualty insurance, property

4.4.3 Reusing Existing Vocabulary

Concept or relation names can also be derived from existing vocabularies, using the approach for concept construction proposed in Section 4.2.3. In both cases we start by mapping either instance profiles or their centroid vector to the concepts from some external vocabulary. The difference comes in how these external concepts are used. In the case of naming, a list of suggestions for the concept name can be assembled from the external concepts' labels.

There is a big speed difference between classifying all instance profiles or just the centroid, especially for concepts with many instances. Former requires the classifier to be applied to each instance, while the later requires application of the classifier only to the centroid vector. Consequently, for scalability reasons only the classification of the centroid vector would be used. We can demonstrate this on the running example, where there are three natural concepts. Table 14 shows the top DMoz concepts for each of the three main concepts from the running example. For the technology instances, the top concept "Computers" is correct. The most appropriate concept for the finance instances is the second concept on their list ("Financial Services") and for the auto manufactures, it is the third concept on their list ("Automotive").

Table 14: List of top DMoz concepts for concepts from the running example.

Instances	Extracted Keywords
$\{APPL, MSFT, GOOG, NOK\}$	Computers, Searching, Search Engines, Internet, Google
$\{F, VOW, GM\}$	Regional, Business and Economy, Automotive, United Kingdom, Motoring, Europe, Oceania
$\{GS, BCS, C\}$	Business, Financial Services, Regional, Banks and Institutions, Banking Services, Investing, Europe

4.5 Visualization

Domain understanding and discovery is an important part of semi-automatic ontology construction. This is not directly covered in the ontology learning framework, which focuses on the formalization of ontology and the definition of operators for modifying the ontology. However, instance and co-occurrence profiles can be used to visualize concepts and relations and help the user with their understanding.

In this section, we describe an approach that we have developed for visualization of text corpora called Document Atlas [60]. The approach uses the analogy of geographical map (or atlas) to visualize topics covered in a large corpus, by mapping all the documents to a two dimensional plane and presenting them to the user in a form of a topic map. In the course of this section we also extend the original Document Atlas approach to cover several aspects of ontology visualization and apply it to learning concept construction

operators. The emphasis is on the visualization of ontology which uses word vectors as feature representations, but some indication will be given, on how it can be used to extend to other modalities/domains than text (i.e. images).

4.5.1 Document Atlas

Document Atlas consists of two major steps. The first step is the mapping of all the documents to two dimensions. Ideally they would be positioned in such a way that the distance between two documents would correspond to the content similarity between them. The second step is enriching and presenting derived two-dimensional representation of documents.

In the following subsections we give a sequence of methods (latent semantic indexing and multidimensional scaling) that transform a text document into a two dimensional point and describe how the two dimensional space can be augmented with additional information, that can be extracted in the case of vector space model. The section concludes with the visual representation of the resulting visualization map.

4.5.1.1 Latent Semantic Indexing

A well-known and used approach for extracting latent semantics (or topics) from text documents is LSI [35]. In this approach we first construct a term-document matrix A from a corpus of text documents. This is a matrix with document vectors as columns. The term-document matrix A is decomposed using singular value decomposition (SVD), so that $A = USV^T$; here matrices U and V are orthogonal and S is a diagonal matrix with ordered singular values on the diagonal. Columns of matrix U form an orthogonal basis of a subspace in the bag-of-words space where vectors with higher singular values carry more information. This follows from the basic theorem about SVD that by setting all but the largest k singular values to 0 we get the best approximation for matrix A with matrix of rank k.

Vectors that form the basis can be also viewed as concepts and the space spanned by these vectors is called the *semantic space*. Each concept is a vector in the bag-of-words space, so the elements of this vector are weights assigned to the words coming from the documents. The words with the highest positive or negative values form a set of words that are found most suitable to describe the corresponding concept.

4.5.1.2 Multidimensional Scaling

Multidimensional scaling (MDS) [61] enables dimensionality reduction by mapping original multidimensional vectors onto two dimensions. Here the points representing documents are positioned into two dimensions so they minimize some energy function. The most common form of this function is

$$E = \sum_{i \neq j} \left(\delta_{ij} - d(x_i, x_j)\right)^2,$$

where x_i are two dimensional points, $d(x_i, x_j)$ denotes Euclidian distance between the two points (we will refere to this with d_{ij} in short) and δ_{ij} represents the similarity between two vectors (in our case cosine similarity between documents i and j). An intuitive description of this optimization problem is: more the distances between points on the plane approximate real similarity between documents, lower the value of the energy function E. Note that the function E is nonnegative and equals zero only when distances between points match exactly with similarity between documents.

A common way of applying multidimensional scaling is by using gradient descent for the optimization step. The problem with this approach is that the energy function is not convex: it usually has many local minima which are not that interesting for us. One could repeat this method several times with different initial states and then choose the results with the lowest energy.

We choose a slightly different approach based on the reformulation of the energy function. Given a placement of points, we calculate for each point how to move it to minimize the energy function. We denote the current positions of points with (x_i, y_j) and the desired position with $(x_i', y_i') = (x_i + \delta x_i, y_i + \delta y_i)$. Then we have

$$d'^2_{ij} - d^2_{ij} =$$
$$= \left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 - \left(x_i + \delta x_i - x_j - \delta x_j\right)^2 + \left(y_i + \delta y_i - y_j - \delta y_j\right)^2 \approx$$
$$\approx \left(x_i - x_j\right)\delta x_i + \left(x_j - x_i\right)\delta x_j + \left(y_i - y_j\right)\delta y_i + \left(y_j - y_i\right)\delta y_j =$$
$$= \left[\left(x_i - x_j\right), \left(x_j - x_i\right), \left(y_i - y_j\right), \left(y_j - y_i\right)\right]\left[\delta x_i, \delta x_j, \delta y_i, \delta y_j\right]^T.$$

By writing this for each pair (i, j) and substituting d'_{ij} with the original distance δ_{ij} between i-th and j-th document we get a system of linear equations which has a vector of moves (δx and δy) for a solution. The derived iteration finds a step towards minimizing the energy function. The iteration requires solving a linear system of equations with a very sparse matrix. This can be done very efficiently using the Conjugate Gradient (CG) method [62]. Finally, the points are normalized to lie in the square $[0,1]^2$.

4.5.1.3 Visualization Using Dimensionality Reduction

The proposed visualization approach is based on a combination of a linear subspace method and multidimensional scaling for reducing document space dimensionality. Both methods can be independently applied to any data set that is represented as a set of vectors in some higher dimensional space. Our goal is to reduce the number of dimensions to two so that the whole corpus of documents can be shown on a computer screen.

Linear subspace methods like LSI focus on finding projections from the original bag-of-words space into a lower dimensional space while trying to preserve as much information as possible. By projecting data (text documents) only on the first two directions we can get the points that live in the two dimensional space. The problem with linear subspace methods is that only the information from the first two directions is preserved. In case of LSI it would mean that all documents are described using only the two main concepts.

In [60] we have proposed combining the two methods (LSI and MDS) in order to take advantage of the both. The algorithm is presented in Figure 19.

Input: Corpus of documents to visualize in form of TFIDF vectors.
Output: Set of two dimensional points representing documents.
Procedure:
1. Calculate k dimensional semantic space using LSI on the input corpus of documents
2. Project documents into the semantic space,
3. Apply MDS on the projected documents using Euclidian distance as the similarity measure.

Figure 19: Algorithm for mapping instance profiles into two-dimensional space by combining LSI and MDS.

There is a parameter k in the first step which has to be given to this procedure. In order to select this parameter we use the following heuristic. Let $\Sigma_k = S_1^2 + S_2^2 + \cdots + S_k^2$, where S_i is i-th singular value. We know that $\Sigma_n = trace(A^T A)$, where n is the number of the documents in the corpus and A is the term-document matrix. From this we can empirically select k by prescribing the ratio Σ_k/Σ_n to some fixed value (e.g. 75% which was found to produce the most useful and visually pleasing results [60]).

4.5.1.4 Semantic Landscape

The procedure defined in the previous section generates a map of topics from a given collection of documents. The map is calculated for each document collection separately with no connection to the other document collections. But in some cases one would like to visualize a given set of documents on a predefined set of topics, e.g. for comparing two document collections on a predefined set of topics. To handle such scenarios we developed a notation of semantic landscape which defines the set of topics over which the documents will be visualized. It works by using a set of documents, which describe the desired topics (e.g. news events), and projecting them onto a two dimensional map using the procedure defined in previous sections. The documents which define the semantic landscape are called *landmarks*.

The landmarks do not appear in the visualization and are used only for positioning

other sets of documents on the map using the following procedure. For each document, a set of the top N most similar landmarks is calculated. The position of the document is defined as a convex combination of landmarks' positions. More specifically, let μ_i be the cosine similarity of the document to the i-th most similar landmark and A_i be the location of the landmark on the 2D map. Then the location of the document is calculated as

$$\left(\frac{1}{\sum_{i=1}^{N}\mu_i}\right)\sum_{i=1}^{N}\mu_i A_i\,.$$

The time complexity of finding top N most similar landmarks is $O(L\,log(N))$, where L is the number of landmarks [75]. There is also an additional cost of $O(N)$ for averaging the positions of top landmarks. The inexpensive positioning in turn means that once the semantic landscape is calculated the visualization of new documents can be done in near real time.

The semantic landscape can be also used for more efficient visualization of larger corpora, when the MDS algorithm would take too long to finish. In such cases a semantic landscape is generated from landmarks, which are extracted from the corpora. In our experiments we used a *k*-means clustering algorithm for clustering the corpora in a larger number of clusters, usually several hundreds. Since *k*-means is more efficient than MDS, this significantly increases the time performance.

4.5.1.5 Visualization Enhancements

Several additions can be added to the visualization to increase its effect. First of all, the background can be used to depict, for example, the density of landscape documents in a particular part of the map. Second, the content of the landscape documents can be used for generating and displaying the keywords which are relevant for a particular part of the map.

The background depicting the density can be either generated based on the documents which were provided for the background landscape. The density is estimated as follows. Each point from the landscape is assigned a height using the formula

$$h(x,y)=\sum_{i} exp(-\sigma\|(x,y)-\,(x_i,y_i)\|^2)\,,$$

where σ defines how wide is the influence of one point.

Similar to density, a set of most important keywords can be assigned to each point of the map. The keywords are selected by averaging the bag-of-words vectors of documents which appear within a predefined distance to the point.

Currently, the keywords are used in two distinct scenarios. First, when the user moves the mouse to a specific point, the list of most important keywords, calculated in real-time, is displayed (Figure 20). Second, for each point from a set of random points, uniformly distributed over the map, the most important keyword is computed and displayed on the map, using white font. These keywords help the user to see the main topics by just

glancing over the map. We call these keywords common words.

There are several parameters which control the calculation and appearance of keywords. First is the radius which defines the documents that are to be used when averaging the bag-of-words vectors. In the case of the dynamically calculated list of keywords, the radius is selected using the mouse scroll wheel. The dark circle shows the area from which the keywords are extracted. Figure 21 shows the effect of this parameter. Second, the density of common words is controlled by sparseness parameter, which defines the minimal distance between any two common words. And finally, the minimal distance between repeating common words can also be specified. Figure 22 shows the effect of the parameter. These three parameters can are predefined when the landscape visualization is created.

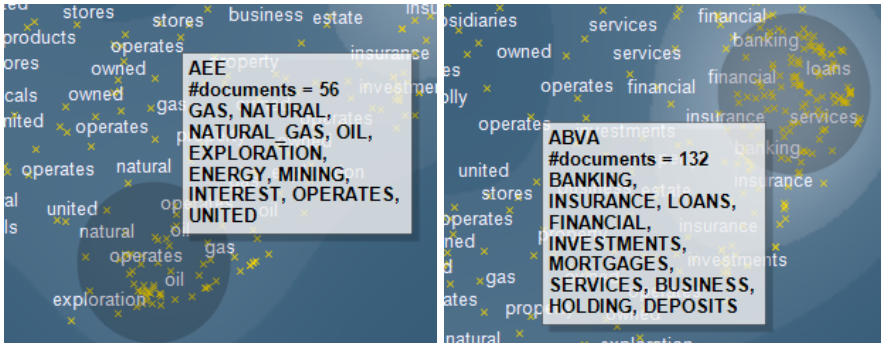


Figure 20: List showing the main keywords for the area marked with the dark circle.

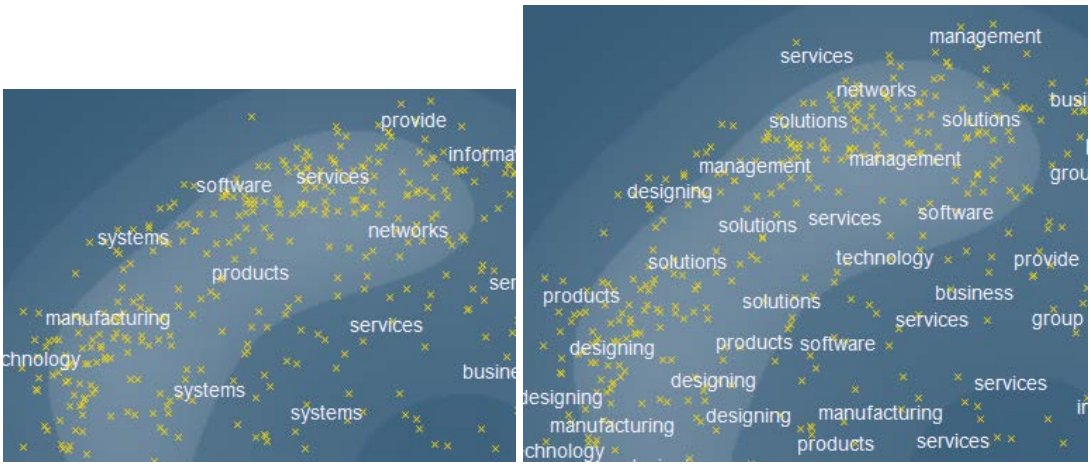


Figure 21: The effect of the sparseness parameter. Common words on the right are more densely spread around the map than the ones on the left.

4.5.2 Concept Analysis

The first task, to which we can apply Document Atlas, is the support of concept analysis and learning of new concept construction operators. We will demonstrate this on the extended version of the running example, which was already used in Section 4.4.2.

The main input to Document Atlas is a corpus of documents, which are represented using the vector space model. This does not restrict the applicability of the presented approach only to the cases where instance profiles are word vectors, but additional

modifications of Document Atlas are required to handle such cases [63].

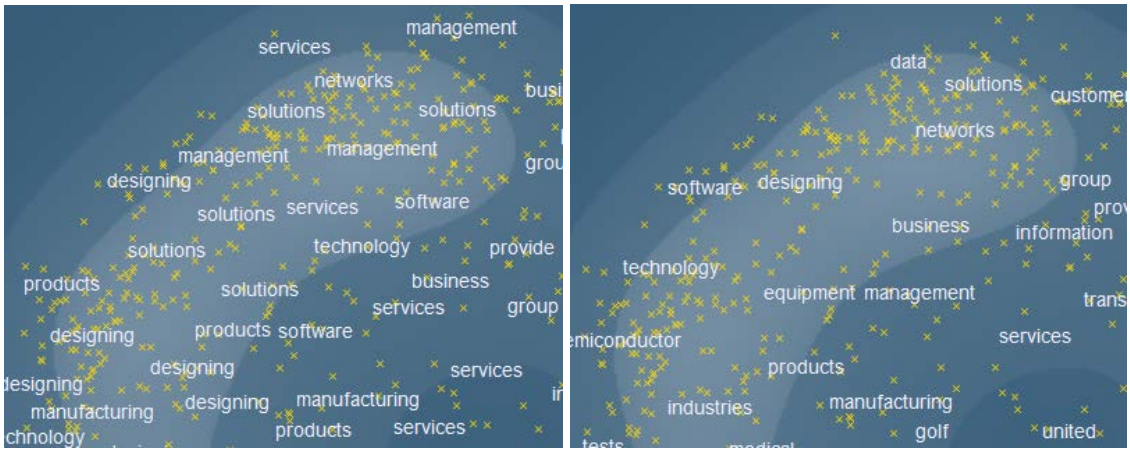


Figure 22: The effect of the minimal distance between repeating common words. Note that increasing the distance decreases the number of common words, since it decreases their selection.

The instances in the extended running example are represented using the vector space model, and are as such a good fit for Document Atlas. Figure 25 shows the visualization of the "Insurance" concept from the ontology presented in Figure 18. Here, each instance is one insurance company, represented by a word vector derived from its description (same as in the running example). The visualization shows how the concept is structured in three major concepts. Concept (a) consists of companies dealing with property, casualty, reinsurance, etc. Opposite is the concept (b) consisting of companies dealing with life and health insurance. Note that there is slightly smooth transition between the two clusters, with companies working in several areas being in the middle. There is a small separate cluster (c), which consists mainly of larger financial institutions, which also deal with insurance or have large investments in insurance companies.

The visualizations, such as the one in Figure 23, can give some understanding of the content of the visualized concept. The user can use this information as an additional guide when using semi-automatic methods for learning concept construction operators or when deciding on how to name the concept.

The visualization can also be used as an input for learning concept construction operators. In this case, preparing the visualization is the automatic part of the procedure (g_A). The user can then use the mouse to mark the areas in the visualization, which should be made into sub-concepts. Figure 24 shows the same visualization of "Insurance" concept, where the user marked all the instances dealing with life insurance. This provides sufficient amount of information to generate the sub-concept "Life Insurance".

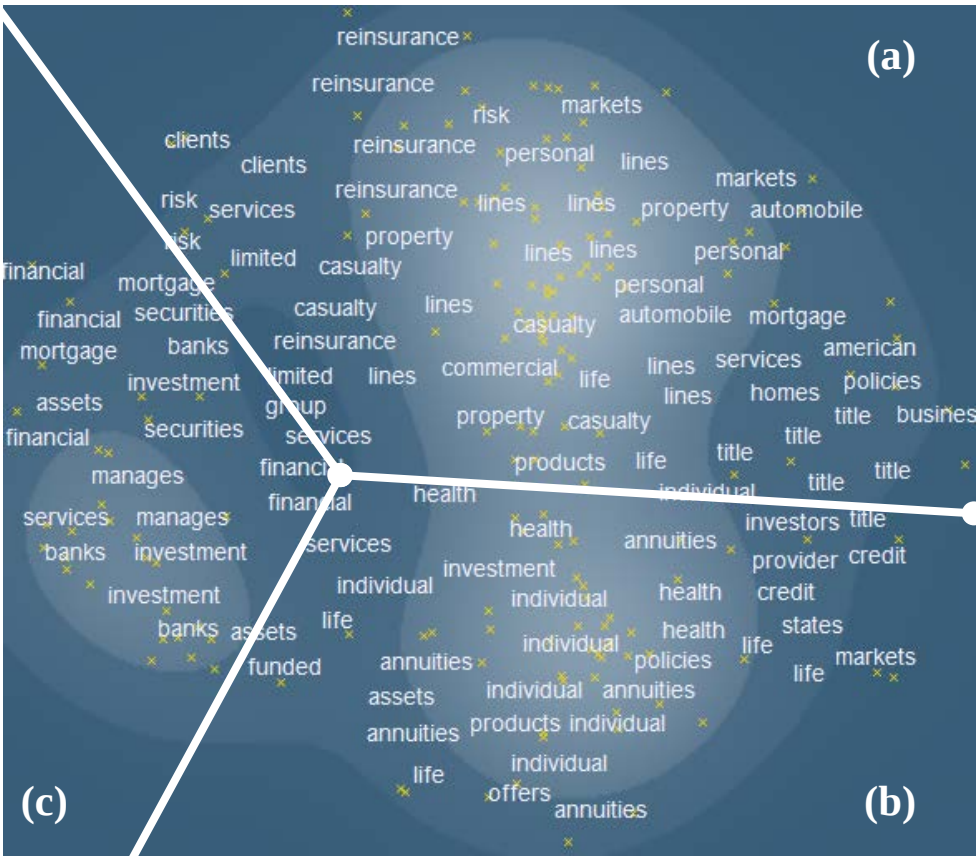


Figure 23: Visualization of instances from the “Insurance” concept, shown in Figure 18. The visualization is manually cut into three parts, to indicate three major clusters.

