U N I V E R Z A V M A R I B O R U

Fakulteta za elektrotehniko, računalništvo in informatiko

Doktorska disertacija

METODA MEJNIH PAROV ZA UČENJE UMETNIH NEVRONSKIH MREŽ

Avtor: mag. Bojan PLOJ

Maribor, julij 2013

Mentor: prof. dr. Milan ZORMAN

Univerza v Mariboru

Maribor, 9. 7. 2012
Številka: DR 158/2012/426-DL

Na osnovi 287., 140., 142. in 144. člena Statuta Univerze v Mariboru (Statut UM-UPB10, Ur. l. RS, št. 46/2012) ter sklepa 12. dopisne seje Senata Univerze v Mariboru, ki je potekala od 4. do 6. 7. 2012 v zvezi z vlogo doktorskega kandidata mag. Bojana Ploja za sprejem odločitve o predlagani temi doktorske disertacije in mentorja

izdajam naslednji

SKLEP

Odobri se tema doktorske disertacije mag. Bojana Ploja s Fakultete za elektrotehniko, računalništvo in informatiko z naslovom »Metoda mejnih parov za učenje umetnih nevronskih mrež«. Za mentorja se imenuje red. prof. dr. Milan Zorman. Kandidat mora članici predložiti izdelano doktorsko disertacijo v zadostnih izvodih najpozneje do 5. 7. 2016.

Obrazložitev:

Kandidat mag. Bojan Ploj je dne 3. 1. 2012 na Fakulteti za elektrotehniko, računalništvo in informatiko vložil vlogo za potrditev teme doktorske disertacije z naslovom »Metoda mejnih parov za učenje umetnih nevronskih mrež«. Za mentorja je bil predlagan red. prof. dr. Milan Zorman.

Senat Fakultete za elektrotehniko, računalništvo in informatiko je na osnovi pozitivnega mnenja komisije za oceno teme doktorske disertacije, ki je ugotovila, da kandidat izpolnjuje pogoje za pridobitev doktorata znanosti, in ocenila, da je predlagana tema ustrezna, sprejel pozitivno mnenje in poslal predlog teme doktorske disertacije s predlogom mentorja v odobritev senatu univerze.

Senat Univerze v Mariboru je po proučitvi vloge in na osnovi določil Statuta Univerze v Mariboru sprejel svojo odločitev o predlagani temi doktorske disertacije in imenoval mentorja, kot izhaja iz izreka.

V skladu s 144. členom Statuta Univerze v Mariboru mora kandidat za pridobitev doktorata znanosti najpozneje v štirih letih od dneva izdaje tega sklepa, članici predložiti izdelano doktorsko disertacijo v zadostnih izvodih. Kandidatu je bil določen rok glede na datum sprejetja teme na pristojnem organu.

Pouk o pravnem sredstvu:
Zoper ta sklep je možna pritožba na Senat Univerze v Mariboru v roku 8 dni od prejema tega sklepa.

UNIVERZA V MARIBORU

Rektor:
Prof. dr. Danijel Rebolj

po pooblastilu rektorja
prorektorica za finančne zadeve
prof. dr. Zdravko Kačič

Obvestiti:
1. Kandidata.
2. Fakulteto.
3. Arhiv.

HR EXCELLENCE IN RESEA

Avtor: mag. Bojan PLOJ

Naslov: **Metoda mejnih parov za učenje umetnih nevronskih mrež**

UDK: 004.032.26:004.8(043.2)

Ključne besede: umetna inteligenca, strojno učenje, nevronske mreže, algoritem, večslojni perceptron, metoda mejnih parov

Število izvodov: 10

Zahvale

Hvala mentorju, prof. dr. Milanu Zormanu, za nesebično pomoč pri radostih odkrivanja novega.

Hvala mami Pavli in očetu Stanislavu za spodbudo in podporo.

Hvala sinu Marku in partnerici Valeriji za razumevanje.

Hvala prijateljici Brigiti za jezikoslovno pomoč.

Hvala vsem, ki vas imam rad in ste mi dajali energijo za znanstveno-raziskovalno delo.

Metoda mejnih parov za učenje umetnih nevronskih mrež

UDK: 004.032.26:004.8(043.2)

Ključne besede: umetna inteligenca, strojno učenje, konstruktivna nevronska mreža, algoritem, večslojni perceptron, metoda mejnih parov

Povzetek: Disertacija opisuje novo metodo strojnega učenja - metodo mejnih parov. Opisana metoda je namenjena učenju večslojnega perceptrona, nevronske mreže s povezavami naprej, ki služi za prepoznavanje oziroma razvrščanje v razrede. Uvodoma je opisana problematika strojnega učenja, nevronska mreža večslojni perceptron (MLP) in njena klasična učna metoda backpropagation s poudarkom na njenih slabostih. V jedru disertacije najprej analiziramo lastnosti naučenega MLP. Pri tem se osredotočimo na učne vzorce v bližini meje in definiramo pojem mejnega para. Sledi analiza lastnosti mejnih parov, ki je podlaga za novo metodo razšumljanja, za novo metodo rojenja, in novo metodo konstruktivnega učenja. To učenje je lahko statično (offline), inkrementalno, dinamično (online) in s postopnim pozabljanjem starih učnih podatkov. Razšumljanje, rojenje in učenje smo testirali s simulacijo na računalniku. V ta namen smo uporabljali uveljavljene nabore učnih podatkov (zaradi primerljivosti rezultatov), realne nabore učnih podatkov (kot dokaz uporabnosti), kot tudi umetne (zaradi prilagajanja učnih podatkov našim potrebam).

Primerjalna analiza je pokazala, da ima metoda mejnih parov nekaj dobrih lastnosti. Z njo smo uspešno razšumljali in rojili podatke, iskali značilke ter razvrščali podatke. Rezultati raziskav kažejo, da je obravnavana metoda zanesljiva, natančna, konstruktivna in odporna na šum in prekomerno učenje.

Border Pairs Method for the Artificial Neural Network learning

UDK: 004.032.26:004.8(043.2)

Key words: artificial intelligence, machine learning, constructive neural network, algorithm, multilayer perceptron, border pairs method

Abstract: Thesis describes a new method of machine learning - Border Pairs Method. The method described is intended for teaching Multi-Layer Perceptron - Feed-Forwarded Artificial Neural Network which is used for recognition or classification. Initially the problems of machine learning are described, as well as a description of the Multi-Layer Perceptron Neural Network and a description of Backpropagation, of its classical teaching method with an emphasis on its weaknesses. Firstly, in the core of the thesis we analyze the properties of trained MLP. In doing so, we focus on the learning patterns near the border and we define the concept of the Border Pair. Furthermore, the analysis of the properties of the border pairs, which is the basis of a new method of the noise reduction, of the clustering and a new method of constructive learning follow. This learning can be offline, incremental, online and with gradually forgetting the old learning patterns (unlearning). Noise cancellation, clustering and learning have been tested by simulation on a computer. Towards the end, we used the established sets of learning data (thanks to the comparability of results), real learning data sets (as a proof of the usability) as well as synthetic ones (due to the adaptation of learning data to our needs).

The comparative analysis showed that the Border Pairs Method has some good characteristics. With it we have successfully reduced the noise, clustered the data, searched the features and classify the data. Results show that the researched method is reliable, accurate, constructive and insensitive to noise and overfitting.

Kazalo vsebine

Kazalo slik

Kazalo tabel

Kazalo algoritmov

Simboli

w_i – utež i-te sinapse

w_0 – prag nevrona

o_j – izhodna vrednost nevrona

x – Vhodna vrednost nevrona

y – vhodna vrednost nevrona

R_i – vhodna matrika evklidskih razdalj

R_n – notranja matrika evklidskih razdalj

R_o – izhodna matrika evklidskih razdalj

Okrajšave

AI – umetna inteligenca, artificial intelligence

ANN – umetna nevronska mreža, artificial neural network

BP – backpropagation

BPM – medota mejnih parov, border pairs method

DT – odočitveno drevo, decision tree

EP – evolucijsko programiranje, evolutionary programming

FFNN – nevronska mreža s povezavami naprej feedforward neural network

GP – genetsko programiranje, genetic programming

HMM – skriti model Markova, hiden Markov model

ILP – induktivno logično programiranje, inductiv logic programming

LTU – linearna pragovna enota, linear treshold unit

MLP – večslojni perceptron, multi layer perceptron

NN – nevronska mreža, neural network

RI – razdalje med vzorci istega razreda

RMSE – srednja kvadratna napaka, root mean square error

RNN – vzvratna nevronska mreže, recurrent neural network

RR – razdalje med vzorci različnih razredov

RTU – krogelna pragovna enota, radial treshold unit

SVM – metoda podpornih vektorjev, support vector machine

1 Uvod

1.1 Umetna inteligenca

Tema te disertacije sodi na področje umetne inteligence. Zato je prav, da se uvodoma seznanimo s tem področjem. Začnimo z razlago osnovnih pojmov. Beseda inteligenca pomeni nadarjenost za umske dejavnosti, bistroumnost [1], ki se pripisuje živim bitjem. Umetna inteligenca (angleško *Artificial Intelligence, AI*) je mlada znanstvena veda, katera proučuje inteligentne naprave [2,43]. To so naprave , ki se samostojno odločajo. Običajno te odločitve temeljijo na izkušnjah iz preteklosti. Za mlade vede je značilno, da se izrazi in definicije na njenem področju šele razvijajo. To velja tudi za umetno inteligenco, zato zanjo najdemo številne različne definicije. Najpogostejše inteligentne naprave so pravzaprav običajni računalniki, ki imajo ustrezno programsko opremo. Manj razširjene, a ne manj pomembne, so naprave oziroma strojna oprema, ki so že v osnovi izdelane za področje umetne inteligence. Primer tovrstne strojne opreme so nevronske mreže. Učenje slednjih je osrednja tema te disertacije.

Za začetek umetne inteligence, kot veje znanosti, šteje znanstvena konferenca v Dartmouth-u (ZDA) leta 1956. Tam je gospod John McCarthy skoval izraz »umetna inteligenca« [12]. Njegova definicija AI se v originalu glasi: "the science and engineering of making intelligent machines". Znanstveniki so na konferenci razpravljali, ali je človeško modrost možno opisati tako podrobno, da bi jo lahko posnemal stroj. Razprava je potekala na filozofskem nivoju, kot je to običajno za komaj rojeno znanost.

Čeprav je AI mlada veda, njeni idejni začetki segajo daleč nazaj pred leto 1956 [13]. Omenimo le nekaj bolj znanih stvaritev na tem področju:

- Prvi znani začetki segajo vsaj v antiko. Antična mitologija pripoveduje o kovinskih bitjih kot sta Talos s Krete [44] in Hefajst [45].
- V devetnajstem stoletju je v Angliji izšel »Frankenstein«, klasični roman pisateljice Mary Shelley [46]. V glavni vlogi romana nastopa bitje, ki ga kirurg sestavi iz več mrtvih trupel.
- Leta 1921 so uprizorili premiero gledališke igre R.U.R., izpod peresa Karla Čapke, kjer je bila prvič uporabljena beseda »robot« [47]. To je kovinsko, človeku podobno bitje v vlogi univerzalnega, ubogljivega in marljivega delavca.

Zgodbe in verovanja omenjenih literarnih junakov so prežete z upanjem, strahom ,etičnimi pomisleki in te teme so v sodobni umetni inteligenci še vedno zelo aktualne.

Umetna inteligenca se deli na več različnih področij [14]. Naštejmo nekaj pomembnejših med njimi:

- **Dedukcija** (angleško *deduction*), **sklepanje** (angleško *reasoning*), **reševanje problemov** (angleško *problem solving*)
- **Predstavitev znanja** (angleško *knowledge representation*)
- **Samodejno načrtovanje in razporejanje** (angleško *automated planning and scheduling*)
- **Strojno učenje** (angleško *machine learning*)
- **Obdelava naravnega jezika** (angleško *natural language processing*)
- **Gibanje** (angleško *motion*) in **rokovanje** (angleško *manipulation*)
- **Strojno zaznavanje** (angleško *machine perception*)

- **Ustvarjalnost** (angleško *creativity*)
- **Socialna inteligenca** (angleško *social intelligence*)
- **Splošna inteligenca** (angleško *general intelligence*)

Meje med temi področji niso strogo določene, saj se področja med seboj nekoliko prekrivajo. V nadaljevanju disertacije se bomo predvsem posvečali tematikam s področja strojnega učenja.

1.2 Strojno učenje

Učenje je, podobno kot inteligenca, širok pojem, ki ga ni lahko natančno opisati. Definicija v slovarju [1] pravi: »Učenje je pridobivanje znanja, razumevanja ali sposobnosti s preučevanjem, poučevanjem ali vadbo. Strojno učenje [48, 49] ima kar nekaj podobnosti z biološkim (ljudskim) učenjem.«

Na splošno lahko rečemo, da se med strojnim učenjem spreminja zgradba stroja, njegov program ali njegovi podatki tako, da se njegovo delovanje izboljšuje. Na primer stroju za prepoznavo govora izboljšujemo delovanje s »predvajanjem« dodatnih govornih učnih vzorcev, ki pripadajo različnim govorcem (stari/mladi, moški/ženske,…).

Lahko se vprašamo po smislu zakaj sploh bi naj stroj učili? Mar ni bolje znanja vgraditi že med načrtovanjem in izdelavo stroja? Včasih je res bolje znanje vgraditi, v nekaterih drugih okoliščinah pa je bolj smiselno uporabiti strojno učenje. To velja predvsem v naslednjih okoliščinah:

- Kadar težko ali pa sploh ne moremo natančno opisati povezave med vhodnimi podatki (vzroki) in izhodnimi podatki (posledicami) drugače kot z množico primerov.
- Kadar je za iskanje vhodno izhodnih povezav potrebno predelati ogromne količine podatkov.

- Kadar ljudje ne najdemo optimalnih rešitev za dan problem.
- Kadar se okoliščine spreminjajo in se naprava s strojnim učenjem lahko samodejno prilagaja novim razmeram.

Kot zanimivost povejmo še to, da z učenjem strojev pridobivamo nova spoznanja, ki veljajo tudi za biološko učenje. Pri strojnem učenju so namreč razmere ponovljive in lahko spreminjamo samo en učni parameter naenkrat, kar pri biološkem učenju (skoraj) ni možno.

Ena od bistvenih lastnosti pri učenju je **generalizacija** [50] oziroma posploševanje znanja. To pomeni, da med učenjem najdemo v učnih podatkih neke splošne zakonitosti. Upoštevanje teh zakonitosti nam omogoča, da pravilno sklepamo tudi takrat, ko so obravnavani podatki nekoliko drugačni od tistih, ki smo jih uporabili med učenjem. Pri biološkem učenju to imenujemo učenje z razumevanjem.

Glede nadzora obstajajo tri vrste strojnega učenja: nenadzorovano (angleško *unsupervised*), okrepčevalno (angleško *reinforcement*) in nadzorovano (angleško *supervised*) učenje.

1.2.1 Nenadzorovano učenje

Pri takšnem učenju so podani le vhodni vzorci, ne pa tudi pripadajoče želene izhodne vrednosti [15, 51, 52]. Naprava, ki se uči, sama išče podobne vhodne vzorce in jih po svojih kriterijih razvrstiti v več razredov. Rezultat nenadzorovanega učenja je nepredvidljiv, kar pri strojnem učenju običajno ni zaželeno. Oglejmo si primer nenadzorovanega strojnega učenja - razvrščanje fotografij ljudi. Stroj razvršča fotografije v več razredov (kategorij), ki jih ustvari po svoje. Ne vemo katere skupne značilnosti bodo imele fotografije posameznega razreda, niti koliko razredov bo med učenjem nastalo. Lahko se zgodi da bo stroj ločil ljudi na fotografijah po velikosti, po barvi las, po spolu ali katerikoli drugi lastnosti. Prav tako ne vemo v koliko razredov bo stroj razdelil ljudi. Samo na male in velike? Morda bo vmes naredil tudi en ali več srednjih razredov. Težave z

nepredvidljivostjo rezultatov učenja lahko odpravi selekcija (genetski algoritmi). Živa bitja se pogosto poslužujemo nenadzorovanega učenja v povezavi z naravno selekcijo.

1.2.2 Okrepčevalno učenje

Pri okrepčevalnem učenju naprava dobiva informacije o pravilnosti odločitev (želene vrednosti so podane), vendar šele po zaključenem zaporedju odločitev [16, 53]. Ta način učenja spominja na ljudsko modrost »Po bitki smo vsi generali«. Želene vrednosti niso znane za vsak posamezen učni vzorec, ampak za njihovo zaključeno zaporedje. Na primer pri šahovski igri lahko šele na koncu zanesljivo ugotovimo kako uspešne so bile šahistove poteze. Odvisno od tega ali je zmagal, igral neodločeno ali izgubil. Med igro ne moremo povsem natančno vrednotiti posameznih potez. Včasih se splača šahistu žrtvovati figuro, kar se mu lahko obrestuje nekaj potez kasneje. Zato je smiselno, da učenje šaha poteka na nivoju celotnih šahovskih partij in ne na nivoju posameznih potez. Intuitivno lahko sklepamo, da je okrepčevalno učenje bolj učinkovito od nenadzorovanega, saj imamo med učenjem vsaj nekaj povratnih informacij o tem kako uspešni smo.

1.2.3 Nadzorovano učenje

Pri nadzorovanem učenju vsebujejo učni podatki vhodne vrednosti in njihove pripadajoče želene izhodne vrednosti za vsak posamezen učni vzorec [17, 54]. Tudi učenje poteka ločeno za vsak učni vzorec posebej. Zaporedje vzorcev med učenjem običajno ni pomembno. Za pojasnilo uporabimo isti primer kot pri nenadzorovanem učenju - fotografije ljudi. Sedaj velja, da že pred pričetkom učenja določimo koliko je razredov in v kateri razred sodi posamezna fotografija. Naprava med učenjem išče skupne značilnosti posameznih pripadnikov različnih razredov (generalizacija). Če je učenje uspešno, potem stroj po končanem učenju pravilno razvrsti tudi fotografije ljudi, ki jih med učenjem še ni »videl«.

1.3 Znani predstavniki strojnega učenja

Obstaja več naprav, strojev oziroma pristopov za strojno učenje. Najbolj znani med njimi so:

1.3.1 Odločitveno drevo

Odločitveno drevo (angleško *Decision Tree, DT*) je matematični model za odločanje [18, 55]. Odločitveno drevo je primerno za razreševanje problemov na področju regresije in razvrščanja. V učnih podatkih najprej poiščemo pravila za odločanje. Ko so pravila znana je odločanje preprosto in hitro. Graf odločitev se veji in je zato po videzu podoben drevesu. Od tod tudi izvira njegovo ime »odločitveno drevo«. Preprost primer uporabe odločitvenega drevesa je lahko odločanje ali bomo šli na sprehod. Odločamo se na podlagi znanih podatkov o temperaturi, vetru, vlagi in oblačnosti. Če so te vrednosti primerne gremo na sprehod, sicer pa ne. Ugotovitev kakšna temperatura, veter, vlaga in oblačnost, oziroma katere njihove kombinacije so primerne za sprehod, izluščimo iz učnih podatkov v fazi učenja.

1.3.2 Asociativna analiza

Asociativna analiza (angleško *Association Rules*) je matematično analitična metoda, za ugotavljanje povezanosti med atributi učnih vzorcev [19, 56]. Na primer trgovci lahko z asociativno analizo ugotavljajo kašne navade imajo kupci. Tipičen rezultat takšne analize bi se lahko glasil: »Kdor kupi oglje in zrezke, običajno kupi tudi pivo«.

1.3.3 Nevronska mreža

Nevronska mreža (angleško *Neural Network, NN*) je naprava narejena po vzoru bioloških možganov [20, 57]. Pred uporabo se nevronska mreža najprej uči s primeri iz preteklosti. Ker se uči samostojno, lahko njeno znanje tudi preseže

učiteljevo. Po končanem učenju se nevronska mreža uporablja kot črna škatla. Uporabniku ni treba ničesar vedeti o dogajanju znotraj nje. Tudi če bi želeli vedeti, kako se je nevronska mreža odločala, tega žal ne moremo ugotoviti. Notranje vrednosti v nevronski mreži, oziroma delni rezultati za nas nimajo nobenega smisla. Tako je vsaj veljalo do sedaj.

1.3.4 Genetsko in evolucijsko programiranje

Genetsko programiranje [21, 58] (angleško *Genetic programming, GP*) in evolucijsko programiranje [22, 59] (angleško *Evolutionary programming, EP*) sta algoritma za optimizacijo po vzoru biološke evolucije. S križanjem lastnosti staršev in mutacijo dobimo novo generacijo primerkov. Potomci so tako hkrati podobni staršema in imajo tudi nekaj izvirnih lastosti. Nad populacijo potomcev se nato izvrši selekcija, ki izloči najslabše primerke potomcev, katerih lastnosti se ne prenašajo več naprej na naslednje generacije. Tako ima vsaka naslednja generacija nekoliko boljše lastnosti od svojih staršev. Ena večjih težav genetskega in evolucijskega programiranja je izbira ustreznega kriterija za selekcijo.

1.3.5 Induktivno logično programiranje

Večina sodobnih programskih jezikov je postopkovnih - programer določi katera opravila se opravljajo med obdelavo podatkov in v kakšnem vrstnem redu. Pri induktivnem logičnem programiranju [23, 60] (angleško *Inductive Logic Programming, ILP*) programer vnese le podatke in pravila, računalnik pa sam najde postopek, kako priti do rešitev. Beseda »induktivno« nam pove, da iščemo splošna pravila, ki jih izluščimo iz množice posameznih primerov.

1.3.6 Metoda podpornih vektorjev

Pri razvrščanju podatkov v razrede je zelo ugodno, če so podatki linearno ločljivi. Metoda podpornih vektorjev [24, 61] (angleško *Support Vector Machine, SVM*) preslika podatke, s pomočjo jedrnih funkcij (angleško *kernel function*), v

večdimenzionalen prostor tako, da ti postanejo linearno ločljivi. Pri tem upošteva le učne vzorce, ki so blizu odločitvene meje in jih imenuje podporni vektorji. Med preslikavo se prosti parametri jedrnih funkcij izberejo tako, da dobimo čim širši rob (angleško *margin*) med podatki različnih razredov.

1.3.7 Bayesova mreža

Bayesova mreža (angleško *Bayesian network*) je verjetnostni grafični model, ki omogoča razdelitev obsežnega, zapletenega problema na več manjših in preprostejših [25, 62]. Ena od bolj znanih Bayesovih mrež je skriti model Markova (angleško *Hidden Markov Model, HMM*).

V nadaljevanju disertacije se bomo osredotočili na obravnavo nevronskih mrež.

1.4 Nevronske mreže

Angleški naziv za nevronsko mrežo je *neural network (NN)*. Lahko jo poimenujemo tudi umetna nevronska mreža (angleško *artificial neural network, ANN*), če želimo poudariti, da gre za neživo napravo oziroma stroj. Nevronska mreža posnema delovanje biološke nevronske mreže – živalskih možganov [63]. Možgani so pri nekaterih opravilih dosti uspešnejši od računalnika. Sodobni računalnik kljub naglemu napredku še vedno potroši mnogo več energije in zaseda neprimerno večjo prostornino od primerljivo zmogljivih bioloških možganov. Biološki možgani so sestavljeni iz ogromnega števila celic, ki so med seboj povezane s tankimi »nitkami«. Te celice se imenujejo nevroni, njihove povezave oziroma »nitke« pa sinapse. Nevroni si med seboj, preko sinaps, pošiljajo električne dražljaje. Značilnost sinaps je, da se med seboj razlikujejo po električni prevodnosti. Med učenjem se ta njihova prevodnost spreminja. Znanje pridobljeno med učenjem je torej nakopičeno v sinapsah oziroma v njihovi prevodnosti. Če je vsota vhodnih signalov posameznega nevrona dovolj velika, pride do vžiga nevrona. To pomeni, da nevron na svoj izhod pošlje signal, ki se preko sinaps prenese naprej v ostale nevrone. Ta proces se v bioloških možganih dogaja od rojstva do smrti. Analiza

zgradbe in delovanja bioloških možganov je zahtevno opravilo. Razlogov zato je več:

- ogromno število nevronov in sinaps v možganih,
- zapletena zgradba možganov (vzvratne sinapse),
- naprave, ki možganov med opazovanje ne poškodujejo in ne vplivajo na njihovo delovanje, so zelo redke, drage in nenatančne.

Moderna tehnologija nam šele zadnjih nekaj let omogoča opazovanje živih možganov brez vpliva na njihovo delovanje, zaradi česar je to še vedno slabo raziskano področje, čeprav je zanimanje znanosti za to veliko. V zadnjih letih je opazen pospešek pri tovrstnih raziskavah. Do zdaj (leto 2013) je znanstvenikom uspelo ugotoviti le kateri centri tvorijo možgane, kako so ti centri razporejeni in povezani med seboj. V preteklosti smo do podobnih spoznanj prihajali večinoma na podlagi opazovanja ljudi in živali, ki so zaradi nesreče ali bolezni ostali brez posameznega dela možganov. Eden od najnovejših in največjih dosežkov pa je prebiranje ljudske misli v živemu človeku v realnem času. V času nastajanja te disertacije je stekla obsežna raziskava z imenom »Human Connectome Project« [3]. To je najbolj obsežna in najbolj natančna raziskava ljudskih možganov do sedaj. Obljubljen rezultat raziskave je kartografiranje celotnih možganov, do nivoja posameznih nevronov in sinaps.

1.4.1 Primerjava nevronskih mrež z računalniki

Pravzaprav gre tukaj za primerjavo strojnega učenja s klasičnim pristopom obdelave podatkov. Nevronske mreže niso, oziroma strojno učenje ni tekmec s klasičnim pristopom, ampak se z njim dopolnjuje. Za reševanje nekaterih problemov so primernejše nevronske mreže, med tem ko je za nekatere druge probleme primernejši klasični pristop. Klasični pristop je osnovan na programu oziroma algoritmu, ki ga ugotovi in vnese v računalnik človek oziroma programer.

Neredko so algoritmi iz resničnega življenja zelo zapleteni in jih ljudje ne uspemo dognati, vsaj ne do zadnje podrobnosti. V tem primeru s klasičnim pristopom, oziroma z običajnim programiranjem problema ne moremo rešiti. To nas omejuje na reševanje problemov, ki jih že dobro razumemo. Seveda bi bilo bolje, če bi lahko rešili tudi probleme, ki jih ljudje še ne razumemo povsem in ne znamo natančno zapisati ustreznih algoritmov za njihovo rešitev. Pri nevronskih mrežah poteka reševanje problema drugače, saj med učenjem same poiščejo algoritem, ki rešuje zadan problem. To naredijo tako, da v kopici učnih podatkov poiščejo splošna pravila. Temu postopku rečemo strojno učenje. Programerji oz. ljudje pri učenju ne sodelujejo, ampak se nevronska mreža uči samostojno. Zaradi tega lahko uporabljamo nevronske mreže tudi takrat, kadar ljudje ne poznamo algoritma, ki je potreben za izračun rezultata. Na ta način lahko znanje nevronskih mrež preseže tudi znanje njenega stvarnika – človeka. Znani so primeri, ko je nevronska mreža podatke napovedovala in razvrščala bolj točno kot eksperti [26].

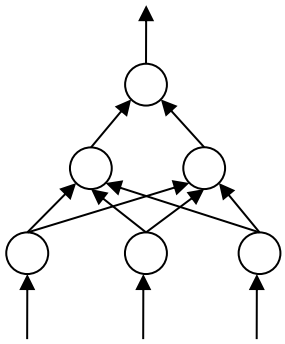
Tudi po končanem učenju je nekaj bistvenih razlik med delovanjem nevronske mreže in klasičnega računalnika. V računalniku je vsak trenutek aktivnih le nekaj odstotkov njegovih gradnikov (tranzistorjev), torej so ti slabo izrabljeni. Če se med njimi pokvari le en sam, je posledično pokvarjen celoten računalnik. Nevronske mreže odpravljajo ti dve pomanjkljivosti računalnikov. Če odpove ena sama sinapsa ali nevron, zaradi tega ne odpove celotna nevronska mreža. Nevronska mreža še vedno zna narediti vse kar je znala že pred odpovedjo, le da sedaj to naredi nekoliko manj natančno. Ljudem nam vsak dan odmre nekaj nevronov, pa še vedno znamo dihati, brati, računati in vse ostalo. Podobno velja tudi za umetne nevronske mreže. Sestavljene so iz velikega števila nevronov in sinaps, ki so vse v vsakem trenutku aktivne. Izguba ene same sinapse ne povzroči usodne napake. Kljub temu, da nevronske mreže izvedejo samo nekaj operacij na sekundo, lahko po zmogljivosti prekašajo računalnike, ki opravijo nekaj milijard operacij na sekundo. Umetne nevronske mreže se učijo iz primerov, ki jih imenujemo učni podatki. Za uspešno učenje morajo biti ti skrbno izbrani. Vsebovati morajo dovolj veliko število učnih primerov, ki morajo biti reprezentativni. Podatki so

reprezentativni, kadar pokrivajo celoten nabor vhodnih podatkov. Učni vzorci morajo biti razpršeni po celem vhodnem prostoru. V nasprotnem primeru lahko učenje hitro postane neučinkovito. Omenili smo že, da NN same najdejo rešitev zadanega problema, kar je sicer izjemno, vseeno pa moramo biti po učenju nekoliko previdni. Zgodi se lahko namreč tudi to, da rezultat učenja ni optimalen, čeprav je na prvi pogled videti tako. Kljub temu, da so odgovori nevronske mreže na vsa učna vprašanja pravilni, se lahko dogaja, da so posamični odgovori na testna vprašanja (nekoliko) napačni. Temu pojavu rečemo prekomerno učenje (angleško *overfitting*).