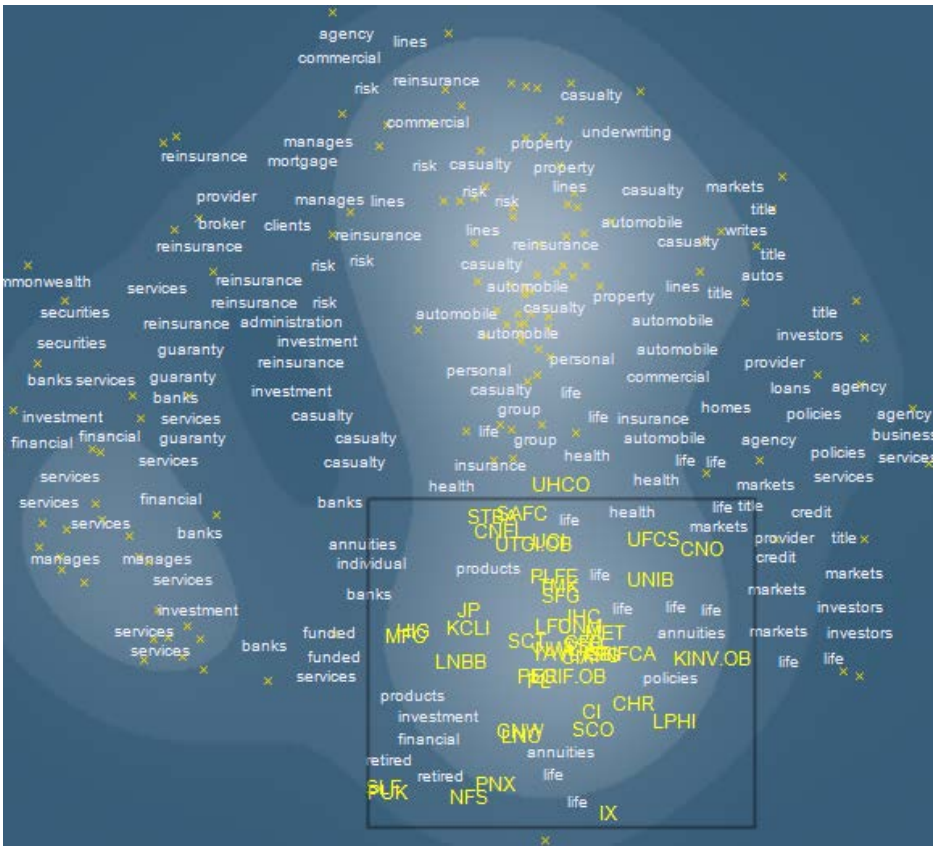


Figure 24: The user selected all the instances dealing with life insurance.

4.5.3 Visualizing Ontologies

Document Atlas was extended for visualization of ontologies. The procedure works by defining the semantic landscape from a set of documents describing a domain, in which we want to visualize the ontology. Each concept from the ontology is mapped to the semantic landscape using the procedure described in Section 4.5.1.4 on concept centroid vector.

We demonstrate this on a sample ontology built around an extended version of the running example. Figure 25 shows the example ontology constructed from instance profiles of the running example (descriptions of public companies). The resulting projection of the sample ontology to the semantic landscape is shown in Figure 26. The content of the ontology (concept names in Figure 26) is presented using yellow font (only for the major concepts) or yellow cross (for lower-level concepts). The links between concepts are shown using green lines. The semantic landscape is visualized with the density and the common keywords.

Figure 27 shows the same ontology projected on a different semantic landscape. The landscape is generated from a corpus of Reuters news articles [56]. Note that the part of ontology describing financial corporations gets spread over larger area, while the technological concepts get densely grouped in the area of technological news. This shows the bias of then selected news article corpus towards news related to the financial sector.

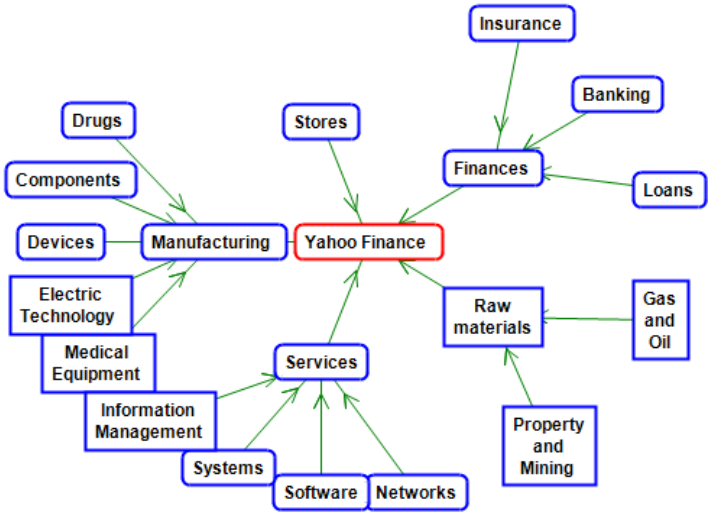


Figure 25: Example ontology built on the instances from the extended version of the running example.

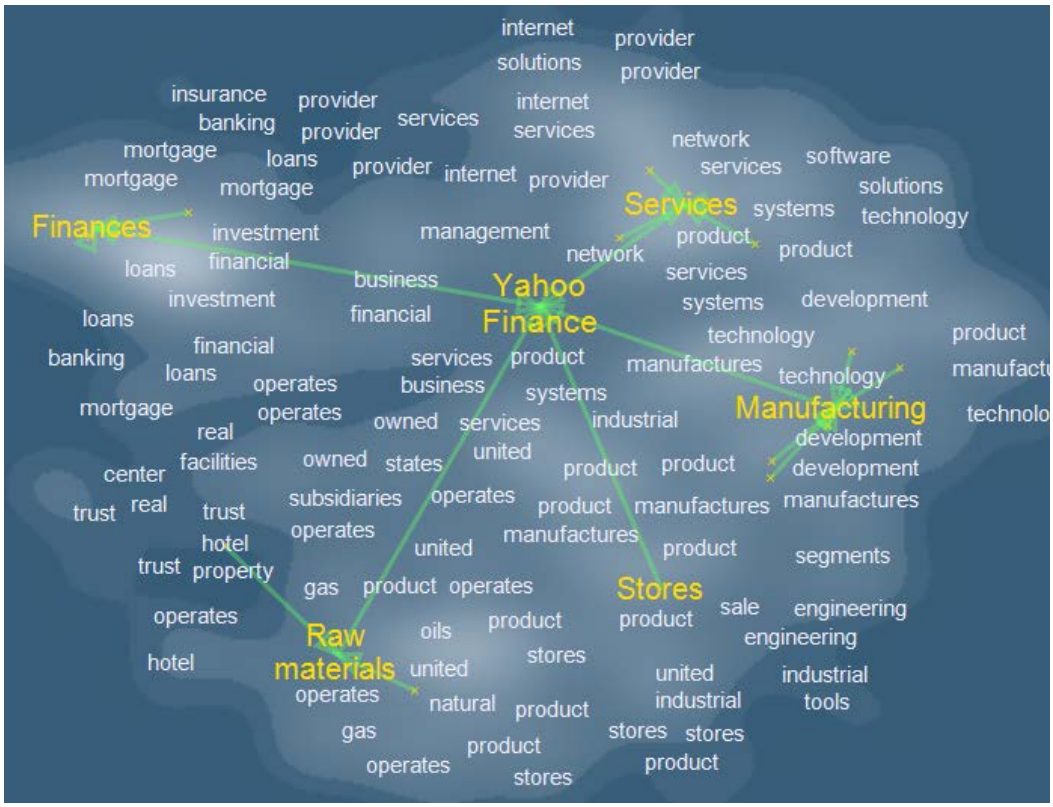


Figure 26: Example ontology projected on the semantic landscape built from company descriptions.

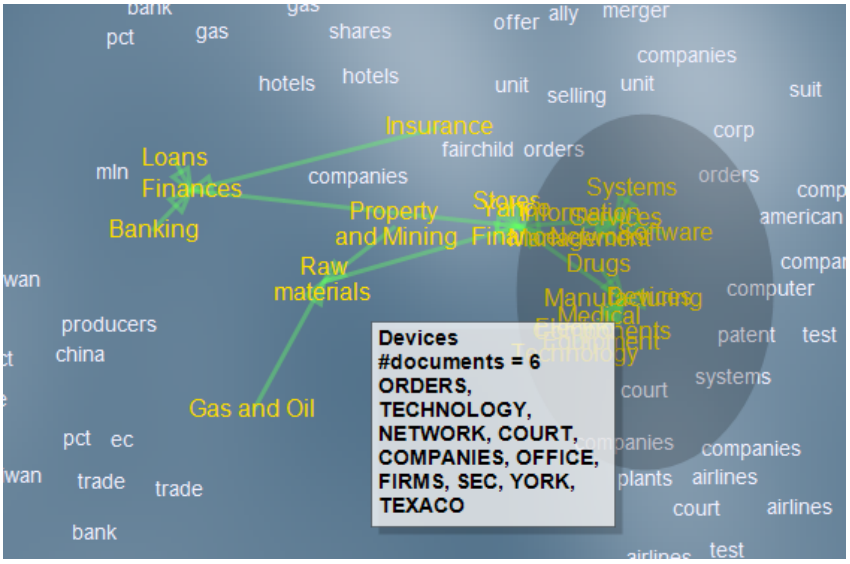


Figure 27: Example ontology projected on the semantic landscape built from a corpus of Reuters news articles.

4.6 Adding Instances to the Ontology

Adding instances to an existing ontology can be a time-consuming task when either ontology has many concepts or there is a large number of new instances. However, with instance profiles it is easy to redefine this task as a straight-forward multi-class

classification [64]. In this section we will demonstrate two approaches, the first one having a higher computation cost but better accuracy, while the second being able to scale to large ontologies.

Adding a new instance e to the ontology requires an update of the instance set I and instance assignment operator $\iota_C(c)$ for all concepts c for which instance e is a member. The first step (updating the instance set) is straight forward. For the second step, we can train a binary classifier f_c for each concept, so that $f_c(e) > 0$ when e is a member of c.

Training a multi-class classifier requires training data, which can be extracted by combining ontology definition and instance profiles. First, the set of all instances is transformed into a set of training examples $\{x_e | e \in I\}$, by taking feature vectors extracted from the instance profiles. Second, each training example is assigned a subset of labels $Y_e \subseteq \{y_c | c \in C\}$, such that for each $y_c \in Y_e \Leftrightarrow e \in \iota_C(c)$. Finally, the resulting training set is a set of pairs $\{(x_e, Y_e) | e \in I\}$ and the classification task is to find a classifier f such that $f(x_e) = Y_e$.

The result of the classifier, Y_e, can be translated to an ontology construction operator

$$f_I: (I, C, R, \leq_C, \leq_R, \iota_C, \sigma_R) \mapsto (I \cup \{e\}, C, R, \leq_C, \leq_R, \iota_C', \sigma_R)$$

where $e \in \iota_C'(c)$ for each c such that $y_c \in Y_e$. Hierarchy constraints require that when $e \in \iota_C'(c)$ and $c \leq_C \hat{c}$, then $e \in \iota_C'(\hat{c})$. This means that the predicted set of $Y_e = f(x_e)$ typically requires to be updated so that

$$Y_e' = \{y_c | \exists \hat{c} \leq_C c \; and \; \hat{c} \in Y_e\}.$$

We assume this correction step as implicit in the following algorithms.

The ontology learning system implemented as a part of this dissertation uses two classification algorithms for adding instances to the concepts. Both translate the multi-class problem to a set of binary classifiers $\{f_c | c \in C\}$, one for each label. We can re-write the classifier as

$$f(x_e) = \{y_c | f_c(x_e) > 0\}.$$

The first classification algorithm is Support Vector Machine [44], which is known for good classification performance, and the second one is centroid-based classifier [50], which can easily scale to large number of concepts due to fast training time, as we have shown in Section 4.2.3.

We demonstrate the addition of instances on the running example. Table 15 shows the ontology built on the running example. The ontology excludes the NOK instance, which is used to demonstrate the addition using the above two algorithms. As the first step, a binary classifier is trained for each concept. Table 16 gives results for SVM and centroid classifier when classifying NOK against each of the concepts. It can be seen, that the SVM correctly classified the instance into the “Technology” concept and gave negative score for other two concepts. The centroid classifier gives positive score for all three concepts by design. This requires, either to provide the classifier with some threshold,

estimated from the training data, or to use the classifier in a semi-automatic scenario, where the classifier is used to generate a list of possible concepts.

Table 15: Ontology built on the running example without the NOK instance.

	Instances
c_1	$\iota_C(c_1) = \{AAPL, MSFT, GOOG\}$
c_2	$\iota_C(c_2) = \{VOW, F, GM\}$
c_3	$\iota_C(c_3) = \{BCS, GS, C\}$

Table 16: Classification scores for the instance NOK for each binary classifier.

	SVM	Centroid
c_1	0.093	0.287
c_2	-0.704	0.055
c_3	-0.526	0.115

4.7 Using Background Knowledge

One of the goals behind the proposed semi-automatic approach is that the combination of domain expert and machine learning can make the task of ontology construction more efficient. One of the requisites for this is the availability of some data, in the form of instance and co-occurrence profiles, which describes the domain. However, the results of unsupervised approaches, such as the case of clustering presented in Section 4.2.2, depend on the feature representation.

The standard approaches for deriving feature representation (e.g. vector space model with TFIDF weighting schema when dealing with textual instance and co-occurrence profiles) are not always suitable for the given data or domain. Also, sometimes the same domain data might be used for deriving several different ontologies. For example, the same document corpus in a company may be viewed differently by marketing, management, and technical staff. This led to the development of approaches for adjusting the feature representations of profiles so that they can incorporate additional background knowledge into ontologies.

The proposed approach for encoding background knowledge requires from the user to label a subset of instances [79]. These labels do not need to describe the data in details, which will be the task of the constructed ontology, but should still be able to show which instances are similar and which are different from the user's perspective. The process of manually labelling the instances is time consuming but can be significantly speeded up by the use of active learning [52]. Labels can also be extracted from some secondary sources. For example, web pages can be labelled with tags from online tagging services, such as, Delicious [76] and the documents can be labelled by the folder in which they appear.

The approach is based on a feature selection method using SVM [59] and we assume vector space model feature representation. However, the approach can be easily generalized to other feature representations presented in Section 4.1. The method starts by

training a SVM classifier for each label using all the features. The classification of instance profile is done by multiplying the instance profile's word vector with the normal vector w computed by SVM,

$$x^T w = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n ,$$

and if the result is above some threshold b then the instance is considered positive. This process can also be seen as voting where each word is assigned a vote weight w_i. When instance x is being classified, each word from the instance votes with $x_i w_i$ and all the votes are summed together to obtain the final classification. A vote can be positive (the instance belongs to the label) or negative (does not belong to the label).

A way of selecting the most important words for the given label would be to select the words with the highest vote values w_i in the classifier for the label. It turns out that it is more stable to select the words with the highest vote $x_i w_i$, averaged over all the positive documents. The votes w_i could also be interpreted as word weights since they are higher for the words which better separate the documents according to the given labels.

The algorithm we developed for assigning weights using SVM feature selection approach works as follows:

1) Calculate a classifier for each label (one-vs-all method for multi-class classification [64]). TFIDF weighting schema can be used at this stage for computing the feature vectors. The result is a set of SVM normal vectors $W = \{w_j; j = 1, \ldots, m\}$, one for each label.
2) Calculate weighting for each of the labels from its classifier weight vector. Weights are calculated by averaging votes $x_i w_i$ across all the documents from the label. Only weights with positive average are kept and the negative ones are set to zero. This results in a separate set of word weights for each label. By μ_k^j we denote weight for the k-th word and j-th label.
3) Weighted feature vectors are calculated for each document. Let $L(e_i)$ be a set of labels of an instance e_i. Elements of vector x_i are calculated as

$$x_i^k = \left(\sum_{j \in L(e_i)} \mu_k^j \right) \cdot TF_k .$$

To demonstrate the effectiveness of the above approach, we used a collection of Reuters news [56]. Each news article from the dataset has two different sets of labels: (1) the topics covered and (2) the countries involved in it. We used a subset of 5000 randomly chosen documents for the experiments.

Figure 28 shows the top 3 concepts discovered with the *k*-means algorithm for weighting schemas trained on either of the label sets. The difference between the concepts is on the instances assignment level. For example, let us take two news articles talking about the stock prices, one at the New York stock exchange and the other at the UK stock

exchange. The New York news article was placed in (1) to the “Market” concept together with the UK document and in (2) to the “USA” concept while the UK document was placed in (2) to the “Europe” concept.

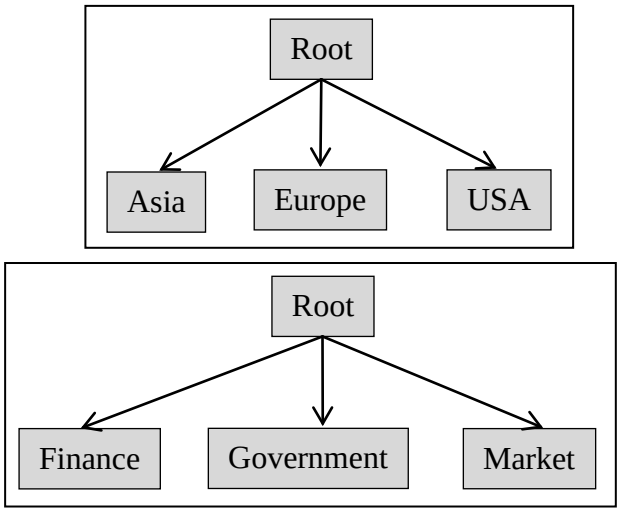


Figure 28: The top 3 discovered concepts for topic labels (left) and for country labels (right).

5 Ontology Learning System

5.1 Overview

This Chapter presents an implementation of the ontology learning framework Chapter 3. The implemented system is called OntoGen [65][66] and will be referred as such henceforward.

The OntoGen system is implemented in C++ and runs as a standalone application under the Windows operating system. The system is developed around Text Garden [67] library of text mining algorithms, which provides the complete pre-processing pipeline for dealing with text, and implementations of several machine learning algorithms. The development of the system required development of several additional components. The most important of the developed components are Ontology Model, implementing ontology learning framework, and Data Layer, providing scalable access to domain data.

The components of the system were reused in applications in areas other than ontology construction. Scenario detailed in [68] and [69] used a combination of Data Layer and Machine Learning Algorithms to model demographics of users from a large online news publisher. In this scenario, the system components were handling millions of instances such as users and web pages. Scenario detailed in [70] used the same two components to deliver real-time news story recommendations to users based on their history and demographic data. Here, the Data Layer showed capable of handling a million of daily events and reacting to them with updated recommendations in real-time.

5.2 Architecture

Figure 29 shows the architecture of the OntoGen system. The central part of the system consists of three components: Data Layer, Ontology Model and Machine Learning Algorithms. Each to these three components can be used independently outside the system, with the Machine Learning Algorithms actually being a collection of independent implementations.

At the bottom of the architecture diagram is a set of data sources (e.g. text document, social networks, images), where the domain data, used for learning ontology construction operators, is stored. Each data source requires an adapter, which maps the data source to a common interface: set of records, each record described by a set of fields. In database terminology, records would correspond to rows and fields to columns. Adapters for several standard data sources are provided: a directory of text documents or images, a

website, unique users accessing the website, etc.