Običajno obravnavamo nevronsko mrežo kot črno škatlo in nevrone v njeni notranjosti kot skrite. V tem primeru ne vemo ničesar o tem kako mreža rešuje zadan problem. Nova učna metoda, ki je opisana v nadaljevanju te disertacije, ne obravnava več nevronske mreže kot črne škatle in posameznih notranjih slojev ne obravnava več kot skritih, kar nam omogoča nekaj razumevanja o tem kako nevronska mreža deluje od znotraj (»razmišlja«).

1.4.2 Vrste nevronskih mrež

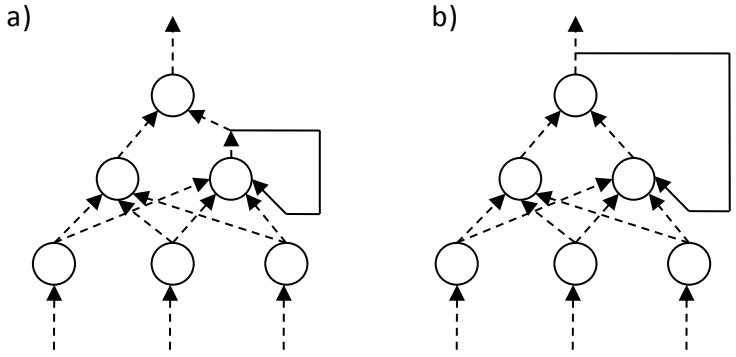
Ločimo več vrst različnih nevronskih mrež, ki se razlikujejo glede na to kakšni nevroni so uporabljeni v njih in kako so ti povezani med seboj. Najbolj preproste



Slika 1 Nevronska mreža s povezavami naprej

Mreža vsebuje tri vhode, dva nevrona v skritem sloju in en izhod. Povezave so puščice, nevroni so krogi.

nevronske mreže imajo vse povezave usmerjene v smeri od vhodov proti izhodom. Imenujejo se nevronske mreže s povezavami naprej [27] (angleško *FeedForward Neural Network, FFNN*). Ko so tovrstne mreže naučene, se njihova izhodna vrednost izračuna v enem samem prehodu in izhodi ne vplivajo več nazaj na lastne vhode. Bolj zapletene nevronske mreže vsebujejo tudi eno ali več povezav v vzvratni smeri in se imenujejo vzvratne nevronske mreže [28] (angleško *recurrent neural network, RNN*). Za vzvratne štejejo tiste povezave, ki potekajo nazaj na vhode enega od predhodnih slojev in tudi tiste, ki potekajo nazaj na vhode istega sloja. Učenje vzvratnih nevronskih mrež je mnogo zahtevnejše kot učenje mrež s povezavami samo naprej, traja dlje časa, pa tudi verjetnost uspešnega zaključka učenja je manjša. To se dogaja zato, ker so izhodne vrednosti nevronov pred pričetkom učenja napačne. Ta napaka se prenese nazaj na vhod in zato se nevroni učijo z napačnimi podatki. Torej med učenjem od nevronske mreže zahtevamo, da iz delno napačnih podatkov ugotovi pravilen rezultat. Pozitivna lastnost vzvratnih nevronskih mrež je v tem, da vzvratne povezave predstavljajo neke vrste spomin.

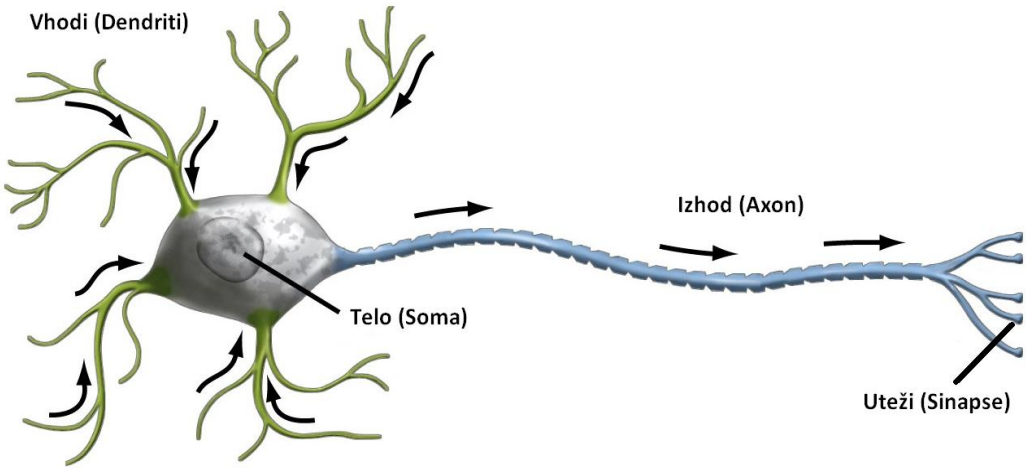


Slika 2 Vzvratna nevronska mreža

a) z vzvratno povezavo na isti sloj
b) z vzvratno povezavo na prejšnji sloj

Torej je izhodna vrednost vzvratne NN posledica trenutne vhodne vrednosti in preteklih vhodnih vrednosti. To nam koristi pri obravnavi časovnih nizov (na

primer napovedovanje vremena, borznih tečajev...). Ker ima vzvratna nevronska mreža lasten spomin, nam na vhodu ni potrebno dovajati starih vrednosti. Število vhodnih priključkov je lahko zaradi tega občutno manjše. Tudi po zaključenem učenju nam lahko vzvratne povezave še vedno nagajajo, saj se med uporabo RNN sprememba na izhodu prenaša nazaj na vhod, kar spet spremeni izhod, ki se spet prenese na vhod,… Posledica tega dejstva je, da se rezultat na izhodu NN spreminja, tudi ko je vrednost na vhodu že nekaj časa stabilna. Rezultat se lahko



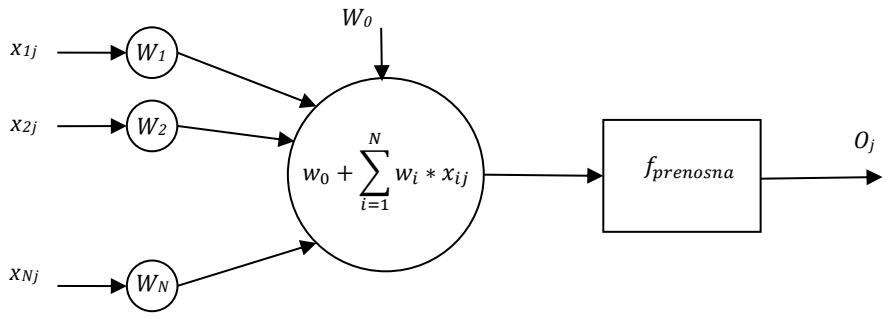
Slika 3 Biološki nevron

čez čas ustali pri neki končni vrednosti (konvergira) ali pa ves čas niha (oscilira). Osciliranje običajno ni zaželeno, saj to pomeni, da se nevronska mreža odloča, ne more pa se dokončno odločiti.

1.4.3 Umetni nevron

Umetni nevron je poenostavljen model biološkega nevrona [64]. Zgradbo biološkega nevrona prikazuje slika 3. Na vhode nevrona preko sinaps prispejo signali iz drugih nevronov. Vsaka sinapsa ima svoj značilni količnik (prosti parameter), ki se imenuje utež.

Signal se v sinapsi ojača za toliko krat, kolikor znaša njena utež w_i. V jedru nevrona se vsi ojačani signali najprej seštejejo, nato pa se rezultatu prišteje še konstanta w_0 imenovana prag. Nazadnje se na izhodu nevrona rezultat še omeji (limitira). To velja za veliko večino nevronov, ki jih imenujemo linearna pragovna



Slika 4 Zgradba umetnega nevrona

W_0 : prag nevrona,

W_1 do W_2 : uteži sinaps

$f_{prenosna}$: prenosna funkcija

enota (angleško *Linear Threshold Unit, LTU*). Redkeje srečamo tudi drugačne nevrone. Na primer nevrone s krogelno pragovno enoto (angleško *Radial Threshold Unit, RTU*), ki jih srečamo pri obravnavi učenja z metodama DistAl in RBF.

$$o_j = f_{prenosna}\left(W_0 + \sum_{i=1}^{n} x_i \cdot W_{ij}\right)$$ **Enačba 1**

i = zaporedna številka nevronovega vhoda

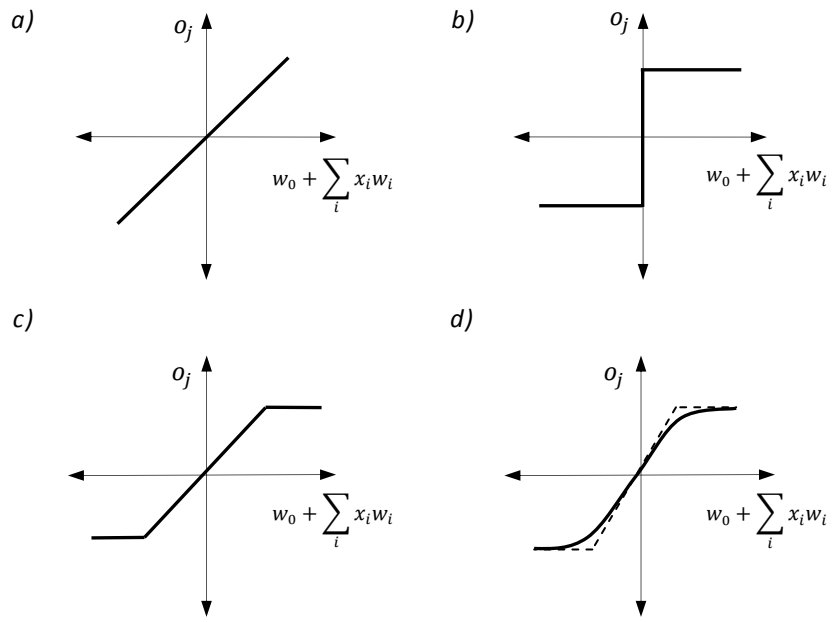
j = zaporedna številka učnega vzorca

Nevroni se razlikujejo tudi v načinu limitiranju izhodnega signala. Linearni limiter signala pravzaprav sploh ne spremeni. Trdi limiter da na izhod samo vrednost 0 ali 1. Vmesna različica med linearnim in trdim limiterjem je odsekovno linearni limiter. Najpogosteje se uporablja sigmoidni limiter, saj ima sigmoidna funkcija dobro lastnost, da je zvezno odvedljiva, sicer pa je sigmoida po obliki zelo

podobna odsekovno zvezni limiterski funkciji. Odvedljivost je koristna lastnost, kadar moramo med učenjem nevronske mreže računati gradient učne napake, ki temelji na parcialnih odvodih funkcije učne napake.

1.5 MLP

Navadni perceptron oziroma perceptron je preprosta in pogosto uporabljana nevronska mreža, ki sodi v kategorijo nevronskih mrež s povezavami naprej (FFNN). Prvi ga je predstavil Frank Rosenblatt leta 1957 [29, 65]. V perceptronu so nevroni razvrščeni v eno samo ravno vrsto, ki se imenuje sloj in so vse povezave



Slika 5 Prenosne funkcije nevrona

a) linearna prenosna funkcija

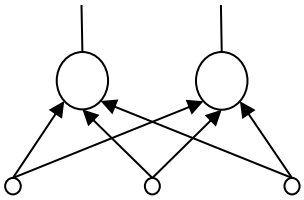
b) trda prenosna funkcija

c) odsekoma linearna prenosna funkcija

d) sigmoidna prenosna funkcija

med nevroni usmerjene naprej v smeri od vhoda proti izhodu (slika 6). Nabor vhodnih in izhodnih vrednosti je pri tej vrsti NN poljuben. Na vhodu in izhodu

perceptrona lahko uporabljamo zvezne in diskretne spremenljivke. To pomeni, da je perceptron primeren za uporabo analognih in digitalnih signalov. Njegovo učenje poteka postopoma (iterativno).



Slika 6 Perceptron s tremi vhodi, dvema nevronoma in dvema izhodoma

Mali krogi so vhodi, veliki krogi so nevroni, puščice pa so sinapse.

Konec učenja lahko sproži več kriterijev:

- Preostala učna napaka postane manjša od v naprej določene vrednosti. To pomeni, da se je perceptron naučil do takšne mere kot smo predvideli.
- Preostala učna napaka se neha zmanjševati, a še ni dosegla predvidene vrednosti. To pomeni, da se perceptron ne zmore naučiti do predvidene mere.
- Zgodnja zaustavitev učenja (angleško *early stoping*). S tem dosežemo, da se perceptron nauči toliko kot mu to dopušča natančnost učnih podatkov. Učenje ustavimo, ko se začne povečevati preostala napaka testnih podatkov. Če bi z učenjem vseeno nadaljevali, bi se preostala napaka učnih podatkov navidezno še naprej zmanjševala. Dejansko pa se ne bi več

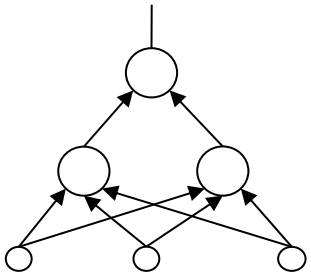
prilagajali učnim podatkom, ampak bi se že prilagajali njihovemu šumu. Pri zbiranju učnih podatkov namreč običajno pride, do različnih šumov - malih napak, ki se jim ne moremo izogniti (merski pogrešek, zaokroževanje, ljudska površnost…).

Perceptron se uči po preprostem algoritmu z imenom pravilo delta [4, 66]. Tudi njegova zgradba je preprosta. Žal so preprosti tudi problemi, ki jih zmore perceptron učinkovito rešiti. Rešti namreč zmore le probleme, kjer so učni vzorci linearno ločljivi. To pomeni, da so učni vzorci različnih razredov razmejeni s premico, ravnino oziroma hiper ravnino. Resnični življenjski problemi običajno niso linearni.

Če zaporedno povežemo več perceptronov nastane večslojni perceptron (angleško *multi-layer perceptron, MLP*), kateri ni več omejen na učenje linearnih problemov. Če uporabimo dovolj velik MLP, se je ta sposoben naučiti česarkoli, žal pa je njegovo učenje bolj zapleteno. Prvo učenje zanj, ki so ga znanstveniki odkrili, se imenuje backpropagation. Ta učna metoda je kasneje doživela številne izboljšave (Quickprop, Quasi-Newton, Levenberg-Marquardt,…). Vse naštete metode imajo eno skupno pomanjkljivost: globalni minimum učne napake iščejo z malimi, postopnimi, iterativnimi gradientnimi popravki. To pomeni troje:

- metoda je časovno zamudna,
- pogosto se ujame v lokalni minimum in
- ne pove nam ničesar o primerni velikosti mreže.

Zaradi takšnih in podobnih težav je učenje MLP-jev z gradientnimi metodami zahtevno in neučinkovito opravilo.



Slika 7 Večslojni perceptron

Ima tri vhode (mali krogi), dva nevrona v skritem sloju (spodnja velika kroga) in en nevron v izhodnem sloju (zgornji veliki krog). Puščice predstavljajo povezave oziroma sinapse.

1.6 Backpropagation

Izraz backpropagation, preveden v slovenščino, pomeni vzvratno napredovanje. To je ime algoritma za nadzorovano učenje MLP. Odkril ga je Paul Werbos leta 1974. Zaslovel pa je šele leta 1986, z objavo članka Learning representations by back-propagating errors [30] Takrat je to povzročilo preporod nevronskih mrež, ki so sicer že počasi začenjale toniti v pozabo. Pojav je bil tako znamenit, da je dobil celo svoje ime - Zima umetne inteligence (angleško *AI winter*). Do leta 1986 namreč ni bilo primernega algoritma za učenje večslojnih NN, enoslojne pa so dajale preskromne rezultate, da bi se uveljavile. Učni algoritem »backpropagation«, namenjen večslojnim perceptronom, je posplošitev že omenjenega učnega algoritma »pravilo delta«, ki je namenjeno navadnim, enoslojnim perceptronom. Gre za nadzorovano učenje, kjer je med učenjem potreben učitelj - nekdo, ki pozna želene izhodne vrednosti za vse vhodne učne podatke. Kot smo že omenili, je backpropagation gradientna optimizacijska metoda s katero optimiziramo (zmanjšujemo) učno napako. Običajno je to srednja

kvadratna napaka (angleško *Root Mean Square Error, RMSE*). Učenje je ponavljajoč oziroma iterativen postopek, kjer se v vsaki ponovitvi učna napaka malenkost zmanjša. To se dogaja tako, da dobivajo uteži nevronskih sinaps in pragov postopoma vedno ustreznejše vrednosti. Vzrok težav med takšno optimizacijo učne napake se skriva v gradientnem bistvu metode, saj med gradientnim učenjem zlahka zaidemo v lokalni minimum, kjer lahko učenje obtiči, še preden je naučenost MLP optimalna ali vsaj zadovoljiva. Optimizacija poteka kot ponavljanje velikega števila drobnih popravkov uteži, ki so bile v prvem trenutku naključno izbrane. Praviloma to pomeni, da so bile izbrane daleč od optimalnih vrednosti. Posledica je počasna konvergenca takšnega učenja, saj lahko med učenjem isto utež nek učni podatek povečuje, drugi pa zmanjšuje, kar lahko povzroči, da učenje obtiči na mestu, kljub temu, da je trenutna naučenost še vedno slaba. Dodatni slabosti med učenjem z metodo backpropagation sta, (a) da ne vemo prav ničesar o primernosti zgradbe MLP in (b) da uporabljamo za učenje vse učne vzorce, tudi tiste ki ne doprinesejo ničesar h kvaliteti učenja. Če zgradba nevronske mreže ni blizu optimalne se to odraža na kvaliteti učenja. Premajhna nevronska mreža se ne more zadovoljivo naučiti zadanega problema, prevelika pa je nagnjena k prekomernemu učenju. To je učenje kjer ne izluščimo bistva. Pri ljudeh takemu učenju rečemo učenje na pamet. Kadar pri učenju sodelujejo tudi nepomembnimi podatki, se s tem po nepotrebnem podaljšuje čas učenja.

1.7 Ostali sorodni pristopi

Pred pričetkom raziskovalnega dela smo se sistematično lotili preiskovanja sodobnih virov, da bi se seznanili z aktulanim stanjem na področju algoritmov za strojno učenje. Našli smo naslednje štiri algoritme, ki so sorodni z obravnavano metodo strojnega učenja:

1.7.1 Metoda DistAl

Odkril jo je Jihoon Yang leta 1998 [39]. Od obravnavane metode se razlikuje v tem, da uporablja drugačne nevrone od klasičnega MLP. Namesto nevronov z linearno pragovno funkcijo (LTU) so tukaj uporabljeni nevroni s krožno pragovno funkcijo (RTU), zato se vzorci ne razmejujejo s premico ampak s krožnico oziroma njenimi večdimenzionalnimi ekvivalenti. Roji (angleško *clusters*) so torej tukaj razmejeni s krožnicami.

1.7.2 Geometrijska Metoda

Odkrila jo je Rita Delogu leta 2006 [40]. Metoda temelji na linearnem programiranju. Je sorodna metodi SVM, saj obe uporabljata jedrne funkcije (angleško *kernel function*). Z jedrnimi funkcijami opravita preslikavo v več dimenzionalen prostor, kjer učni vzorci postanejo linearno ločljivi. Posebnost geometrijske metode je uporaba stopničaste jedrne funkcije. Od obravnavane metode se razlikuje tudi v tem, da namesto mejnih parov uporablja poliedre in da namesto izbranih uporablja vse učne podatke.

1.7.3 Inicializacije uteži z uporabo pravil

Odkril jo je Arunava Banerjee leta 1994 [41]. Od metode BPM se razlikuje v tem, da za izhodišče vzame pravila odločitvenega drevesa. Žal ta metoda ne zna poiskati zgradbe nevronske mreže, ki je blizu minimalne (angleško *near minimal architecture*). Njena dobra lastnost je konstruktivnost in to navkljub dejstvu, da temelji na nekonstruktivni metodi backpropagation.

1.7.4 Metoda Bipropagation

Odkril jo je Bojan Ploj leta 2009 [3]. Od obravnavane metode se razlikuje v dveh lastnostih. 1.) Metoda Bipropagation ni konstruktivna oziroma ne poišče zgradbe nevronske mreže, ki je optimalna za podane učne podatke. 2.) Za učenje uporablja vse učne podatke, tudi tiste ki nič ne prispevajo h kvaliteti učenja. Ta metoda je predhodnica metode BPM. Njena dobra lastnost je , da omogoča hitro učenje velikega MLP z velikim številom učnih vzorcev, pri čemer se dobro izogiba lokalnim minimumom.

1.8 Pričakovani rezultati

Cilj te disertacije je predstaviti nov algoritem za učenje umetnih nevronskih mrež s povezavami naprej vrste večslojni perceptron, ki smo ga poimenovali: »**metoda mejnih parov**« (angleško *Border Pairs Method*, okrajšano *BPM*). Med obravnavo nas bo predvsem zanimala primerjava nove metode z obstoječimi pristopi strojnega učenja. Ideja za metodo BPM je nastala med raziskovalnim delom, ko smo se soočali s težavami, ki jih povzroča metoda Backpropagation. Originalni prispevki metode BPM so:

- Pred pričetkom učenja izločimo jalove učne podatke in si izoblikujemo mnenje o zahtevnosti učnih vzorcev.
- Učenje poteka ločeno za posamezne nevrone in sloje. Problem učenja je s tem razstavljen na več manjših, ki jih obravnavamo ločeno.
- Med učenjem lako odpravljamo šum ter najdemo značilke in roje učnih podatkov.
- Neiterativno, konstruktivno in zanesljivo učenje.

Vse naše ugotovitve in pričakovanja smo po temeljitem premisleku strnili v glavno tezo.

Glavna teza: Z uporabo nadzorovanega konektivističnega pristopa BMP k strojnemu učenju, lahko dosežemo uspešnejšo izgradnjo in boljše učne rezultate v realnih primerih odločanja, kot pri običajnih in že obstoječih pristopih nadzorovanega strojnega učenja.

Glavno tezo smo razdelili na šest hipotez, s čimer smo jo podrobneje opredelili in olajšali njeno potrjevanje:

Hipoteza 1: Z metodo BPM lahko najdemo zgradbo MLP, ki je blizu optimalne oziroma minimalne zgradbe za podane učne podatke.

Hipoteza 2: Z metodo BPM lahko zapleten in negotov iterativni učni postopek razdelimo na več manjših, preprostejših in zanesljivejših neiterativnih postopkov.

Hipoteza 3: Z metodo BPM lahko dosežemo manjšo preostalo učno napako ob hkratnem izogibanju prekomerne naučenosti.

Hipoteza 4: Z metodo BPM lahko poiščemo značilke v vhodnem prostoru.

Hipoteza 5: Z metodo BPM lahko poiščemo roje v vhodnem prostoru.

Hipoteza 6: Z metodo BPM lahko zmanjšamo šum v učnih vzorcih.

O potrditvi oziroma zavrnitvi glavne teze se bomo odločali na podlagi rezultatov potrjevanja pravkar omenjenih šestih hipotez.

2 Bipropagation

2.1 Razvoj algoritma bipropagation

Algoritem bipropagation, ki je izboljšava algoritma backpropagation in predhodnik Metode mejnih parov, je odkril Bojan Ploj leta 2009 [3]. Med učenjem s tem agoritmom se popravki uteži širijo v obeh smereh, naprej in nazaj, kar nam sporoča tudi predpona »bi« v njegovem imenu. Skriti sloji se, pri algoritmu bipropagation, preimenujejo v notranje sloje, saj so njihove želene vrednosti med učenjem znane. Zaradi tega novega dejstva lahko ločeno učimo vsak sloj zase. Namesto enega N slojnega perceptrona dobimo N navadnih enoslojnih perceptronov. S tem smo zapleten problem razdelili na več manjših medsebojno neodvisnih problemov. Učenje perceptrona je tako opravljeno mnogo hitreje kot z metodo backpropagation in predvsem brez zapletanja v lokalne minimume.

2.2 Osnovna ideja algoritma bipropagation

Osnovna ideja algoritma bipropagation je, da namesto naključnih uporabimo premišljeno izbrane notranje želene vrednosti. Na tak način poteka učenje zadanega problema v MLP ločeno in postopono iz sloja v sloj. Ta postopnost je podana z matrikami evklidskih razdalj med učnimi vzorci. Isto ležni elementi matrike evklidskih razdalj postopoma (monotono) spreminjajo svoje vrednosti od vhodnega sloja proti izhodnemu.

2.3 Primer uporabe algoritma bipropagation

Oglejmo si uporabo algoritma bipropagation na primeru učenja z naborom podatkov za logično funkcijo XOR. Za ta nabor učnih podatkov smo se odločili, ker vsebuje lokalni minimum, ki med učenjem z metodo backpropagation povzroča težave.

Tabela 1 Logična funkcija XOR

X	y	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Matrika evklidskih razdalj R_i (enačba 2) vhodnega sloja za podatke iz tabele 1 zgleda tako:

$$R_i = \begin{bmatrix} 0 & 1 & 1 & \sqrt{2} \\ 1 & 0 & \sqrt{2} & 1 \\ 1 & \sqrt{2} & 0 & 1 \\ \sqrt{2} & 1 & 1 & 0 \end{bmatrix}$$

Enačba 2

Elementi matrike R_i so evklidske razdalje med posameznimi učnimi vzorci. Razdalja med vzorcema n in m se v matriki nahaja v vrstici n in stolpcu m. Ker je evklidska razdalja komutativna je matrika zrcalna glede na svojo glavno diagonalo. Na glavni diagonali so same ničle, saj so tam podane razdalje učnih vzorcev do samega sebe. Na podoben način nastane tudi matrika evklidskih razdalj za podatke izhodnega sloja:

$$R_o = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$ **Enačba 3**

Tabela 2 Prikaz notranjih vrednosti

X	Y	X_n	Y_n	XOR
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

Če izberemo notranje vrednosti (X_n in Y_n) kot je prikazano v tabeli 2, dobimo naslednjo matriko evklidskih razdalj v notranjem sloju:

$$R_n = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & \sqrt{2} & 1 \\ 1 & \sqrt{2} & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$ **Enačba 4**

Opazimo lahko podobnost med matrikami R_i, R_n in R_o. Če zanemarimo pomožno diagonalo, so vrednosti isto ležnih elementov v vseh treh matrikah enake. Poglejmo si podrobneje še pomožno diagonalo. Vhodna matrika R_i ima vse diagonalne vrednosti kvadratni koren iz števila dva, notranja matrika R_n ima že na mesto dveh korenov ničli, izhodna matrika R_o pa ima samo še ničle. Temu rečemo, da so vrednosti matrike R_n vmes med isto ležnimi vrednostmi matrik R_i in R_o. Isto ležne vrednosti matrik se od sloja do sloja spreminjajo postopoma, s čemer se z vsakim slojem vedno bolj približamo k rešitvi. To nam daje upanje, da smo na dobri poti do pravilne rešitve.

Največja težava metode bipropagation je iskanje takšnih optimalnih notranjih želenih vrednosti (X_n in Y_n), da se bodo istoležne vrednosti matrike razdalj R

dejansko postopoma spreminjale (monotono upadale). Avtor algoritma predlaga dva načina iskanja optimalnih notranjih želenih vrednosti:

1. **Analitično**. Pri tem načinu določanja želenih vrednosti notranjih slojev uvedemo nov pojem, ki ga poimenujemo kvaliteta vzorcev (KV).

$$KV = \frac{\sum RR}{\sum RI} \qquad \textbf{Enačba 5}$$

Pri tem velja :

ΣRR = vsota razdalj med vzorci različnih razredov

ΣRI = vsota razdalj med vzorci istega razreda

Želimo, da se kvaliteta vzorcev poveča, pri vsakem prehodu v naslednji sloj. To pomeni, da se pripadniki istega razreda z vsakim slojem med seboj zbližajo, pripadniki različnih razredov pa med seboj oddaljijo. Za izračun vrednosti notranjih slojev uporabimo nelinearni sistem enačb s prostimi parametri (jedrne funkcije). Ko računamo kvaliteto vzorcev za notranji sloj, dobimo izraz, ki vsebuje le konstante in proste parametre. Z izbiro ustreznih prostih parametrov maksimiramo kvaliteto notranjih vzorcev.

2. Grafično. Pri tem načinu izberemo želeno vrednost notranjega sloja na intervalu med vhodno vrednostjo in želeno izhodno vrednostjo. Lahko uporabimo kar njuno aritmetično sredino. Če je notranjih slojev več, uporabimo načelo postopnosti. Za prvi notranji sloj izberemo vrednosti, ki so vmes med vhodnimi vrednostmi in aritmetičnimi

sredinami, za drugi notranji sloj pa vrednosti med aritmetičnimi sredinami in želenimi izhodnimi vrednostmi.

Algoritem bipropagation se je izkazal kot hiter in zanesljiv. Učenje logične funkcije XOR z metodo bipropagation poteka več kot 25 krat hitreje od metode backpropagation [4]. Pri tem je število potrebnih učnih iteracij (angleško *epoch*) tudi mnogo bolj stalno (manjši standardni odklon), kar nam pove, da smo manj odvisni od sreče pri izbiri začetnih vrednosti uteži.

Tudi testiranje metode bipropagation z realnimi učnimi podatki daje spodbudne rezultate. V preizkusu smo razvrščali (prepoznavali) prostoročno zapisane desetiške številke. Učni nabor je obsegal približno 60 tisoč številk, ki so bile prebrane z ločljivostjo 28 x 28 točk in jih je prispevalo več kot 500 različnih piscev. Kljub obsežni podatkovni zbirki so bile vse učne številke pravilno prepoznane že po nekaj desetih iteracijah učenja z metodo bipropagation. Tudi testne številke so dale dobre rezultate. Metoda backpropagation je v enakih okoliščinah v celoti odpovedala, saj se njena učna napaka ni niti začela zmanjševati. V tej primerjavi ni bila uporabljena osnovna metoda backpropagation ampak njena izpeljanka z imenom Levenberg–Marquardt, ki je velja za eno najbolj učinkovitih gradientnih učnih metod za MLP-je.

Metoda bipropagation je od backpropagation podedovala tudi njeno nekonstruktivnost. Zato med učenjem z metodo bipropagation žal ne ugotovimo ničesar o optimalni zgradbi MLP-ja. Ne izvemo niti koliko slojev ima optimalni MLP, niti koliko nevronov vsebujejo posamezni sloji optimalnega MLP.

3 Analiza gradientnih učnih metod

3.1 Pomanjkljivosti gradientnih metod

Večina metod za učenje MLP izvira iz prvotne, gradientne, iterativne metode z imenom backpropagation, ki ima na žalost mnogo pomanjkljivosti. Med bolj opazne in moteče pomanjkljivosti sodijo naslednje lastnosti:

- **Počasnost** - za uspešno učenje je potrebno veliko število iteracij. Proste parametre računamo postopoma v majhnih korakih. Običajno je potrebno več tisoč korakov, da dobimo ustrezno naučeno nevronsko mrežo. To pomeni zadovoljivo majhno preostalo učno napako testnih vzorcev in dovolj visok odstotek pravilno razvrščenih testnih vzorcev. To pa ni edini vzrok počasnosti. Namesto da bi uporabljali samo podporne vektorje (izbrane učne vzorce) učimo z vsemi vzorci.

- **Nezanesljivost** - med učenjem iterativno računamo skupno napako vseh učnih vzorcev. Z gradientnimi korekcijami to napako počasi zmanjšujemo, iščemo njen globalni minimum. Zelo pogosto se dogaja, da gradientno zmanjševanje napake nevronske mreže trajno obtiči v lokalnem minimumu. Tako se učenje že konča, ko je preostala učna napaka še vedno mnogo večja od optimalne. V tem primeru je potrebno učenje ponoviti z drugačnimi začetnimi vrednostmi prostih parametrov. Zaradi nekonstruktivnosti (glej naslednjo alinejo) se lahko zgodi, da uporabimo neustrezno zgradbo MLP in je lahko učenje po številnih poskusih še vedno

neuspešno. Obstaja tudi možnost, da učenje zaustavimo prepozno. V tem primeru govorimo o prekomernem učenju.

- **Nekonstruktivnost** - med učenjem nevronske mreže z gradientnim učnim algoritmom ne ugotovimo ničesar o optimalnem številu slojev in optimalnem številu nevronov v posameznih slojih nevronske mreže. Optimalno zgradbo iščemo z ugibanjem na slepo, tako da učimo številne raznolike MLP. Zaradi nezanesljivosti učenja (glej prejšnjo alinejo) moramo vsak MLP učiti večkrat.

Zadnji dve med omenjenimi pomanjkljivostmi se medsebojno podpihujeta. Lahko se zgodi, da se med učenjem znajdemo v začaranem krogu iz katerega ne najdemo izhoda.

3.2 Pomisleki o lastnostih gradientnih učnih metod

Med preteklim raziskovalnim delom so se nam, razen že omenjenih pomanjkljivosti gradientnih metod, porajala še mnoga druga vprašanja oziroma pomisleki povezani z učenjem MLP:

- **Ali je potrebno za učenje uporabljati vse učne vzorce**, ali pa morda zadostujejo že izbrani vzorci? Kako izbrati te vzorce? Če nam bo uspelo izločiti redundančne učne vzorce lahko pričakujemo, da bo učenje potekalo hitreje in z manj zapleti. Podobno idejo uporablja metoda SVM [24]. Tam se izbrani učni vzorci imenujejo podporni vektorji .

- **Ali je morda možno poiskati začetne vrednosti prostih parametrov MLP-ja**, ki so boljše od naključno izbranih? Ali je morda možno določiti končne vrednosti prostih parametrov v enem samem koraku (neiterativno)? V omejenem obsegu to velja za metodo bipropagation, kjer so nekatere uteži določene že v naprej [4].

- **Ali morda lahko značilke (angleško *feature*) z večjo dimenzijo od originalnega vzorca poenostavijo učenje in izboljšajo stopnjo naučenosti**?

Kako najti te morebitne značilke? Običajno ima značilka manjšo dimenzijo kot originalni učni vzorec, saj z njo želimo odpraviti redundanco v podatkih. Obratno pot ubira metoda podpornih vektorjev, ki skuša poiskati velike in hkrati linearno ločljive značilke, ki imajo po vrh vsega še prazen prostor (rob) v neposredni bližini mejne črte [24].

- **Ali obstajajo v skritih slojih naučenega MLP-ja kakšne zakonitosti**? Ali nam te morebitne zakonitosti morda sporočajo kako MLP deluje? Do sedaj je prevladovalo mnenje, da MLP črna škatla, ki nam da na izhodu rezultat, ki ni pospremljen z razlago [31]. Izhodišče naše raziskove bo ravno dogajanje v notranjih slojih MLP.

- **Ali je med učenjem možno poiskati (skoraj) optimalno zgradbo MLP za podane učne vzorce?** Običajno pod pojmom optimalna zgradba razumemo čim manjši MLP, ki še rešuje zadani problem, saj so majhni MLP manj podvrženi prekomernemu učenju [26]. S tališča robustnosti pa optimalni MLP vsebuje še nekaj redundančnih nevronov, ki lahko nadomestijo poškodovane.

- **Ali je možno med učenjem zmanjšati šum učnih vzorcev in s tem poenostaviti učenje ter povečati natančnost rezultatov?** Obstaja več metod odpravljanja šuma, ki se izvedejo pred pričetkom učenja. Optimalno razšumljanje poteka med učenjem in vpliva samo na ustrezne učne zvorce.

- **Ali je možno že pred učenjem ugotoviti kako kvalitetni so učni vzorci, oziroma kako težavno bo učenje?** Dobro je že med učenjem vedeti s kašnimi učnimi podatki imamo opravek, da lahko prilagodimo naše zahteve in pričakovanja. Če so učni podatki slabi (šum, prekrivanje, nereprezentativnost,...) ne moremo pričakovati dobrega učnega rezultata [26].

- **Ali je možno rojenje podatkov med učenjem**? Rojenje podatkov med učenjem pomeni, da zapleten problem razdelimo na več preprostih. To nam tudi olajša razumevanje kako MLP rešuje zadan problem.

- **Ali je možno učinkovito inkrementalno in online učenje MLP?** Pri algoritmu backpropagation je pogosto potrebno ponoviti celoten postopek učenja že zaradi enega samega dodatnega učnega vzorca [32]. Idealen algoritem omogoča nadaljevanje predhodnega učenja.

- **Ali je med učenjem možna spremeba učnega koncepta (angleško *concept drift*)?** Enako kot pri dodajanju novih učnih vzorcev velja tudi pri odvzemanju zastarelih. Zaradi samo enega odvzetega zastarelega učnega vzorca ne želimo ponavljati celotnega učnega postopka.

Na podlagi omenjenih pomanjkljivosti in pomislekov smo izoblikovali hipoteze, ki smo jih podali v uvodu.

3.3 Analiza učnih rezultatov gradientnih metod

Da bi odpravili omenjene pomanjkljivosti gradientnih učnih metod in da bi našli odgovore na pomisleke o njih, smo najprej analizirali njihove učne rezultate. Zato smo se lotili učenja MLP - jev z različnimi, že uveljavljenimi, gradientnimi učnimi metodami. Rezultati učenja z različnimi metodami so se med seboj razlikovali po številu potrebnih iteracij, po porabljenem času in po številu neuspešnih poskusov učenja. Običajno smo dosegali najboljše rezultate z Levenberg-Marquardtovo izpeljanko metode backpropagation. Pri tej raziskavi nas je predvsem zanimala stopnja naučenosti po uspešno zaključenem učenju. Želeli smo dobiti čim bolj natančno naučen MLP, pri čemer sta bili hitrost in zanesljivost učenja drugotnega pomena. Različno zgrajene MLP smo učili z več različnimi dvodimenzionalnimi nabori učnih podatkov (angleško *data set*), ki ne vsebujejo šuma in se ne prekrivajo. Začeli smo z najmanj zahtevnim naborom, nato pa smo postopoma uporabljali vedno zahtevnejše nabore učnih podatkov. Pri tem smo pričakovali, da

bomo lahko kasneje ugotovitve iz dvodimenzionalnega sveta posplošili tudi v tri in večdimenzionalni prostor. Uporaba »samo« dvodimenzionalnega prostora, nikakor ne pomeni, da so obravnavani problemi preprosti. Spomnimo se že omenjene »zime umetne inteligence«, ki jo je povzročilo dejstvo, da perceptron ne zmore rešiti »zgolj« dvodimenzionalnega problema logične operacije XOR, ki vsebuje »zgolj« štiri učne vzorce.

Po uspešno končanih učenjih, z večimi različnimi metodami, smo ugotovili, da so se nevroni v skritih slojih vedli vedno enako, neodvisno od uporabljene učne metode. Morebitna razlika je bila samo v vrstnem redu nevronov znotraj sloja, kar pa ne vpliva na delovanje MLP, saj so nevroni znotraj posameznih slojev enakovredni. Poglejmo si podrobneje učne rezultate za posamezne nabore učnih vzorcev:

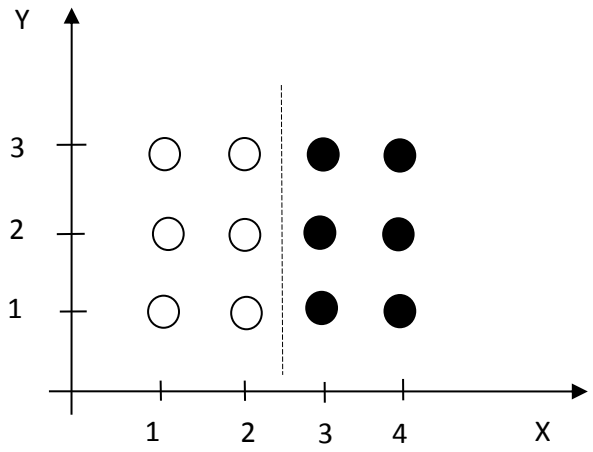
3.3.1 Učenje z linearno ločljivimi učnimi vzorci

Začeli smo z najpreprostejšim naborom podatkov – z dvodimenzionalnimi, dvorazrednimi linearno ločljivimi učnimi vzorci. Njihova pripadnost je podana v tabeli 3 in na sliki 8, ter je določena z barvo. Posamezni vzorci lahko pripadajo belemu ali črnemu razredu, ter tako tvorijo dvodimenzionalno črno belo sliko.

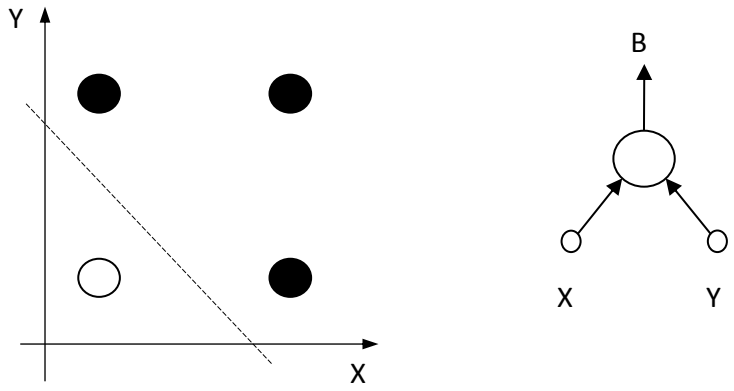
Tabela 3 Preprosti učni vzorci

X	Y	B
1	1	0
1	2	0
1	3	0
2	1	0
2	2	0
2	3	0
3	1	1
3	2	1
3	3	1
4	1	1
4	2	1
4	3	1

V vsaki vrstici tabele 3 sta podani X in Y komponenti enega učnega vzorca ter njegov razred B (barva), ki mu pripada. Beli krogi predstavljajo negativne vzorce



Slika 8 Preprosti dvodimenzionalni učni vzorci



Slika 9 Učni vzorci logične funkcije ALI in ustrezen perceptron

(B=0), črni pa pozitivne (B=1). Evklidske razdalje med sosednjimi učnimi vzorci so v tem primeru enake, v splošnem pa se lahko tudi razlikujejo.

Vhodni spremenljivki X in Y sta zvezni: $x \in \Re,\ y \in \Re$

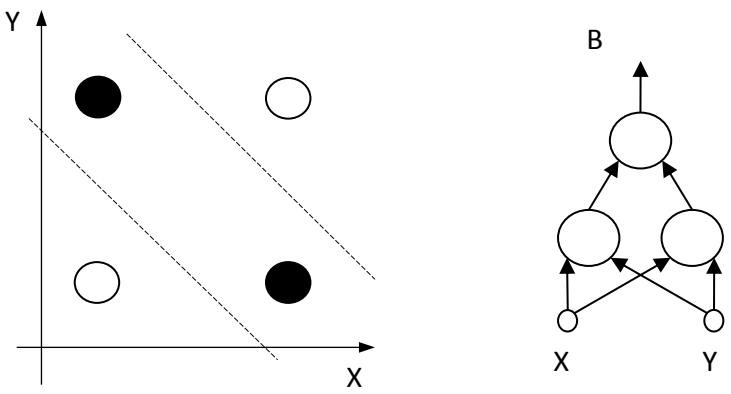
Njuno definicijsko območje sega od minus neskončno do plus neskončno. Število različnih barv, ki jih imajo krogi oziroma število razredov je lahko tudi večje kot dva, vendar mora biti končno naravno število. Torej je izhodna spremenljivka diskretna. V našem primeru, ko imamo dva razreda, zato lahko rečemo, da je izhodna spremenljivka binarna. Podani učni vzorci so linearno ločljivi. To pomeni, da lahko vse bele kroge ločimo od črnih z eno samo premico (črtkana črta na sliki 8). Takšni učni vzorci veljajo za nezahtevne, kar so potrdili tudi naši eksperimenti. Učenje s takšnimi vzorci je bilo vsakokrat opravljeno hitro in zanesljivo.

3.3.2 Učenje logične operacije ALI

Učenje smo nadaljevali s podatki logične operacije ALI z dvema spremenljivkama. Tudi ta nabor podatkov sodi med linearno ločljive in zato ne zahteva velikega števila slojev ter nevronov v MLP. Kot so pred nami potrdili že številni raziskovalci, zadostuje za uspešno učenje operacije ALI že en sam sloj, z enim samim nevronom. Tudi dejstvo, da uporabljamo le dve vhodni spremenljivki je blagodejno za naše raziskovanje, saj si lahko dvodimezionalni vhodni prostor brez težav predočimo, oziroma ga upodobimo na papirju ali zaslonu. Zahtevnost učnih vzorcev smo nato počasi stopnjevali.

3.3.3 Učenje logične operacije XOR (izključni ALI)

Sledilo je učenje MLP z naborom podatkov logične operacije XOR, ki je za učenje

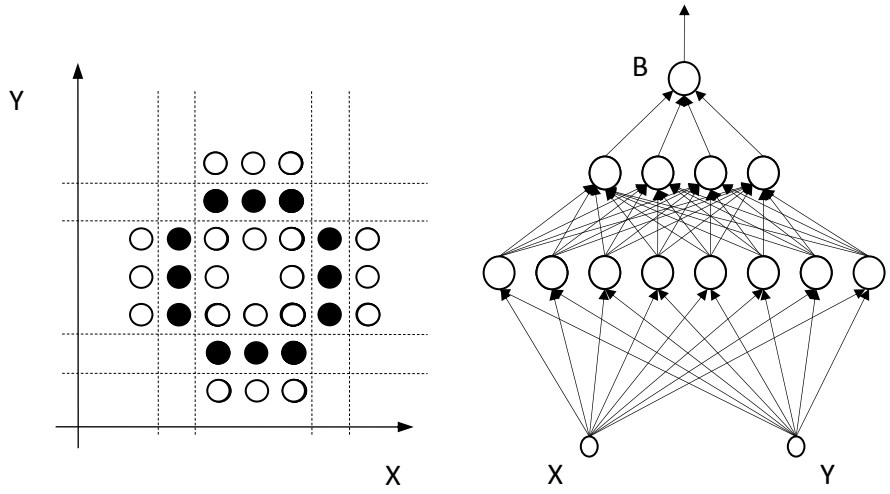


Slika 10 Učni vzorci logične funkcije XOR in ustrezen MLP

MLP znamenita. Ta nabor se razlikuje od predhodnih po tem, da ni več linearno ločljiv. Običajno povzroča učenje funkcije XOR z gradientnimi metodami težave [67], saj se tovrstno učenje neredko konča v slepi ulici, ko navkljub slabi naučenosti (veliki preostali učni napaki), ni več možen nadaljnji napredek pri učenju. Prav tako obstaja tudi dvom glede ustrezne velikosti MLP. Šele po številnih poskusih učenja z raznoliko zgrajenimi MLP-ji smo našli optimalnega - najmanjšega, ki še lahko uspešno reši zadan problem. Torej smo optimalno zgradbo MLP našli z ugibanjem na slepo. Zaradi teh in podobnih težav se je zgodila tudi znamenita zima umetne inteligence (angleško *AI winter*).

3.3.4 Učenje črke »O«

Po logični funkciji XOR, je sledilo učenje z naborom podatkov črke »O«, ki jih prikazuje levi del slike 11. Ponovno smo uporabili metodo poskusov in napak ter



Slika 11 Učni vzorci črke O in ustrezen MLP

Črtkane črte na sliki so skica rešitve, ki jo bomo spoznali v nadaljevanju disertacije.

ugotovili, da je za uspešno učenje potrebno dosti večje število nevronov, kot pri prejšnjih učnih problemih. Tokrat smo potrebovali vsaj trinajst nevronov razporejenih v tri sloje, kot je prikazano na desnem delu slike 11. V nadaljevanju

disertacije bomo ugotovili, da osem nevronov v prvem skritem sloju ustreza osmim črtkanim črtam, štirje nevroni v drugem skritem sloju pa ustrezajo štirim področjem, ki vsebujejo črne kroge.

3.4 Opazovanje lastnosti skritih slojev

V prejšnjem naslovu smo opisali učenje MLP-jev s štirimi različnimi nabori podatkov. Po uspešno zaključenem učenju nas zanima ali je možno v naučenih skritih slojih najti kakšne zakonitosti, ki bi nam omogočile drugačno, neiterativno in bolj učinkovito učenje. Zelo ugodno bi bilo, če bi nam najdene zakonitosti omogočale učenje posameznih slojev MLP-ja, od prvega do zadnjega, ločeno vsakega zase.

Skrite sloje naučenega MLP smo preučevali tako, da smo spreminjali vhodne vrednosti in opazovali odziv posameznih nevronov v skritih slojih. Med preučevanjem se nismo omejili samo na obstoječi nabor učnih vzorcev, temveč nas je še posebej zanimalo mejno področje oziroma vhodni prostor v bližini mejne črte. Na sliki 8 se nahaja mejno področje tam, kjer ima spremenljivka X vrednost blizu 2,5 (črtkana črta). Pri opazovanju smo izhajali iz predpostavke, da enake vrednosti na vhodu nekega sloja povzročijo enake rezultate na izhodu tega sloja, oziroma na vhodu naslednjega sloja, saj se lastnosti posameznih nevronov med uporabo ne spreminjajo, oziroma velja, da so nevroni časovno neodvisni, kar imenujemo časovna invariantnost nevronov. Najprej smo na vhod MLP dali učni podatek, ki je blizu mejne črte. Nato smo vhodno vrednost postopoma spreminjali tako, da smo se približevali najbližjemu učnemu podatku na drugi strani mejne črte. To smo poimenovali pomikanje učnih podatkov. Zaradi invariantnosti nevronov je za izhodno spremembo nujno potrebna tudi sprememba v vseh skritih slojih MLP-ja. Prva ugotovitev med opazovanjem rezultatov je bila, da so vsi učni vzorci dajali v vseh skritih slojih le binarne rezultate (0 ali 1). Druga ugotovitev je bila, da se je praviloma pri vsakem pomikanju učnih podatkov spremenila vrednost le nekaterim posameznim nevronom v prvem skritem sloju. Skoraj vedno

enemu samemu nevronu, le redko dvema ali večim. Ko smo izvajali pomikanje med bližnjimi učnimi vzorci istega razreda, se v skritih slojih običajno niso dogajale spremembe (rojenje), na izhodu MLP pa zagotovo ni bilo sprememb.

Strnimo ugotovitve:

- Vsi nevroni, v vseh slojih dajo na svojem izhodu binarno vrednosti 0 ali 1 oziroma delujejo v zasičenju. Izjema je le zelo ozek pas tik ob mejni črti.
- Bližnji vzorci istega razreda dajejo na izhodu skritih slojev enak rezultat - rojenje (angleško *clustering*).
- Bližnja vzorca dveh različnih razredov povzročata na izhodu prvega skritega sloja zelo podoben rezultat in se med seboj razlikujeta največkrat samo v enem bitu.
- Nevroni v drugem skritem sloju in v vseh morebitnih naslednjih slojih na svojem vhodu dobijo in dajo na svoj izhod izključno binarne vrednosti. Lahko rečemo, da se od drugega skritega sloja naprej izvršujejo logične operacije s podatki iz predhodnega sloja.

3.5 Definicija mejnega para

Pri opazovanju skritih slojev so bili omenjeni bližnji vzorci različnih razredov, ki se med seboj razlikujejo samo v enem bitu. Ker je izraz »bližnji vzorci« zelo ohlapen in ga vsakdo lahko razume po svoje, ga želimo določiti bolj natančno, čim bolj enostavno, čim bolj univerzalno in hkrati primerno za vhodni prostor s poljubnim število dimenzij. S tem bomo dosegli, da bo metoda primerna za nevronske mreže s poljubnim številom vhodov. V enodimenzionalnem prostoru je najbolj preprosta definicija razdalje [68] med dvema točkama določena z enačbo 6.

$$d_{AB} = f(x_A, x_B) = x_B - x_A \qquad \textbf{Enačba 6}$$

Ker nas ne zanima razdalja od točke A do B niti od točke B do A, ampak zgolj njuna medsebojna oddaljenost ne glede na smer, vpeljemo v enačbo operator za absolutno vrednost. Za takšno pojmovanje razdalje velja enačba 7.

$$d_{AB} = f(x_A, x_B) = |x_B - x_A| \qquad \textbf{Enačba 7}$$

V večdimenzionalnem prostoru lahko skalarja x_A in x_B nadomestita vektorja X_A in X_B (enačba 8).

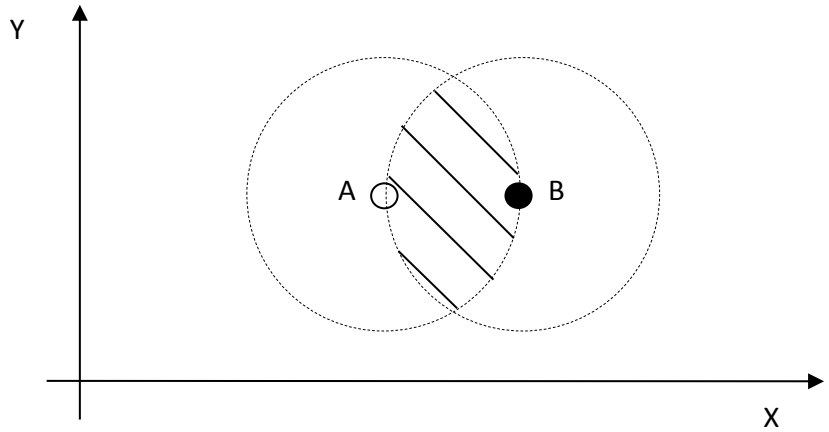
$$D_{AB} = f(X_A, X_B) = |X_B - X_A| \qquad \textbf{Enačba 8}$$

V tem primeru je razdalja D_{AB} vektor. Ker nas zanima le dolžina tega vektorja, ne pa tudi smer, se zadovoljimo s skalarjem d_{AB} iz enačbe 9, oziroma z evklidsko razdaljo med točkama A in B:

$$d_{AB} = f(X_A, X_B) = \sqrt{\sum_{i=1}^{i=N} (x_{iB} - x_{iA})^2} \qquad \textbf{Enačba 9}$$

N = število dimenzij vhodnega prostora

Osredotočimo se sedaj na del vhodnega prostora, ki se nahaja med dvema učnima vzorcema. Zaradi lažje predstave za razlago uporabimo primer



Slika 12 Dve točki v dvodimenzionalnem prostoru

dvodimenzionalnega vhodnega prostora (N=2), ki ga prikazuje slika 12. Točki A in B smo poimenovali bližnji (angleško *near points*), če se znotraj preseka njunih krožnic s polmerom, ki ustreza njuni medsebojni razdalji (šrafirano polje na sliki 12), ne nahaja nobena tretja točka. Bližnji točki, ki pripadata različnima razredoma poimenujemo **mejni par** (angleško *border pair*). V trodimenzionalnem prostoru krožnici na sliki 12 nadomestita sferi, v prostoru s še več dimenzijami pa dobimo hipersferi. Trodimenzionalen prostor in presek dveh sfer v njem si še lahko predstavljamo in je podoben konveksni leči. Pri hipersferah predstava preseka žal ne deluje. Kot smo že ugotavljali v enačbi 9, je računsko določanje razdalj in presekov hipersfer možno za poljubno število dimenzij.

4 Razšumljanje po metodi mejnih parov

4.1 Splošno o šumu

Šum je nekaj, kar se nam nehote prikrade v podatke med njihovim zajemanjem in obdelavo. Zaradi šuma se vrednost podatkov nekoliko spremeni, čeprav si tega ne želimo [69]. Ko je šum že prisoten v podatkih, ga je težko ublažiti, še težje pa ga je popolnoma odstraniti. To velja predvsem za zvezne oziroma analogno zapisane podatke. Prisotnost šuma v podatkih nam otežuje njihovo obdelavo. Iz vsakodnevnih izkušenj vemo, da se je v hrupnem prostoru (šum) težje pogovarjati (obdelava podatkov) kot v tišini (brez šuma). V hrupu se moramo bolj potruditi, da bi razumeli sogovornika in lažje pride do nesporazuma. Kadar je hrup premočan, sporazumevanje sploh ni več možno. To velja enako za ljudi kot tudi za inteligentne sisteme. Prisotnost šuma v signalu otežuje, v skrajnem primeru pa celo onemogoča, delovanje inteligentnih sistemov. Da bi se izognili tem težavam, poskušamo pri vsakem koraku obdelave signala, od zajemanja naprej, preprečiti oziroma omejiti vdor šuma.

Beseda šum izvira iz avdio tehnike. Pri poslušanju zvočnih zapisov se v ozadju pojavlja zvok podoben fonemu »š«. Temu pojavu so dali ime šumenje, dodatnemu zvoku pa šum. Kasneje se je ta naziv uveljavil tudi na drugih področjih rabe signala, kot je na primer slikovni signal. V animiranem slikovnem signalu se šum pojavi kot sneženje. Slika dobi premične bele pike, ki spominjajo na ples snežink [33].

Želimo si, da šum sploh ne bi »onesnažil« signala. Kadar pa se šum kljub vsemu pojavi, ga skušamo izločiti. Čeprav je preprečevanje in izločanje šuma stara toliko kot je stara obdelava signalov, še to ne pomeni, da je to preprosto opravilo. Omenimo dva izmed bolj znanih pristopov za odpravljanje šuma:

1.) V analogni avdio tehniki je znan sistem Dolby [34]. Ker se pri zapisovanju signala prikrade šum, ga predhodno ojačamo za nek faktor. Pri branju ga, skupaj s šumom, nazaj oslabimo za isti faktor. Na koncu ima koristni signal spet enako jakost kot jo imel na začetku pred zapisovanjem, šum pa je oslabljen za faktor ojačanja. S tem izboljšamo razmerje med koristnim signalom in šumom. Praktična posledica uporabe sistema Dolby je pristnejša reprodukcija zvoka, saj tako poslušalec sliši manj šumenja oziroma motenj v predvajanem zvoku.