The Data Layer provides unified access to all the data sources from the higher architectural layers, and includes integrated inverted index and feature extractors. Integrated inverted index is used to provide faceted search functionality over the records from the data sources. Integrated feature extractors provide functionality for extracting feature vectors from raw data provided by the data sources. An example of feature extractor would be the vector-space model (for text data) or visual words (for images). Feature extractors provide an abstraction layer required by machine learning algorithms.

The methods for learning the ontology construction operators require instance and co-occurrence profiles. The profiles can have a direct mapping to the records from data sources, or can be a result of queries executed by the Data Layer, which combine several records.

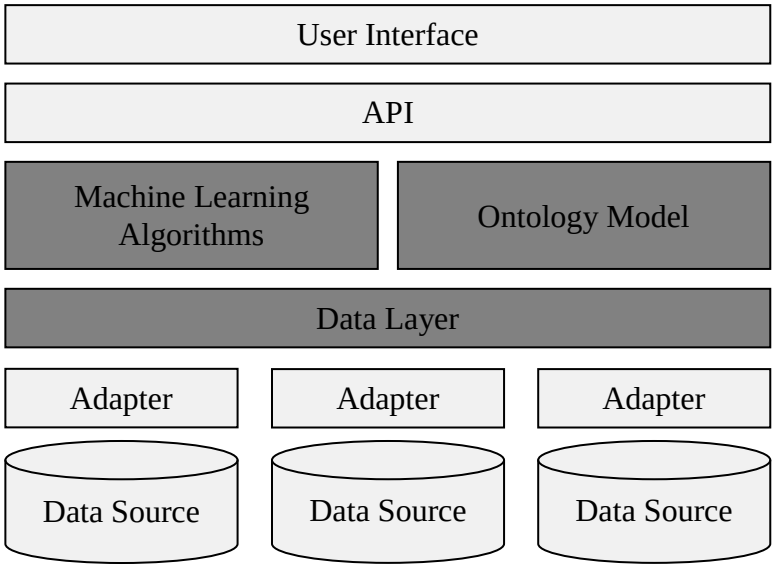


Figure 29: OntoGen system architecture diagram.

In the running example, the data source would be a set of documents, each document describing one public company. The adapter would map each document to one record, with its content corresponding to one field. The Data Layer would index each document by its keywords and generate word vector using the feature extractor for text. Figure 30 demonstrates the mapping.

On the other hand, the example from Section 4.3.3 would result in a slightly more complex data model. In the example, each entity and each sentence, in which at least one named entity occurs, is a separate record. The inverted index is used to index the occurrence of entities in sentences. Figure 31 demonstrates the mapping. In this scenario both instance and co-occurrence profiles are represented by searches inside the Data Layer. Instance profile is a result of search, where the query is the entity corresponding to

the instance, and co-occurrence profile is a result of search, where the query consists of a pair of entities. The result in both cases is a profile represented by a set of sentences, which can be transformed into word vector using feature extractor for text.

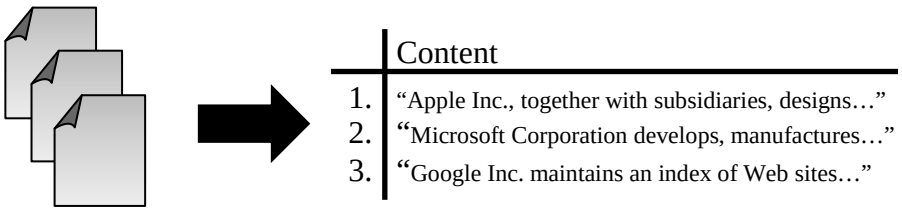


Figure 30: Running example, mapped to Data Layer abstractions. The left side corresponds to the data source (set of documents) and the right side corresponds to the data model within the Data Layer.

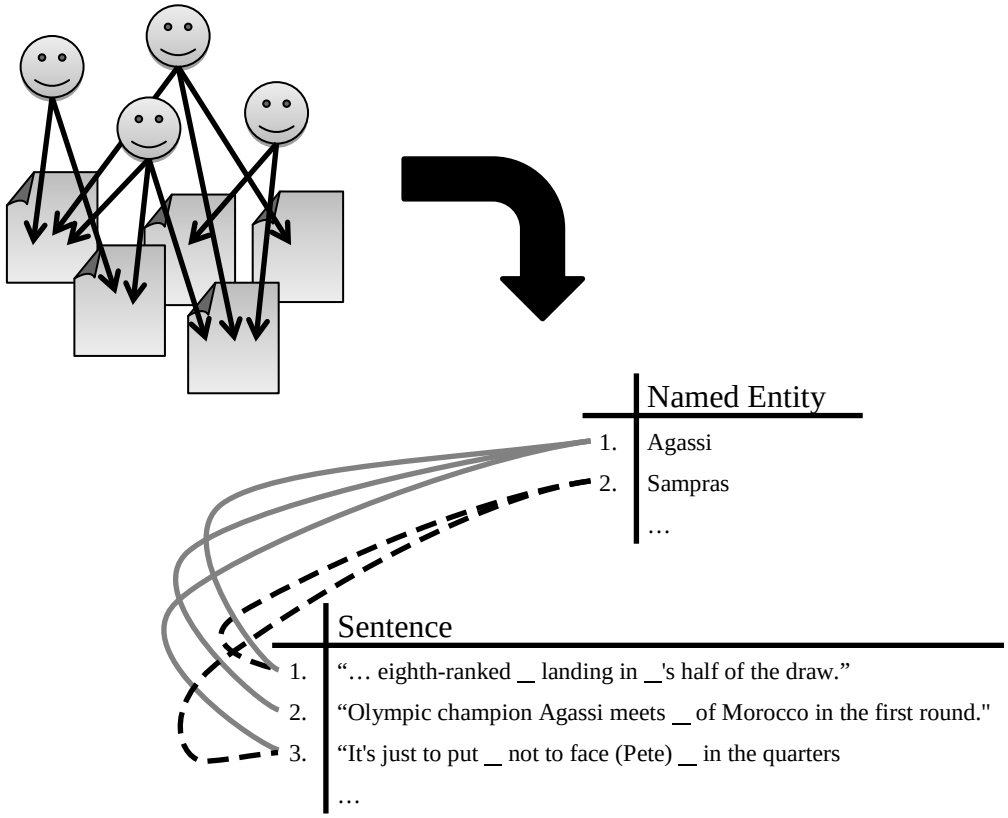


Figure 31: Named entity scenario mapped to Data Layer abstractions. Left side corresponds to the data source (set of entities and sentences, with occurrence links) and the right side corresponds to the data model within the Data Layer. The data model consists of two sets of records: named-entities and sentences. Records are connected through the inverted index.

The Ontology Model is a direct implementation of the ontology definition presented in Section 3.1. The model also stores additional information, which required by either the user interface or machine learning algorithms. The former refers to information, such as, friendly concept and relation names, required in the user interactions. The latter refers to information, such as links from instances (stored in the Ontology Model) to the instance profiles (accessible through the Data Layer), and links from instance pairs to co-occurrence profiles.

The Machine Learning Algorithms layer consists of algorithms, which were described in Chapter 4. The algorithms have access to the ontology structure within the Ontology Model and can retrieve feature vectors from the Data Layer. The algorithms can be either off-the-shelf implementations, or can be tightly integrated with the Data Layer. For example, the classification approach described at the end of Section 4.2.3.1 relies on the inverted index as the first layer in the cascade classification. Another example is a machine learning algorithm with inherent support for sampling, such as stochastic gradient descent [71]. In such case, the Data Layer does not need to pre-compute all feature vectors, but can instead compute them as they are requested.

The User Interface was implemented in C#, and accesses the lower layers through DLL API. The interface was developed to be used by domain experts, with little or no experience in ontology engineering or text mining. The evaluation of the interface is presented in Section 6.4. Figure 32 shows an example screenshot of the developed user interface. The main window (Figure 32) is divided into three main areas. The largest part of the window is dedicated to the ontology visualization and the document management part (the right side of the window). On the upper left side is the concept tree showing all the concepts from the ontology and on the bottom left side is the area where the user can check details and manage properties of the selected concept and get suggestions for its sub-concepts.

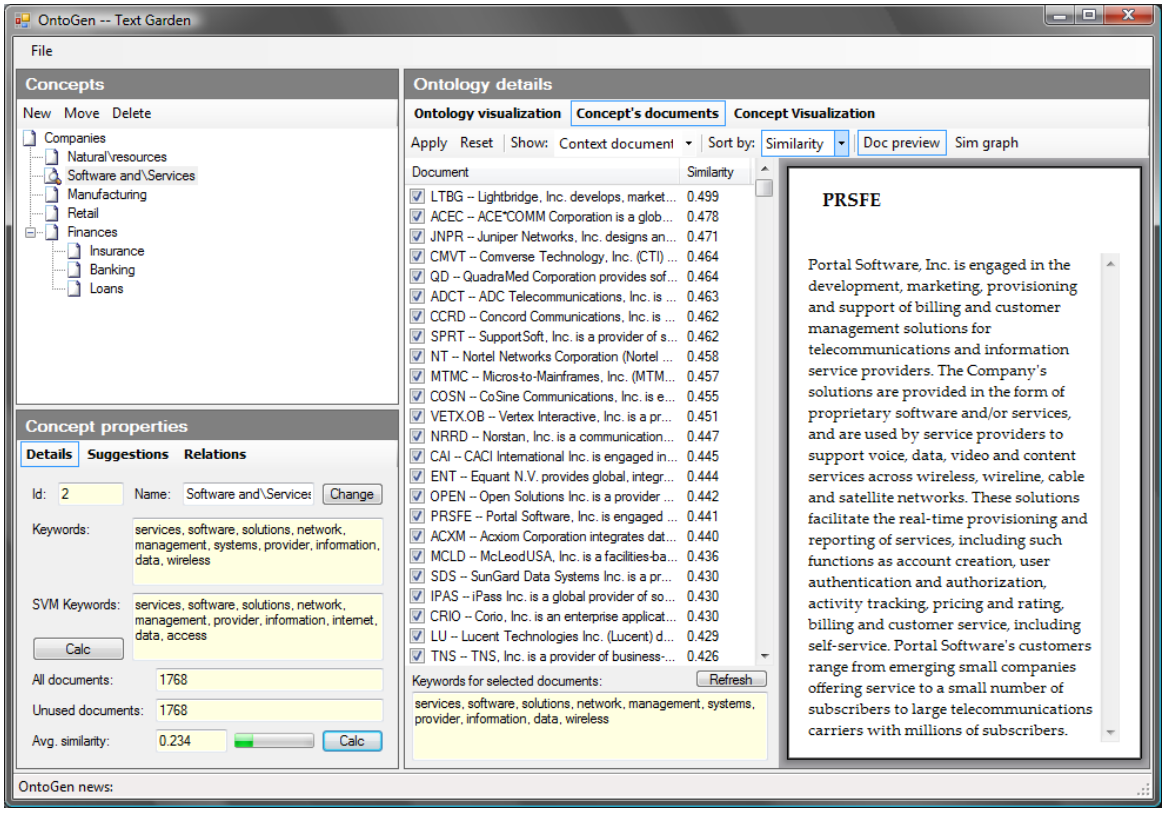


Figure 32: Example screenshot of the OntoGen user interface.

5.3 Scalability of Components

The Data Layer and Machine Learning Algorithms components were used in several scenarios, outside of semiautomatic ontology construction. This section will outline parts of these scenarios, to demonstrate the scalability of the developed components. In both cases, the experiments were done on a standard workstation.

In [68] and [69], the Data Layer and Machine Learning Algorithms components were used to store and operate with the access logs of a large online news publisher with 100 million daily page-views. The task addressed in the scenario was modelling of visitor groups, defined through an interactive user interface using ad-hoc queries. The task required storing millions of instances, such as web pages and visitors. The requirements for the Data Layer component in this scenario were ability to retrieve millions of instances using the built-in inverted index, and extraction of their feature vectors suited to the ad-hoc queries. The Machine Learning Algorithm component was required to react in the range of minutes to the queries, to support the interactive user interface, used to define the queries and analyse the results.

The scenario detailed in [70] used the Data Layer and Machine Learning Algorithms components to deliver real-time news story recommendations. This scenario required real-time processing of the access logs, processing a million of daily visits. Here, the Data Layer showed capable of handling a million of daily events and reacting to them with updated recommendations in real-time.

6 Evaluation

In this Chapter we demonstrate the use of the developed system in several use-cases. First, we investigate how OntoGen can be used to derive an ontology of financial domain, using the data from the extended version of our running example, and how the system can be applied to the social network data. The Chapter concludes with the analysis of ontologies developed by the users of the system and a more thorough evaluation of the developed system with respect to the qualitative analysis of user feedback gathered in a controlled environment.

6.1 Yahoo! Finance

In Section 3.2 we introduced a running example, which was used to showcase particular approaches for learning the ontology construction operators throughout the dissertation. In this section we use the extended version of this example to demonstrate the developed system OntoGen as a whole.

The extended version of the example contains 7177 instances, with each instance being one public company. Instance profiles are short textual descriptions of the companies represented using the vector space model. Examples of company descriptions can be seen in Appendix A. To demonstrate the functionality of the system, we first use a random subset of 6177 instances, to construct a finance domain ontology. The remaining 1000 instances are used to demonstrate the addition of new instances to the developed ontology.

We begin by first loading the instances into OntoGen. At this stage the system asks for parameters related to the vector space model, such as stemmer, *n*-gram length and frequency, and stop word list. Additional stop words can be provided if needed. Figure 33 shows the interface developed for specifying these parameters.

After the instances are loaded into OntoGen, we can start constructing new concepts. The first option is to create new concepts using the clustering approach. First, the user selects the concept, which he/she wants to further refine and selects the parameters required by the clustering algorithm. In the case of *k*-means, the parameter required is the number of clusters. Another parameter, which does not depend on the clustering algorithm, is which instances should be used. One option is to use all the instances from the concept, $\iota_C(c)$, or just the instances which are not yet assigned to any sub-concept, $\{e \in \iota_C(c) | \nexists \hat{c}: e \in \iota_C(\hat{c}) \; and \; \hat{c} \leq_C c\}$. After the parameters are selected, the clustering algorithm returns a set of concept suggestions. Each suggestion is described with three

most characteristic keywords (extracted using the approach presented in Section 4.4.1). Additional keywords can be seen by hovering over the suggestion with the mouse. The user can create a concept construction operator by selecting the suggestions he/she is satisfied with, and apply it to the ontology by clicking the "Add" button. Figure 34 demonstrates this procedure with a sequence of screenshots of the process.

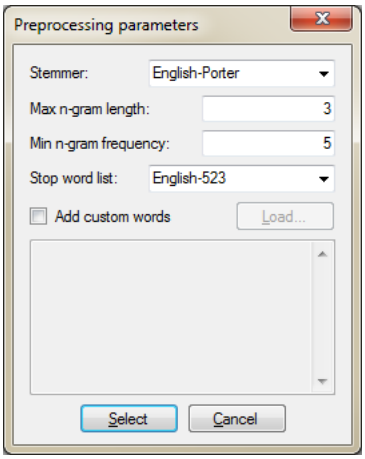


Figure 33: The interface for specifying the vector space model parameters.

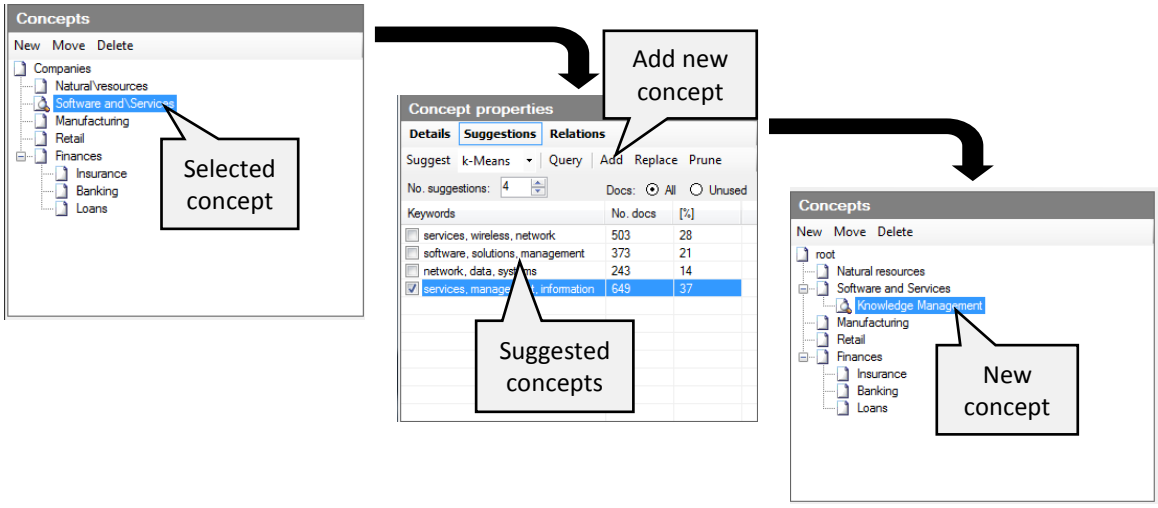


Figure 34: The interface for learning the concept construction operators using clustering.

The second option for concept construction is to use active learning. The user again starts by selecting the concept he/she wants to refine. The active learning loop is started by specifying the query, which describes the desired concept. For example, to add the sub-concept of "companies producing equipment for wireless networks" to the concept of "Manufacturing", an appropriate query would be "wireless networks". Same as with the clustering, the user can specify whether to use all the instances or just the ones, not yet assigned to any sub-concept. After the query is initiated, the system starts by asking the user to identify for a small subgroup of instances whether they should belong to the sub-concept. At each step, the system presents the number of instances, identified so far by

the system as belonging to the sub-concept, and the top ten keywords extracted from these instances. Each step results in a new concept construction task. Once the user is satisfied with these parameters (typically after few minutes of interaction), he/she can apply the developed concept construction task to the ontology by clicking “Finish”. Figure 35 demonstrates this procedure with a sequence of screenshots of the process.

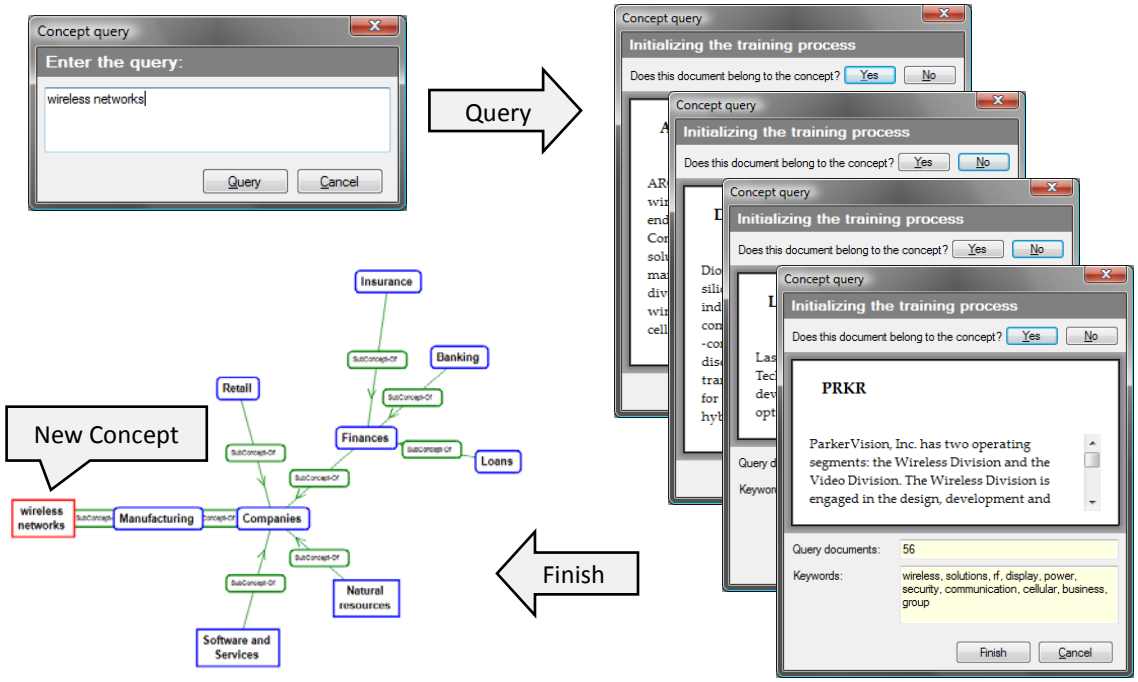


Figure 35: The interface for learning the concept construction operators with active learning.