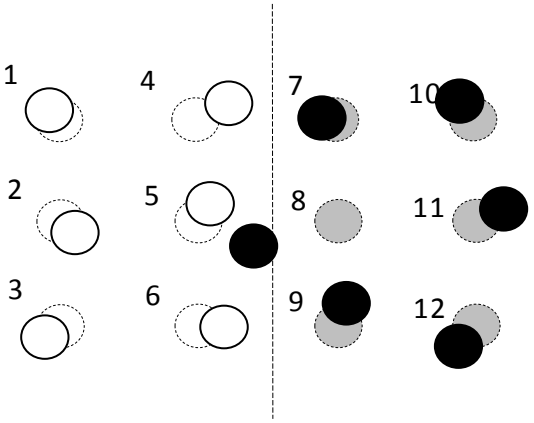
2.) V digitalni tehniki je znano dodajanje paritetnih bitov [35]. Digitalni signal je že v svojem temelju bolj odporen na šum, saj je bil že zasnovan s tem namenom. Če se šum v digitalnem signalu vseeno pojavi, nam paritetni biti omogočajo njegovo zaznavanje in odpravljanje. To poteka tako, da pri pošiljanju zaporedja bitov znane dolžine dodamo en bit. Po dodajanju bita je skupno število enk v zaporedju vedno parno. Če pri sprejemanju bitov naštejemo neparno število enk, potem vemo, da je med prenosom signala nekje prišlo do napake in pošiljanje bitov ponovimo.

4.2 Šum in strojno učenje

Pri strojnem učenju nas predvsem zanima:

- vpliv šuma na lastnosti učnih vzorcev,
- vpliv šuma na potek učenja in
- vpliv šuma na rezultate učenja.

Včasih avtorji različnih raziskav namenoma dodajo nekaj šuma h koristnemu signalu. To se imenuje tudi stresanje (angleško *jitter*) signala. Zaradi lažjega



Slika 13 Učni vzorci s šumom

Črtkano so narisani učni vzorci pred dodajanjem šuma. Vzorec št. 8 je po dodajanju šuma prestopil

predstavljanja bomo spet uporabili čim bolj preproste učne podatke (dvodimenzionalne, dvorazredne podatke),čeprav bo veljalo vse povedano tudi za kompleksne podatke z več dimenzijami ter več razredi, saj vso teorijo gradimo na evklidskih razdaljah. Opazovali bomo vedenje učnih vzorcev, ki se pred dodajanjem šuma ne prekrivajo oziroma jih lahko loči premica (linearno ločljivi vzorci). Po dodajanju šuma (stresanju) se položaj posameznih učnih vzorcev nekoliko spremeni. Večji kot je šum, močneje se spremeni začetni razpored učnih vzorcev. Omenili smo že, da je šum moteča komponenta signala z naključnim značajem. To pomeni, da šum premakne učne vzorce v poljubno smer za poljubno razdaljo. Jakost šuma smo med raziskovanjem omejili in jo postopoma povečevali, ter tako ugotavljali kako vpliva šum različnih jakosti na mejno črto in prekrivanje vzorcev. Pri šibkem šumu je bil premik učnih vzorcev majhen. V tem primeru so (skoraj) vsi učni vzorci ostali na isti strani mejne premice, kot so bili že pred dodajanjem šuma. Pri naraščanju šuma se je število učnih vzorcev, ki so prestopili

mejo, povečevalo. Tiste vzorce, ki so prestopili izhodiščno mejno črto smo poimenovali »prestopniki«.

Učni vzorec 8 na sliki 13 (črn krog) je zaradi šuma prestopil na drugo (napačno) stran mejne črte in se s tem približal vzorcem nasprotnega razreda (beli krogi). Črtkana črta prikazuje vzorec pred stresanjem, polna pa po njem. Predvsem se je vzorec 8 približal vzorcu številka 5. Ta dva vzorca sta tvorila mejni par že pred dodajanjem šuma, zdaj pa sta si še bližje drug drugemu. Tudi vzorca 4 in 7 tvorita mejni par, čeprav nobeden o njiju ni prestopnik. V obravnavanem primeru bi torej zadostovalo že, če bi razšumljali (vračali proti izhodišču) samo vzorec 8.

V resničnih okoliščinah ne poznamo izhodiščnega položaja učnih vzorcev (njihovega položaja pred stresanjem), zato tudi ne moremo določiti pravilnega položaja mejne črte (črtkana črta na sliki 13). Posledično tudi ne vemo kateri učni vzorci so prestopniki. Med strojnim učenjem iščemo takšno mejno črto, ki bi čim bolje razmejila originalne (neznane) učne vzorce, oziroma bo čim bolje razmejila testne vzorce obeh razredov med seboj. Pri tem nam prestopniki povzročajo dve težavi:

- Mejna črta se je zaradi stresanja premaknila.
- Mejna črta zaradi stresanja ni več premica.

Za uspešno razmejitev premaknjenih učnih vzorcev pogosto potrebujemo eno ali več dodatnih mejnih premic. Mejna črta postane odsekoma linearna. Večje kot je število prestopnikov, več dodatnih odsekov oziroma več mejnih premic je potrebnih.

Pri vrednotenju učnih rezultatov se pojavi še dodatna težava. Šum vsakega vzorca se razlikuje. To ne velja samo za učne vzorce, ampak tudi za testne. Zato se lahko zgodi, da šum pomakne učne vzorce in s tem mejno črto v eno smer, testne vzorce pa v drugo smer. Vzroka za napačno razvrščanje testnih vzorcev sta torej dva:

- Šum učnih vzorcev povzroči, da med učenjem določimo napačno lego mejne črte.

- Šum testnih vzorcev povzroči, da ti prestopijo mejno črto.

Med učenjem želimo mejno črto postaviti čim bližje položaju, kot bi ga dobili, če učni vzorci ne bi vsebovali šuma. Zavedati se moramo, da bo del testnih vzorcev, zaradi njihovega lastnega šuma, napačno razvrščen tudi, če bi nam uspelo postaviti mejno črto povsem natančno (idealna naučenost). Pri gradientni učni metodi backpropagation te težave blažimo z zgodnjo ustavitvijo učenja (angleško *early stopping*). Učenje zaustavimo takoj, ko začne naraščati testna napaka, čeprav se učna napaka še vedno zmanjšuje. Iskanje optimalnega položaja mejne črte ni enostavno, ker ne vemo za koliko in v katero smer so bili premaknjeni posamezni učni vzorci. Dobra novica pri odpravljanju šuma je ta, da zadostuje že, če se učni vzorci vrnejo na pravo stran optimalne mejne črte. Pri tem ni potrebno, da se vrnejo natančno v izhodiščni položaj, ampak je dovoljen zamik. Poglejmo si primer. Vzorec 8 na sliki 13 se mora vrniti nazaj na desno stran črtkane črte, pri tem njegova višina ni tako pomembna. Če dobimo po razšumljanju, kljub zamaknjenim učnim vzorcem, lego mejne črte, ki je podobna originalni, je bilo odpravljanje šuma uspešno.

Sedaj smo seznanjeni z okoliščinami razšumljanja, še vedno pa se nismo odločili kako bomo izvedli razšumljanje učnih vzorcev. Odločiti se moramo katere učne vzorce bomo premikali med razšumljanjem, v katero smer in za koliko, da bo razšumljanje čim bolj učinkovito. Možni so različni pristopi. Nekoliko kasneje bomo predlagali pristop razšumljanja, ki temelji na uporabi mejnih parov. Za njegovo razumevanje se moramo najprej seznaniti z rezultati raziskave o tem, kako šum in še nekateri drugi parametri vplivajo na mejne pare. Preden se seznanimo z raziskavo še enkrat poudarimo, da želimo razšumljati podatke iz resničnega sveta, za katere je značilno:

- Ne poznamo pravih vrednosti učnih vzorcev.
- Ne vemo kateri učni vzorci so prestopniki.
- Jakost šuma ni znana.
- Ni znan položaj optimalne mejne črte.

4.3 Vpliv šuma in drugih lastnosti podatkov na mejne pare in prestopnike

V splošnem se nabori učnih vzorcev med seboj razlikujejo po številnih lastnostih:

- po številu dimenzij vhodnega prostora
- po številu dimenzij izhodnega prostora
- po zalogi vrednosti vhodnih podatkov (zvezni, diskretni)
- po definicijskem območju izhodnih podatkov (razvrščanje, regresija)
- po razmerju števila pozitivnih in negativnih vzorcev
- po številu učnih vzorcev
- po vsebnosti šuma
- po vsebnosti časovne komponente (časovno urejeni podatki, brezčasni podatki)
- po časovni spremenljivosti (angleško *concept drift*).

Zanima nas kako naštete lastnosti vplivajo na mejne pare in na prestopnike. Iz ugotovitev v tej raziskavi bomo poskušali izluščiti informacije, ki bi nam lahko koristile pri razšumljanju, rojenju in razvrščanju vzorcev. Nekatere od dobljenih informacije so lahko koristne že v fazi učenja:

- kako zahtevni so učni vzorci,
- ali je smiselno razšumljanje učnih vzorcev,
- kakšna zgradba MLP ustreza učnim vzorcem,...

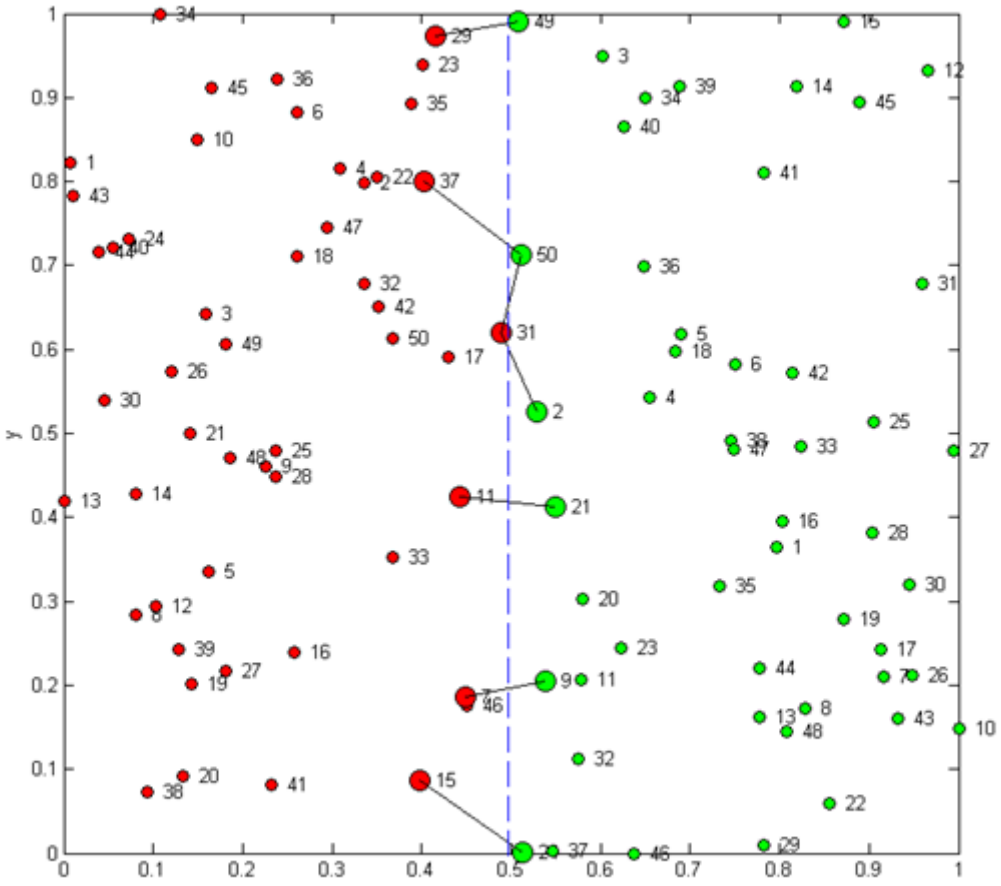
Pri algoritmih, ki temeljijo na iterativnem gradientnem zmanjševanju učne napake (backpropagation in njegove izpeljanke), tovrstnih informacij ne pridobimo niti po končanem učenju.

Celotno raziskavo bomo izvajali s sintetičnimi učnimi podatki, saj je edino tako mogoče pridobiti dovolj veliko količino dovolj raznolikih učnih podatkov, ki jim ločeno določamo posamične parametre. Da bomo vseeno ostali zvesti ideji razšumljanja resničnih učnih vzorcev, bomo pred vsako raziskavo učne podatke stresali. Za izhodišče bomo vedno uporabljali iste podatke, ki jim bomo v vsaki posamezni raziskavi spreminjali le en parameter. Poglejmo si lastnosti izhodiščnih podatkov:

- so dvodimenzionalni,
- pripadajo dvema razredoma in
- so linearno ločljivi.

Podatkovna razreda se imenujeta pozitivni in negativni, ter oba štejeta po 50 vzorcev. Vzorci so naključno generirani in enakomerno porazdeljeni na intervalu med 0 in 1 po obeh dimenzijah. Primer takšnih vzorcev prikazuje slika 14. Modra črtkana črta prikazuje optimalno mejno črto, rdeči krogi predstavljajo negativne učne vzorce, zeleni pa pozitivne. Udeleženci mejnih parov so ponazorjeni z večjimi krogi. Udeleženca istega mejnega para sta med seboj povezana s črno črto. Na sliki vidimo sedem mejnih parov. Isti učni vzorec je lahko udeležen tudi v več parih. V primeru na sliki 14 velja to za pozitivni vzorec št. 50. Ker podatki ne vsebujejo šuma, so vsi zeleni krogi na desni strani mejne črte, vsi rdeči pa na levi.

V resničnem svetu imajo učni vzorci največkrat naravno porazdelitev. To pomeni, da so učni vzorci istega razreda zgoščeni okoli svojega središča, v neposredni bližini mejne črte pa že nekoliko razredčeni. Pri enakomerni porazdelitvi učnih



Slika 14 Primer izhodiščnih učnih vzorcev

podatkov je torej gneča vzorcev v bližini mejne črte večja kot bi bila pri normalni porazdelitvi. Zato dobimo po dodajanju šuma enakomerno porazdeljenim vzorcem tudi več prestopnikov. Posledično je težje tudi vračanje prestopnikov na pravo stran mejne črte.

4.3.1 Vpliv števila vzorcev na število mejnih parov

V prvi raziskavi nas zanima kako vpliva število učnih vzorcev na število mejnih parov. Izhodiščnim podatkom smo spreminjali število učnih vzorcev od deset do dvesto po logaritemski skali, pri čemer je bilo število pozitivnih vzorcev vedno enako številu negativnih. Podatki v tabeli 4 prikazujejo rezultate te raziskave. V prvem stolpcu tabele je podano število vzorcev v enem razredu. Ugotavljamo število učnih vzorcev (absolutno in relativno), ki so udeleženi v mejnih parih. Relativno število oziroma delež smo izračunali po enačbi 10.

$$relativni\ delež\ udeležencev = \frac{število\ mejnih\ parov}{število\ vzorcev\ v\ razredu} \qquad \textbf{Enačba 10}$$

Tabela 4 Vpliv števila vzorcev na delež mejnih parov

Število vzorcev v enem razredu	Povprečno število mejnih parov	Relativni delež udeležencev
10	2,96	0,2960
20	4,08	0,2040
50	6,25	0,1250
100	9,42	0,0942
200	13,0	0,0650

Število mejnih parov in relativni delež udeležencev sta bila izračunana z računalniško simulacijo. Ker lega vzorcev v simulaciji temelji na uporabi naključnih števil, smo jo 100 krat ponovili in na koncu izračunali povprečne rezultate. Prva vrstica tabele 4 prikazuje rezultat za 10 vzorcev v enem razredu oziroma na vsaki strani mejne črte. Pri takšni količini učnih vzorcev so povprečno nastali malo manj kot trije mejni pari, oziroma malo manj kot šest izmed dvajsetih vzorcev je

udeleženih v mejnih parih. Z večanjem števila vzorcev se tudi število mejnih parov zvišuje, vendar ne tako hitro kot število vzorcev. Relativni delež udeležencev se z naraščanjem števila vzorcev zmanjšuje. Intuitivno pričakujemo, da za določitev mejne črte zadostujejo že vzorci oziroma mejni pari, ki ležijo v njeni bližini. V tem primeru bo zmanševanje relativnega deleža udeležencev v mejnih parih olajšalo načrtovanje mejne črte. V tabeli 4 lahko vidimo, da dvajsetkratno povečanje števila učnih vzorcev (iz 10 na 200) pomeni le približno štirikratno povečanje števila mejnih parov (iz 2,96 na 13,00).

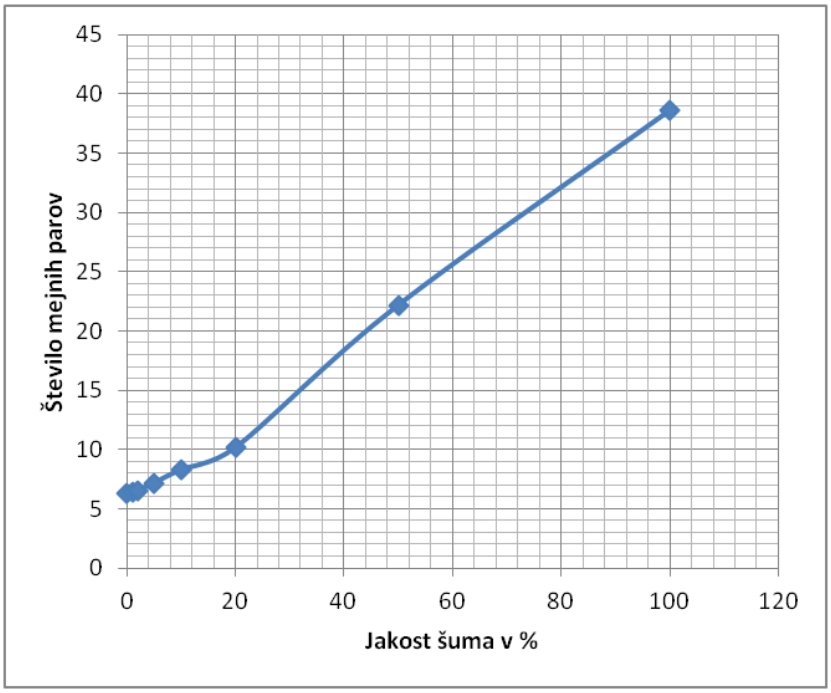
4.3.2 Vpliv šuma na število mejnih parov

V tej raziskavi nas zanima kako jakost šuma vpliva na število mejnih parov. Rezultati raziskave so podani v tabeli 5. Ugotovimo lahko, da s povečevanjem šuma narašča tudi število mejnih parov. Jakost šuma je podana v odstotkih od celotnega razpona vhodne vrednosti. V primeru, ko dimenzija učnega vzorca X zavzema vrednosti med 0 in 1, je njen celotni razpon enak 1 in če znaša šum 2% to pomeni, da je vrednost te dimenzije po dodajanju šuma znotraj intervala X-0,01 in X+0,01. Enako velja tudi za dimenzijo Y.

Tabela 5 Vpliv jakosti šuma na število mejnih parov

Jakost šuma v %	Povprečno število mejnih parov
0	6,25
1	6,38
2	6,49
5	7,14
10	8,27
20	10,13
50	22,17
100	38,62

Posamezne dimenzije podatkov se lahko torej pri 2 odstotnem šumu spremenijo v pozitivno ali negativno smer za največ en odstotek od celotnega razpona dimenzije. Za vsako jakost šuma je bilo opravljeno sto simulacij. V tabeli 5 so podane povprečne vrednosti za vseh 100 rezultatov simulacije. Iz dobljenih rezultatov (tabela 5) je razvidno, da se s povečevanjem šuma, monotono povečuje tudi število mejnih parov.



Slika 15 Število mejnih parov v odvisnosti o jakosti šuma

4.3.3 Vpliv razmerja učnih vzorcev na število mejnih parov

Pri tej raziskavi nas je zanimalo kako vpliva razmerje med številom pozitivnih in negativnih učnih vzorcev na število mejnih parov. Začeli smo z desetimi pozitivnimi in devetdesetimi negativnimi vzorci. Nato smo število pozitivnih vzorcev povečevali za deset, število negativnih pa zmanjševali za deset. Z

raziskavo smo ugotovili, da razmerje med številom pozitivnih in negativnih vzorcev vpliva na število mejnih parov. To je bilo tudi pričakovano. Namreč, če bi imeli vse vzorce iz samo enega razreda, ne bi mogli tvoriti niti enega mejnega para. Največje število mejnih parov dobimo kadar je število vzorcev obeh razredov enako (glej tabelo 6).

Tabela 6 Vpliv razmerja med vzorci na število mejnih parov

Število pozitivnih vzorcev	Število negativnih vzorcev	Delež udeležencev
10	90	0,0774
20	80	0,1034
30	70	0,1222
40	60	0,1236
50	50	0,1250
60	40	0,1237
70	30	0,1225
80	20	0,1031
90	10	0,0779

Iz te raziskave ugotavljamo, da je v manjšinskem razredu smiselno imeti vsaj 20 odstotkov učnih vzorcev, sicer nam število mejnih parov močno upade. Z malim številom mejnih parov je določanje mejne črte oteženo oziroma manj natančno.

4.3.4 Vpliv šuma na prestopnike

Pri tej raziskavi nas zanima kako vpliva jakost šuma na število prestopnikov. Rezultati te raziskave nam bi utegnili koristiti pri odpravljanju šuma. Ločeno smo opazovali dve vrsti prestopnikov – udeležence in neudeležence v mejnih parih. Rezultati raziskave so podani v tabeli 7. Med to raziskavo smo prišli do naslednjih zanimivih spoznanj:

1.) Število prestopnikov narašča s stopnjo šuma. Pri malem šumu ostanejo skoraj vsi učni vzorci na isti strani mejne črte (skorajda ni prestopnikov). Pri večjem šumu se število prestopnikov poveča. Povečanje števila prestopnikov je približno premo sorazmerno z jakostjo šuma.

2.) Velika večina prestopnikov je udeležena v mejnih parih (pri 10% šumu je udeležba 95,58%).

3.) Velika večina vzorcev, ki niso udeleženci mejnih parov, niso prestopniki (pri 10% šumu je med neudeleženci ~99,87% neprestopnikov).

Tabela 7 Vpliv šuma na prestopnike

Šum (%)	Povprečno število prestopnikov	Povprečno št. udeleženih prestopnikov	Povpr. št. mejnih parov	Povprečno št. udeležencev	Delež prestopnikov v mejnem paru (%)	Delež neprestopnikov med vzorci izven mejnih parov (%)
0	0,00	0,00	6,37	11,33	-	100,00
1	0,26	0,25	6,44	11,41	96,15	99,99
2	0,62	0,59	6,55	11,55	95,16	99,97
5	1,19	1,12	7,07	12,11	94,12	99,92
10	2,49	2,38	7,46	12,46	95,58	99,87
20	4,69	4,27	10,32	15,57	91,04	99,50
50	12,43	11,31	22,26	30,15	90,99	98,40
100	25,38	22,66	39,90	52,93	89,28	94,22

4.4 Princip razšumljanja po metodi mejnih parov

S pomočjo ugotovitev v točkah 2.) in 3.), v naslovu Vpliv šuma na prestopnike, smo prišli do naslednjih dveh sklepov:

1.) Učnih vzorcev, ki niso udeleženi v mejnih parih ni smiselno razšumljati (premikati), saj je zelo velika verjetnost, da niso prestopniki.

2.) Učne vzorce, ki so udeleženi v mejnih parih bomo razšumljali (premikali). To bomo naredili tako, da jih bomo približevali najbližjemu vzorcu (najbližjim vzorcem) iz istega razreda, ki niso udeleženi v mejnem paru. Lahko jih tudi oddaljujemo od najbližjih neudeleženih vzorcev nasprotnega razreda. V primeru vzorca 8 na sliki 13 bi to pomenilo, da ga premikamo k vzorcu 11 in vstran od vzorca 2.

Oglejmo si podrobneje kaj pomenita ta dva sklepa za razšumljanje pri 10% šumu. Razšumljali bi le 12,46% od vseh učnih vzorcev, s tem bi zajeli 95,58% prestopnikov in bi jih v 99,87% premikali v pravo smer. Za koliko jih bomo premikali k zanesljivim vzorcem je stvar naše odločitve. Kadar je šum močnejši, je morda smiselno močnejše razšumljanje (večji premiki učnih vzorcev). Skrajna možnost razšumljanja podatkov je, da podatke iz mejnih parov izločimo iz učnih vzorcev. To je enako, kot če bi prestopnika v celoti približali zanesljivemu vzorcu. Vse kar smo omenili glede razšumljanja je bilo vezano le na določanje evklidske razdalje, včasih neposredno, pri pojmu mejni par pa posredno, zato veljajo vse te ugotovitve tudi za tri in več dimenzionalne podatke.

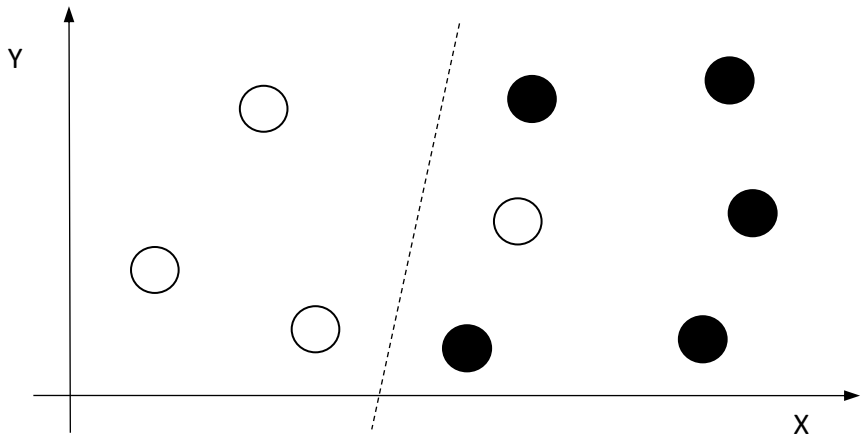
Potrjevanje hipoteze 6: »Z metodo BPM lahko zmanjšamo šum v učnih vzorcih.«

Ugotovili smo, da s premikom učnih vzorcev, ki so udeleženi v mejnih parih, k isto razrednim vzorcem, ki niso udeleženi v mejnih parih, izboljšamo njihovo lego oziroma zmanjšamo njihov šum. S tem je hipoteza 6 potrjena.

5 Rojenje podatkov z metodo mejnih parov

5.1 Splošno o rojenju

Rojenje učnih vzorcev (angleško *clustering*), imenujemo postopek s katerim razdelimo učne vzorce v vhodnem prostoru na dve ali več homogenih področij [70]. Neko področje imenujemo homogeno, kadar vsebuje učne vzorce samo



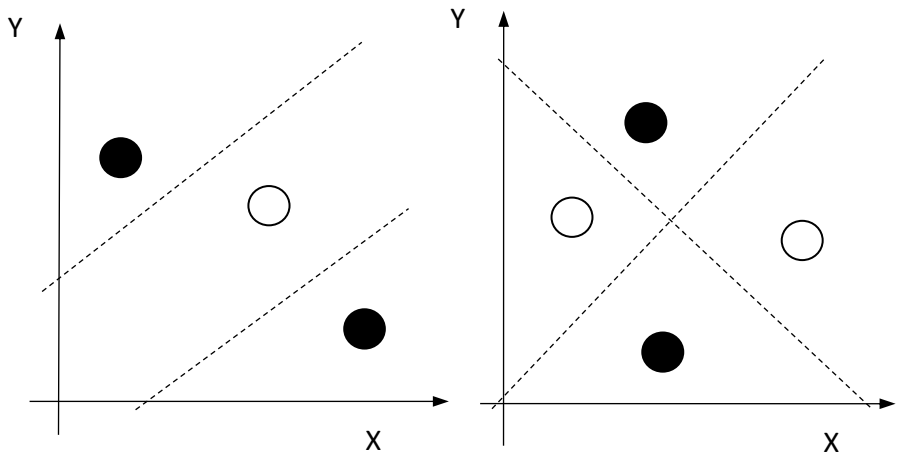
Slika 16 Homogenost področij

Področje desno od črtkane črte ni homogeno, ker vsebuje en vzorec nasprotnega razreda (bel krog).

enega razreda. Znova bomo obravnavo začeli z dvodimenzionalnimi učnimi vzorci in jo kasneje posplošili na prostor z več dimenzijami. Meja med področji v dvodimenzionalnem prostoru je črta, v tridimenzionalnem prostoru črte nadomestijo ploskve, v štiri in v več dimenzionalnem prostoru pa ploskve

nadomestijo hiper ploskve. Črte, ploskve in hiper ploskve so v splošnem lahko tudi ukrivljene oz. nelinearne. V naši disertaciji se omejujemo na linearno razmejevanje, saj bomo v nadaljevanju ta algoritem uporabili za učenje MLP, kjer se uporabljajo le LTU nevroni, ki imajo linearno mejno črto oziroma (hiper)ploskev.

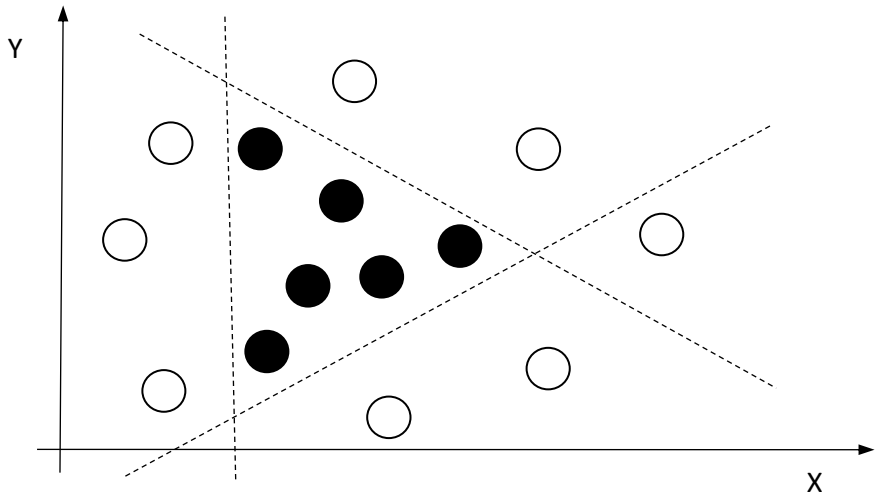
V najbolj ugodnem primeru lahko ločimo vzorce med seboj z eno samo premico



Slika 17 Področji z dvema mejnima črtama.

Leva slika vsebuje tri homogena področja, desna pa štiri

(linearna ločljivost) in tako dobimo dva roja. Kadar ena sama mejna premica ne zadostuje, jih lahko uporabimo tudi več. S tem, ko povečamo število premic naraste tudi število področij. Če vsaka dodana premica preseka vsa dotedanja področja v ravnini, se število področij podvoji. Tako dobimo z N premicami 2^N področij. To je največje možno število področij za dano število premic. Kadar se premice v ravnini ne sekajo, nam vsaka dodatna premica doda eno področje. Tako dobimo z N premicami N+1 področij. To je najmanjše možno število področij za dano število premic.



Slika 18 Področje v obliki konveksnega otoka.

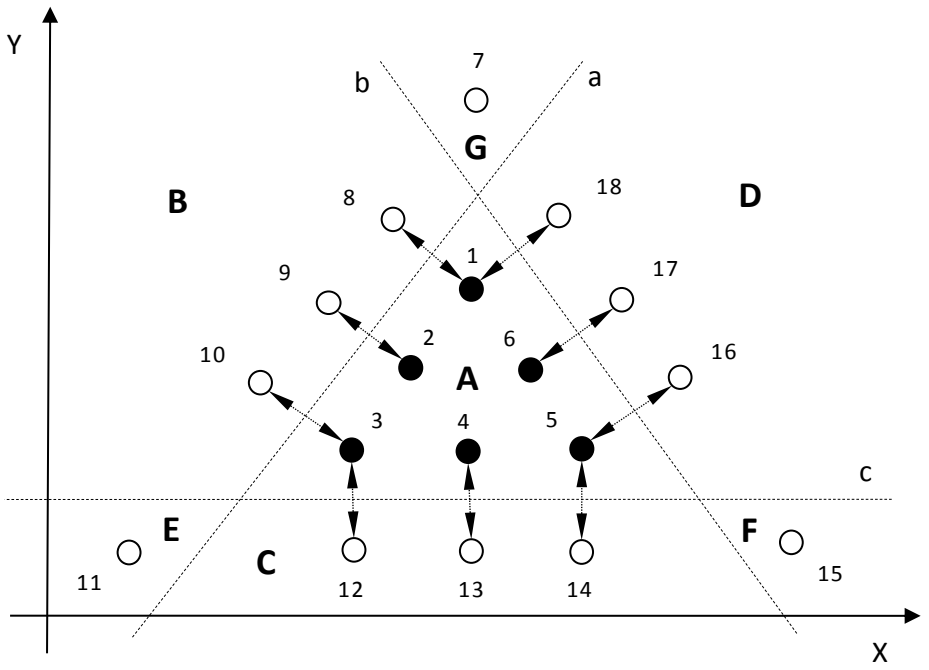
Mejna črta je sklenjena in vsi vogali so izbočeni

Za dve premici torej velja, da je število področij od 3 (=2+1) do 4 (=2^2). Slika 17 prikazuje ti dve možnosti razmejevanja vhodnega prostora z dvema premicama. Slika 17 levo prikazuje primer, ko se mejni premici ne sekata in se vhodni prostor razdeli na tri dele. V primeru na sliki 17 desno se premici sekata in vhodni prostor razdelita na štiri področja. Pri treh in več premicah lahko nastanejo zaključena področja, ki jih imenujemo otoki. Primer takega otoka s polnimi krogi, ki je obdan s tremi premicami, prikazuje slika številka 18. Otoki so v splošnem različni mnogokotniki. Če so vsa oglišča otoka izbočena, je otok konveksen, sicer je konkaven.

5.2 Rojenje in mejni pari

Osnovna ideja rojenja z mejnimi pari je preprosta. Med oba vzorca istega mejnega para narišemo črto (hiperploskev), ki ju razmejuje in jo poimenujemo »mejna črta« (»mejna hiperploskev«). Mejne črte razdelijo celotni vhodni prostor na več manjših prostorov oziroma področij. Če so posamezna področja učnih vzorcev ali več sosednjih področij homogena, jih imenujemo »roj«. Poučen primer nabora učnih vzorcev je prikazan na sliki 19, ki vsebuje 18 vzorcev, od tega jih je

šest pozitivnih (polni krogi) in dvanajst negativnih. Vzorci tvorijo devet mejnih parov ((1, 8), (2,9),…, (1,18)), ki so na sliki označeni z dvoglavo puščico. Nekateri učni vzorci niso udeleženi v nobenem mejnem paru (npr.: vzorec 7), nekateri drugi pa so udeleženi celo večkrat. Vzorec 1 je na primer udeležen v dveh mejnih parih. Mejni pari na sliki 19 so razporejeni v obliki trikotnika. Udeležence posameznih mejnih parov ločimo z mejno črto, pri čemer prekrižamo dvoglave puščice . Ista



Slika 19 Nabor učnih podatkov z mejnimi črtami in področji

1, 2, 3,…18: učni podatki

A, B, C,…G: področja

a, b, c: mejne črte

mejna črta lahko loči tudi več mejnih parov. Pri tem velja, da morajo biti ločeni vzorci istega razreda v isti polravnini. Za ločitev vseh parov je torej potrebno kvečjemu toliko mejnih črt kolikor je mejnih parov. Če z isto črto ločimo dva ali več mejnih parov, smo s tem uspeli zmanjšati število potrebnih mejnih črt. V primeru vzorcev na sliki 19 zadostujejo tri mejne črte, ki so poimenovane z malimi tiskanimi črkami a, b in c. Vsaka od njih ločuje tri mejne pare. Na primer črta a

ločuje pare (1,8), (2,9) in (3,10). Tako je nastalo sedem področij, ki so poimenovana z velikimi tiskanimi črkami A, B, C, D, E, F in G.

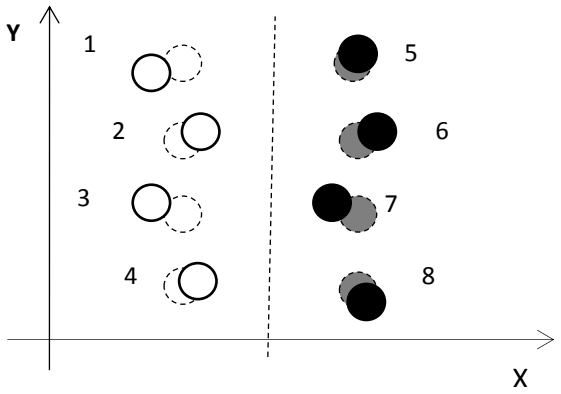
Ideja za pravkar opisan postopek rojenja z metodo mejnih parov je nastala med preučevanjem delovanja naučenega MLP. Takrat je bilo namreč ugotovljeno, da se med učenjem MLP v prvem sloju odvija pravkar opisan postopek rojenja. Pri proučevanju se nismo omejili zgolj na učne podatke. Še posebej smo se osredotočili na dogajanje v bližini mejnih črt, kjer smo v majhnih korakih spreminjali vhodno vrednost MLP (pomikanje učnih podatkov) in opazovali odziv posameznih nevronov v notranjih slojih, ter pri tem prišli do naslednjih ugotovitev:

- Kljub temu, da je izhodna vrednost nevrona zvezna funkcija vhodnih spremenljivk, je na njegovem izhodu skoraj vedno le vrednost zelo blizu 0 in 1. Izhodne spremembe nevronov so sunkovite - nevroni delujejo v zasičenju. Izjema je zelo ozko področje v neposredni bližini mejne črte.

- Dokler ostanemo znotraj istega področja se vrednost na izhodu prvega sloja ne spreminja in posledično tudi ostali sloji ostanejo nespremenjeni. Vhodni prostor je razdeljen na več homogenih področij - rojev.

- Ko vhodni signal leze od nekega učnega vzorca do bližnjega vzorca nasprotnega razreda, eden od nevronov v prvem sloju preklopi. To pomeni, da prečkamo mejno črto oziroma preidemo iz enege roja v drugega.

Če pravkar omenjene ugotovitve posplošimo na celoten vhodni prostor velja, da vsakemu področju pripada binarna koda. Dolžina kode je enaka številu mejnih črt v vhodnem prostoru oziroma številu nevronov v prvem sloju. Posamezni biti te kode nam povedo na katerem bregu posamezne mejne črte se nahajamo.



Slika 20 šumni podatki

Črtkana črta - vzorec brez šuma

Polna črta – vzorec s šumom

Področne kode so značilke. Vse povedano o določanju značilk temelji zgolj na uporabi mejnih parov, ki so določeni na osnovi evklidske razdalje, zato lahko teorijo določanja značilk posplošimo tudi na prostor s tremi in več dimenzijami.

Potrjevanje hipoteze 4: »Z metodo BPM lahko poiščemo značilke v vhodnem prostoru.«

Na podlagi pravkar omenjega ugotavljamo, da je hipoteza 4 potrjena.

Sedaj si oglejmo vpliv šuma na položaj mejne črte. Vpliv šuma na posamezne učne vzorce že poznamo. Če učnim podatkom dodamo malo šuma, se učni vzorci nekoliko premaknejo in sicer vsak v drugo smer – slika 20. Šum nekaterih vzorcev želi premakniti mejno premico v levo (vzorci 1, 3 in 7), šum ostalih vzorcev (vzorci 2, 4, 5, 6 in 8) pa v desno.

Tabela 8 Kode področij vhodnega prostora

področje	mejna črta		
	a	b	c
A	1	1	1
B	0	1	1
C	1	1	0
D	1	0	1
E	0	1	0
F	1	0	0
G	0	0	1

Ker je njihov vpliv na mejno črto nasprotujoč, se delno izničuje in se zato položaj mejne premice, ki je obdana s številnimi mejnimi pari, zaradi šuma skorajda ne spremeni. To velja dokler je šum dovolj majhen, da učni vzorci ne prestopijo mejne črte. Pravkar smo ugotovili, da rojenje z metodo mejnih parov ni občutljivo na mali šum v učnih podatkih.

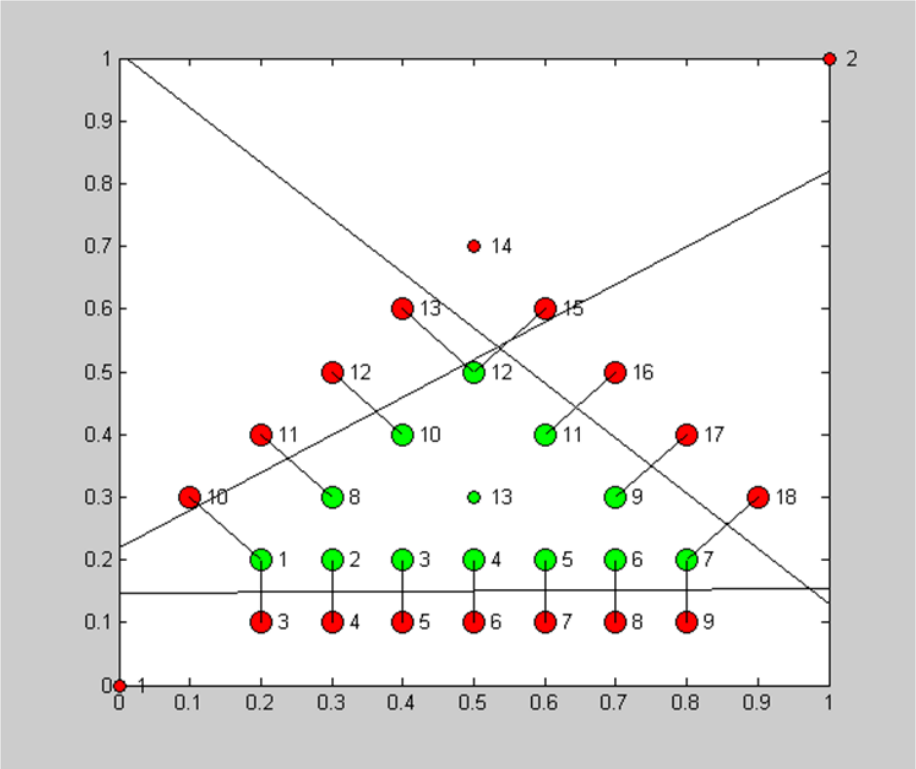
5.3 Preverjanje ideje rojenja z mejnimi pari

Rojenje nabora podatkov na sliki 19 je dalo pričakovan rezultat. Kar pa še ne pomeni, da podana ideja deluje za poljuben nabor učnih podatkov. Zato smo omenjeno idejo dodatno testirali s podatki mnogih različnih učnih naborov. V ta namen smo napisali simulacijski program, ki poišče mejne črte (ploskve, hiper ploskve) z razmejevanjem mejnih parov. Pri tem nas zanima ali so res vsa dobljena področja homogena. To bi namreč pomenilo, da je bilo rojenje uspešno. Za testiranje homogenosti smo uporabili iste nabore učnih vzorcev, ki smo jih že spoznali v poglavju »Analiza gradientnih učnih metod« pod naslovom »Učenje z gradientnimi metodami«. To so logična funkcija ALI, logična funkcija XOR ter črka O. Tudi te nabore je obravnavana metoda rojila uspešno, saj so vsa področja, ki so nastala z razmejevanjem mejnih parov, bila homogena. Sprva smo že dobili vtis, da je opisani postopek rojenja dober za vse nabore učnih vzorcev, vendar se je

kmalu izkazalo, da obstajajo tudi primeri, ko takšno rojenje ne deluje brezhibno. Poglejmo si kakšne rezultate je dala računalniška simulacija.

5.3.1 Rojenje trikotnika

Začnimo s trikotniškim naborom učnih podatkov, ki se tokrat nekoliko razlikuje od prejšnjega trikotniškega nabora podatkov. Na sliki 21 vidimo, da so mejni pari



Slika 21 Uspešno rojenje trikotnikovih učnih podatkov

tokrat nekoliko bolj razmaknjeni. Še posebej to velja za pare v bližini poševnih stranic trikotnika. Na primer par (1, 10) je širši od para (1, 3). Mejne črte na sliki so rezultat računalniške simulacije rojenja. Dobili smo pričakovano število mejnih črt (tri). Kljub temu so razmere nekoliko drugačne kot na sliki 19. Par (12, 15) je razmejen dvakrat. Celotna površina vhodnega prostora je tokrat razdeljena na pet področij, če štejemo samo tista, ki vsebujejo vsaj en učni vzorec. Vseh pet

področij je homogenih, saj področje na sredi slike vsebuje samo zelene kroge, preostala štiri področja pa samo rdeče. Torej je bilo tudi v tem primeru rojenje uspešno. Novo spoznanje pa je, da je možno razmejiti mejne pare na več različnih načinov.

V začetni fazi raziskav, ko je hipoteza rojenja z mejnimi pari šele nastajala, smo predvidevali, da bomo z razmejitvijo vseh mejnih parov dobili zgolj homogena področja. Zato smo sprva uporabljali preprost algoritem za rojenje učnih vzorcev:

Algoritem 1 Preprost algoritem za rojenje nabora učnih podatkov

1. korak: *Poišči mejne pare.*

2. korak: *Loči vse mejne pare s čim manj mejnimi črtami.*

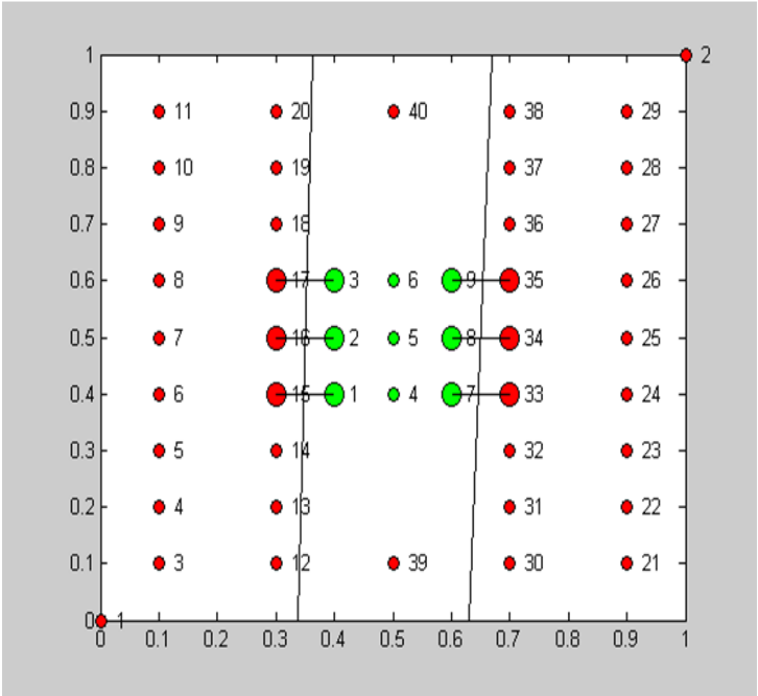
5.3.2 Rojenje kvadrata

V mnogih primerih je bilo rojenje z algoritmom 1 uspešno. Na primer, pri ekvidistančnih učnih vzorcih geometrijskih likov, kot sta trikotnik in kvadrat. Ko smo uporabili bolj zahtevne nabore učnih vzorcev pa smo ugotovili, da algoritem 1 ni vedno uspešen. Včasih je po risanju mejnih črt ostalo nekaj učnih vzorcev še vedno na napačni strani črte, čeprav so bili razmejeni že vsi mejni pari. Tako so nekatera področja ostala heterogena. Poglejmo si podrobneje en tak primer: učne vzorce na sliki 22 smo dobili tako, da smo najprej vzeli vse ekvidistančne vzorce kvadrata in nato nekaj vzorcev odstranili. V zgornjem delu slike so odstranjeni vzorci med pozitivnim, zelenim vzorcem številka 6 in negativnim, rdečim vzorcem številka 40. Na podoben način smo odstranili tudi nekaj vzorcev pod kvadratom.

Z algoritmom 1 najdemo le mejne pare na levi in desni strani kvadrata. Za njihovo razmejitev zadostujeta že dve navpični premici, ena na levi strani kvadrata in druga na desni. Ker se ne sekata, razdelita vhodni prostor na tri dele. Na sliki 22 opazimo, da sta področji na levi in desni strani homogeni, saj vsebujeta le negativne, rdeče vzorce. Žal pa to ne velja za področje na sredi, ki je ostalo

heterogeno, saj vsebuje rdeče in zelene vzorce. V srednjem področju sicer prevladujejo zeleni, pozitivni učni vzorci, vendar sta med njimi tudi dva »tujca« - rdeča, negativna vzorca številka 39 in 40.

Poraja se vprašanje, zakaj ta dva vzorca nista udeležena v nobenem od mejnih parov oziroma zakaj smo jih spregledali med tvorbo mejnih parov. Če ločeno opazujemo zgolj srednje področje ugotovimo, da sta oba tukajšnja rdeča, negativna učna vzorca (vzorec številka 39 in 40) dejansko udeležena vsak v svojem mejnem paru. Negativni vzorec 40 tvori mejni par s pozitivnim vzorcem 6, vzorec 39 pa z vzorcem 4. Pri iskanju teh dveh mejnih parov so nas sprva zmotili učni vzorci v sosednjih področjih. Negativni vzorci številka 18, 19, 36 in 37 so ovirali



Slika 22 Pomanjkljivo rojenje

Sredinsko področje s pozitivnimi, zelenimi vzorci ni homogeno. Moteča sta negativna, rdeča vzorca številka 39 in 40. Pozitivni, zeleni vzorec številka 6 in negativni, rdeči vzorec številka 40 pa nista prepoznana kot mejni par, ker so jima na poti negativni, rdeči vzorci številka 18, 19, 36 in 37.

tvorbo gornjega mejnega para (6, 40), negativni vzorci številka 13, 14, 31 in 32 pa so ovirali tvorbo mejnega para (4, 39). Na podlagi tega spoznanja smo naredili sklep, kako izboljšati algoritem tako, da bo našel vse mejne pare med učnimi vzorci. Kadar po razmejevanju mejnih parov ostane kakšno področje heterogeno, ga znova obravnavamo, a tokrat ločeno od ostalih področij. S to nadgradnjo algoritma v vsakem heterogenem področju najdemo nekaj dodatnih mejnih parov, ki smo jih sprva spregledali. Iskanje mejnih parov je zaključeno šele, ko so vsa področja homogena. Izboljšan algoritem iskanja mejnih parov se glasi:

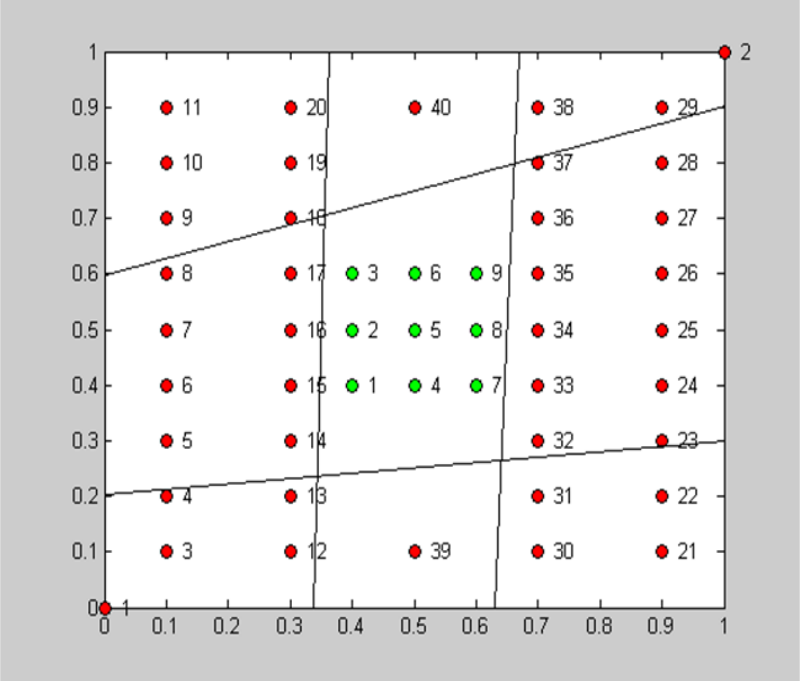
Algoritem 2 Izboljšan algoritem za rojenje nabora učnih podatkov

1. korak: *Poišči mejne pare.*

2. korak: *Loči mejne pare z mejnimi črtami.*

3. korak: *Preveri homogenost nastalih področij.*

4. korak: *Če najdeš heterogeno področje, poišči v njem dodatne mejne pare in nadaljuj z 2. korakom.*



Slika 23 Uspešno rojenje učnih vzorcev

Prikazanih je devet homogenih področij, eno pozitivno in osem negativnih.

Izboljšan algoritem iskanja mejnih parov smo preverili s številnimi nabori učni podatkov, tako z umetnimi kot tudi realnimi. Omenimo samo nekatere med njimi:

- Nabori umetnih podatkov - ALI, XOR, trikotnik, kvadrat in črka O.
- Nabori realnih podatkov – Perunike (angleško *Iris*) [36], Vino (angleško *Wine*) [37] in Petrovo uho (angleško *Abalone*) [38].

Izboljšana metoda rojenja podatkov z mejnimi pari je delovala pravilno, brez izjeme pri vseh testnih naborih podatkov. Rezultati rojenja za vseh šestnajst MLP-jev z metodo mejnih parov z obema algoritmoma so podani v tabeli 9, kjer je razvidno, da izboljšani algoritem (algoritem 2) običajno najde nekaj več mejnih parov od preprostega algoritma (algoritem 1). Razen v enem primeru (nabor podatkov za številko 6) so dodatni mejni pari povzročili tudi dodatne mejne črte. V ekstremnih primerih (Vino 2, Iris Versicolor) je skoraj vsak dodatni mejni par

prispeval dodatno mejno črto. Vse povedano o rojenju temelji zgolj na uporabi mejnih parov, ki so določeni na osnovi evklidske razdalje, zato lahko teorijo rojenja posplošimo tudi na prostor s tremi in več dimenzijami.

Tabela 9 Število najdenih mejnih parov pri različnih učnih vzorcih

Ime nevronske mreže	Število mejnih parov		Število področij	
	algoritem 1	algoritem 2	algoritem 1	algoritem 2
Vino 1	14	18	1	2
Vino 2	25	27	2	4
Vino 3	11	11	1	1
Iris Setosa	2	2	1	1
Iris Versicolor	25	27	2	4
Iris Virginica	12	15	3	5
Števka 0	4	5	2	3
Števka 1	12	16	2	5
Števka 2	5	6	1	2
Števka 3	5	11	1	4
Števka 4	8	11	2	5
Števka 5	8	9	2	3
Števka 6	5	6	1	1
Števka 7	8	10	2	3
Števka 8	6	12	2	6
Števka 9	9	10	2	3

Potrjevanje hipoteze 5: Z metodo BPM lahko poiščemo roje v vhodnem prostoru.

Na podlagi pravkar omenjenih dejstev ugotavljamo, da je rojenje z metodo mejnih parov uspešno in s tem tudi, da je hipoteza 5 potrjena.

5.4 Združevanje mejnih parov

Omenili smo že, da lahko ista mejna črta ločuje dva ali več mejnih parov. Na primer na sliki 19 je prikazano dvanajst mejnih parov: (2, 14); (3, 16); (4, 19); (10, 17); (11, 17); (11, 18); (13, 18); (13, 19); (20, 19); (6, 14); (7, 15) in (8, 17), ki jih uspešno razmejujejo tri mejne črte (črte a, b, in c). Prav tako vemo, da število mejnih črt ustreza številu nevronov v prvem sloju MLP-ja. Torej moramo, če želimo ustvariti majhen MLP, čigar zgradba je blizu optimalne (angleško *near optimal construction*), postaviti mejne črte tako, da bo po tvorbi homogenih področij čim manj mejnih premic oziroma, da posamezne premice ločujejo čim več mejnih parov. Katere mejne pare, oziroma koliko jih naj razmejuje ista premica, je zahtevno vprašanje. Podobno vprašanje se pojavi tudi pri metodi podpornih vektorjev (SVM). Tam se vhodni prostor transformira z uporabo jedrnih funkcij (angleško *kernel function*), v še več dimenzionalen prostor, kjer učni vzorci postanejo linearno ločljivi. Jedrna funkcija je izbrana tako, da je rob (angleško *margin*), oziroma prazen prostor med pozitivnimi in negativnimi vzorci čim širši [42]. Tudi metoda bipropagation, ki je bila omenjena uvodoma, naredi v prvem sloju MLP nekaj podobnega. Medsebojno oddalji vzorce nasprotnega razreda in zbliža vzorce istega razreda. S tem se mejni pari razširijo in njihovo razmejevanje je poenostavljeno.

Pri metodi BPM se transformacije podatkov iz vhodnega prostora v prostor značilk (angleško *features space*) izvede v prvem sloju, kar se opravi med združevanjem mejnih parov vhodnega prostora v skupine, ki jih razmejuje skupna premica. Možnih je več načinov kako združevati mejne pare. Po našem mnenju je to obetavno in pestro raziskovalno področje in hkrati ena od številnih možnosti za nadgradnjo te disertacije. Mi smo izbrali preprosto metodo združevanja, ki deluje po naslednjem algoritmu:

Algoritem 3 Algoritem za združevanje mejnih parov

***1. korak**: Poišči mejne pare.*

***2. korak**: Vzemi nov nevron in ga uči s podatki naslednjega mejnega para.*

***3. korak**: Dodaj nov (bližnji) mejni par in z njim uči isti nevron. Če učenje uspe, potem nov par obdrži in ga označi kot uporabljenega, drugače ga zavrži.*

***4. korak**: Nadaljuj s 3. korakom do zadnjega mejnega para.*

***5. korak**: Nadaljuj z 2. korakom dokler niso uporabljeni vsi mejni pari.*

***6. korak**: Preveri vsa področja vhodnega prostora. Če je katero področje heterogeno, poišči v njem dodatne mejne pare in nadaljuj z 2. korakom.*

Z uporabo algoritma 3 dobimo v vhodnem prostoru mejne črte oziroma nevrone v prvem sloju večslojnega perceptrona. Število tako dobljenih mejnih črt oziroma nevronov je blizu minimalnega. Vrednosti na izhodu pravkar dobljenega prvega sloja so binarne, saj nevroni delujejo v zasičenju. Biologi temu pojavu rečejo vžiganje nevronov. Tisti učni podatki, ki pripadajo istemu roju, dobijo na izhodu prvega sloja enako binarno vrednost (značilko) oziroma se rojio. Število nevronov v prvem sloju je odvisno od zahtevnosti učnih vzorcev (od razporeda mejnih parov) in je lahko tudi mnogo večje kot je število vhodov.