The third option for concept construction is to reuse an existing external vocabulary. The procedure for this is very similar to using clustering. The user selects the concept he/she wants to further refine, and selects the desired external vocabulary. The system presents the user with a series of concept suggestions, based on the external vocabulary, and the user defines a concept construction task by selecting some of suggestions. The external vocabulary can be used for naming concepts constructed with other methods. In this case, the system provides a list of concept names (using methods described in Section 4.4.3), from which the user can select a name. Figure 36 demonstrates both procedures with a screenshot from the system.

The fourth option for concept construction is through the visualization. The user starts by selecting the concept he/she wants to further refine, and visualizes it using the approach presented in Section 4.5.2. One of the possible actions on the visualization is to mark a part of the area (as demonstrated in Figure 37) and use the selected instances to define a new sub-concept. The selected instances appear as a suggestion in the bottom right area of the interface, similar to the clustering approach.

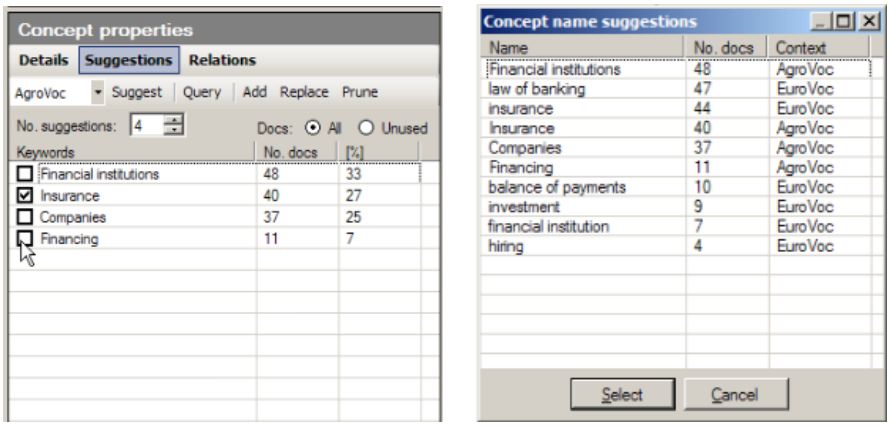


Figure 36: The interface for learning the concept construction operators by reusing external vocabulary (left) and reusing external vocabulary for concept naming (right).

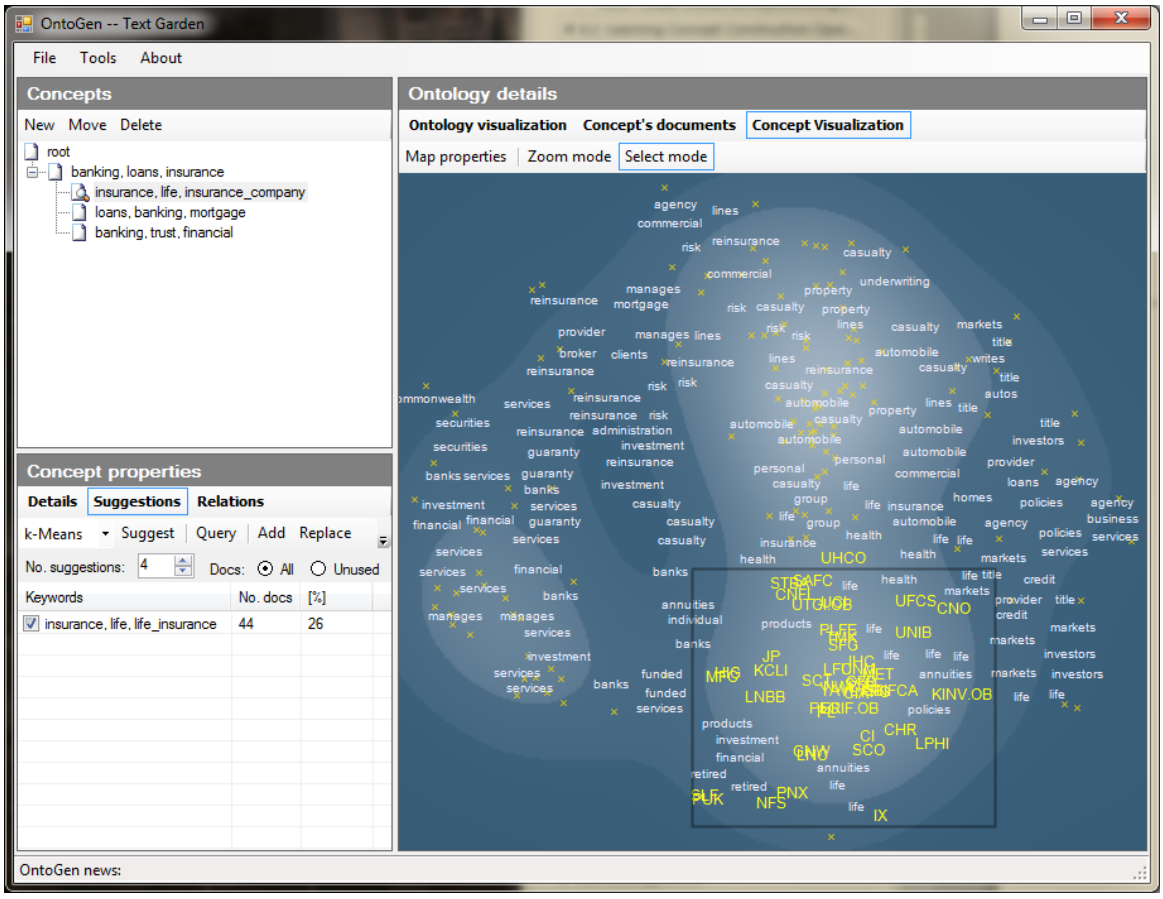


Figure 37: The interface for learning the concept construction operators through concept visualization.

In case new instances become available, they can be added to the ontology using the approach presented in Section 4.6. Figure 38 presents the interface implementing the approach. The top part of the window contains a list of new instances. Each instance can be selected, and the bottom part of the window shows its instance profile (company description in this case), and the concept hierarchy. The classification score for each concept is shown and the concepts, for which the score is above some threshold, are automatically selected.

The final ontology is shown in Figure 39. The whole process took around 30 minutes of effort by an ontology engineer without any specific domain knowledge. The manual organization and structuring of more than 7000 instances would require greater effort, first to define the concept hierarchy which suites the domain and the given set of instances, and second to manually map each instance to one or more concepts.

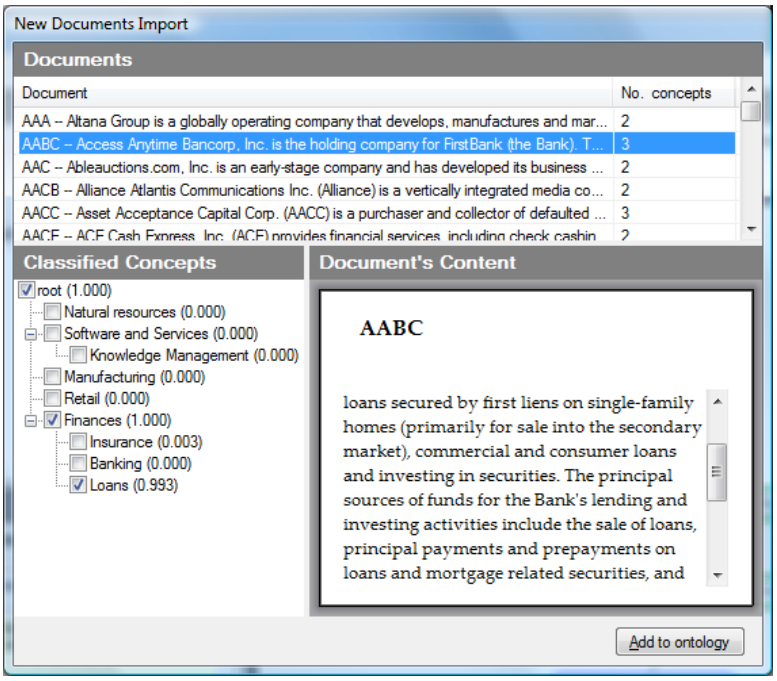


Figure 38: The interface for adding additional instances to the constructed ontology.

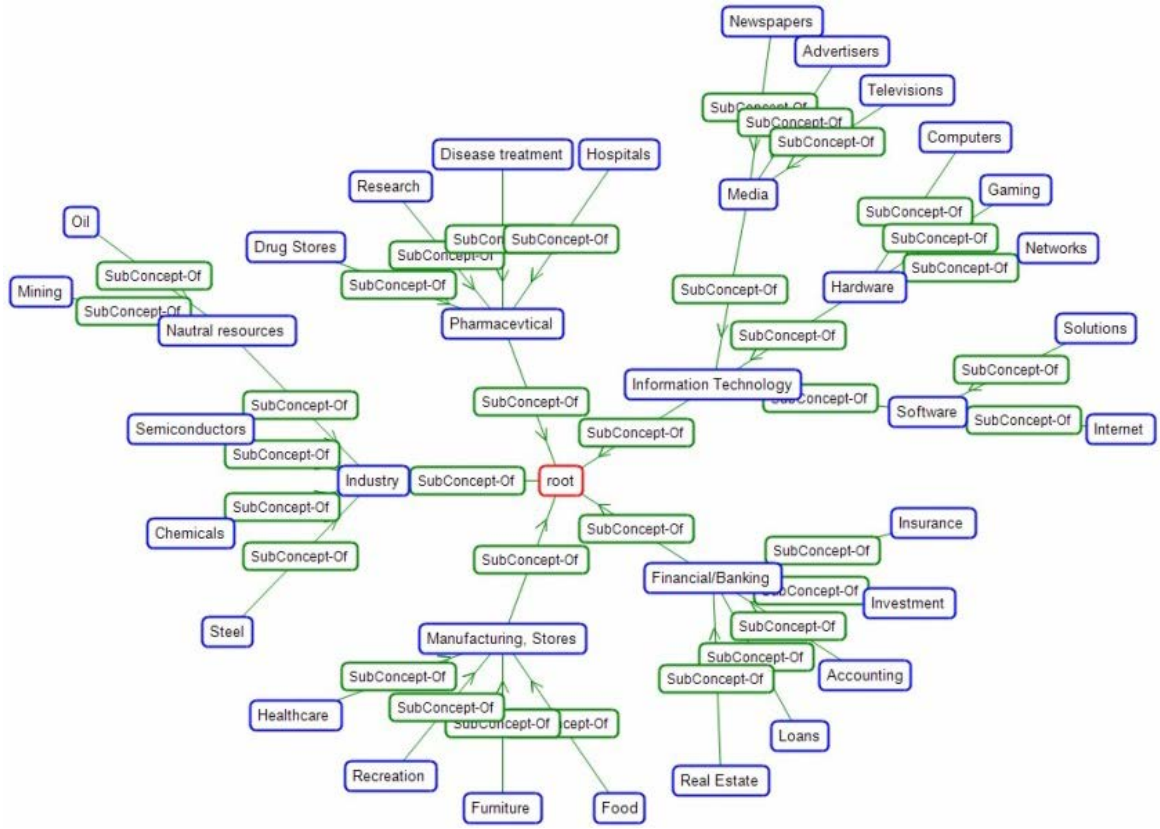


Figure 39: An example of the finance ontology derived from the extended running example.

6.2 Communication Network

In the previous section we demonstrated how can the developed system be used to construct an ontology out of text documents. This section demonstrates the application of OntoGen to the organization ontology, developed from a social graph.

6.2.1 Data Description

The data used in the experiment is a collection of log files with e-mail transactions from a mid-size research institution. It was obtained from spam filter software. Each line of the log files denotes one event at the spam filter software. For the purpose of the experiment, only the events on successful e-mail transactions which include information on the time when the event happened, who sent the e-mail (sender), and who received it (a list of receivers) was used. The advantage of using spam filter log files is additional information provided by spam filter software which tags each e-mail transaction as being “CLEAN” or “SPAM”.

The log files include e-mail data from Sep 5th 2003 to Mar 28th 2005 which sums up to 12.8 GB of data. This was reduced to 564 MB after filtering out successful e-mail transactions, containing approx. 2.7 million of successful e-mail transitions. These 2.7 million of transactions were used for further processing. The whole dataset contains references to approx. 45,000 e-mail addresses. After the data cleaning phase, the number is reduced to approx. 17,000 e-mail addresses out of which 770 e-mail addresses are internal to the observed institution.

There are two significant data transformations in processing the original log data towards ontological representation. The first transformation transforms a collection of e-mail transactions into an e-mail social network. From the set of transactions we construct a graph where vertices are e-mail addresses and edges between vertices represent the communication between the e-mail addresses. In other words, there is an edge between two vertices representing two e-mail addresses if there are e-mail transactions between them. Edges are additionally labelled with the intensity of communication (number of transactions between e-mail addresses representing both vertices). The result of the first transformation is a social network between senders and recipients of the email.

The social network serves as the input to the ontology learning system, with each node in the network corresponding to one instance. The second transformation transforms the social network into a set of sparse vectors, as described in Section 4.1.2, to be used as instance profiles.

6.2.2 Ontology Construction

In our case of modelling a social network of co-workers inside the same organization [26] each instance corresponds to one e-mail address within the organization (one row from

the matrix generated in step 4 in Section 3.3). On the output we want to get an ontology in the form of taxonomy modelling the relation “sub-community-of” which would correspond to the organizational structure of the institution where the e-mail data is coming from. The actual experiment performed on the data described in 6.2.1consisted of 770 e-mail addresses from a mid-size research institution. Each of these e-mail addresses was described with the subset of 17000 e-mail addresses being in the direct or indirect contact with the target e-mail address. Each e-mail address was represented as sparse vector from the e-mail graph.

The result of approximately 20 minute session with OntoGen is shown in Figure 40. The background information that we have used there is associating e-mail addresses with organizational units, which was used to come-up with the concept labels. We can see that the whole organizational structure can on the top level be represented by four communities: electronics and information technologies, chemistry, physics and administration. As expected the administration consists of two sub-communities: management and accounting, where management can be further split into secretariat, legal and commercial support. However, at this point it is not clear if the administration is mainly communicating internally or fulfils its natural role of supporting other organizational units. From the visualization in Section 6.2.3 we can see that indeed it fulfils its role of connecting the research groups. Inside each research group we can see sub-communities that are related to topics, such as, telecommunications, systems and control, artificial intelligence. On a finer level these topic oriented communities have their sub-communities that capture the role of different people in the organization. For instance, artificial intelligence has a sub-community of PhD students, a sub-community of employees and a small sub-community getting mailing lists bounces. Closer inspection using OntoGen shows that this small sub-community consists of researchers also associated to some other institution (having different e-mail address there) sending e-mails to internal mailing lists that bounced (because they were sent from their e-mail address that belongs to the other institution).

An interesting hidden pattern surfaces here showing a sub-community of physics is closely related to electronic and IT community (appears as its sub-community in Figure 40). Investigating the status of those members several years later, we found that most of them became members of the same spin-out company.

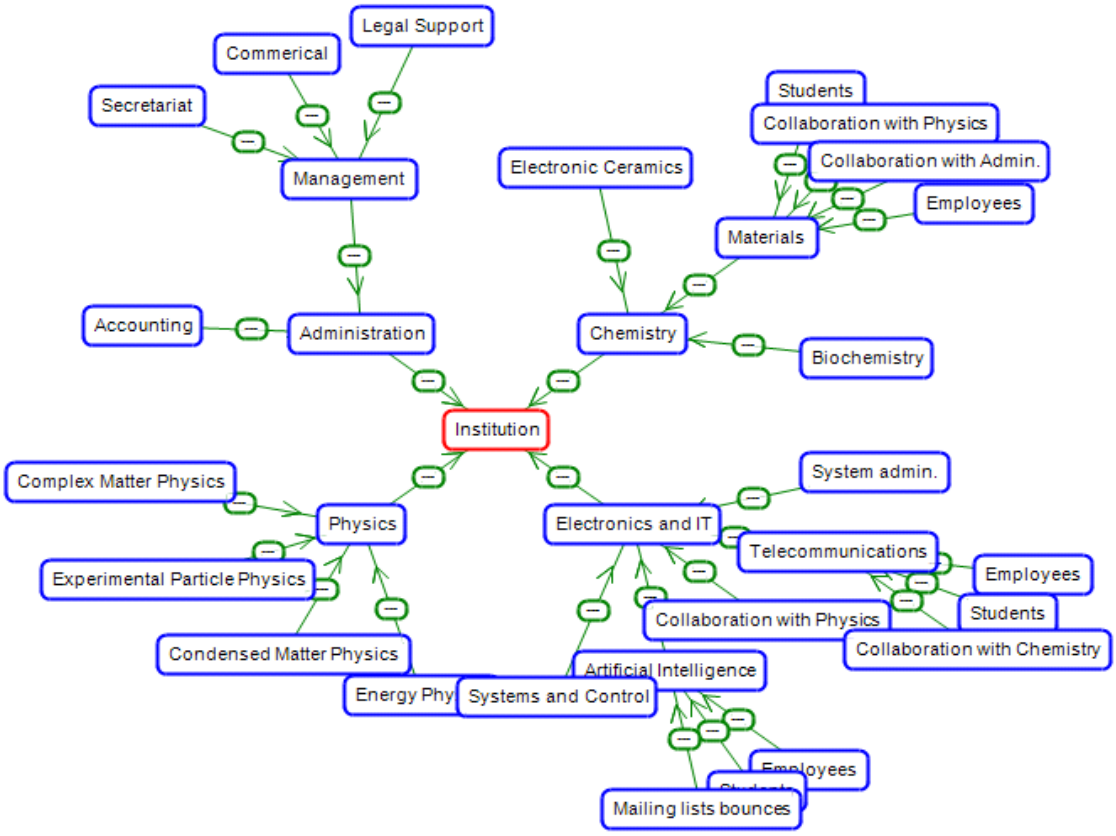


Figure 40: The organizational structure modelled from the e-mail data in a 20 minute session with the ontology learning system OntoGen (as provided in [26]).

6.2.3 Visualization

Figure 41 shows the annonymized dataset from the previous section in the geographical relief visualization, computed using the approach presented in Section 4.5. Each data point represents one e-mail address, e-mail addresses having similar communication are closer on the image and the density of points results in higher elevation of the terrain. From the image it can be seen the top level structure of the communication, corresponding to the top level concepts in the generated ontology. We can see three major “mountains” representing the major research topics of the institution functioning: computer science (electronics and IT), chemistry, and physics. Management/administration is placed in the middle of the terrain serving as a connector between the three main research areas. It is interesting to notice that the institution management not only formally but also actually has its role in communication with all the other organizational units. On the other hand computer science has a rather pointed “mountain” showing intensive communication inside the computer science community.