6 Razvrščanje podatkov z metodo mejnih parov

6.1 Opis metode razvrščanja podatkov z metodo mejnih parov

Binarno vrednost na izhodu perceptrona, ki smo jo dobili z rojenjem podatkov z metodo mejnih parov, lahko uporabimo za razvrščanje podatkov v razrede, kar je izvedljivo na več načinov. Eden od njih je Boolova algebra (uporaba logičnih funkcij), saj so podatki od rojenja naprej binarni. V tej raziskavi bomo nadaljevali z uporabo dodatnih perceptronov. Perceptronu ki roji vhodne podatke, kaskadno dodamo več novih perceptronov in tako dobljeno nevronsko mrežo poimenujemo preostanek večslojnega perceptrona (preostanek MLP). Raziskali smo dve možnosti konstruiranja preostanka MLP:

1. Vse nadaljnje sloje smo ustvarili na povsem enak način, kot smo ustvarili prvega. Prvi sloj je imel na vhodu zvezne vrednosti, zato je lahko število nevronov po rojenju podatkov presegalo število vhodov (MLP se lahko v prvem sloju razširi). Naslednji sloji imajo binarne vhodne vrednosti, zato se število nevronov postopoma zmanjšuje ali kvečjemu ostane enako, kot je bilo v predhodnem sloju. Ko v naslednjem sloju ostane le še en nevron, je načrtovanje MLP zaključeno. Nastali MLP ima piramidno obliko.

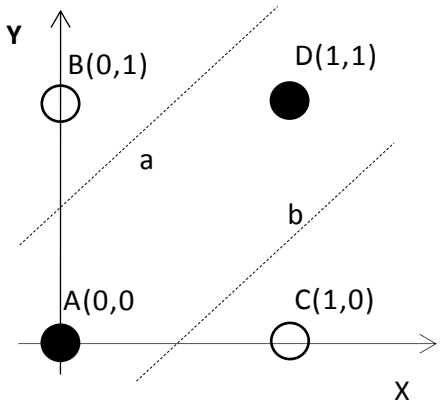
2. Vse nadaljnje sloje obravnavamo kot dodatni MLP, ki ga učimo z eno od uveljavljenih gradientnih metod. Metoda poskusov je pokazala, da je bilo morebitno ozko grlo odpravljeno že v prvem sloju, zato je bilo učenje dodatnega MLP-ja vedno opravljeno zelo hitro in zanesljivo. Za vse nabore učnih vzorcev je krivulja preostale učne napake med učenjem strmo in monotono padala. Pri tem smo imeli občutek, kot da nobena funkcija preostale učne napake dodatnega MLP ne vsebuje lokalnega minimuma.

S tem smo postavili temelje za potrjevanje hipoteze 2: »Z metodo BPM lahko zapleten in negotov iterativni učni postopek razdelimo na več manjših, preprostejših in zanesljivejših neiterativnih postopkov.« Hipotezo bomo potrjevali kasneje z rezultati učenja.

6.2 Učenje z umetnimi učnimi podatki

6.2.1 XOR

Pravkar opisani metodi razvrščanja smo preizkusili s številnimi uveljavljenimi,



Slika 24 Graf učnih vzorcev XOR

A,B, C, D: učni vorci

a, b: mejne črte

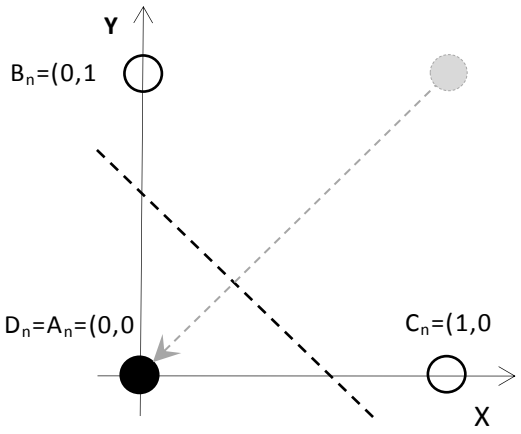
realnimi in sintetičnimi nabori učnih podatkov. Linearno ločljivi nabori učnih vzorcev so razvrščeni že v prvem nivoju oziroma z navadnim peceptronom in ugotavljanje naslednjih nivojev za njih ni potrebno. Zato smo logično funkcijo ALI izpustili in raziskavo začeli z naborom učnih podatkov, ki ni linearno ločljiv – z že večkrat omenjenim XOR naborom. Navkljub dejstvu, da vsebuje samo štiri učne vzorce in da so ti samo dvodimenzionalni, povzroča ta nabor številnim učnim algoritmom težave. Funkcija XOR ima namreč lokalni minimum, v katerega gradientne metode pogosto zaidejo in v njem običajno tudi obtičijo. Tako se učenje ustavi že, ko je preostala učna napaka še vedno zelo velika oziroma prevelika. Razvrščanje podatkov s tako naučenim MLP ni uspešno, saj nekateri učni vzorci niso pravilno naučeni. Pri tako malem številu učnih vzorcev (štirje), pomeni vsak napačno razvrščen vzorec 25% relativno napako.

Poglejmo si kako se obnese s tem naborom učnih podatkov metoda mejnih parov. Najprej ugotavljamo kateri učni vzorci tvorijo mejne pare. Iz slike 24 je razvidno, da med točkama A in B ni vmesne točke. Enako velja za pare točk A in C, B in D ter C in D. Torej imamo štiri mejne pare: AB, AC, BD in CD. Po iskanju mejnih parov je na vrsti njihovo združevanje. Par AB in AC lahko loči ista premica oziroma en nevron. Tudi druga dva para uspe ločiti ena skupna premica. Vse mejne pare smo torej uspeli razmejiti s samo dvema premicama, iz česar sledi, da prvi sloj MLP vsebuje le dva nevrona, kar se zgolj slučajno ujema s številom vhodov. Iskanje, ločevanje in združevanje mejnih parov je opravljeno enostavno, hitro in brez stranpoti. Na izhodu prvega sloja tako dobimo:

Tabela 10 Notranje vrednosti

Učni vzorec	X	Y	Xn	Yn
A	0	0	0	0
B	0	1	0	1
C	1	0	1	0
D	1	1	0	0

Ker imamo zgolj dvodimenzionalne učne podatke, si lahko razmere v notranjem sloju tudi narišemo.



Slika 25 Prostor značilk za funkcijo XOR

Točke A, B in C ostanejo na svojem položaju, točka D pa se premakne na točko A.

Iz slike 25 je razvidno, da so podatki na izhodu notranjega sloja linearno ločljivi, zato za končen rezultat zadostuje že en sam dodaten nevron v izhodnem sloju. Do enakih ugotovitev pridemo tudi, če zgradimo drugi sloj na enak način kot smo prvega. Preoblikovane učne vzorce, ki jih dobimo na izhodu prvega sloja in so podani v tabeli 11, pripeljemo na notranji sloj MLP.

Tabela 11 Učni podatki notranjega in izhodnega sloja MLP

Učni vzorec	Notranji sloj		Izhodni sloj
	X_n	Y_n	XOR
A_n=A=D	0	0	0
B_n=B	0	1	1
C_n=C	1	0	1

Prvi in četrti učni vzorec (A in D) se oba preslikata v vrednost A_n= (0,0). Tako nam za drugi sloj MLP ostanejo le še trije učni vzorci, s katerimi lahko tvorimo samo dva mejna para: (A_n,B_n) in (A_n,C_n). Ker lahko oba ločimo z isto premico, znova ugotavljamo, da zadostuje en nevron v drugem oziroma izhodnem sloju.

6.2.2 Trikotnik

V primeru učnega nabora trikotnikovih podatkov ugotavljamo katere točke ležijo znotraj trikotnika. Tudi ta nabor učnih vzorcev je dvodimenzionalen, ter je podoben naboru podatkov na sliki 19 in 21. Razlika je le v številu in razporedu učnih podatkov. Tokrat smo uporabili mnogo več (200) učnih vzorcev, ki so razporejeni naključno (niso več enakomerno oddaljeni med seboj). Približno četrtina teh vzorcev je pozitivnih oziroma ležijo znotraj trikotnika. Zaradi naključne lege učnih vzorcev smo učni postopek desetkrat ponovili in na koncu izračunali povprečen rezultat ter standardno odstopanje. Dobljene rezultate smo primerjali z rezultati dobljenimi po metodi backpropagation. Iz tabele 12 je razvidno, da so rezultati po metodi mejnih parov občutno boljši. V resnici je prednost metode mejnih parov še večja, kot je videti iz tabele 12. Z metodo BPM smo namreč našli zgradbo MLP-ja blizu optimalne, ki smo jo nato uporabili tudi za učenje z metodo backpropagation. Tako je bilo slednje učenje uspešnejše, kot bi bilo brez poznavanja optimalne zgradbe MLP. Kot zanimivost omenimo še ugotovitev, da BPM trikotnika ni vedno omejila s tremi premicami, ampak so včasih bile tudi štiri. Do tega pojava pride zaradi naključne lege učnih vzorcev in še ne povsem dodelanega algoritma za združevanje mejnih parov. Vpliv naključne lege vzorcev bi lahko omilili z večjim številom učnih vzorcev, kar pa bi pomenilo tudi več mejnih parov. Zaradi neoptimiranega algoritma za združevanje mejnih parov, se je čas izvajanja s številom vzorcev učnega nabora naglo povečeval.

Tabela 12 Primerjalni rezultati učenja trikotnika

Številka testa	Napaka RMSE	
	BPM	BP
1	0,036	0,200
2	0,075	0,128
3	0,008	0,417
4	0,016	0,072
5	0,024	0,240
6	0,048	0,051
7	0,048	0,339
8	0,115	0,026
9	0,008	0,155
10	0,016	0,042
Povprečje	0,039	0,167
Standardno odstopanje	0,034	0,133

6.3 Učenje z resničnimi učnimi podatki

6.3.1 Prepoznavanje Perunik

Za prvi nabor resničnih podatkov namenjenih testiranju metode BPM smo izbrali perunike (angleško *Iris*). Ta nabor podatkov je eden izmed najbolj priljubljenih in najstarejših naborov resničnih učnih podatkov za strojno učenje in sorodne discipline [8]. Zapisan je bil že pred rojstvom umetne inteligence, leta 1936 za potrebe diskriminantne analize. Nabor učnih podatkov vsebuje podatke o treh vrstah perunik -Iris Setosa, Iris Virginica in Iris Versicolor.

Slika 26 Perunika Iris Setosa

Slika 27 Perunika Iris versicolor

Slika 28 Perunika Iris virginica

Za vsako vrsto perunik je v naboru zajetih 50 cvetlic, torej skupno 150 primerkov. Vsaka cvetlica ima podane štiri parametre: dolžino in širino čašnega lista (latinsko petal) ter dolžino in širino venčnega lista (latinsko sepal). Obe vrsti listov se nahajata v cvetu. Čašni listi so običajno živahne barve in jih vidimo šele ko se cvet razcveti. Venčni listi so običajno zeleni in obdajajo cvetne liste, ko je cvet še v popku. Pri perunikah se barva omenjenih listov med seboj ne razlikuje, ločimo jih lahko le po obliki. Zaradi te podobnosti nepoznavalec cvetnih listov perunike med seboj ne loči.

Meritve dolžine in širine cvetnih listov so bile opravljene za ves nabor cvetlic na enak način. Nekateri raziskovalci s področja analize rojenja (angleško *cluster analysis*) navajajo perunike, zaradi delnega prekrivanja rojev, kot težaven nabor podatkov. Prekrivajo se predvsem perunike vrste Iris Versicolor in Iris Virginica [9].

Slika 29 Cvetni listi

Pri vrtnici so cvetni listi razpoznavni,
zeleni so venčni, rožnati pa čašni.

Obravnavana metoda BPM je v osnovi prirejena za bipolarno razvrščanje oziroma za razvrščanje v dva razreda. Ker imajo Perunike tri razrede, smo morali dva razreda združiti skupaj in razvrščanje ponoviti tri krat. Pri vsaki ponovitvi smo ločevali posamezno vrsto perunik od preostalih dveh. Tak pristop se imenuje »eden proti vsem« (angleško *one against all*). Tako smo ustvarili tri ločene MLP-je. Pri tej raziskavi nas je predvsem zanimalo kako uspešno loči metoda BPM vzorce, ki se nekoliko prekrivajo, zato smo za učenje uporabili celoten nabor podatkov oziroma smo učili in testirali z istimi podatki. Začeli smo s prepoznavanjem perunik vrste Iris Setosa in pri tem se je izkazalo, da celoten nabor učnih podatkov vsebuje samo dva mejna para. Zgolj štirje od 150-ih učnih vzorcev povsem zadostujejo za uspešno učenje MLP. Ker je tudi razpored mejnih

parov ugoden, velja, da za razmejitev mejnih parov zadostuje že ena sama mejna hiper ploskev.

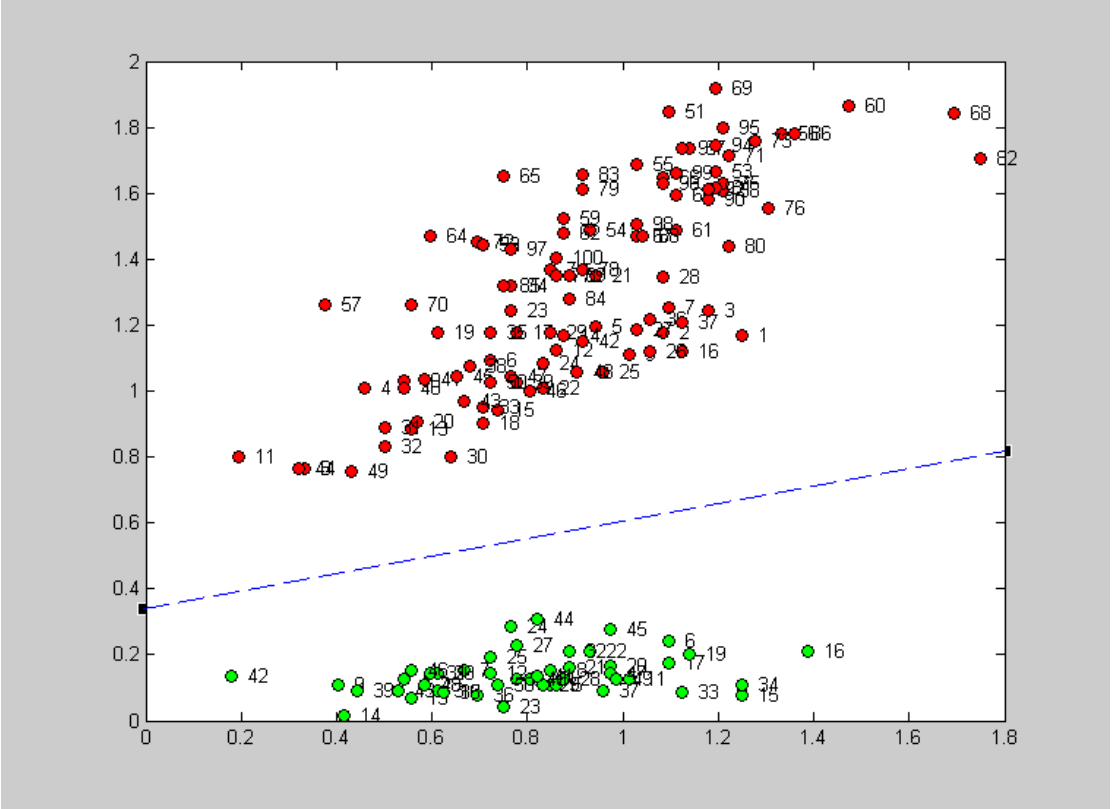
Razvrščanje preostalih dveh vrst perunik (Iris Virginica in Iris Versicolor) je potekalo na enak način. Dobili smo le nekoliko več mejnih parov in mejnih hiper ravnin, kot v pri Setosi. V vsah treh primerih je metoda BPM uspela pravilno ločiti vse učne podatke. Podatki o preosatli učni napaki RMSE so podani v tabeli 13.

Tabela 13 MLP-ji primerni za učenje za nabor podatkov »Perunike«

Podan razred	Mejnih parov	Nevronov v notran. sloju	Napaka RMSE			
			BPM	BP	SVM	DT
Setosa	2	1	0,0000	0,0071	0,0000	0,0000
Versicolor	14	5	0,0000	0,1114	0,0816	0,1323
Virginica	12	4	0,0000	0,1129	0,1155	0,1325

Za primejavo smo učili tudi z drugimi metodami (BP, SVM in DT), ki so se vse izkazale kot slabše, saj imajo večjo preostalo RSME. Zaradi razmeroma majhnega števila učnih vzorcev je pri tem naboru težko ugotavljati ali je MLP prekomerno naučen. Ko smo podatke naključno prepolovili in eno polovico podatkov uporabili za učenje, drugo polovico pa za testiranje, se je izkazalo, da smo pri vseh metodah dobili le eno ali dve napačno razvrščeni peruniki, kar pomeni dobro generalizacijo. To nas ni presenetilo, saj običajno velja, da imajo MLP-ji z malim številom nevronov dobro generalizacijo.

Slika 30 prikazuje ločevanje perunik vrste Iris setosa od preostalih vrst. Štiri dimenzionalne podatke smo za risanje transformirali v dvodimenzionalne. Na osi X



Slika 30 Rojenje perunik

Zeleni krogi so perunike Iris setosa, rdeči pa Iris virginica in Iris versicolor.

smo sešteli širino in dolžino venčnih listov, na osi Y pa širino in dolžino čašnih listov. Kljub tej redukciji dimenzij je na sliki še vseeno vidna ločenost perunik vrste setosa od ostalih (rdeči in zeleni krogi niso pomešani med seboj).

6.3.2 Prepoznavanje števk zapisanih z elektronskim peresom

Prepoznavanje števk zapisanih z elektronskim peresom (angleško *Pen-Based Recognition of Handwritten Digits*) je ime obsežnega, resničnega, uveljavljenega in preverjenega nabora učnih podatkov [10]. Kot pove že ime, so v naboru podatki o

številkah, ki so jih ljudje zapisali z elektronskim peresom, ki je podobno peresu na sliki 31. Omenjeno pisalo ima ločljivost petsto krat petsto oziroma dvesto petdeset tisoč slikovnih točk. Pri pisanju številk je sodelovalo štiriinštirideset različnih oseb. Vsak pisec je prispeval dvesto petdeset številk, kar skupno znese nekaj več kot deset tisoč številk. Nabor podatkov so avtorji predhodno obdelali, tako da so podatki že normalizirani in centrirani. Z normalizacijo so izenačili velikost številk, s centriranjem pa so jih premaknili v sredino okvirja. Posamezna številka je opisana s sedemnajstimi atributi, od katerih predstavljajta prvih šestnajst števil koordinati X in Y za osem točk. To so cela števila med nič in sto. Te točke so zabeležene s peresom v enakomernih časovnih presledkih po sto milisekund. Sedemnajsto število pa predstavlja razred v katerega sodi številka.

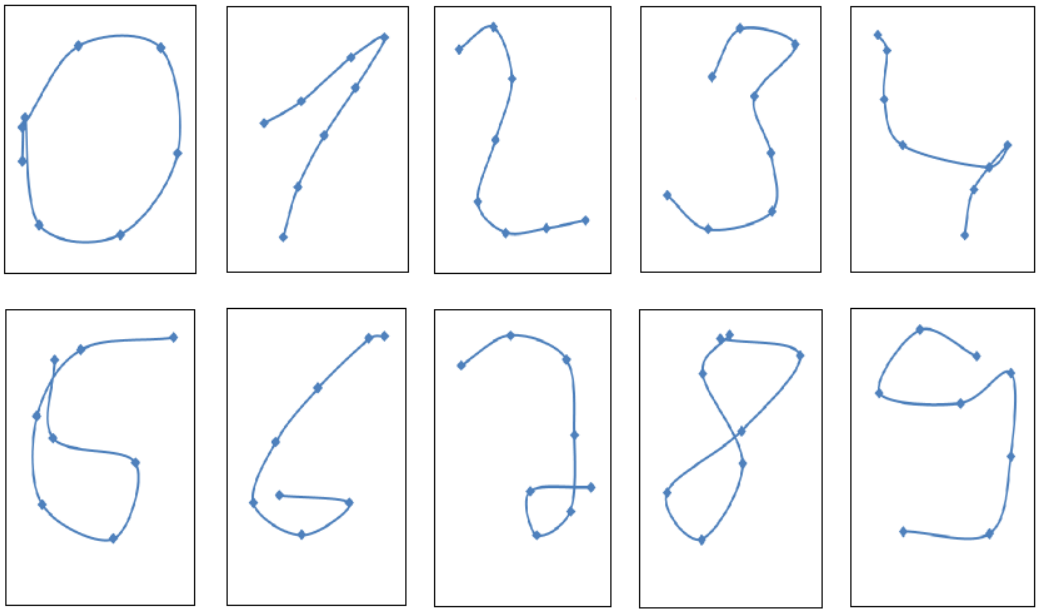
Slika 31 Elektronsko pero

S podobnim peresom so bile zapisane vse številke iz nabora učnih podatkov

Primeri učnih vzorcev za vseh deset različnih številk so prikazani v tabeli 14.

Tabela 14 Primeri ročno zapisanih desetiških števk

Koordinatne točke elektronskega peresa																Števka
1. točka		2. točka		3. točka		4. točka		5. točka		6. točka		7. točka		8. točka		
0	39	2	62	11	5	63	0	100	43	89	99	36	100	0	57	0
0	57	31	68	72	90	100	100	76	75	50	51	28	25	16	0	1
0	89	27	100	42	75	29	45	15	15	37	0	69	2	100	6	2
35	76	57	100	100	92	68	66	81	38	82	9	32	0	0	17	3
0	100	7	92	5	68	19	45	86	34	100	45	74	23	67	0	4
13	89	12	50	72	38	56	0	4	17	0	61	32	94	100	100	5
100	100	88	99	49	74	17	47	0	16	37	0	73	16	20	20	6
0	85	38	100	81	88	87	50	84	12	58	0	53	22	100	24	7
47	100	27	81	57	37	26	0	0	23	56	53	100	90	40	98	8
74	87	31	100	0	69	62	64	100	79	100	38	84	0	18	1	9



Slika 32 Slikovni prikaz števk, ki so podane v tabeli 14

Razvrščanje smo najprej opravili z metodo BPM in nato še s tremi kontrolnimi metodami. Kontrolne metode so bile backpropagation (BP), metoda podpornih vektorjev (SVM) in odločitveno drevo (DT). Avtorji nabora podatkov predlagajo uporabo dveh tretjin števk za učenje, preostalo tretjino pa za validacijo učenja (preverjanje uspešnosti). Predlog smo le delno upoštevali, saj smo število učnih vzorcev zmanjšali tako, da smo jih uporabili le dvesto. Pri številu validacijskih vzorcev smo se držali predloga in uporabili vseh 3498 za to predvidenih vzorcev.

Enako število vzorcev smo uporabili tudi pri obeh kontrolnih metodah, ki smo ju izvajali v programski zbirki Weka [11]. To je uveljavljena odprtokodna zbirka algoritmov za strojno učenje oziroma rudarjenje podatkov. Primerjalni rezultati učenja so podani v tabeli 15.

Razlog za uporabo malega število učnih vzorcev je tem, da izvorna koda učnega programa ni optimizirana, saj je program za učenje napisan v tolmačenem programskem jeziku (angleško *interpreter*), zaradi česar je počasen. Iz iztega razloga tudi nismo opravljali nobenih hitrostnih meritev in primerjav. Kljub malemu naboru učnih podatkov je iz tabele 15 razvidno, da so izbrani učni vzorci reprezentativni in da je učenje uspelo. Odstotek napačno prepoznanih števk je podoben kot pri metodi podpornih vektorjev in je za dober odstotek boljši kot pri odločitvenem drevesu. Metoda backpropagation se je odrezala bolje, razlog pa je verjetno v tem, da pri BPM ni bilo razšumljanja podatkov in da so bili po učenju vsi učni vzorci pravilno prepoznani (prekomerno učenje).

Tabela 15 Primerjava učnih rezultatov prepoznavanja ročno zapisanih števk

Števka	Napačno razvrščeni vzorci [%]			
	BPM	BP	SVM	DT
0	2.5	2,2	2.0	1.7
1	6.8	7,1	7.7	9.6
2	4.2	4,4	4.9	4.5
3	5.2	1,8	1.5	5.2
4	1.7	0,7	1.3	2.6
5	4.8	5,5	8.3	6.0
6	0.8	0,4	0.3	2.8
7	2.2	1,2	3.9	7.1
8	3.4	1,7	4.8	5.5
9	4.7	3,2	2.9	5.2
Povprečje	3,63	2,82	3,76	5,02

Predvidevamo lahko, da bi pri BPM z razšumljanjem podatkov in z izvedbo zgodnje ustavitve učenja, dobili nekoliko boljše rezultate. Za bolj celovito sliko o uspešnosti učenja povejmo še to, da tudi človek ne prepozna pravilno vseh števk v celotnem naboru podatkov.

6.3.3 Ionosfera

Ionosfera je nabor podatkov dobljen med uporabo radarja v Gosjem zalivu, v pokrajini Labrador, v državi Kanada [72]. Radar je sestavljen iz 16 visoko frekvenčnih, fazno zamaknjenih radijskih anten s skupno oddajno močjo 6,4 kW. Oddana energija je usmerjena v proste elektrone, ki se nahahjo v iononosferi (v sloju atmosfere, ki se nahaja med 50 in 600 km nadmorske višine). Pozitivne vzorce dobimo, kadar je v ionosferi prisoten električni prevodnik, saj se radijski valovi od njega odbijajo, negativne pa dobimo, ko ni prevodnika in gre signal skozi ionosfero.

Tabela 16 Rezultati učenja z naborom podatkov za ionosfero

Učni nabor	Napačno razvrščenih vzorcev			
	BPM	BP	SVM	DT
1	65	50	81	106
2	50	47	59	33
3	71	110	72	82
4	62	76	64	50
5	55	48	66	53
6	111	104	126	69
7	126	126	126	126
Povprečje	77,14	80,14	84,86	74,14

S prejetim signalom je izvršena avtokorelacija, katere rezultat je 34 zveznih atributov, ki predstavljajo števila impulzov. V naboru je 351 vzorcev brez manjkajočih vrednosti, ki so setavljeni iz že omenjenih 34 atributov ter razreda, ki je lahko pozitiven ali negativen. Celoten nabor vzorcev smo razdelili na sedem delov: 6 delnih naborov po 50 vzorcev in enegan z 51-imi. Učenje smo izvršili z posamičnimi delnimi nabori, testirali pa smo vedno s celotnim naborom. S tovrstnim učenjem smo izločili dejavnik naključja pri delitvi vzorcev na učne in testne. Zaradi primerjave smo z istimi podatki učili MLP najprej z metodo BPM nato pa še z backpropagation. Rezultati razvrščanja so prikazani v tabeli 16.

Iz tabel 11, 12, 13, 15 in 16 je razvidno, da je metoda BPM delovala v vseh poskusih dobro, oziroma nekoliko bolje od večine primerjalnih metode in to na učnih kot tudi na testnih naborih podatkov.

Potrjevanje hipoteze 1: »Z metodo BPM lahko najdemo zgradbo MLP, ki je blizu optimalne oziroma minimalne zgradbe za podane učne podatke.«

Ugotovili smo, da morajo biti med rojenjem vsi mejni pari ločeni. Z maksimalnim združevanjem parov smo poskrbeli, da smo dobili minimalno število mejnih premic in s tem minimalno število nevronov, ki razmejujeo roje, s čemer je hipoteza 1 potrjena.

Potrjevanje hipoteze 2: »Z metodo BPM lahko zapleten in negotov iterativni učni postopek razdelimo na več manjših, preprostejših in zanesljivejših neiterativnih postopkov.«

Ugotovili smo, da lahko vsak sloj obravnavamo ločeno in da učimo naenkrat le posamezne nevrone. Del postopka učenja med združevanjem parov sicer nekoliko ponavljamo, vendar le toliko krat kolikor je mejnih parov, ki jih poskušamo združiti ob eni mejni premici. To pa je dosti ugodneje kot kot pri metodi bacpropagation, kjer smo vse nevrone učili z vsemi učnimi vzorci dosti večkrat.

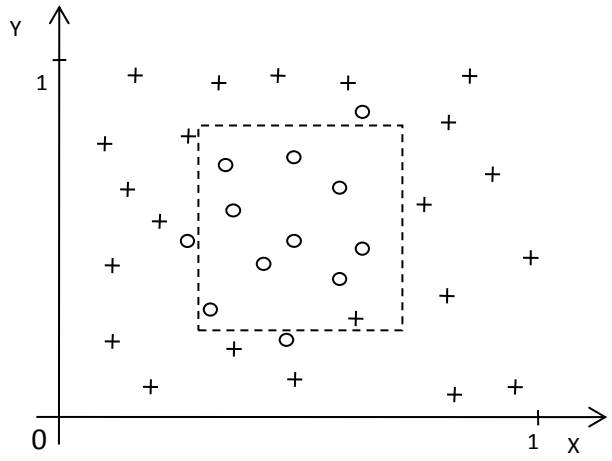
Algoritem BPM se ni zataknil niti v enem poskusu in je vedno je našel rešitev. S tem je hipoteza 2 potrjena.

Potrjevanje hipoteze 3: »Z metodo BPM lahko dosežemo manjšo preostalo učno napako ob hkratnem izogibanju prekomerne naučenosti«.

Rezultati poskusov v tabelah 12, 13, 15 in 16 kažejo, da je napaka po učenju pri metodi BPM manjša kot pri primerjalnih metodah in sicer pri učnih in testnih podatkih. To nam kaže, da ni prišlo do prekomernega učenja. S tem je hipoteza 3 potrjena.

6.4 Šumni podatki

Naslednjo raziskavo smo opravili, da bi ugotovili kako je metoda mejnih parov odporna na šum v učnih podatkih. Natančno vsebnost šuma v učnih podatkih lahko poznamo le, če uporabljamo umetne nabore učnih podatkov. Stopnjo



Slika 33 Učni vzorci, ki vsebujejo šum

Trije krogi so izstopili iz kvadrata, en križec je vstopil.

dodanega šuma v umetnih učnih podatkih smo postopoma povečevali in ugotavljali, kakšen je pri tem odstotek napačno prepoznanih testnih vzorcev.

Zgodnje ustavitve učenja namenoma nismo uporabljali. Znova smo uporabili dvodimenzionalen nabor učnih vzorcev, ki so tokrat normalizirani in predstavljajo sliko kvadrata. Njihova lega je naključna, porazdelitev v vhodnem prostoru pa enakomerna. Vseh vzorcev je 500, od tega jih je ena polovica namenjena za učenje, druga pa za vrednotenje učnih rezultatov. Podobno velja tudi za pripadnost vzorcev razredoma, saj se jih približno pol nahaja v kvadratu, ostali pa so izven. Prvo učenje je potekalo brez dodanega šuma v podatkih, nakar smo šum stopnjevali po naslednjih stopnjah: 1%, 2%, 5% in 10%. Vsako učenje smo ponavljali 10 krat in nato izračunali povprečje ter standardno odstopanje.

Dobljene rezultate raziskave smo primerjali z rezultati metode backpropagation. Ta primerjava je še posebej zanimiva, ker gre v obeh primerih za učenje povsem enakega MLP.

Tabela 17 Napaka RMSE pri različnih stopnjah šuma in z različnimi metodami

Algoritem		Stopnja šuma (%) 0	1	2	5	10
Back-Propagation	Povprečna RMSE	0,0692	0,1257	0,1094	0,1333	0,1519
	Standardno odstopanje	0,0290	0,1136	0,0953	0,1146	0,0782
Metoda mejnih parov	Povprečna RMSE	0,0399	0,0308	0,0497	0,0302	0,1046
	Standardno odstopanje	0,0304	0,0250	0,0491	0,0235	0,0280

Ne pozabimo, da metoda BPM sama ugotovi podatke o skoraj optimalni zgradbi MLP, medtem ko metoda backpropagation te podatke dobi od BPM. Tako je metoda backpropagation pridobila na račun BPM, vendar so kljub temu njeni rezultati slabši, kakor pri metodi BPM.

7 Dinamično učenje z metodo mejnih parov

Uvodoma smo že omenili, da je strojno učenje smotrno uporabiti, ko so razmere dinamične [71], saj lahko inteligentnim sistemom dodajamo nove učne podatke kar med delovanjem in tako dosežemo dve prednosti:

- **Povečamo nabor učnih podatkov**. Razen podatkov, ki so bili na voljo že pred zagonom inteligentnega sistema, vključimo še podatke, ki smo jih pridobili med njegovim delovanjem in tako dobimo večji nabor učnih podatkov. To je še posebno koristno kadar smo na začetku imeli majhen nabor. Zbiranje učnih podatkov je pogosto povezano z različnimi težavami kot so veliki stroški, ogrožanje zdravja, oškodovanje premoženja, varovanje zaupnih podatkov,… V teh primerih je vsak dodaten vzorec dobrodošel.

- **Prilagajamo se novim okoliščinam**. Pogosto pri strojnem učenju ne moremo zajeti vseh dejavnikov, ki vplivajo na rezultat razvrščanja ali regresije. V tem primeru poznamo le del vhodnega vektorja oziroma le del vrednosti učnega vzorca. Med neznanimi dejavniki so tudi takšni, ki se spreminjajo zelo počasi, kar ima za posledico, da novejši učni vzorci prispevajo več h kvaliteti učenja od starejših. Lahko tudi rečemo, da učni vzorci zastarevajo oziroma postajajo neaktualni. Nekateri avtorji to imenujejo sprememba koncepta (angleško *concept drift*). V takih razmerah je smiselno k naboru učnih podatkov

dodajati nove vzorce, stare vzorce pa (postopoma) odstranjevati.

7.1 Pristopi dinamičnega učenja z metodo mejnih parov

Različni pristopi strojnega učenja so različno prilagojeni dodajanju in odvzemanju učnih vzorcev. Pri najbolj »okorelih« pristopih je treba zaradi enega samega dodanega ali odvzetega vzorca ponoviti celotno učenje. Tak pristop prav gotovo ni najbolj ustrezen za dinamične inteligentne sisteme. Žal v to skupino pristopov sodijo tudi vse gradientne učne metode z metodo backpropagation na čelu.

Ločimo dve strategiji učenja dinamičnih inteligentnih sistemov:

- **Inkrementalno učenje**. Pri tej strategiji najprej zberemo učne podatke, ki so do tistega trenutka dostopni in z njimi učimo inteligentni sistem. Po učenju inteligentni sistem nekaj časa uporabljamo in beležimo nove podatke, ki so nastali med uporabo. Ko se nabere dovolj novih učnih podatkov, prekinemo uporabo inteligentnega sistema in ga dodatno učimo. Dodatno učenje se lahko ponavlja po potrebi. Vnaprej določimo kriterije kdaj bomo opravili dodatno učenje. Na primer, ko se nabere določeno število novih učnih vzorcev, ali ko zaznamo večjo spremembo lastnosti učnih podatkov.

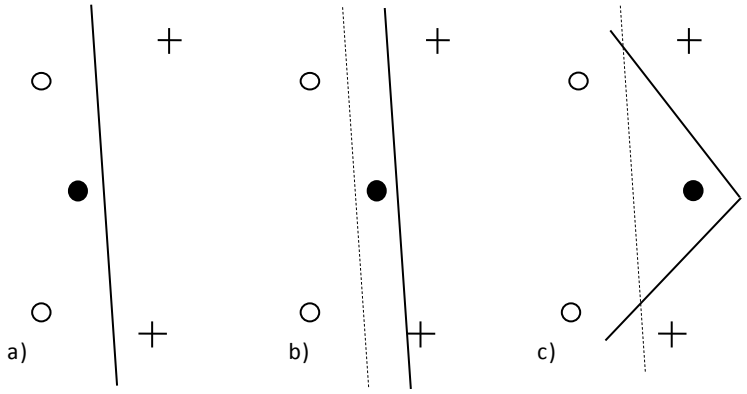
- **Online učenje**. Ta strategija je podobna inkrementalnemu učenju, saj gre tudi tukaj za dodatno učenje, pomembna razlika je le v sprotnosti. Pri tej metodi uporabe inteligentnega sistema ne prekinjamo. Ko se pojavi nov učni vzorec ga takoj uporabimo za dodatno učenje. S tem dosežemo, da je inteligentni sistem maksimalno ažuren. Edina slabost online inteligentnega sistema

je, da se zaradi nenehnega sprotnega dodatnega učenja upočasni njegovo delovanje.

Katera strategija je v danih okoliščinah primernejša, je odvisno od dinamike učnih podatkov in razpoložljivega časa. Kadar je dinamika sistema velika in je dovolj časa za učenje, je smiselno uporabiti online dodatno učenje. Če je dinamika sistema mala ali pa ni dovolj časa za nenehno učenje, je inkrementalno učenje boljša izbira. Poglejmo kako primerno je dodatno učenje za večslojni perceptron. Metoda backpropagation ima z dodatnim učenjem težave. Kadar pride do spremembe koncepta v učnih podatkih, je potrebno izvesti dodatno učenje z vsemi učnimi vzorci. Rezultat dodatnega učenja je pogosto neuspešen. Novi učni vzorci močno povečajo preostalo učno napako, ki jo dodatno učenje ne zmore več zmanjšati. Sinapse v nevronski mreži se vedejo, kot da so »olesenele« in njihove vrednosti se le stežka spreminjajo. Mreža se pri dodatnem učenju praviloma znajde v lokalnem minimumu in preostala učna napaka se sploh ne začne več zmanjševati. Včasih že zadostuje, da obstoječim utežem naključno malenkostno spremenimo vrednosti in tako uidemo iz lokalnega minimuma. Če ta ukrep ne zaleže, damo utežem popolnoma nove naključne vrednosti. To pomeni, da nevronsko mrežo učimo znova od začetka in v tem primeru ne moremo več govoriti o dodatnem učenju.

7.2 Inkrementalno učenje z metodo mejnih parov

Inkrementalno učenje z metodo mejnih parov poteka zelo podobno običajnemu učenju z isto metodo. Pravzaprav najprej MLP učimo z običajno metodo mejnih parov in pri tem uporabimo podatke, ki so bili na voljo pred pričetkom učenja. Učenju sledi uporaba MLP, med katero nastajajo novi učni podatki. Med tem ko rezultate sproti vrednotimo, nas med drugim zanima ali napaka rezultata s časom



Slika 34 Različni razporedi učnih vzorcev med online doučevanjem.

Polni krog prikazuje nov učni vzorec, črtkana črta prikazuje bivšo mejno črto.

a) Ni potrebno premikati mejne črte.
b) Potrebno je premakniti mejno črto.
c) Premik ne zadošča, potrebna je dodatna mejna črta.

narašča. Če napaka narašča, potem govorimo o spremembi koncepta. Če pa spremembe koncepta ni, potem lahko staremu naboru učnih podatkov enostavno dodamo nove vzorce, ki so nastali med uporabo nevronske mreže. V primeru, ko se zgodi večja sprememba koncepta, uporabimo samo nove podatke, stare pa izločimo. Pri blagi spremembi koncepta običajno naredimo kompromis in izločimo le zelo stare podatke. Dodajanju učnih podatkov sledi dodatno učenje, pri katerem poiščemo le dodatne mejne pare in morebitne dodatne mejne črte (ploskve). Velika večina podatkov iz starega učnega nabora pri dodatnem učenju ne rabi več sodelovati. Spremeni se le ustrezen del prvega sloja MLP, ki je odgovoren za področje vhodnega prostora, ki se je dehomogeniziralo z

dodajanjem učnih vzorcev. Pri gradientnih učnih metodah je bilo to drugače. Ker te niso konstruktivne in pri njih ne vemo kateri vzorec vpliva na kateri del nevronske mreže, moramo pri dodatnem učenju uporabiti celoten nabor učnih vzorcev. Povedano drugače: dodatno učenje pri gradientnih metodah v resnici pomeni celovito ponovno učenje od začetka naprej.

Dodatno učenje lahko ponavljamo toliko krat kot si to želimo. Na primer: borzni posrednik lahko svoj MLP za napovedovanje borznih tečajev dodatno uči vsak teden, vsak mesec, oziroma vsakokrat, ko zazna spremembo koncepta v ceni vrednostnih papirjev.

7.3 Online učenje z metodo mejnih parov

Princip online učenja z metodo mejnih parov prikazuje slika 34. MLP razvršča neznane vzorce v razrede, pri čemer sproti ugotavlja ali je bilo zadnje razvrščanje opravljeno pravilno. Kadar je pravilno (slika 34 a), ni potreben noben dodaten ukrep, saj se mejna črta še vedno nahaja na pravem položaju in zgradba MLP-ja ter vrednosti njegovih uteži lahko ostanejo nespremenjeni. V nasprotnem primeru, ko je razvrščanje napačno, je potrebno dodatno učenje. Takšno dodatno online učenje lahko opravimo na več različnih načinov:

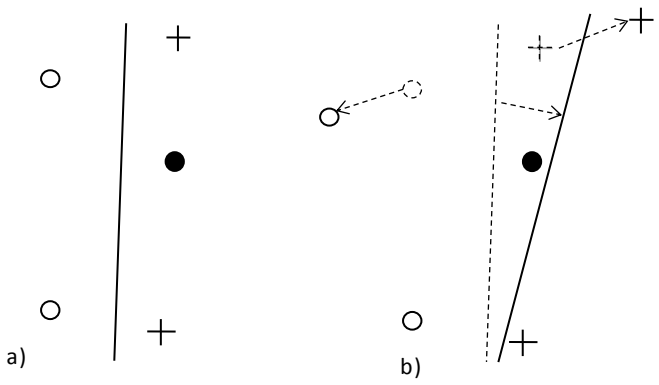
A) Z ali brez rekonstrukcije MLP-ja.
B) Z ali brez pozabljanja (angleško *forgetting*) oziroma odučevanja (angleško *unlearning*).

ad A) Na sliki 34a) je prikazan razpored učnih vzorcev, kjer ni potrebno nobeno dodatno učenje, saj je nov učni vzorec (polni krog) pravilno razvrščen (nahaja se na pravi strani mejne črte). Ko je razvrščanje napačno, najprej ugotovimo katera mejna črta je najbližja novemu učnemu vzorcu. Nevron v prvem sloju, ki ustreza tej črti, dodatno učimo in s tem premikamo pripadajočo črto. Za premikanje črte uporabimo le tiste mejne pare, ki smo jih že uporabili pri

določanju njene lege in nov učni vzorec. S premikanjem mejne črte poskušamo dobiti nov vzorec na pravo stan črte ali pa vsaj čim bližje njej. To je edini možni ukrep, kadar ni dovoljena rekonstrukcija MLP-ja (dodajanje nevronov). Slika 34b) nam prikazuje primer učnih vzorcev, kjer je tak premik možen. Na sliki 34c) so prikazane razmere, kjer premikanje mejne črte ne zadostuje. V tem primeru se moramo odločiti:

- Ali bomo dodali novo mejno črto (nevron)?
- Ali se bomo sprijaznili z napačno razvrstitvijo?

Kadar učni vzorci vsebujejo šum, je običajno smotrno, da se z manjšimi napakami pri razvrščanju sprijaznimo. S tem ukrepom



Slika 35 Učenje s pozabljanjem (odučenjem)

a) Okoliščine pred pozabljanjem, ko mejne črte ni možno prestaviti.

b) Okoliščine po pozabljanju (gornji mejni par je razširjen), ko je mejno črto možno premakniti.

zmanjšamo prekomerno naučenost in izboljšamo generalizacijo učenja. Kadar je napaka večja, oziroma je napačno razvrščeni vzorec daleč preko mejne črte je bolje, da premaknemo mejno črto. Če to ni

izvedljivo, moramo mejno črto nadomestiti z dvema (lomljenje mejne črte). Takšen ukrep doda nov nevron v prvem sloju MLP-ja.

ad B) Vse zapisano v točki A) velja za učenje brez pozabljanja (odučevanja). Oboji učni vzorci - starejši in novejši, so pri tovrstnem učenju enakovredno vplivali na potek učenja. V primeru učnih podatkov prikazanih na sliki 19 smo ugotavljali ali se vzorec nahaja znotraj trikotnika. Ti vzorci vsebujejo le krajevne podatke, brez časovne razsežnosti. V tem primeru ne moremo govoriti o zastaranju podatkov in zato odučevanje (pozabljanje) ni smiselno. V nasprotnem primeru, ko podatki vsebujejo tudi časovno razsežnost, so okoliščine drugačne. Pri takšnih podatkih je pozabljanje lahko dobrodošla, zaželena lastnost, kot primer omenimo razmere pri napovedovanju borznih tečajev vrednostnih papirjev. Napovedovanje tečajev, ki vsebujejo tudi časovno komponento, temelji na izkušnjah iz preteklosti. Težava borznega napovedovanje je v tem, da poznamo le del dejavnikov, ki vplivajo na borzne tečaje. Vse neznane dejavnike lahko združimo v enega, ki ga poimenujemo »duh časa«. Borzni posredniki so v ta namen skovali izraze kot so konjunktura, recesija, stagnacija in podobno. Stari učni vzorci so bili pridobljeni v drugačnih gospodarskih razmerah, v drugačnem duhu časa kot novi. Starejše učne vzorce lahko zato smatramo za (delno) zastarele in jih pri učenju obravnavamo zadržano, ali pa jih celo popolnoma izločimo. Slika 35a) nam prikazuje razmere brez pozabljanja. Oba mejna para ostaneta na svojih položajih. Nov vzorec je moral ostati na napačni strani mejne črte, saj položaja črte ni več možno izboljšati. Če je odučevanje dovoljeno, potem lahko razrahljamo stare mejne pare (povečamo razdaljo med obema udeležencema v mejnem paru). S tem ukrepom pridobimo nekaj dodatnega manevrskega prostora za premikanje mejne črte. Tako smo omogočili, da se je nov vzorec na sliki 35 b) lahko znašel na pravi strani mejne črte.

Poglejmo si algoritem za online učenje z metodo mejnih parov:

Algoritem 4 Algoritem za online učenje z metodo mejnih parov

1. **korak:** Vzemi nov učni vzorec in ga razvrsti.
2. **korak:** Če je razvrščanje uspelo, nadaljuj s 6. korakom.
3. **korak:** Poišči vse vzorce v tem roju.
4. **korak:** Poišči vse mejne pare v tem roju.
5. **korak:** Najdene mejne pare uporabi za učenje nevronov v prvem sloju MLP-ja.
6. **korak:** Nadaljuj s 1. korakom dokler ne obdelaš zadnjega online učnega vzorca.

7.3.1 Online prepoznavanje števk

Za testiranje in vrednotenje rezultatov online algoritma (algoritem 4) smo znova uporabili nabor podatkov o ročno zapisnih številkah, ki smo ga že spoznali v prejšnjem poglavju. Vzeli smo MLP, ki je prehodno že bil naučen z offline metodo in ga dodatno učili z online metodo. Tako smo lahko po zaključenem učenju med seboj primerjali online in offline rezultate. Za dodatno online učenje smo uporabili sto novih učnih vzorcev, ki pri offline učenju in testiranju še niso sodelovali. Ker je metoda BPM konstruktivna, se zgradba MLP med online učenjem lahko spreminja. Po potrebi se dodajo novi nevroni v posameznih slojih oziroma se MLP širi. Običajno se širi samo prvi sloj. Postopek testiranja naučenosti je ostal enak kot pri offline metodi, za kar smo uporabili tudi istih 3498 testnih vzorcev. Dobljeni rezultati dodatnega online učenja so prikazani v tabeli 18. Pričakovali smo, da bo se bo rezultat po dodatnem učenju izboljšal oziroma, da bo po njem manj

napačno razvrščenih števk, kar se je v večini primerov tudi zgodilo. V posameznih primerih, kot je števka 4, pa se je rezultat tudi nekoliko poslabšal. Gledano v celoti (povprečen rezultat) se je po dodatnem učenju število napačno prepoznanih števk zmanjšalo. Poslabšanje rezultatov, ki se je zgodilo pri nekaterih posameznih števkah, pripisujemo šumu v učnih vzorcih, ki smo jih uporabili pri dodatnem učenju.

Tabela 18 Rezultati online učenja

	Napačno razvrščenih (%)	
Števka	prej	potem
0	2,5	2,5
1	6,8	6,0
2	4,2	3,3
3	5,2	4,2
4	1,7	2,1
5	4,8	4,8
6	0,8	1,1
7	2,2	4,4
8	3,4	3,0
9	4,7	3,9
Povprečje	3,63	3,50

8 Rezultati in analiza

Z rezultati online učenja se naše raziskovalno delo zaključuje. Ugotavljamo, da smo bili pri tem uspešni in smo odkrili nekaj zanimivih rezultatov. Naredimo povzetek pomembnejših:

Razšumljanje. Prva pomembna ugotovitev je, da je z metodo mejnih parov možno uspešno razšumljati podatke. To smo naredili tako, da smo poiskali udeležence mejnih parov in jih približali najbližjim neudeležencem mejnih parov istega razreda. Na ta način nismo razšumljali vseh učnih vzorcev ampak le izbrane. Ko smo obravnavali vpliv šuma na mejne pare smo ugotovili, da so skoraj vsi prestopniki udeleženi v mejnih parih. Podobno smo tudi ugotovili, da izven mejnih parov skorajda ni prestopnikov. Tako smo se odločili, da je smiselno razšumljati le udeležence mejnih parov in sicer v smeri najbližjih neudeležencev. Tovrstno razšumljanje učnih vzorcev se je izkazalo za uspešno v zelo visokem odstotku. Ta odstotek je odvisen od stopnje šuma. Pri 10% šumu smo za razšumljanje uporabili le 12,46% učnih vzorcev in tako zajeli 95,58% prestopnikov. Pri 99,87% od razšumljanih vzorcev smo šum zmanjšali. Podobno smo v nadaljevanju ugotovili še, da je generalizacija učenja s šumnimi podatki pri metodi mejnih parov boljša kot pri metodi backpropagation.

Rojenje. Druga pomembna ugotovitev je, da je z metodo mejnih parov možno uspešno rojiti podatke. To smo naredili tako, da smo poiskali vse mejne pare in jih linearno razmejili. Če je po razmejevanju ostalo kakšno heterogeno področje, smo ga znova razmejili na podpodročja. To smo ponavljali, dokler niso bila vsa

področja homogena. Ker smo imeli vedno opravka s končno velikimi nabori učnih vzorcev, smo vedno prišli do popolne homogenosti. To potrjujejo tudi testiranja. Opravili smo več deset primerov rojenj z metodo mejnih parov. Rojenje z mejnimi pari je uspelo v vseh opravljenih poskusih.

Iskanje značilk. Tretja pomembna ugotovitev je, da je možno z metodo mejnih parov poiskati kvalitetne značilke. Vsaka mejna črta ima dva bregova. Dvojiško ju poimenujemo z 0 in 1. Značilko tvorimo tako, da zapišemo na katerem bregu posameznih mejnih črt leži vzorec. Ker se roj nahaja znotraj istega področja imajo pripadniki istega roja tudi isto značilko.

Zanesljivost učenja. Četrta pomembna ugotovitev je, da je učenje z metodo mejnih parov zanesljivo. To pomeni, da se učenje ni nikoli zataknilo oziroma se je vedno končalo uspešno. Značilke, ki nastanejo v prvem sloju, so dvojiška števila. Nadaljnji sloji poiščejo logično operacijo, ki iz značilke izračuna, če vzorec pripada nekemu razredu. V digitalni tehniki se to imenuje sinteza logične funkcije. Iz Boolove algebre je znano dejstvo, da lahko za vsako pravilnostno tabelo poiščemo ustrezno logično funkcijo.

Natančnost učenja. Peta pomembna ugotovitev je, da je učenje z metodo mejnih parov natačnejše ali vsaj podobno natančno kot učenje s sorodnimi metodami. Največ primerjalnih analiz smo opravili z metodo backpropagation. Povprečna preostala učna napaka je bila praviloma pri metodi mejnih parov znatno manjša.

Konstruktivnost. Šesta pomembna ugotovitev je, da je učenje z metodo mejnih parov, za razliko od gradientnih metod, konstruktivno. Med učenjem z njo najdemo (skoraj) minimalno zgradbo MLP za dane učne podatke. Minimalna zgradba MPL hkrati pomeni dobro generalizacijo učnih podatkov. Slabost minimalne zgradbe MLP se pokaže le v primeru njegove delne okvare, ker v takšnem MLP ni redundančnih gradnikov in ni prostora za podvojen zapis naučenega znanja in zato okvara vsakega nevrona napako rezultata močno poveča. Biološke nevronske mreže imajo pogosto redundančen zapis znanja oziroma nimajo minimalne zgradbe.

Prekomerno učenje. Sedma ugotovitev je, da metoda mejnih parov nima težav s prekomernim učenjem. Vzrok temu je dejstvo, da je linearna mejna črta (ploskev) hkrati prilagojena številnim učnim podatkom, zato ni nevarnosti pretiranega prilagajanja posameznemu podatku. Metoda backpropagation se ukvarja z učenjem posameznih podatkov in ko en podatek prilagaja položaj meje zase, istočasno pokvari položaj meje pri več sosednjih podatkih.

Izbrani učni vzorci. Pri učenju sodelujejo le tisti učni vzorci, ki vsebujejo informacijo. V primeru prepoznavanja perunik vrste Setosa je za popolno naučenost zadostovala uporaba zgolj 3% učnih vzorcev. Med izločanjem koristnih učnih vzorcev, si lahko že pred učenjem ustvarimo sliko o zahtevnosti učnega nabora podatkov.

Postopnost in modularnost učenja. Osma ugotovitev je, da učenje poteka postopoma od vhodnega sloja proti izhodnemu, saj smo zahteven učni postopek razdelili na več manjših neodvisnih postopkov, ki jih lahko rešujemo ločeno (modularno). Pomembna razlika nove metode v primerjavi z gradientnimi je tudi to, da jo izvedemo v enem prehodu, kajti metoda mejnih parov ni iterativna.

Primernost metode za inkrementalno in online učenje. Deveta ugotovitev je, da metoda BPM omogoča inkrementalno in online učenje. Kljub temu da je MLP bil že naučen za neke podatke smo ga lako z metodo BPM brez težav dodatno učili. Metoda backpropagation ima s tem velike težave. Ko je MLP enkrat naučen, z metodo backpropagation le s težavo opravimo dodatno učenje. Običajno se v ta namen postavi nekaj uteži nazaj na naključno vrednost. Temu pojavu pravimo olesenelost sinaps, triku pa stresanje MLP.

Ugotovitve o lastnostih metode BPM so strnjene v tabeli 19.

Tabela 19 Pregled lastnosti metode BPM

Možnost rašumljanja podatkov	DA
Možnost rojenja podatkov	DA
Možnost iskanja značilk	DA
Zanesljivost učenja	DA
Natančnost učenja	DA
Konstruktivnost učenja	DA
Občutljivost na prekomerno učenje	NE
Izločanje jalovih vzorcev	DA
Modularnost učenja	DA
Možnost dinamičnega učenja	DA

Primerjava lastnosti metode BPM z uveljavljenimi metodami je podana v tabeli 20. Krepka pisava označuje pristop z najmanjšim odstotkom napačno prepoznanih testnih vzorcev, oziroma najboljši rezultat.

Tabela 20 Sumarna tabela rezultatov

	Napačno prepoznani vzorci (%)			
	BPM	BP	SVM	DT
Perunike*	**0,00**	1,33	1,33	2
Števke	3,63	**2,82**	3,76	5,02
Ionosfera	77,14	80,14	84,86	**74,14**

*- testni nabor podatkov je bil enak učnemu naboru

Iz sumarne tabele je razvidno, da je metoda BPM enkrat dosegla najboljši, dvakrat pa drugi rezultat. Ker je vsakokrat zmagal drug pristop lahko rečemo, da je gledano v celoti bil pristop BPM najuspešnejši med vsemi, vendar zaradi razmeroma malega števila naborov učnih vzorcev obravnavamo to ugotovitev z zadržkom.

Naredimo še povzetek potrjevanja hipotez:

Hipoteza 1: »Z metodo BPM lahko najdemo zgradbo MLP, ki je blizu optimalne oziroma minimalne zgradbe za podane učne podatke.«

Hipoteza 2: Z metodo BPM lahko zapleten in negotov iterativni učni postopek razdelimo na več manjših, preprostejših in zanesljivejših neiterativnih postopkov.

Hipoteza 3: Z metodo BPM lahko dosežemo manjšo preostalo učno napako ob hkratnem izogibanju prekomerne naučenosti.

Hipoteze 1, 2 in 3 se nanašajo na razvrščanje z metodo mejnih parov, zato smo jih potrjevali na koncu poglavja »Razvrščanje podatkov z metodo mejnih parov«, kjer smo z metodo poskusov ugotovili, da je metoda mejnih parov uspešno delovala na vseh raznolikih učnih naborih podatkov. Z združevanjem mejnih parov smo našli skoraj najmanjše možno število premic, ki omejujejo roje in stem zgradbo, ki je blizu minimalne. Ločeno smo učili posamezne nevrone oziroma sloje, česar ni bilo potrebno ponavljati. Ker smo posamezne nevrone hkrati učili s podatki večih mejnih parov, smo se izognili prekomerni naučenosti. Zaradi omenjenih dejstev ugotavljamo, da so hipoteze 1, 2 in 3 potrjene.

Hipoteza 4: Z metodo BPM lahko poiščemo značilke v vhodnem prostoru.

Hipoteza 5: Z metodo BPM lahko poiščemo roje v vhodnem prostoru.

Hipotezi 4 in 5 se nanašata na razmejevanje vhodnega prostora, zato smo ju potrjevali v poglavju »Rojenje podatkov z metodo mejnih parov«. Učne vzorce različnih razredov smo v vhodnem prostoru linearno razmejevali. Če je pri tem nastalo nehomogeno področje, smo ga nadalje razmejevali, dokler nismo dobili samo homogenih področij - rojev. Po vsakem linearnem razmejevanju dobimo dva bregova, ki ju poimenujemo z 0 in 1. Vsako področje tako dobi unikatno dvojiško kodo - značilko, ki nam pove na katerem bregu vseh posameznih meja se nahaja nek roj. Na koncu smo to teorijo potrdili tudi z metodo poskusov. Zaradi omenjenih dejstev sta hipotezi 4 in 5 potrjeni.

Hipoteza 6: Z metodo BPM lahko zmanjšamo šum v učnih vzorcih.