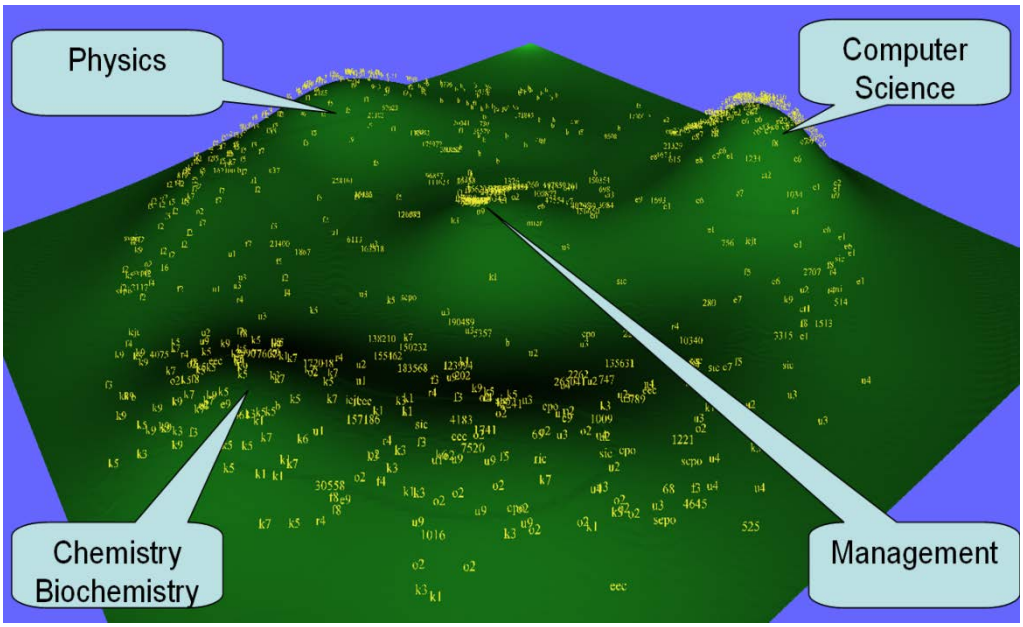


Figure 41: Visualization of e-mail communication records as geographical terrain. The high level structure of the data shows 4 major areas on the map corresponding to the four major groups, three research units (computer science, chemistry, physics) and management.

6.2.4 Evaluation

We perform evaluation of the approach described in the previous sections indirectly by showing the compactness of the clusters as produced by the ontology learning system OntoGen. The main hypothesis we want to prove is that the communication intensity follows organizational structure of an organization – in other words, people inside the same organizational unit are communicating more intensively between each other than to the people outside their organizational unit. To show that, we perform a "gold standard" style of comparison, where we compare the ontological structure obtained from the communication record using the proposed approach to the formal organizational structure of the institution.

Table 17 shows the result of *k*-means clustering producing 10 clusters, as provided by OntoGen on the first ontology level. Columns correspond to the clusters and rows correspond to the organizational units of the institution. For each cluster we give the percentage of e-mail addresses falling into the individual organizational unit – the sum of each column is 1. The analysis of the table shows that individual clusters obtained from the e-mail exchange graph actually contain e-mails belonging mainly to one of the formal groups. For instance, cluster C-0 contains mainly e-mail addresses associated to organizational unit AI1 (Artificial Intelligence 1), C-1 mainly form SC (Systems and Control), C-3 mainly from (Ce) Ceramics, etc.

At a deeper inspection of cluster C-0, we can see that, 36.8% of all the e-mail addresses in this cluster are from AI1(Artificial Intelligence), 21% of e-mail addresses are from BC (Biochemistry) and 15% are from AI2 (Artificial Intelligence). Knowing the background of the institutional organization, it is not surprising, as AI1 and AI2 used to

be the same organizational unit shortly prior to our data collection and still have some common mailing lists and collaborations. The analysis suggests that the community formed around this computer science community contains considerable proportion of members that belong to the organizational unit covering biochemistry, which is a kind of hidden pattern suggesting collaboration between them. After some discussions with the members of the identified community it turned out that indeed some researchers collaborate and work on bioinformatics.

Table 17: Part of the results for data grouped into 10 clusters (C-0, C-1 … C-9) showing the distribution of the clustered e-mails over the formal groups inside the institution. The largest group in each automatically obtained cluster is marked with (!).

Group	C-0	C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9
AI1	0.368!				0.03					
Ch1			0.04	0.02			0.1	0.04	0.06	0.01
BC	0.21	0.05	0.01		0.03		0.361!	0.07	0.04	0.07
SA						0.200!			0.02	
SC		0.381!					0.01			
Ch2			0.14		0.06		0.01			
Ph					0.06		0.03	0.08		0.261!
Ch3	0.01	0.02	0.235!		0.03	0.1	0.01	0.01	0.06	0.02
Ce	0.01		0.01	0.448!		0.05	0.01			
MPh		0.03	0.03		0.444!	0.15	0.01	0.347!		0.17
AI2	0.15		0.01	0.02			0.01	0.01		
EPh			0.01		0.08		0.01	0.03	0.471!	0.01
…	…	…								

Looking more closely into the identified communities, we can see that the most compact are clusters C-3, C-4 and C-8 where over 40% of all the communication is inside one formal organizational unit. On the other hand, C-2, C-5 and C-9 represent communities that connect different formal organizational units. As expected, if we increase the number of groups the compactness of clusters gets better.

6.3 Analysis of the Developed Ontologies

To analyse the influence of the OntoGen system on the developed ontology we compared the ontology, developed using OntoGen, to a manually developed ontology over the same domain. As a manually developed ontology we used a subset of DMoz taxonomy related to physics [72].

The whole dataset contained 4,883 web pages, which were manually categorized by DMoz editors into 37 categories. The dataset was given to a group of computer science graduate students, of which all had a one year university course in physics. The students had a basic training in OntoGen, and had partial understanding of text mining algorithms.

The manually constructed ontology is shown in Figure 42. It can be seen that the ontology is very shallow, most of the concepts being on the first level. Only some of the concepts have sub-concepts. This can be attributed to the collaborative development of DMoz, where each category is assigned to one or more editors. Increasing the depth of

ontology would require more hierarchical editorial structure. It can be seen that some of the concepts also repeat. For example, content of the “Events” sub-concept of the “Mathematical Physics” is strongly related to the “Conferences” concept.

On the other hand, an example the ontology constructed by the students, using OntoGen system (see Figure 43), makes higher use of the sub-concept structure. There are less first level concepts, which are mostly further refined into second and, in few cases, third level sub-concepts. The students did not use the DMoz ontology or any other external vocabulary to help them during the construction period.

Both manually and an example of semi-automatically constructed ontologies can be seen side-by-side in Figure 44.

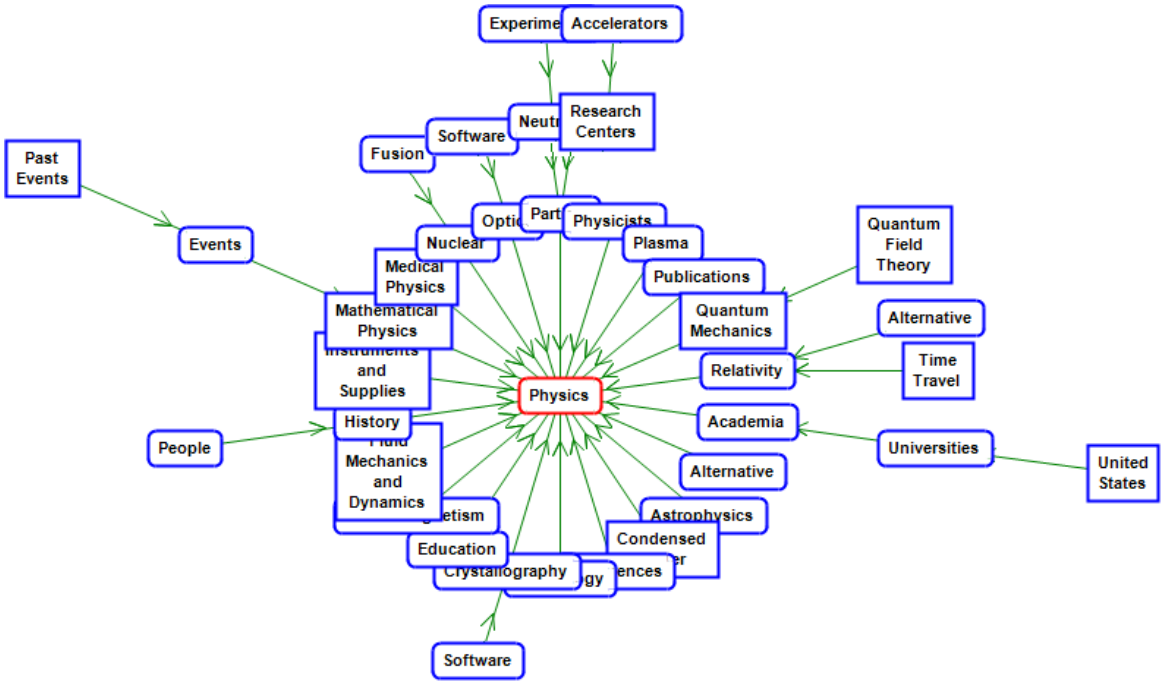


Figure 42: Part of DMoz taxonomy, categorizing a set of 4883 web pages related to physics.

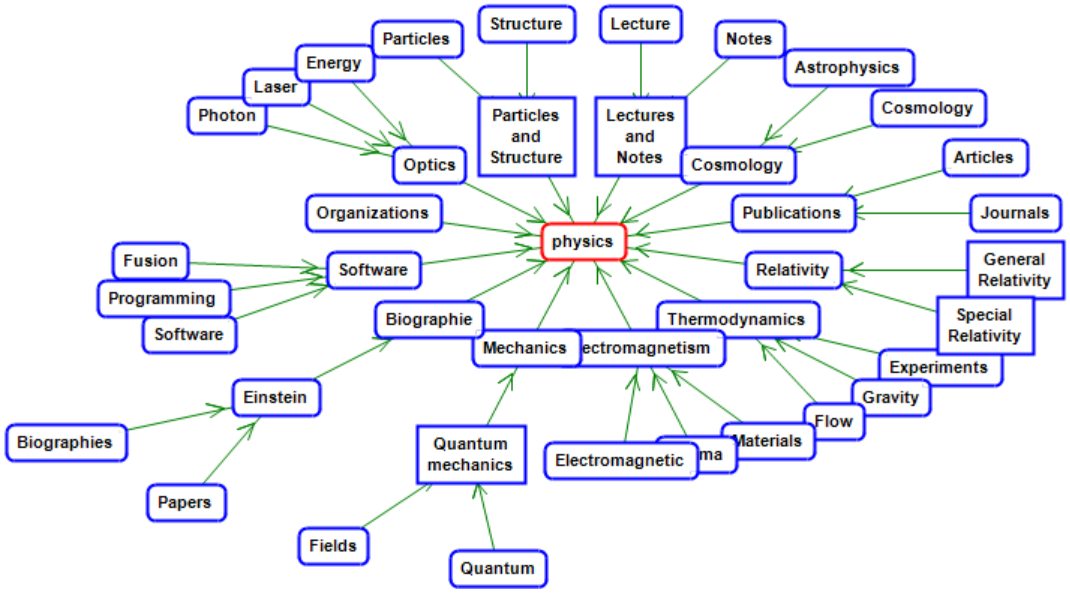


Figure 43: Ontology built using OntoGen, categorizing 4883 web pages related to physics.

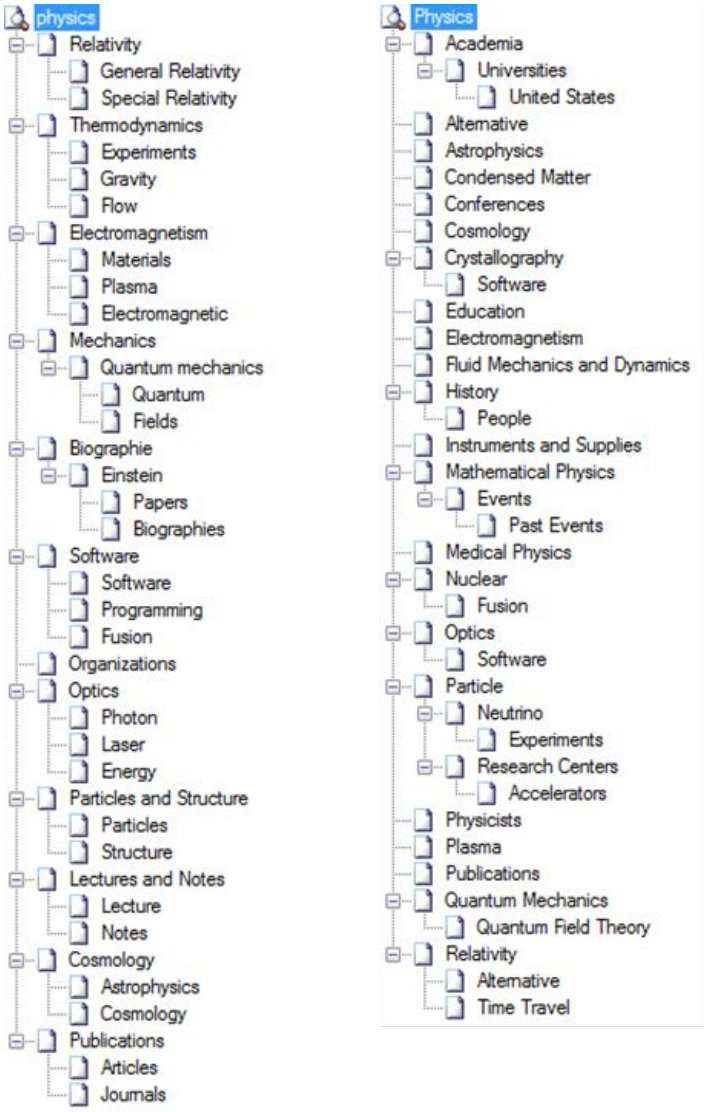


Figure 44: Ontology produced using the OntoGen system (left) and manually constructed one, extracted from DMoz taxonomy (right).

An interesting element that can be analysed is the instance overlap between concepts from the two ontologies. The overlap is computed as *Jaccard similarity coefficient*,

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|},$$

which measures the ratio of instances that two sets share versus all the unique instances in the two sets. The value coefficient is between zero ($A \cap B = \emptyset$) and one ($A = B$).

Figure 45 shows for each concept from the OntoGen ontology the nearest DMoz concept, based on their Jaccard Similarity Coefficient. It can be seen that most concepts have a match to semantically similar concepts (e.g. “Relativity” is mapped to “Relativity” with high overlap). There are also some concepts from the OntoGen side that have no good match on the DMoz side. For example, “Software” instances were grouped into one concept on the OntoGen side and are spread over several concepts on the DMoz side (e.g. “Optics/Software” and “Crystallography/Software”).

OG/Relativity → DMoz/Physics/Relativity [0.29]
OG/Relativity/General Relativity → DMoz/Physics/Relativity [0.24]
OG/Relativity/Special Relativity → DMoz/Physics/Relativity/Time Travel [0.08]
OG/Thermodynamics → DMoz/Physics/Fluid Mechanics and Dynamics [0.16]
OG/Thermodynamics/Experiments → DMoz/Physics/Relativity [0.03]
OG/Thermodynamics/Gravity → DMoz/Physics/Cosmology [0.06]
OG/Thermodynamics/Flow → DMoz/Physics/Fluid Mechanics and Dynamics [0.22]
OG/Electromagnetism → DMoz/Physics/Electromagnetism [0.13]
OG/Electromagnetism/Materials → DMoz/Physics/Condensed Matter [0.10]
OG/Electromagnetism/Plasma → DMoz/Physics/Plasma [0.13]
OG/Electromagnetism/Electromagnetic → DMoz/Physics/Electromagnetism [0.13]
OG/Mechanics → DMoz/Physics/Quantum Mechanics [0.24]
OG/Mechanics/Quantum mechanics → DMoz/Physics/Quantum Mechanics [0.28]
OG/Mechanics/Quantum mechanics/Quantum → DMoz/Physics/Quantum Mechanics [0.27]
OG/Mechanics/Quantum mechanics/Fields → DMoz/Physics/Quantum Mechanics [0.09]
OG/Biographies → DMoz/Physics/Physicists [0.15]
OG/Biographies/Einstein → DMoz/Physics/History/People [0.22]
OG/Biographies/Einstein/Papers → DMoz/Physics/History/People [0.10]
OG/Biographies/Einstein/Biographies → DMoz/Physics/History/People [0.17]
OG/Software → Physics [0.16]
OG/Software/Software → DMoz/Physics/Fluid Mechanics and Dynamics [0.08]
OG/Software/Programming → DMoz/Physics/Academia/Universities [0.10]
OG/Software/Fusion → DMoz/Physics/Nuclear [0.09]
OG/Organizations → DMoz/Physics/Plasma [0.07]
OG/Optics → DMoz/Physics/Optics [0.19]
OG/Optics/Photon → DMoz/Physics/Optics [0.06]
OG/Optics/Laser → DMoz/Physics/Optics [0.05]
OG/Optics/Energy → DMoz/Physics/Optics/Software [0.14]
OG/Particles and Structure → DMoz/Physics/Particle/Neutrino [0.13]
OG/Particles and Structure/Particles → DMoz/Physics/Particle/Neutrino [0.26]
OG/Particles and Structure/Structure → DMoz/Physics/Crystallography [0.14]
OG/Lectures and Notes → DMoz/Physics/ Quantum Mechanics/Quantum Field Theory [0.07]
OG/Lectures and Notes/Lecture → DMoz/Physics/Cosmology [0.07]
OG/Lectures and Notes/Notes → DMoz/Physics/Quantum Mechanics/Quantum Field Theory [0.05]
OG/Cosmology → DMoz/Physics/Cosmology [0.19]
OG/Cosmology/Astrophysics → DMoz/Physics/Astrophysics [0.26]
OG/Cosmology/Cosmology → DMoz/Physics/Cosmology [0.29]
OG/Publications → DMoz/Physics/Mathematical Physics [0.10]
OG/Publications/Articles → DMoz/Physics/History/People [0.06]
OG/Publications/Journals → DMoz/Physics/Publications [0.11]

Figure 45: List of concepts from the ontology developed with OntoGen, mapped to the highest matching concept from the DMoz ontology based on Jaccard coefficient.

6.4 Analysis of User Feedback

During the development we used the OntoGen system [66] in domains such as business, legislation and digital libraries within several research projects (SEKT, IMAGINATION, NEON, ENVISION, TAO). The users participating in the case studies were domain experts with limited experience in ontology construction. The feedback we got from the users was used to improve the user interface. In general, the system was found to enable the users to model ontologies which would be significantly more difficult and expensive to model otherwise.

The freely available version of the system [73] was extensively tested through user study at Faculty of Arts and Sciences in Rijeka [74][66][80]. The general goal of user study was to gain as additional information in order to objectively assess the current stage of development of the OntoGen and its functionalities from the end-users' point of view. The reminder of this section describes the methodology of the user study and the analysis of the results.

6.4.1 Methodology

For that purpose two different groups of users were chosen to provide the useful data from two different points of view. The first group consisted of 48 Computer sciences students and the second from 43 Psychology and Pedagogy sciences students. Such of user selection enabled the gathering of data from subjects who, generally, have good computer skills. The first group has more than average informational knowledge and understands the technical background of software development .The second group has more than common knowledge about cognitive processes of mental data structuring and organizing.

Methodology of our user study involved two questionnaires that we have developed for the purpose of this study. One was used to assess the user preferences and computer usage skills, as well as their knowledge on cognitive processes (Initial questionnaire). It contains 15 questions and was filled by the users prior to interacting with the tool. The second questionnaire contains 24 questions and was designed for assessing the users experience with the tool and was given to the users after the testing (End questionnaire). Both questionnaires are presented in Appendix B.

The initial questionnaire was the first step in the user study. The questionnaire was organized into several topics covering the computer usage, learning habits and knowledge organization.

Computer usage was tested with various questions in the area of computer experience and skills. This involved computer skills and time spent on a computer in different areas: Email, Chat, web surfing, web searching, e-Learning system, Windows, Linux and Mac.

Learning habits part examined their learning preferences: learning in group, learning

alone, by reading, by talking to others, listening, through practice. It also included questions capturing their opinion on using technology while learning.

Knowledge organization part of the questionnaire covered mental processes they recognize while learning and organizing knowledge, as well as their formal knowledge about that area (e.g. cognitive maps). It included also qualitative questions addressing their opinion on the term “cognitive maps”, their methods of learning and organizing knowledge, and the idea about possible computer tool which would ease knowledge organization.

The end questionnaire was given to the users at the end of the testing phase was used to obtain the final feedback on the tested tool. The questions included overall impression of the tool, the layout assessment of the main functionalities and their graphical interface, and concept and document management. Their experience in creating and saving ontologies was qualitatively addressed, as well as concept suggestion methods and ontology visualizations.

An average length of user trail session took 90 minutes and consisted of three phases. The first phase was used to introduce the users to the purpose of the task and to explain the procedures which will follow in the second and third phases of the trail. The main data collection happened in the second phase of user trials, which consisted of the following steps:

1. Filling in the Initial questionnaire
2. Demonstration of the OntoGen system by the facilitator.
3. Testing the ontology construction system OntoGen:
    a. Construct ontology which captures the areas covered by the companies in the given collection of descriptions of 7177 companies.
    b. Construct two different ontologies on top of the same collection of 5000 news articles from Reuters news agency. The first ontology should group instances based on geographical properties, and the second based on topic of news articles.
4. Filling in the End questionnaire. After the trial students also had a possibility to discuss their experiences with other participants involved in trials.

Drawing conclusion is the third phase of user trial in which it is necessary to thank all the users for participating in user trial. Also, at this moment, it is very important to discus, if necessary, any question or thought that roused from the tasks during the trials.

6.4.2 Results

Results of Initial and End questionnaire are presented in Appendix C.

Analysis of the Initial questionnaire revealed statistically significant differences in frequency of some of the programs and services usage (e-mail, chat and surfing the web),

but there were no significant difference in web searching, e-learning system, Linux and mac OSX usage. The significant difference occurred in formal knowledge of learning processes, as expected; most of Psychology and Pedagogy students knew exactly what "Cognitive maps" are. In the same way there was significant difference in computer usage in favour of Computer sciences students as well as the average amount of time spent working on their computers. There are no differences in Linux and Mac OSX usage because students from all involved Departments are normally not using it. They are not different in time spent searching the Web.

Analysis of the End questionnaire revealed that in general the users had good impression on the OntoGen tool (75% marked it "good" to "very good"). The most of the users think that the tool is useful (93%), and around half of them referred to the layout of the three major interface parts as "mostly good"(Ontology visualisation - 44%, Concept tree – 45%, Concept Properties – 50%). The similar situation occurred in assessment of graphical interface. The overall impression is "mostly good", but there were objections expressed concerning the attractiveness (36% find it not sufficiently attractive or not attractive at all). Users repeatedly pointed out that there should be more colours.

Qualitative analysis revealed that one of the main advantages of OntoGen is managing large data bases with easiness. According to users, the tool is efficient, saves time and effort and gives a lot of space for user intervention. Disadvantages, as perceived, mostly concern unattractive look, abstract conception, occasional slowness and the need to learn how to use it first.

After testing the system for only two hours, the 75% of the users could easily identify distinction between the two ways of ontology creation; namely, the unsupervised learning (clustering) and the semi-supervised learning (active learning) methods.

Lack of detailed instructions and need for all the options in one place were pointed out as the main difficulties in generating new concepts. They also found the basic concept management is "mostly" to "very easy" with qualitative remark of making it simpler if possible.

Visualization of concepts proved to be of a great help in choosing sub-concepts for majority (90%) of the users. Most of the users found the current state of visualization appropriate, but in need of more attractive graphical solutions. Furthermore, users think that with adding of concepts, picture becomes cluttered which makes it hard to inspect. They also think that the active learning part can be improved (but are not certain how to do it) and they propose to extend it by manual placing of new documents.

Visualization of Ontologies qualitative analysis revealed that the current state of its development is satisfactory, but could include search by a simple mouse click.

When it comes to management of the concept documents, the users think that it is quite good. There are no significant differences in assessing that part of OntoGen. General mark for all the operations included in the document management functionalities is "very

good”. The smaller objections concerned relation management, where we got suggestions to include some simple handling of documents, e.g., by dragging a mouse. There was a general comment that the presentation of similarity of a document to a concept should be made simpler.

7 Conclusions

This thesis proposes an ontology learning framework, designed to cover semi-automatic scenario, combining machine learning with interactive user interface. The framework provides the ability to incorporate domain data, which can be exploited by machine learning techniques to help the users during the construction process with concept and relation suggestions, summarization and visualization of instances, concepts and developed ontology, and ability to easily categorize new instances within the developed ontology.

The framework and machine learning techniques were implemented within an interactive system OntoGen. There is inherent danger that introducing machine learning methods would make the whole system hard to comprehend and to use by the domain experts without previous experience with machine learning. However, the user trails show that our system managed to avoid most such missteps and that it can be used by people without background in machine learning.

7.1 Scientific Contributions

The work presented in this thesis compromises of the following contributions to ontology learning, text mining and visualization.

- Definition of Ontology Learning Framework that supports interactive semi-automatic ontology construction, and provides the mechanisms for incorporating domain data and machine learning algorithms.
- Population of the ontology learning framework with existing machine learning techniques and feature representations. This includes necessary adaptations of the existing algorithms to fit the proposed framework.
- Development of novel algorithms and approaches for interactive visualization of text corpora and ontologies.
- Development of algorithms and approaches which enabled the usage of background knowledge for optimizing word weighting schema.

7.2 Future Work

There are several potential directions for future work. The list of machine learning algorithms implementing various aspects of the ontology learning framework can be extended with the introduction of additional or alternative machine learning algorithms,

such as Latent Dirichlet Allocation, for learning ontology construction operators. With the current focus on textual data within the ontology learning community there is a great potential for extending the system to better address specificities of different feature representations covering other data modalities. Some of this was already presented in this thesis (social networks) or is currently under development [63]. Additionally, the ability to easily combining several feature representations (e.g. text and image features) can result in a whole new class of ontologies, which are not “learnable” with the current techniques. There are still additional improvements that can be done on the user interaction side, most important being the ability for collaborative editing of larger ontologies.

The developed system with a subset of the presented functionality is freely available online. Since it was first made available on the Web in 2006, reports that we received from various users downloading the system indicate the usefulness of the proposed approach, and its applicability in practice. The system was downloaded more than thousand times and is used by organizations ranging from New York Times, Bloomberg, Toyota, Microsoft, to Katholieke Universiteit Leuven, Mehiläinen Medical Center, University of Washington, and University of Melbourne. As such, one potential future direction is also the development of a commercial product based on the present work.

8 Acknowledgements

First, I wish to thank my advisor Professor Dunja Mladenić, for her support and guidance during my doctoral studies, which resulted in this thesis. I would also like to thank my group of co-authors and collaborators, and my colleges from Jožef Stefan Institute for many valuable discussions and contributions.

I would also like to thank the members of my doctoral committee, Irena Nančovska Šerbec, Marko Bohanec and John Davies, for their valuable comments and remarks.

Special thanks go to my wife Carolina, my parents Franci and Anica, and my sister Helena for their support and encouragement.

This work was supported by the Slovenian Research Agency and the European Community under SEKT Semantically Enabled Knowledge Technologies (IST-1-506826-IP), NeOn Lifecycle Support for Networked Ontologies (IST-4-027595-IP), PASCAL Network of Excellence (IST-2002-506778), and PASCAL2 (IST-NoE-216886).

9 References

[1] Gruber, T. R. A translation approach to portable ontology specifications. *Knowledge Acquisition* **5**, 2 (1993).

[2] Lenat, D. B.; Guha, R. V. *Building large knowledge-based systems: representation and inference in the Cyc project* (Addison-Wesley, Reading, Massachusetts, 1990).

[3] Witbrock, M.; Baxter, D.; Curtis, J.; Schneider, D.; Kahlert, R.; Miraglia, P.; Wagner, P.; Panton, K.; Matthews, G.; Vizedom, A. An Interactive Dialogue System for Knowledge Acquisition in Cyc. In: *Proceedings of the IJCAI-2003 Workshop on Mixed-Initiative Intelligent Systems* (Acapulco, 2003).

[4] Sarjant S.; Legg, C.; Robinson, M.; Medelyan, O. "All You Can Eat" Ontology-Building: Feeding Wikipedia to Cyc. In: *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence* (Italy, 2009).

[5] Uschold, M.; King, M.; Moralee, S.; Zargois, Y. The Enterprise Ontology. *The Knowledge Engineering Review* **13**, 1 (1998).

[6] Fernández, M.; Gómez-Pérez, A.; Pazos, J.; Pazos, A. Building a chemical ontology using MethOntology and the ontology design environment. *IEEE Intelligent Systems Applications* **14**, 1 (1999)

[7] Grobelnik, M.; Mladenić, D. Knowledge Discovery for Ontology Construction. In *Semantic Web Technologies: Trends and Research in Ontology-based Systems*. (John Wiley, New York, 2006)

[8] Maedche, A.; Staab, S. Ontology Learning for the Semantic Web. *IEEE Intelligent Systems* **16**, 2 (2001).

[9] Cimiano, P.; Völker, J. Text2Onto - A Framework for Ontology Learning and Data-driven Change Discovery. In: *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems, volume 3513 of Lecture Notes in Computer Science* (Spain, 2005).

[10] Missikof, M.; Navigli, R.; Velardi, P. Integrated Approach to Web Ontology Learning and Engineering. *Computer* **35**, 11 (2002).

[11] Buitelaar, P.; Cimiano, P. Ontology Learning and Population: Bridging the Gap between Text and Knowledge. *Frontiers in Artificial Intelligence and Applications* **167** (2008).

[12] Tang, J.; Leung, H.; Luo, Q.; Chen, D.; Gong, J. Towards Ontology Learning from Folksonomies. In: *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09)* (San Francisco, 2009).

[13] Gulla, J. A.; Borch, H. O.; Ingvaldsen, J. E. Ontology Learning for Search Applications. In: *Proceedings of 2007 OTM Confederated internatlional conference on On the move to meaningful internet systems* (Berlin, 2007).

[14] Carlson, A.; Betteridge, J.; Wang, R.C.; Hruschka Jr., E.R.; Mitchell, T.M. Coupled Semi-Supervised Learning for Information Extraction. In: *Proceedings of the ACM International Conference on Web Search and Data Mining* (New York, 2010).

[15] Yates, A.; Cafarella, M.; Banko, M.; Etzioni, O.; Broadhead, M.; Soderland, S. TextRunner: open information extraction on the web. In: *NAACL-Demonstrations '07 Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations* (2007).

[16] Agirre E.; Ansa, O.; Hovy, E.; Martínez, D. Enriching very large ontologies using the WWW. In: *Proceedings of ECAI 2000 Workshop on Ontology Learning* (2000).

[17] Cimiano, P.; Hotho, A.; Staab, S. Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. *Journal of Artificial Intelligence Research* **24**, 1 (2005).

[18] Bisson G.; Nédellec, C.; Cañamero, D. Designing clustering methods for ontology building: The Mo'K workbench. In: *Proceedings of ECAI 2000 Workshop on Ontology Learning* (2000).

[19] Reinberger M. L.; Spyns, P. Discovering Knowledge in Texts for the learning of DOGMA-inspired ontologies. In: *Proceedings of ECAI 2004 Workshop on Ontology Learning and Population* (2004)

[20] Grobelnik, M.; Mladenić, D. Visualization of news articles. *Informatica journal* **28**, 4 (2004).

[21] Grobelnik, M.; Brank, J.; Mladenić, D.; Novak, B.; Fortuna, B. Using DMoz for constructing ontology from data stream. In: *28th International Conference on Information Technology Interfaces* (Croatia, 2006)

[22] Ehrig M.; Haase, P.; Hefke, M.; Stojanovic, N. Similarity for ontologies: a comprehensive framework. In: Proceedings of the 13th European Conference on Information Systems (2005).

[23] Brickely, D.; Guha, R. V.; McBride, B. RDF Vocabulary Description Language 1.0: RDF Schema. http://www.w3.org/TR/rdf-schema/ (accessed July 2011).

[24] Manning, C.D.; Schutze, H. *Foundations of statistical Natural Language Processing* (MIT Press, 1999).

[25] Jiang, Y. G.; Ngo, C. W.; Yang, J. Towards optimal bag-of-features for object categorization and semantic video retrieval. In ACM Internationall Conference on Image and Video Retrieval (2007).

[26] Grobelnik, M.; Mladenić, D.; Fortuna, B. Semantic Technology for Capturing Communication Inside an Organization. *IEEE Internet Computing* **13**, 4 (2009).

[27] Fortuna, B.; Cristianini, N.; Shawe-Taylor, J. A Kernel Canonical Correlation

Analysis For Learning The Semantics Of Text. In: Campos-Valls et al (ed.) *Kernel methods in bioengineering, communications and image processing* (IGI Global, 2006).

[28] Tan, P. N.; Steinbach, M.; Kumar, K. *Introduction to Data Mining* (Addison-Wesley, 2005).

[29] Cohen W.W.; Ravikumar P.; Fienberg S.E. A comparison of string distance metrics for name-matching tasks. In: *Proceedings of IJCAI03 Workshop on Information Integration on the Web* (Mexico, 2003).

[30] Mladenić, D.; Grobelnik, M. Feature selection on hierarchy of web documents. *Decision Support Systems* **35**, 1 (2003).

[31] Porter, M. F. An algorithm for suffix stripping. *Program* **14**, 3 (1980).

[32] Mladenić, D.; Grobelnik, M. Visualizing very large graphs using clustering neighborhoods. In: Morik et al (ed.) *Local pattern detection: international seminar, revised selected papers, Lecture notes in computer science, Lecture notes in artificial intelligence, 3539, State-of-the-art survey*. (Springer, 2005).

[33] Olston, C.; Chi, H. E.; ScentTrails: Integrating browsing and searching on the Web. *ACM Transactions on Computer-Human Interaction* **10**, 3 (2003).

[34] Jain, A. K.; Murty, M. N.; Flynn, P. J. Data clustering: a review. *ACM Computing Surveys* **31**, 3 (1999).

[35] Deerwester, S.; Dumais, S. T; Furnas, G. W.; Landauer, T. K.; Harshman, R. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* **41**, 6 (1990).

[36] Blei, D. M.; Ng, A. Y.; Jordan, M. I. Latent dirichlet allocation. *The Journal of Machine Learning Research* **3**, 3 (2003).

[37] Grobelnik, M.; Brank, J.; Fortuna, B.; Mozetic, I. Contextualizing ontologies with ontolight: a pragmatic approach. In: *Proceedings of the 10th International multi-conference Information Society* (Slovenia, 2007).

[38] DMoz: Open Directory Project. http://www.dmoz.org/ (accessed July 2011).

[39] Settles, B. Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2010).

[40] Steinbach, M.; Karypis, G.; Kumar, V. A comparison of document clustering techniques. In: *Proceedings of KDD Workshop on Text Mining* (2000).

[41] Grobelnik, M.; Mladenić, D. Efficient visualization of large text corpora. In: *Proceedings of the Seventh TELRI seminar* (Croatia, 2002).

[42] Arthur, D.; Vassilvitskii, S. *k*-means++: the advantages of careful seeding. In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (2007).

[43] Arthur, D.; Vassilvitskii, S. How slow is the *k*-means method? In: *Proceedings of the 22nd Symposium on Computational Geometry* (2006).

[44] Shawe-Taylor, J.; Cristianini, N. *Kernel Methods for Pattern Analysis*. (Cambridge

University Press, 2004).

[45] EuroVoc. http://europa.eu/eurovoc/ (accessed July 2011).

[46] Lenat, D. B. Cyc: A Large-Scale Investment in Knowledge Infrastructure. *Communication of ACM* **38**, 11 (1995).

[47] Bizer, C.; Heath, T.; Berners-Lee, T. Linked Data – The Story So Far. *International Journal on Semantic Web and Information Systems* **5**, 3 (2009).

[48] AgroVoc. http://www.fao.org/aims/ag_intro.htm (accessed July 2011).

[49] ASFA: http://www.fao.org/fi/asfa/partners.asp (accessed July 2011).

[50] Grobelnik, M.; D. Mladenić. Simple classification into large topic ontology of Web documents. In: *Proceedings of 27th International Conference on Information Technology Interfaces* (Croatia, 2005).

[51] Mladenić, D.; Grobelnik, M. Mapping documents onto web page ontology. *Web mining: from web to semantic web, Lecture notes in artificial inteligence, Lecture notes in computer science* **3209** (2004).

[52] Settles, B. Active Learning Literature Survey. *Computer Sciences Technical Report 1648, University of Wisconsin–Madison* (2009).

[53] Tong, S.; Koller, D. Support Vector Machine Active Learning with Applications to Text Classification. In: *Proceedings of the Seventeenth International Conference on Machine Learning* (2000).

[54] Fortuna, B.; Mladenić, D.; Grobelnik, M. Visualization of Temporal Semantic Spaces. In: Davies, J. et al (ed.) *Semantic Knowledge Management* (Springer, 2008).

[55] Grobelnik, M.; Brank, J.; Mladenić, D.; Novak, B.; Fortuna, B. Using DMoz for Constructing Ontology from Data Stream. In: *Proceedings of 28th Int. Conference on Information Technology Interfaces* (2005).

[56] Lewis, D.; Yang, Y.; Rose, T.; Li, F. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research* **5** (2004).

[57] Stajner, T.; Rusu, D.; Dali, L,; Fortuna, B.; Mladenić, D.; Grobelnik, M. A service oriented framework for natural language text enrichment. *Informatica* **34**, 3 (2010).

[58] OpenCyc. http://sw.opencyc.org/ (accessed July 2011).

[59] Brank, J.; Grobelnik, M.; Milic-Frayling, N.; Mladenić, D. Feature Selection Using Support Vector Machines. In: *Proceedings of the 3rd International Conference on Data Mining Methods and Databases for Engineering, Finance, and Other Fields* (2002).

[60] Fortuna, B.; Mladenić, D.; Grobelnik, M. Visualization of text document corpus. *Informatica* **29** (2006).

[61] Carroll , J. D.; Arabie, P. Multidimensional scaling. In M.R. Rosenzweig and L.W. Porter (Eds.): *Annual Review of Psychology* **31** (1980).

[62] Barret, R.; Berry, M.; Chan, T.; Demmel, J.; Donato, J.; Dongarra, J.; Eijkhout, V.; Pozo, R.; Romine, C.; Vorst, H. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* (SIAM, Philadelphia, 1993).

[63] Tomašev, N.; Fortuna, B.; Mladenić. D. OntoGen Extension for Exploring Image Collections. IEEE International Conference on Intelligent Computer Communication and Processing (Romania, 2011).

[64] Hsu, C. W.; Lin, C. J. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks* **13**, 2 (2002).

[65] Fortuna, B.; Mladenić, D.; Grobelnik, M. Semi-automatic Construction of Topic Ontologies. *Semantics, Web and Mining. Lecture Notes in Computer Science* **4289** (Springer, 2006).

[66] Fortuna, B.; Grobelnik, M.; Mladenić, D. OntoGen: Semi-automatic Ontology Editor. In: *Human interface and the management of information. Interacting in information environments. Lecture Notes in Computer Science* **4558** (Springer, 2007).

[67] Text Garden. http://videolectures.net/stw07_grobelnik_tg/ (accessed July 2011).

[68] Fortuna, B.; Mladenić, D.; Grobelnik, M. Application of semantic annotations to predicting users' demographics. In: *Proceeding of the third workshop on Exploiting semantic annotations in information retrieval* (2010).

[69] Fortuna, B.; Mladenić, D.; Grobelnik, M. User Modeling Combining Access Logs, Page Content and Semantics. In: *Proceeding of 1st International Workshop on Usage Analysis and the Web of Data in the 20th International World Wide Web Conference* (India, 2011).

[70] Fortuna, B.; Fortuna, C.; Mladenić, D. Real-time News Recommender System. In: *Machine Learning and Knowledge Discovery in Databases. Lecture Notes in Computer Science* **6323** (2010).

[71] Bartkute-Norkuniene, V. Stochastic Optimization Algorithms for Support Vector Machines Classification. *Informatica* **20**, 2 (IOS Press Amsterdam, The Netherlands, 2009).

[72] DMoz Science/Physics category. http://www.dmoz.org/Science/Physics/ (accessed July 2011).

[73] OntoGen website. http://ontogen.ijs.si/ (accessed July 2011).

[74] Ilijasic Misic, I.; Taksic, V.; Kovacic, B.; Mohoric, T. Design of a user study and evaluation of software tool OntoGen V 2.0. *Technical Report FAS Rijeka* (2006).

[75] Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C. Introduction to Algorithms, Third Edition (MIT Press, 2009).

[76] Delicious. http://delicious.com (accessed July 2011).

[77] Novalija, I.; Mladenić, D.; Bradesko, L. OntoPlus: Text-driven ontology extension using ontology content, structure and co-occurrence information. *Knowledge-Based Systems* (Elsevier, 2011).

[78] Shah, P.; Schneider, D.; Matuszek, C.; Kahlert, R. C.; Aldag, B.; Baxter, D.; Cabral, J.; Witbrock, M.; Curtis, J. Automated population of Cyc: Extracting information about named-entities from the web. In: *Proceedings of the Nineteenth International*

FLAIRS Conference (2006).

[79] Fortuna, B.; Grobelnik, M.; Mladenić, D. Background Knowledge for Ontology Construction. In: *Proceedings of the 15th international conference on World Wide Web* (Edinburgh, Scotland, 2006).

[80] Ilijasic Misic, I.; Kovacic, B.; Mohoric, T.; Mladenić, D.; Fortuna, B.; Grobelnik, M. User Study of Ontology Generation Tool. In: *29th International Conference on Information Technology Interfaces* (Croatia, 2007).

Index of Figures

Index of Tables

Appendix A: Running Example

Ticker	Profile
APPL[3]	"Apple Inc., together with subsidiaries, designs, manufactures, and markets personal computers, mobile communication and media devices, and portable digital music players, as well as sells related software, services, peripherals, networking solutions, and third-party digital content and applications worldwide. The company sells its products worldwide through its online stores, retail stores, direct sales force, third-party wholesalers, resellers, and value-added resellers. In addition, it sells third-party Mac, iPhone, iPad, and iPod compatible products, including application software, printers, storage devices, speakers, headphones, and other accessories and peripherals through its online and retail stores; and digital content and applications through the iTunes Store. The company sells its products to consumer, small and mid-sized business, education, enterprise, government, and creative markets. As of September 25, 2010, it had 317 retail stores, including 233 stores in the United States and 84 stores internationally. The company, formerly known as Apple Computer, Inc., was founded in 1976 and is headquartered in Cupertino, California."
MSFT[4]	"Microsoft Corporation develops, manufactures, licenses, and supports a range of software products and services for various computing devices worldwide. The company?s Windows & Windows Live Division segment offers Windows operating system, Windows Live, and Internet Explorer. It offers Windows operating system, which include Windows 7, Windows Vista, and Windows XP Home, as well as Windows Live suite of applications and Web services. Microsoft?s Server and Tools segment provides Windows Server operating systems, Windows Azure, Microsoft SQL Server, SQL Azure, Visual Studio, Silverlight, System Center products, Biz Talk server, Microsoft consulting services, and product support services. This segment also provides enterprise consulting and product support services; and training and certification to developers and information technology professionals, as well as builds standalone and software development lifecycle tools for software architects, developers, testers, and project managers. The company?s Online Services Division segment offers online information products, such as Bing, MSN portals, and channels; and an online advertising platform for publishers and advertisers. Microsoft?s Microsoft Business Division segment offers Microsoft Office, Microsoft SharePoint, and Microsoft Dynamics ERP and CRM, as well as Microsoft Office Web Apps. The company?s Entertainment and Devices Division segment develops, produces, and markets the Xbox 360 platform; PC software games; online games and services; Mediaroom, an Internet protocol television software; Windows Phone and Windows Embedded device platforms; the Zune digital music and entertainment platform; and application software for Apple?s Macintosh computers, Microsoft PC hardware products, and other devices. This segment also involves in retail and marketing of packaged versions of the Microsoft Office system and the Windows operating systems. Microsoft was founded in 1975 and is headquartered in Redmond, Washington."
GOOG[5]	"Google Inc. maintains an index of Web sites and other online content for users, advertisers, and Google network members and other content providers. It offers AdWords, an auction-based advertising program; AdSense program, which enables Web sites that are part of the Google Network to deliver ads from its AdWords advertisers; Google Display, a display advertising network that comprises the videos,

[3] http://finance.yahoo.com/q/pr?s=AAPL+Profile
[4] http://finance.yahoo.com/q/pr?s=MSFT+Profile
[5] http://finance.yahoo.com/q/pr?s=GOOG+Profile

	text, images, and other interactive ads; DoubleClick Ad Exchange, a real-time auction marketplace for the trading of display ad space; and YouTube that provides video, interactive, and other ad formats for advertisers. The company also provides Google Mobile that optimizes Google?s applications for mobile devices in browser and downloadable form; and enables advertisers to run search ad campaigns on mobile devices, as well as Google Local that provides local information on the Web; and Google Boost for small businesses to participate in the ads auction. In addition, it offers Android, an open source mobile software platform; Google Chrome OS, an open source operating system; Google Chrome, a Web browser; Google TV, a platform for the consumers to use the television and the Internet on a single screen; and Google Books platform to discover, search, and consume content from printed books online. Further, the company provides Google Apps, a cloud computing suite of message and collaboration tools, which includes Gmail, Google Docs, Google Calendar, and Google Sites; Google Search Appliance that offers real-time search of business and intranet applications, and public Web sites; Google Site Search, a custom search engine; Google Commerce Search for online retail enterprises; Google Checkout to make online shopping and payments streamlined and secure; Google Maps Application Programming Interface; and Google Earth Enterprise, a firewall software solution for imagery and data visualization. Google Inc. was founded in 1998 and is headquartered in Mountain View, California."
NOK[6]	"Nokia Corporation manufactures and sells mobile devices, and provides Internet and digital mapping and navigation services worldwide. Its Devices & Services segment develops and manages a portfolio of mobile devices, such as mobile phones, smartphones, and mobile computers; services; applications; and content. It also offers Internet services focusing on music, navigation, media, and messaging, as well as on the tools that enable developers to create applications under the Ovi brand name. The company?s NAVTEQ segment provides various digital map information and related location-based content and services to mobile device and handset manufacturers, automobile manufacturers and dealers, navigation systems manufacturers, software developers, Internet portals, parcel and overnight delivery services companies, and governmental and quasi-governmental entities. Its map database enables its customers to offer advanced driver assistance systems, dynamic navigation, route planning, location-based services, and other geographic information-based products and services to consumer and commercial users. Its Nokia Siemens Networks segment provides mobile and fixed network solutions and related services to operators and service providers. This segment offers various business solutions, such as consulting and systems integration; network and service management, and charging and billing software; and subscriber database management. It also provides managed services, such as network planning, optimization, and network operations; software and hardware maintenance, proactive, and multi-vendor care, as well as competence development services; and project management, turnkey implementations, and energy efficient sites. In addition, this segment offers fixed and mobile network infrastructure, including Flexi base stations, optical transport systems, and broadband access equipment, as well as network solutions. Nokia Corporation was founded in 1865 and is based in Espoo, Finland."
F[7]	Ford Motor Company primarily develops, manufactures, distributes, and services vehicles and parts worldwide. It operates in two sectors, Automotive and Financial Services. The Automotive sector offers vehicles primarily under the Ford and Lincoln brand names. This sector markets cars, trucks, and parts through retail dealers in North America, and through distributors and dealers outside of North America. It also sells cars and trucks to dealers for sale to fleet customers, including daily rental car companies, commercial fleet customers, leasing companies, and governments. In addition, this sector provides retail customers with a range of after-sale vehicle services and products in the areas, such as maintenance and light repair, heavy repair, collision repair, vehicle accessories, and extended service contracts under the Ford Service, Lincoln Service, Ford Custom Accessories, Ford Extended Service Plan, and Motorcraft brand names. The Financial Services sector offers various automotive financing products to and through

6 http://finance.yahoo.com/q/pr?s=NOK+Profile
7 http://finance.yahoo.com/q/pr?s=F+Profile

	automotive dealers. It offers retail financing, which includes retail installment contracts for new and used vehicles; direct financing leases; wholesale financing products that comprise loans to dealers to finance the purchase of vehicle inventory; loans to dealers to finance working capital, purchase real estate dealership, and/or make improvements to dealership facilities; and other financing products, as well as provides insurance services. Ford Motor Company was founded in 1903 and is based in Dearborn, Michigan.
VOW[8]	"Volkswagen Aktiengesellschaft manufactures automobiles. The company operates in two divisions, Automotive and Financial Services. The Automotive division engages in the development of vehicles and engines, as well as the production and sale of passenger cars, commercial vehicles, trucks and buses, pick-ups, heavy trucks, and parts. Its product range extends from low-consumption small cars to luxury class vehicles. The Financial Services division offers dealer and customer financing, leasing, banking and insurance, and fleet management services. The company offers its products under Volkswagen Passenger Cars, Audi, SEAT, Bentley, Volkswagen Commercial Vehicles, Scania, Skoda, Bugatti, and Lamborghini brand names, as well as services under Volkswagen Financial Services brand name. Volkswagen Aktiengesellschaft sells its products in various markets in Europe, North America, South America, the Asia-Pacific, and internationally. It has strategic alliances with Higer; Suzuki Motor Corporation; Dr. Ing. h.o. F. Porsohe AG; Daimler AG; Chrysler Group; CHOREN Industries; and IOGEN. The company was founded in 1937 and is headquartered in Wolfsburg, Germany."
GM[9]	"General Motors Company (GM) operates as a global automaker. It produces cars and trucks and sells them under the brand names Baojun, Buick, Cadillac, Chevrolet, GMC, Daewoo, Holden, Opel, Isuzu, Vauxhall, Jiefang, FAW, and Wuling. The company sells its cars and trucks to dealers for consumer retail sales, as well as to fleet customers, including daily rental car companies, commercial fleet customers, leasing companies, and governments. Its major markets include China, the United States, Brazil, the United Kingdom, Germany, Canada, and Italy. GM?s OnStar subsidiary provides vehicle safety, security, and information services. In addition, it provides automotive financing services through its subsidiary, General Motors Financial Company, Inc., which purchases automobile finance contracts for new and used vehicles purchased by consumers primarily from franchised and select independent dealerships. GM was founded in 1908 and is headquartered in Detroit, Michigan."
GS[10]	"The Goldman Sachs Group, Inc., together with its subsidiaries, provides investment banking, securities, and investment management services to corporations, financial institutions, governments, and high-net-worth individuals worldwide. Its Investment Banking segment offers financial advisory, including advisory assignments with respect to mergers and acquisitions, divestitures, corporate defense, risk management, restructurings, and spin-offs; and underwriting securities, loans and other financial instruments, and derivative transactions. The company?s Institutional Client Services segment provides client execution activities, such as fixed income, currency, and commodities client execution related to making markets in interest rate products, credit products, mortgages, currencies, and commodities; and equities related to making markets in equity products, as well as commissions and fees from executing and clearing institutional client transactions on stock, options, and futures exchanges worldwide. This segment also engages in the securities services business providing financing, securities lending, and other prime brokerage services to institutional clients, including hedge funds, mutual funds, pension funds, and foundations. Its Investing and Lending segment invests in debt securities, loans, public and private equity securities, real estate, consolidated investment entities, and power generation facilities. This segment also involves in the origination of loans to provide financing to clients. The company?s Investment Management segment provides investment management services and investment products to institutional and individual clients. This segment also offers wealth advisory services, including portfolio management and financial counseling, and brokerage and other transaction services to high-net-worth

[8] http://finance.yahoo.com/q/pr?s=VOW.DE+Profile
[9] http://finance.yahoo.com/q/pr?s=GM+Profile
[10] http://finance.yahoo.com/q/pr?s=GS+Profile

	individuals and families. It also provides global investment research services. The company was founded in 1896 and is headquartered in New York, New York."
BCS[11]	"Barclays PLC provides various financial products and services in Europe, the United States, Africa, and Asia. It offers retail and commercial banking, credit cards, investment banking, wealth management, and investment management services. The company?s products include current account and savings products, Woolwich branded mortgages, unsecured loans, protection products, general insurance, credit cards, Sharia-compliant products, installment finance and commercial property finance, commercial loans, and personal loans. It also offers money transmission, international and private banking, investment management, fiduciary, and brokerage services, as well as payment solutions and mobile banking services. In addition, the company provides fixed income, currency and commodities, foreign exchange, emerging markets, money markets, and credit services; equities, which include cash and equity derivatives and prime services; investment banking products and services that comprise financial advisory, and equity and debt underwriting; and advisory services. It serves individual, commercial, corporate, institutional, retail, and mass affluent customers. The company was formerly known as Barclays Bank Limited and changed its name to Barclays PLC in January 1985. Barclays PLC was founded in 1896 and is headquartered in London, the United Kingdom."
C[12]	"Citigroup, Inc., a global financial services company, provides consumers, corporations, governments, and institutions with a range of financial products and services. The company operates through two segments, Citicorp and Citi Holdings. The Citicorp segment operates as a global bank for businesses and consumers with two primary businesses, Regional Consumer Banking and Institutional Clients Group. The Regional Consumer Banking business provides traditional banking services, including retail banking, and branded cards in North America, Asia, Latin America, Europe, the Middle East, and Africa. The Institutional Clients Group business provides securities and banking services comprising investment banking and advisory services, lending, debt and equity sales and trading, institutional brokerage, foreign exchange, structured products, cash instruments and related derivatives, and private banking; and transaction services consisting of treasury and trade solutions, and securities and fund services. The Citi Holdings segment operates Brokerage and Asset Management, Local Consumer Lending, and Special Asset Pool businesses. The Brokerage and Asset Management Business, through its 49% stake in Morgan Stanley Smith Barney joint venture and Nikko Cordial Securities, offers retail brokerage and asset management services. The Local Consumer Lending business provides residential mortgage loans, retail partner card loans, personal loans, commercial real estate, and other consumer loans, as well as western European cards and retail banking services. The Special Asset Pool business is a portfolio of securities, loans, and other assets. Citigroup Inc. has approximately 200 million customer accounts and operates in approximately 160 countries. The company was founded in 1812 and is based in New York, New York."

11 http://finance.yahoo.com/q/pr?s=BCS+Profile
12 http://finance.yahoo.com/q/pr?s=C+Profile

Appendix B: User Feedback Questionnaires

INITIAL QUESTIONNAIRE

Software tool OntoGen 2.0 provides creation of ontologies and bonds between concepts and therefore has the major role in the process of learning. The aim of user trial is analysis of OntoGen V 2.0 and evaluation of current state of its development by the users.

Participation makes you part of the user trials process taking place in Slovenia and Croatia.

Name: _____
Country: _____
University: _____
Faculty (Major) _____
Age: _____
Gender: ☐ M ☐ F

The questionnaire is filled by placing mark **x,** or by providing the explanation where it is required.

COMPUTER EXPERIENCE

Do you have computer at home?

☐ Yes ☐ No

If YES, what kind?

PC/Mac	☐
Laptop	☐
Palm	☐

According to scale below, assess the frequency of usage of different services and programs on the computer:

0 – never
1 – once a month
2 – once a week
3 – several days in a week
4 – every day

	0	1	2	3	4
E-mail	☐	☐	☐	☐	☐
Chat	☐	☐	☐	☐	☐
Surfing the Web	☐	☐	☐	☐	☐
Searching the Web	☐	☐	☐	☐	☐
eLearning system	☐	☐	☐	☐	☐
Windows	☐	☐	☐	☐	☐
Linux	☐	☐	☐	☐	☐
Mac OSX	☐	☐	☐	☐	☐

How many hours a day (in weekly average) you spend on your computer?

less than 1h	1-2h	3-5h	6-10 h	more than 10h
☐	☐	☐	☐	☐

How many hours a day (in weekly average) you browse Internet at home and/or at work Internet?

less than 1h	1-2h	3-5h	6-10 h	more than 10h
☐	☐	☐	☐	☐

LEARNING HABITS

Do you learn best when (mark everything that applies on you):

- ☐ alone
- ☐ in group
- ☐ depending on situation and need

You learn most efficiently when:

- ☐ reading
- ☐ speaking to others/repeating out loud
- ☐ listening to others
- ☐ through practice
- ☐ enjoying yourself
- ☐ exploring
- ☐ using graphics
- ☐ using multimedia

Do you prefer technological or human interaction during learning?

- ☐ only human lectures
- ☐ 20% technology, 80% human lectures
- ☐ 50% technology, 50% human lectures
- ☐ 80% technology, 20% human lectures
- ☐ only technology

What do you think about using technology in learning:

a) the human interaction is irreplaceable ☐
b) it is good to have the help from both sides ☐
c) it is inevitable to use technology in learning ☐
d) today learning is possible only by using technology ☐

Choose all the processes you recognize in your learning:

Coding ☐
Saving ☐
Recall ☐
Decoding ☐
Perception ☐
Contextualization ☐
Simplifying ☐

Do you know what „**cognitive maps**“ are? YES ☐ NO ☐

If YES, explain.

_____
_____
_____

Is there a method you most often use in learning (mnemotechnics, visualization...)? Explain.

_____
_____
_____

Do you, and in which way, organize your knowledge? Explain.

_____
_____
_____

Do you think that creation, manipulation and visualization of knowledge concepts by software tools can additionally improve your learning?

☐ Yes ☐ No ☐ I don’t know

Do you have any idea about software tool which would make easier creation of relations between concepts and documents which interest you?

_____
_____
_____

Thank you for your collaboration and help!

END QUESTIONNAIRE

Main areas of OntoGen 2.0.

Overall impression (circle the number next the answer that mostly represents your opinion)

1- bad
2- sufficient
3- good
4- very good
5- excellent

Do you consider OntoGen 2.0 useful in organizing data?

Yes ☐ No ☐

Which are, by your opinion, the advantages of OntoGen 2.0?

_____
_____
_____

Which are, by your opinion, the disadvantages of OntoGen 2.0?

_____
_____
_____

Do you consider OntoGen 2.0 an interesting way of organizing data and knowledge? Explain.

_____
_____
_____

On the scale 1-4, where numbers mean:

1- not good at all
2- sufficienty
3- mostly good
4- very good
5- excellent

assess the content presentation of the main areas in OntoGen 2.0.

a) Ontology visualization and document management .....
b) Concept tree .....
c) Concept properties .....

On the scale 1-5 assess which of stated can be applied on the interface of OntoGen 2.0.

a) ease of use …….
b) Understandable ….
c) Meaningful …….
d) Logical …….
e) Attractive …….

Creating and saving ontologies

Is there a clear distinction in understanding of two ways of creating ontologies?
Yes ☐ No ☐

What do you think about Query method?
Advantages:_____
_____
Disadvantages:_____
_____

What do you think about *k*-means method?
Advantages:_____
_____
Disadvantages:_____
_____

Which difficulties are present in generating new concepts by OntoGen 2.0. tool? Explain.
_____
_____
_____

Basic concept management

On scale 1-5 assess the ease of concept management in OntoGen 2.0.
a) Adding …….
b) Deleting …….
c) similarity graph ….

What might, in your opinion, ease the concept management in OntoGen 2.0? Explain.
_____
_____
_____

Does visualization help in choosing sub-concepts?
Yes ☐ No ☐

Can it be improved? Explain.
_____
_____
_____

Does inclusion of new documents (Include option) in already existing concepts can be improved?
Yes ☐ No ☐

Try to think of a way to do it.
_____
_____
_____

Concept suggestion (learning)

Which one of two presented ways (k-means and Query) you like better? Why?

_____
_____
_____

Which are the advantages and disadvantages of "unsupervised" approach?
Advantages:_____
_____
Disadvantages:_____
_____

Which are the advantages and disadvantages of "supervised" approach?
Advantages:_____
_____
Disadvantages:_____
_____

Ontology visualization

Do you consider the current way of ontology visualization suitable for use?

Yes ☐ No ☐

Can it be improved? How? Describe.

_____
_____
_____

Concept document management

On scale 1-5 assess the simplicity of document management within ontologies:

a) Relation management (creating new, changing existing, deleting) …
b) Similarity calculation …….
c) Concept visualization …….
d) Assignment of new instances …….

Which of listed processes can be improved? How? Explain.

_____
_____
_____

Thank you for your collaboration and help!

Appendix C: User Feedback Results

C.1 Initial Questionnaire

C.1.1 Quantitative analysis of Initial questionnaire

All subjects have computers at home; 83,5% have their own *PC*, 39,6% have *laptop* and 9,9% *palm*.

Comparison of usage frequencies of different services and programs: there are statistically significant differences in frequency of usage programs and services, except in Linux and Mac OS X, which are usually not used by students.

Sample (N=91)	1	2	3	4	5	Significance	Mean value
	E-MAIL						
Psychology/Pedagogy	0	1	10	11	21		3,21
Computer sciences	0	0	3	10	35	χ^2 **= 8,066, p< 0,05**	3,67
	CHAT						
Psychology/Pedagogy	32	6	2	2	1		0,47
Computer sciences	20	7	6	9	6	χ^2 **= 12,63, p< 0,05**	1,46
	SURFING THE WEB						
Psychology/Pedagogy	0	4	13	16	10		2,74
Computer sciences	0	1	2	20	25	χ^2 **= 16,51, p< 0,01**	3,44
	SEARCHING THE WEB						
Psychology/Pedagogy	0	0	7	28	8		3,02
Computer sciences	0	1	6	19	22	χ^2 **= 9,08, p< 0,05**	3,29
	e-LEARNING SYSTEM						
Psychology/Pedagogy	34	8	1	0	0		0,23
Computer sciences	22	17	5	2	2	χ^2 **= 12,24, p< 0,05**	0,85
	WINDOWS						
Psychology/Pedagogy	1	1	2	14	25		3,42
Computer sciences	1	0	0	6	41	χ^2 **= 9,83, p< 0,05**	3,79
	LINUX						
Psychology/Pedagogy	40	2	1	0	0		0,09
Computer sciences	35	8	2	01	2	χ^2 = 7,01, p > 0,05	0,48
	MAC OS X						
Psychology/Pedagogy	43	0	0	0	0		0,00
Computer sciences	46	1	0	0	1	χ^2 = 1,83, p > 0,05	0,10

Time spent by the computer: There are differences between students in overall number of hours spent by the computer, abut there are no differences in number of hours spent in searching the Web.

Sample (N=91)	0-1 h	1-2 h	3-5 h	6-10 h	10 h or more	Significance	Mean value
	No. OF HOURS SPENT BY THE COMPUTER						
Psychology/ Pedagogy	0	26	13	1	3		2,56
Computer sciences	1	7	25	11	4	**χ^2 = 24,00 p< 0,01**	3,21
	No. OF HOURS SPENT SERACHING THE WEB						
Psychology/ Pedagogy	13	21	7	2	0		1,95
Computer sciences	6	23	15	3	1	χ^2 = 6,52 p > 0,05	2,38

Learning habits

Sample (N=91)		Frequency	**%**
Psychology/Pedagogy	ALONE	24	55,8
	IN A GROUP	0	0
	DEPENDING ON THE SITUATION AND NEED	19	44,2
Computer sciences	ALONE	18	37,5
	IN A GROUP	4	8,3
	DEPENDING ON THE SITUATION AND NEED	26	54,2

How do you learn best?

Sample (N=91)	Psychology/Pedagogy	Informational science
	Frequency (N=43)	Frequency (N=48)
reading	28	33
speaking to others/repeating out loud	23	19
listening to others	9	18
through practice	25	32
enjoying yourself	13	13
exploring	23	25
using graphics	21	24
using multimedia	9	16

Preferred way of knowledge transfer

	Knowledge transfer				Significance
Sample (N=91)	Only lectures	20% technology 80% lectures	50-50%	80% technology 20% lectures	
Psychology/ Pedagogy	1	12	28	2	
Computer sciences	1	11	30	6	χ2 = 1,84, p > 0,05

Opinion about using technology in Learning

Sample (N=91)	human interaction is irreplaceable	it is good to have the help from both sides	it is inevitable to use technology in learning	today learning is possible only by using technology	Significance
Psychology/ Pedagogy	0	20	22	1	
Computer sciences	2	17	27	2	$\chi 2 = 2,82$ $p > 0,05$

Learning processes

Sample (N=91)	Psychology/Pedagogy	Informational science
	Frequency (N=43)	Frequency (N=48)
Coding	26	17
Saving	36	35
Recalling	39	35
Decoding	12	17
Perception	26	30
Contextualization	28	29
Simplifying	39	41

Understanding of the term “Cognitive maps”

Sample (N=91)	Understanding “cognitive maps”		Significance
	Yes	No	
Psychology/Pedagogy	38	5	
Computer sciences	11	36	$\chi 2 = 38,21$, $p< 0,01$

Improving learning by computer tools

Sample (N=91)	Yes	No	I don’t know	Significance
Psychology/Pedagogy	28	1	14	
Computer sciences	30	1	17	$\chi 2 = 0,08$, $p > 0,05$

There is no difference in student's opinion about improving learning processes using computer tools. Both, Psychology/Pedagogy students and Informatics sciences students in most cases consider creation, manipulation and visualization of concepts by computer tools can additionally enhance learning.

C.1.2 Qualitative analysis of Initial questionnaire

Computer sciences students - analysis of qualitative data concerning their opinions on “Cognitive maps”:

- I don't know [68%]
- Cognitive structures [10%]
- Graphical layout of knowledge [8%]
- Linked data maps [12%]
- Other [2%]

Computer sciences students - analysis of qualitative data representing their ideas about computer tool that helps improving learning process:

- No idea [76%]
- Other [24%]

Computer sciences students - analysis of qualitative data representing learning methods students most frequently use:

- I don't know [20%]
- Visualization [39%]
- Mnemotechnics [15%]
- Associations, connections to previous knowledge [22%]
- Other [22%]

Computer sciences students - analysis of qualitative data representing their ways of organizing knowledge:

- Categorization simplification [12%]
- I don't know how [32%]
- No organization [17%]
- Contextualization, connection [22%]
- Other [17%]

Psychology/Pedagogy students- analysis of qualitative data concerning their opinions on “Cognitive maps”:

- Mental schemes of memory [32%]
- Mental image of space [29%]
- Mental data layout [10%]
- Organized knowledge systems [27%]
- Other [2%]

Psychology/Pedagogy students - analysis of qualitative data representing learning methods students most frequently use:

- Visualization [35%]
- Associations, connections to previous knowledge [21%]
- Other [15%]

Psychology/Pedagogy students - analysis of qualitative data representing their ways of organizing knowledge:

- Cathegorization simplification [38%]
- I don't know how [9%]
- No organization [18%]
- Contextualization, connection [35%]

Psychology/Pedagogy students - analysis of qualitative data representing their ideas about computer tool that helps improving learning process:

- No idea [59%]
- Visualization software [18%]
- Linkage of terms by similarity [12%]
- Other [24%]

C.2 End Questionnaire

C.2.1 Quantitative analysis of End questionnaire

General impression

	General impression					
Sample (N=91)	Sufficient	Good	Very good	Excellent	Significance	Mean value
Psychology/Pedagogy	3	20	12	6		3,51
Computer science	9	27	9	2	$\chi 2 = 6,09$ $p > 0,05$	3,09

Assessment of software usefulness

Sample (N=91)	Usefulness		
	Yes	No	Significance
Psychology/Pedagogy	40	1	
Computer science	45	3	$\chi 2 = 0,75,\ p > 0,05$

Content presentation assessment of the main areas of OntoGen 2.0

Sample (N=91)	not good at all	sufficient	mostly good	very good	excellent	Significance	Mean value
	Ontology Visualization and document management						
Psychology/Pedagogy	0	11	22	6	0		2,87
Computer science	3	13	18	8	1	$\chi 2 = 4,66$ $p > 0,05$	2,79
	Concept tree						
Psychology/Pedagogy	0	6	20	12	1		3,21
Computer science	1	7	21	12	2	$\chi 2 = 1,24$ $p > 0,05$	3,16
	Concept properties						
Psychology/Pedagogy	0	8	22	8	1		3,05
Computer science	5	5	23	8	1	$\chi 2 = 5,61$ $p > 0,05$	2,88

Assessment of graphical interface

	not good at all	sufficient	mostly good	very good	excellent	Significance	Mean value
Sample (N=91)	Easy to use						
Psychology/Pedagogy	0	7	11	12	3		3,33
Computer science	1	6	10	18	3	$\chi^2 = 1,98$ $p > 0,05$	3,42
	Understandable						
Psychology/Pedagogy	0	7	12	12	2		3,27
Computer science	0	7	17	13	1	$\chi^2 = 0,89$ $p > 0,05$	3,21
	Meaningful						
Psychology/Pedagogy	0	1	14	15	3		3,61
Computer science	1	4	15	12	6	$\chi^2 = 3,83$ $p > 0,05$	3,47
	Logical						
Psychology/Pedagogy	2	3	13	13	2		3,30
Computer science	2	3	12	18	4	$\chi^2 = 1,02$ $p > 0,05$	3,49
	Attractive						
Psychology/Pedagogy	4	12	10	6	1		2,64
Computer science	3	12	13	8	2	$\chi^2 = 5,61$ $p > 0,05$	2,84

Distinction between the two ways of creating ontology

Sample (N=91)	Clear distinction		
	Yes	No	Significance
Psychology/ Pedagogy	34	7	
Computer science	34	12	$\chi^2 = 1,03,\ p > 0,05$

Basic concept management

Sample (N=91)	1	2	3	4	5	Significance	Mean value
	Adding						
Psychology/Pedagogy	0	1	9	12	18		4,18
Computer science	0	4	6	11	23	$\chi^2 = 2,87,\ p > 0,05$	4,20
	Deleting						
Psychology/Pedagogy	0	0	7	15	18		4,27
Computer science	0	2	6	13	23	$\chi^2 = 2,64,\ p > 0,05$	4,30
	Similarity graph						
Psychology/Pedagogy	0	5	15	12	7		3,54
Computer science	1	8	12	18	4	$\chi^2 = 3,85,\ p > 0,05$	3,37

Does visualization help in choosing sub-concepts?

Sample (N=91)	Yes	No	Significance
Psychology/Pedagogy	40	1	
Computer science	42	5	$\chi 2 = 2{,}31,\ p > 0{,}05$

Can we improve inclusion of new documents (option Include) in existing ontology (concepts)?

	Yes	No	Significance
Psychology/Pedagogy	20	21	
Computer science	21	18	$\chi 2 = 0{,}21,\ p > 0{,}05$

Suitability of current ontology visualization

Sample (N=91)	Yes	No	Significance
Psychology/ Pedagogy	36	5	
Computer Science	38	8	$\chi 2 = 0{,}46,\ p > 0{,}05$

Concept document management

	1	2	3	4	5	Significance	Mean value
	Relation management						
Psychology/Pedagogy	0	3	10	13	13		3,92
Computer science	0	5	6	18	15	$\chi 2 = 2{,}12,\ p > 0{,}05$	3,98
	Similarity calculation						
Psychology/Pedagogy	0	5	13	12	7		3,57
Computer science	0	7	14	15	7	$\chi 2 = 0{,}25,\ p > 0{,}05$	3,51
	Concept visualization						
Psychology/Pedagogy	0	3	14	17	5		3,62
Computer science	2	6	11	18	6	$\chi 2 = 3{,}23,\ p > 0{,}05$	3,47
	Assignment of new instances						
Psychology/Pedagogy	0	5	12	16	6		3,59
Computer science	0	5	7	24	8	$\chi 2 = 2{,}91,\ p > 0{,}05$	3,80

C.2.2 Qualitative analysis of End questionnaire

Because of the large amount of suggestions, it wasn't so easy to determine categories and the percentages of users, as it was done in the Initial questionnaire. But, since all the information is extremely valuable, it was decided to point out the suggestions divided into large categories of answers (where possible) in order to sort out the different ideas coming from the end users. It was also noticed in the early stage of analysis that data coming from Computer sciences students and Psychology and Pedagogy students was quite similar so it made sense for all the data to be brought together and represented in that manner.

Analysis of qualitative data representing advantages of OntoGen 2.0:

- Managing large data bases
- Saves time & effort
- Efficient, possibility of intervention
- Organizes, searches, easy to handle
- Other

Analysis of qualitative data representing disadvantages of OntoGen 2.0:

- Have to learn how to use it
- Not so attractive
- Abstract
- Not connected to Internet
- Occasionally slow
- I don't know

Analysis of qualitative data representing advantages and disadvantages of Query method:

- Connecting data from a different view
- Not appropriate if you don't have any idea
- Spending lot of time reading
- Large amount of data can mislead
- If you know what to look for
- Our choice

Analysis of qualitative data representing advantages and disadvantages of K-means method:

- If it's a large field of interest
- Saves time, useful and practical
- Better graphical layout
- Don't know what key words represent
- Less control over organization
- Large amount of data we don't need
- Key words ease the choice
- System suggests the data

Analysis of qualitative data representing perceived difficulties in generating new concepts:

- Didn't notice any difficulties
- Lack of detailed instructions
- How to connect new data with existing
- All the options in one place
- I don’t know

Analysis of qualitative data representing ideas about Basic concept management improvement:

- Usage of mouse
- Simplifying the approach to large number of data
- Better computer knowledge
- No improvement needed
- I don't know

Analysis of qualitative data representing ideas about Concept visualization improvement:

- I don't know how
- More colours
- Better layout
- Key words can hardly be seen
- Different kind of organization
- Analysis of qualitative data representing ideas about Include option improvement:
- Manual placing of new documents
- I don’t know how

Analysis of qualitative data representing advantages and disadvantages of unsupervised method:

- The ideas are given to users
- System suggests the data
- Easier, faster, simpler
- Some amount of useless data
- Some data isn't available
- I don't know what I'll get

- No own ideas
- I don’t know

Analysis of qualitative data representing advantages and disadvantages of supervised method:

- Autonomy
- We choose the data
- Bigger sense of control
- Some previous knowledge needed
- Don’t know if my choices are right
- Large amount of data
- Slow
- I don’t know

Analysis of qualitative data representing Visualization of ontologies improvements:

- Unnecessary
- Continue search by mouse click
- With adding picture becomes cluttered
- I don’t know how

Appendix D: Publications Related to this Thesis

Journal papers:

Grobelnik, M.; Mladenić, D.; Fortuna, B. Semantic Technology for Capturing Communication Inside an Organization. *IEEE Internet Computing* **13**, 4 (2009).

Fortuna, B.; Mladenić, D.; Grobelnik, M. Visualization of text document corpus. *Informatica* **29** (2006).

Stajner, T.; Rusu, D.; Dali, L,; Fortuna, B.; Mladenić, D.; Grobelnik, M. A service oriented framework for natural language text enrichment. *Informatica* **34**, 3 (2010).

Book chapters:

Fortuna, B.; Cristianini, N.; Shawe-Taylor, J. A Kernel Canonical Correlation Analysis For Learning The Semantics Of Text. In: Campos-Valls et al (ed.) *Kernel methods in bioengineering, communications and image processing* (IGI Global, 2006).

Fortuna, B.; Mladenić, D.; Grobelnik, M. Visualization of Temporal Semantic Spaces. In: Davies, J. et al (ed.) *Semantic Knowledge Management* (Springer, 2008).

Conference and workshop papers:

Grobelnik, M.; Brank, J.; Mladenić, D.; Novak, B.; Fortuna, B. Using DMoz for constructing ontology from data stream. In: *28th International Conference on Information Technology Interfaces* (Croatia, 2006).

Fortuna, B.; Grobelnik, M.; Mladenić, D. Background Knowledge for Ontology Construction. In: *Proceedings of the 15th international conference on World Wide Web* (Edinburgh, Scotland, 2006).

Fortuna, B.; Mladenić, D.; Grobelnik, M. Semi-automatic Construction of Topic Ontologies. *Semantics, Web and Mining. Lecture Notes in Computer Science* **4289** (Springer, 2006).

Fortuna, B.; Grobelnik, M.; Mladenić, D. OntoGen: Semi-automatic Ontology Editor. In: *Human interface and the management of information. Interacting in information environments. Lecture Notes in Computer Science* **4558** (Springer, 2007).

Grobelnik, M.; Brank, J.; Fortuna, B.; Mozetic, I. Contextualizing ontologies with ontolight: a pragmatic approach. In: *Proceedings of the 10th International multi-conference Information Society* (Slovenia, 2007).

Fortuna, B.; Fortuna, C.; Mladenić, D. Real-time News Recommender System. In: *Machine Learning and Knowledge Discovery in Databases. Lecture Notes in Computer Science* **6323** (2010).

Fortuna, B.; Mladenić, D.; Grobelnik, M. Application of semantic annotations to predicting users' demographics. In: *Proceeding of the third workshop on Exploiting semantic annotations in information retrieval* (2010).

Fortuna, B.; Mladenić, D.; Grobelnik, M. User Modeling Combining Access Logs, Page Content and Semantics. In: *Proceeding of 1st International Workshop on Usage Analysis and the Web of Data in the 20th International World Wide Web Conference* (India, 2011).

Tomašev, N.; Fortuna, B.; Mladenić. D. OntoGen Extension for Exploring Image Collections. IEEE International Conference on Intelligent Computer Communication and Processing (Romania, 2011).

Ilijasic Misic, I.; Kovacic, B.; Mohoric, T.; Mladenić, D.; Fortuna, B.; Grobelnik, M. User Study of Ontology Generation Tool. In: *29th International Conference on Information Technology Interfaces* (Croatia, 2007).

Appendix E: Biography

Blaž Fortuna was born in Kranj, Slovenia, on December 30, 1982.

He completed the Bachelor of Science degree in applied mathematics at the Faculty of Mathematics and Physics, University of Ljubljana, in 2006. Afterwards, he enrolled at the PhD programme New Media and E-science at the Jožef Stefan International Postgraduate School.

During his PhD studies he was employed as a research assistant at Department of Knowledge Technologies and Artificial Intelligence Laboratory, at Jožef Stefan Institute, Ljubljana, Slovenia. He was a teaching assistant at Jožef Stefan International Postgraduate School and interned at Microsoft Research, Cambridge, UK, and at Bloomberg Inc, New York, USA. He collaborated on EU funded projects SEKT (SEmantic Knowledge Technologies), SMART (Statistical Multilingual Analysis for Retrieval and Translation), NEON, RENDER and PASCAL (Pattern Analysis, Statistical Modelling and Computational Learning).