Hipoteza 6 se nanaša na raziskave o šumu, zato smo jo potrjevali v naslovu: »Vpliv šuma in drugih lastnosti podatkov na mejne pare in prestopnike«. Z metodo poskusov smo opazovali vedenje učnih vzorcev v neposredni bližini mejne črte, pri čemer smo ločeno obravnavali udeležence mejnih parov in ostale. Prišli smo do dveh ugotovitev, da so skoraj vsi prestopniki udeleženci mejnih parov in da skoraj vsi neprestopniki niso udeleženci. Iz tega smo naredili sklep, da je smotrno razšumljati le udeležence in sicer v smeri najbližjega neudeleženca. Zaradi omenjenih dejstev je hipoteza 6 potrjena.

Glavna teza: Z uporabo nadzorovanega konektivističnega pristopa BMP k strojnemu učenju, lahko dosežemo uspešnejšo izgradnjo in boljše učne rezultate v realnih primerih odločanja, kot pri običajnih in že obstoječih pristopih nadzorovanega strojnega učenja.

Glavno tezo smo, zaradi lažjega potrjevanja, uvodoma razdelili na šest hipotez. Ker smo potrdili vseh šest hipotez je s tem potrjena tudi glavna teza.

9 Zaključek in načrti za nadaljnje delo

Ideja za temo te disertacije je nastala med predhodnim raziskovalnim delom, ki je temeljilo na uporabi gradientnega učenja večslojnega perceptrona z do sedaj uveljavljeno učno metodo backpropagation, pri čemer so nas spremljale različne težave kot so: počasnost, nezanesljivost in nenatančnost. Po daljšem in temeljitem razmisleku se je porodila ideja za novo, povsem drugačno učno metodo, ki bi se naj izognila omenjenim težavam. Ideja je nastajala postopoma. Najprej smo opazovali lastnosti naučenih MLP, predvsem lastnosti notranjih slojev MLP, ki se pri metodi bacpropagation imenujejo skriti sloji. Pri tem smo ugotovili, da veljajo v notranjih slojih neke zakonitosti. Ko smo analizirali te zakonitosti smo ugotovili, da lahko učimo posamezne sloje MLP oziroma celo posamezne nevrone. Na učni rezultat vplivajo le vzorci, ki se nahajajo v bližini mejne črte. Posamezen par takih vzorcev, eden z vsake strani mejne črte, tvori mejni par, zato smo opisano metodo poimenovali metoda mejnih parov. Z razmejevanjem učnih vzorcev so nastala homogena področja oziroma roji. Za vzorce znotraj istega roja velja, da so na istem bregu vseh polravnin. Če zapišemo bregove vseh rojevih polravnin dobimo enoznačen podatek - značilko celotnega roja oziroma vzorcev v tem roju. Nevroni delujejo v zasičenju, tako da imamo od prvega notranjega sloja naprej samo še logične operacije. Logična operacija IN nam tvori konveksne otoke, logična operacija ALI pa jih združuje v zaključene celote. Nevron v zadnjem sloju pa se odloča, ali vhodni podatek pripada pozitivnemu ali negativnemu razredu.

V glavni tezi smo predpostavili, da je ta metoda boljša od svoje uveljavljene prehodnice in vzornice. Nato smo glavno tezo razdelili na šest hipotez, ki so bile v nadaljevanju vse uspešno potrjene. S tem smo posredno potrdili tudi glavna tezo. Rezultate smo podkrepili s številnimi poskusi. Obravnavana metoda mejnih parov

se je tako izkazala za uspešno in nas je zaradi tega navdihnila za nadaljnje raziskovalno delo, saj se nam za to ponujajo številne možnosti:

- Optimizacija iskanja mejnih parov: V obsežnih naborih podatkov je lahko iskanje mejnih parov zelo zamudno opravilo in je zato smiselno uporabljati optimiran algoritem.
- Optimizacija združevanja mejnih parov: Kadar je v podatkih mnogo mejnih parov je mnogo tudi različnih kombinacij, kako jih lahko združimo med seboj. Med eksperimentalnim delom smo ugotovili, da je združevanje mejnih parov pri zapletenih učnih podatkih v celotnem procesu učenja najbolj zamudno opravilo. Optimizacija bi pomenila hitrejše učenje.
- Regresijska različica metode BPM: obravnavana metoda BPM omogoča samo razvrščanje med dvema razredoma, kjer imamo na izhodu binarno oziroma diskretno vrednost. V naravi so mnoge stvari zvezne in poznajo med odgovorom nič in vse mnogo vmesnih vrednosti. Na primer napovedovanje vremena, regulacija temperature,... Obravnavani algoritem ni primeren za učenje zveznih vrednosti, pedvidoma pa bi ga lahko predelali tako, da bi postal primeren.
- Večrazredna metoda: Obravnavana metoda je imela na izhodu vedno le en nevron, ki je sporočal ali vzorec ustreza nekemu razredu. MLP z večimi izhodnimi nevroni, bi se lahko odločal med večimi razredi.
- Vgradnja metode BPM v program Weka in sorodna uveljavljena orodja za strojno učenje: Vgradnja metode BPM bi zagotovo olajšala in približala raziskovalno delo številnim zaintersiranim znanstvenikom.

K nadaljnemu raziskovanju ste vljudno vabljeni tudi vsi ostali, ki vas to zanima.

10 Literatura

1. Slovar slovenskega knjižnega jezika , http://bos.zrc-sazu.si/sskj.html, 25. 4. 2013

2. What is AI, http://www-formal.stanford.edu/jmc/whatisai/, 12. 3. 2013

3. Human connectome project, http://www.humanconnectomeproject.org, 15. 3. 2013

4. B. Zajc, A. Trost, Zbornik osemnajste mednarodne Elektrotehniške in računalniške konference ERK 2009, Slovenian section IEEE, 2009

5. Hebb, Donald Olding, The Organization of Behaviour, 1949

6. Yann LeCun, Corinna Cortes: yan.lecun.com/exdb/mnist, Handwritten digit database

7. Jihoon Yang, Rajesh Parekh, Vasant Honavar: DistAl: An inter-pattern distance-based constructive learning algorithm, Intelligent Data Analysis, Volume 3, Issue 1, May 1999, Pages 55–73

8. Data set, http://en.wikipedia.org/wiki/Data_set, 12. 1. 2013

9. Iris data set, http://en.wikipedia.org/wiki/Iris_flower_data_set, 12. 1. 2013

10. Pen based handwriten digits data set, http://archive.ics.uci.

edu/ml/support/Pen-Based+Recognition+of+Handwritten+Digits, 12. 1. 2013

11. Weka software. http://en.wikipedia.org/wiki/Weka_(machine_learning) , 18. 4. 2013

12. John McCarty, http://en.wikipedia.org/wiki/John_McCarthy_(computer_scientist) , 12. 8. 2012

13. History of artificial intelligence, http://en.wikipedia.org/wiki/History_of artificial_intelligence, 12. 3. 2013

14. Artificial inteligence, http://en.wikipedia.org/wiki/Artificial_intelligence, 12. 3. 2013

15. Unsupervised learning, http://en.wikipedia.org/wiki/Unsupervised_learning, 12. 3. 2013

16. Reinforcement learning, http://en.wikipedia.org/wiki/Reinforcement_learning, 12. 3. 2013

17. Supervised learning, http://en.wikipedia.org/wiki/Supervised_learning, 12. 3. 2013

18. Decision tree, http://en.wikipedia.org/wiki/Decision_tree, 12. 3. 2013

19. Association rules, http://en.wikipedia.org/wiki/Association_rules, 12. 3. 2013

20. Neural network, http://en.wikipedia.org/wiki/Neural_network, 12. 3. 2013

21. Genetic programming, http://en.wikipedia.org/wiki/Genetic_programming, 12. 3. 2013

22. Evolutionary programing, http://en.wikipedia.org/wiki/Evolutionary_programming, 12. 3. 2013

23. Inductive logic Programming, http://en.wikipedia.org/wiki/Inductive_Logic_Programming, 12. 3. 2013

24. Suport vector machine, http://en.wikipedia.org/wiki/Support_vector_machine, 12. 3. 2013

25. Baysian network, http://en.wikipedia.org/wiki/Baysian_network, 12. 3. 2013

26. Ian H. Witten, Eibe Frank, Mark A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, Third Edition,The Morgan Kaufmann Series in Data Management Systems

27.Feed forward neural network, http://en.wikipedia.org/wiki/Feed-forward_neural_networks, 12. 3. 2013

28. Recurrent neural network, http://en.wikipedia.org/wiki/Recurrent_neural_networks, 12. 3. 2013

29. Perceptron, http://en.wikipedia.org/wiki/Perceptron, 12. 3. 2013

30. Davida E. Rumelharta, Geoffreya E. Hintona in Ronalda J. Williamsa, Learning representations by back-propagating errors, Nature, October 1986

31. P.J.G. Lisboa, T.A. Etchells and D.C. Pountney, Minimal MLPs do not model the XOR logic, School of Computing and Mathematical Sciences

32. T. L. Andersen, T.R. Martinez, DMP3: A Dynamic Multilayer Perceptron Construction Algorithm, Brigham Young University, Utah USA

33. Audio noise, http://en.wikipedia.org/wiki/Noise#Audio_noise, 12. 3. 2013

34. Dolby, http://en.wikipedia.org/wiki/Dolby, 12. 3. 2013

35. Parity bit, http://en.wikipedia.org/wiki/Parity_bit, 12. 3. 2013

36. Iris data set, http://archive.ics.uci.edu/ml/datasets/Iris, 12. 3. 2013

37. Wine data set, http://archive.ics.uci.edu/ml/datasets/Wine, 12. 3. 2013

38. Abalone dataset, http://archive.ics.uci.edu/ml/datasets/Abalone, 12. 3. 2013

39. DistAl: An inter-pattern distance-based constructive learning algorithm, Jihoon Yang, Rajesh Parekh, Vasant Honavar, Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference, 4-9 May 1998, Volume: 3, On Pages: 2208 - 2213 vol.3

40. Geometrical synthesis of MLP neural networks, Rita Delogu, Alessandra Fanni and Augusto Montisci, Neurocomputing,Volume 71, Issues 4–6, January 2008, Pages

919–930,Neural Networks: Algorithms and Applications, 4th International Symposium on Neural Networks

41. Arunava Banerjee, Initializing Neural Networks using Decision Trees, Computational learning theory and natural learning systems: Volume IV, MIT Press Cambridge, 1997, ISBN:0-262-57118-8

42. Cortes, Corinna; and Vapnik, Vladimir N.; "Support-Vector Networks", Machine Learning, 20, 1995, http://www.springerlink.com/content/k238jx04hm87j80g/, 12. 3. 2013

43. Neapolitan, Richard; Jiang, Xia (2012). Contemporary Artificial Intelligence. Chapman & Hall/CRC. ISBN 978-1-4398-4469-4.

44. Talos, http://en.wikipedia.org/wiki/Talos, 12. 3. 2013

45. Hephaestus, https://en.wikipedia.org/wiki/Hephaestus, 12. 3. 2013

46. Frankenstein, http://www.planetebook.com/Frankenstein.asp, 12. 3. 2013

47. R.U.R., http://en.wikipedia.org/wiki/R.U.R. , 12. 3. 2013

48. Mitchell, T.: Machine Learning, McGraw Hill, 1997, ISBN 0-07-042807-7, p.2.

49. Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar: Foundations of Machine Learning, The MIT Press, 2012, ISBN 9780262018258.

50. Ross, Brian H.; Kennedy, Patrick T: Generalizing from the use of earlier examples in problem solving, Journal of Experimental Psychology: Learning, Memory, and Cognition, Vol 16(1), Jan 1990, strani 42-55.

51 Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar Foundations of Machine Learning, The MIT Press, 2012, ISBN 9780262018258.

52. Hinton, Geoffrey; Sejnowski, Terrence J.; Unsupervised Learning: Foundations of Neural Computation, MIT Press, 1999 ISBN 0-262-58168-X

53. Sutton, Richard S. "Learning to predict by the method of temporal differences". Machine Learning (Springer) 3: 9–44. Doi, 1988, 10.1007/BF00115009

54. Vapnik, V. N. The Nature of Statistical Learning Theory (2nd Ed.), Springer Verlag, 2000

55. Oded Maimon and Lior Rokach: DATA MINING AND KNOWLEDGE DISCOVERY HANDBOOK, Springer, 2010

56. Hipp, J.; Güntzer, U.; Nakhaeizadeh, G.: "Algorithms for association rule mining - a general survey and comparison". ACM SIGKDD Explorations Newsletter 2: 58. doi:10.1145/360402.360421, 2000

57. J. J. HOPFIELD Neural networks and physical systems with emergent collective computational abilities. Proc. NatL Acad. Sci. USA Vol. 79, pp. 2554-2558, April 1982 Biophysics

58. Banzhaf, W., Nordin, P., Keller, R.E., and Francone, F.D. (1998), Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications, Morgan Kaufmann

59. Fogel, L.J., Owens, A.J., Walsh, M.J. (1966), Artificial Intelligence through Simulated Evolution, John Wiley

60. Muggleton, S. (1994). "Inductive Logic Programming: Theory and methods". The Journal of Logic Programming. 19-20: 629–679. doi:10.1016/0743-1066(94)90035-3

61. Cortes, Corinna; and Vapnik, Vladimir N.; "Support-Vector Networks", Machine Learning, 20, 1995. http://www.springerlink.com/content/k238jx04hm87j80g/

62. Ben-Gal, Irad (2007). Bayesian Networks (PDF). In Ruggeri, Fabrizio; Kennett, Ron S.; Faltin, Frederick W. "Encyclopedia of Statistics in Quality and Reliability". Encyclopedia of Statistics in Quality and Reliability. John Wiley & Sons. doi:10.1002/9780470061572.eqr089. ISBN 978-0-470-01861-3.

63. John Peter Jesan, Donald M. Lauro: Human Brain and Neural Network behavior a comparison, Ubiquity, Volume 2003 Issue November

64. McCulloch, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 7:115 - 133.

65. Rosenblatt, Frank (1957), The Perceptron--a perceiving and recognizing automaton. Report 85-460-1, Cornell Aeronautical Laboratory

66. Russell, Ingrid. "The Delta Rule". University of Hartford, November 2012

67. P.J.G. Lisboa, T.A. Etchells, D.C. Pountney: Minimal MLPs do not model the XOR logic, Neurocomputing, Volume 48, Issues 1–4, October 2002, Pages 1033–1037

68. Deza, E.; Deza, M.: Dictionary of Distances, Elsevier, ISBN 0-444-52087-2, 2006

69. Roland Priemer : Introductory Signal Processing. World Scientific. p. 1. ISBN 9971509199, 1991

70. Estivill-Castro, V.: "Why so many clustering algorithms". ACM SIGKDD Explorations Newsletter 4: 65. doi:10.1145/568574.568575, 2002

71. Borodin, A.; El-Yaniv, R.: Online Computation and Competitive Analysis. Cambridge University Press. ISBN 0-521-56392-5, 1998

72. Ionosphere data set, http://archive.ics.uci.edu/ml/machine-learning-databases/ionosphere, 12. 3. 2013

73. Alsmadi M. S., Omar B. K. :Back Propagation Agorithm: The Best Algorithm Among the Multi-layer Perceptron Algorithm, IJCSNS, April 2009

74. Sharma K. S., Constractive Neural Networks: a reiew, International Journal of Engineerinf Science and Technology, 2010, pp. 7847-7855

75. Aizenbeg I., Moraga C.: Multilayer Feedforward Neural Network Based on Multi-Valued Neurons and Backpropagation Learning Algorithm, Soft Computing, January 2007, pp. 169-183

76. P. A. Castillo, J. Carpio, J. J. Merelo, A. Prieto, V. Rivas, G, Romero: Evolving Multilayer Perceptrons, Neural Processing Letters, 2000, pp. 115-127

77. J. L. Subirat, L. Franco, I. Molina, J. M. Jerez:Active Learning Using a Constructive Neural Network Algorithm, Constructive Neural Networks, pp. 193-206, 2009, Springer Verlag

78.Y. G. Smetanin: Neural Networks as system for recognizing patterns, Journal of Mathematical Science, 1998

79. E. Ferrari, M. Muselli:Efficient Constructiv Tecniques for Training Switching Neural Networks, Constructive Neural Networks, pp. 24-48, 2009, Springer Verlag

80. J. F. C. Khaw, B. S. Lim, L. E. N. Lim: Optimal Design of Neural Networks Using the Taguchi Method, Neorocomputing, 1995, pp. 225-245

11 Življenjepis

Ime in priimek: Bojan PLOJ

Rojen: 16. 6. 1965, Maribor, Slovenija

Šolanje:

1972 – 1980 Osnovna šola Slava Klavora, Maribor, Slovenija

1980 – 1984 Srednja elektro in računalniška šola, Maribor, Slovenija (elektrotehnik šibki tok)

1984 – 1985 Vojna akademija, Zagreb, Hrvaška (šola rezervnih častnikov)

1985 – 1990 Tehniška fakulteta Maribor, Slovenija (UNI študij – inženir elektrotehnike)

1992 – 1993 Pedagoška fakulteta, Maribor, Slovenija (pedagoško andragoška dokvalifikacija)

2005 – 2008 Fakulteta za elektrotehniko in računalništvo, Maribor, Slovenija (magisterij iz računalništva in informatike)

Zaposlitev:

1991 – 1992 Razvojni inženir, Birostroj Computers, Maribor Slovenija

1992 – > Učitelj strokovno teoretičnih predmetov, Šolski center Ptuj, Ptuj, Slovenija

2002 – 2005 Asistent, Fakulteta za elektrotehniko, računalništvo in informatiko, Maribor, Slovenija

2005 –> Višješolski predavatelj, Šolski center Ptuj, Ptuj, Slovenija

12 Bibliografija

ČLANKI IN DRUGI SESTAVNI DELI

1.01 Izvirni znanstveni članek

1. PLOJ, Bojan, ZORMAN, Milan, KOKOL, Peter. Border Pairs Method - Constructive MLP Learning Classification Algorithm. Lecture notes computer science, str. 297-307, tabele, grafi. [COBISS.SI-ID 285567]

1.08 Objavljeni znanstveni prispevek na konferenci

2. PLOJ, Bojan. Načrtovanje umetne nevronske mreže za nadzor robotske mravlje. V: ZAJC, Baldomir (urednik), Zbornik dvanajste mednarodne Elektrotehniške in računalniške konference ERK 2003, 25. - 26. september 2003, Ljubljana, Slovenija, (Zbornik ... Elektrotehniške in računalniške konference ERK ..., 1581-4572). Ljubljana: IEEE Region 8, Slovenska sekcija IEEE, 2003, str. 463-466. [COBISS.SI-ID 8237846]

3. PLOJ, Bojan. Artificial Neural Network for Motion Control of a Six-legged Robot. V: BUDIN, Leo (ur.), RIBARIĆ, Slobodan (ur.). MIPRO 2007 : 30th Jubilee International Convention, May 21-25, 2007, Opatija, Croatia : proceedings = 30. Jubilarni međunarodni skup : zbornik radova. Vol. III, Computers in Technical Systems, = Računala u tehničkim sustavima, Intelligent Systems, = Inteligentni sustavi. Rijeka: MIPRO, 2007, str. 182-186. [COBISS.SI-ID 62335]

4. PLOJ, Bojan. Bipropagation-nov način učenja večslojnega perceptrona (MLP). V: Zbornik ... Elektrotehniške in računalniške konference ERK .... Ljubljana: Slovenska sekcija IEEE, IEEE Region 8, 1992-, str. 199-202. [COBISS.SI-ID 121727]

MONOGRAFIJE IN DRUGA ZAKLJUČENA DELA

2.09 Magistrsko delo

5. PLOJ, Bojan. Nadzor biološko inspiriranega gibanja šestnožnega mehanizma : magistrska naloga, (Fakulteta za elektrotehniko, računalništvo in informatiko, Magistrska dela). Maribor: [B. Ploj], 2006. 88 f., ilustr. [COBISS.SI-ID 10174998]

2.11 Diplomsko delo

6. PLOJ, Bojan. Izbira nevronske mreže za razpoznavanje govornih signalov, (Tehniška fakulteta, VTO elektrotehnika, računalništvo in informatika). Maribor: [s.n.]. VIII, 118 str., ilustr. [COBISS.SI-ID 739606]

SEKUNDARNO AVTORSTVO

Mentor pri diplomskih delih (višješolski študij)

7. KIRIČ, Denis. Manipulacija s 5-osnim polarnim robotom : diplomska naloga. Ptuj: [D. Kirič], 2009. VIII, 45 str., ilustr. [COBISS.SI-ID 139903]

Prevajalec

8. Mehatronika : [celovit, strokoven in didaktičen priročnik : učbenik v programih Mehatronik operater in Tehnik mehatronike]. 2. izd. Ljubljana: Pasadena, 2009. 624 str., ilustr. ISBN 978-961-6361-87-3. [COBISS.SI-ID 244880384]

NERAZPOREJENO

9. PLOJ, Bojan. Elektrotehnika 2 : [skripta za interno uporabo]. Maribor: samozal., 2009. 42 f., ilustr. ISBN 978-961-245-727-3. [COBISS.SI-ID 62892801]

10. PLOJ, Bojan. Informatika, Program Ekonomist. Ptuj: samozal., 2009. 30 f., ilustr. ISBN 978-961-245-728-0. [COBISS.SI-ID 62893569]

13 Priloga

Izvorna koda programa

Glavna datoteka: BPManaliza.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% branje podatkov
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[podatki,online,test]=nalozi;
normalizacijaPodatkov;
[pozitivni,negativni]=separacija(podatki);
zacetniPozitivni=pozitivni;
zacetniNegativni=negativni;

[steviloPozitivnih,x]=size(pozitivni);
[steviloNegativnih,x]=size(negativni);
disp('Po separaciji je ')
disp(int2str(steviloPozitivnih))
disp('pozitivnih in ')
disp(steviloNegativnih)
disp(' negativnih vzorcev ')

vsiPozitivni=pozitivni;
vsiNegativni=negativni;
[steviloPozitivnih,dolzina]=size(pozitivni);
[steviloNegativnih,dolzina]=size(negativni);
stVsehPozitivnih=steviloPozitivnih;
stVsehNegativnih=steviloNegativnih;
RisanjeVzorcev;
sosedje=IskanjeMejnihParov(pozitivni,negativni)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% risanje sosedov
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


RisanjeSosedov;

[utezi,pragovi]=GrupiranjeMejnihParov(pozitivni,negativni,sosedje);
disp('nevronov je ')
disp(size(pragovi))
'nekaj stisni'
%pause;

TvorbaPerceptrona;
sosedov=size(sosedje)
RisanjeMej;

'meje pred dograjevanjem stisni za dalje'
%pause;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Iskanje dodatnih mejnih parov
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Dograjevanje;
pozitivni=zacetniPozitivni;
negativni=zacetniNegativni;

RisanjeVzorcev;
%RisanjeSosedov;
RisanjeMej;
'meje narisane'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% izgradnja in ucenje
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TvorbaOstalihNivojev;
felerof=sum(abs(round(sim(ostanek,sim(prvinivo,podatki(:,1:end-1)'))-
podatki(:,end)')'))
felerofPred=sum(abs(round(sim(ostanek,sim(prvinivo,test(:,1:end-1)'))-
test(:,end)')'))
%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% online ucnje
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

OnlineUcenje;


felerofPo=sum(abs(round(sim(ostanek,sim(prvinivo,test(:,1:end-1)'))-
test(:,end)')'))
podatki=test;
[pozitivni,negativni]=separacija(podatki);
RisanjeVzorcev;
RisanjeMej;
```

datoteka nalozi.m

```
%%%%%%%%%%%%%%%%%%

function[podatki,online,test]=nalozi;
    dir *.txt

    datoteka=input('Vnesi ime datoteke s ofline podatki? (v enojnih
narekovajih)');
    ukaz=strcat(['load ',datoteka]);
    eval(ukaz);
    %datoteka
    datoteka=datoteka(1:length(datoteka)-4);
    ukaz=strcat(['podatki=',datoteka,';']);
    eval(ukaz);

    datoteka=input('Vnesi ime datoteke s online podatki? (v enojnih
narekovajih)');
    ukaz=strcat(['load ',datoteka]);
    eval(ukaz);
    %datoteka
    datoteka=datoteka(1:length(datoteka)-4);
    ukaz=strcat(['online=',datoteka,';']);
    eval(ukaz);

    datoteka=input('Vnesi ime datoteke s testnimi podatki? (v enojnih
narekovajih)');
    ukaz=strcat(['load ',datoteka]);
    eval(ukaz);
    %datoteka
    datoteka=datoteka(1:length(datoteka)-4);
    ukaz=strcat(['test=',datoteka,';']);
    eval(ukaz);

end
%%%%%%%%%%%%%%%%%%'
```

datoteka normalizacijaPodatkov.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% normalizacija podatkov na interval   med 0 in 1 za vsak stoplpec
posebej
% in doloèitev pozitivnega razreda (pozitiven je 1, ostali 0)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

[steviloVzorcev,velikostVzorca]=size(podatki);
[steviloOnline,velikostOnline]=size(online);
[steviloTest,velikostTest]=size(test);

if (velikostVzorca~=velikostOnline)||(velikostVzorca~=velikostTest)
    'online ali testni podatki niso skladni z ofline podatki'
    return
end %if
vhodov=velikostVzorca-1;
for i=1:vhodov
    minpodatek(i)=min(podatki(:,i));
    podatki(:,i)=podatki(:,i)-minpodatek(i);    %min(podatki(:,i));
    online(:,i)=online(:,i)-minpodatek(i);
    test(:,i)=test(:,i)-minpodatek(i);
    maxpodatek(i)=0;
    if max(podatki(:,i))~=0
        maxpodatek(i)=max(podatki(:,i));
        podatki(:,i)=podatki(:,i)/maxpodatek(i);
        online(:,i)=online(:,i)/maxpodatek(i);
        test(:,i)=test(:,i)/maxpodatek(i);
    end %if
end

if max(podatki(:,velikostVzorca))>1
   'Razredi       so       med:',       [min(podatki(:,velikostVzorca)),
max(podatki(:,velikostVzorca))]
   [prebrano]=input('Kateri razred je pozitiven? ');
else
    prebrano=1;
end %if


for i=1:steviloVzorcev

    if podatki(i,velikostVzorca)==prebrano
        podatki(i,velikostVzorca)=1;
    else
        podatki(i,velikostVzorca)=0;
    end %if

end


for i=1:steviloOnline

    if online(i,velikostVzorca)==prebrano
        online(i,velikostVzorca)=1;
    else
        online(i,velikostVzorca)=0;
    end %if

end
```

```
for i=1:steviloTest

    if test(i,velikostVzorca)==prebrano
        test(i,velikostVzorca)=1;
    else
        test(i,velikostVzorca)=0;
    end %if

end


%konec normalizacije podatkov
```

datoteka TvorbaOstalihNivojev.m

```
metoda='trainlm';              % 'traingdm'  'trainlm'  'traingd'

if stNevronov>=1
    %stevilo_nevronov_tretji_nivo=round(stNevronov/1);
    vmesniRezultat=sim(prvinivo,podatki(:,1:end-1)');
    ostanek=newff(vmesniRezultat,podatki(:,end)',[stNevronov+1]);
    ostanek.name='ostanek';
    ostanek.layers{1}.transferFcn='logsig';
    ostanek.layers{2}.transferFcn='logsig';
    ostanek.trainFcn=metoda;
    ostanek.divideParam.trainRatio=1;
    ostanek.divideParam.valRatio=0;
    ostanek.divideParam.testRatio=0;
    ostanek.inputs{1}.processFcns={};% izklop pred procesiranja
    ostanek.outputs{2}.processFcns={};% izkolp po procesiranja
    ostanek.trainParam.Epochs=1010;
    ostanek=train(ostanek,vmesniRezultat,podatki(:,end)');
    rezultat=sim(ostanek,vmesniRezultat)';
else
    rezultat=sim(prvinivo,podatki(:,1:end-1)')';
end %if
```

datoteka RisanjeVzorcev.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% risanje podatkov preslikanih v 2D
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mejaPodatkov=round(vhodov/2);
[steviloPozitivnih,velikost]=size(pozitivni);
[steviloNegativnih,velikost]=size(negativni);

if (steviloPozitivnih>0 )
if vhodov>2
    xpozitivni=sum(pozitivni(:,1:mejaPodatkov)')'; %pretvorba podatkov v
2D
    if vhodov>mejaPodatkov+1
        ypozitivni=sum(pozitivni(:,mejaPodatkov+1:vhodov)')'; %pretvorba
podatkov v 2D
    else
        ypozitivni=pozitivni(:,mejaPodatkov+1:vhodov);        %pretvorba
podatkov v 2D
    end
else
```

```
    xpozitivni=pozitivni(:,1);
    ypozitivni=pozitivni(:,2);
end %if


hold off

[steviloPozitivnih,dolzina]=size(pozitivni);

    for i=1:steviloPozitivnih

            plot(xpozitivni(i),ypozitivni(i),'o','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor','g',...
                    'MarkerSize',7);
                hold on
        text(xpozitivni(i),ypozitivni(i),strcat(['    ',int2str(i)]));
    end %fori
end %if



if (steviloNegativnih>0 )
if vhodov>2
    xnegativni=sum(negativni(:,1:mejaPodatkov)')'; %pretvorba podatkov v
2D
    if vhodov>mejaPodatkov+1
        ynegativni=sum(negativni(:,mejaPodatkov+1:vhodov)')'; %pretvorba
podatkov v 2D
    else
        ynegativni=negativni(:,mejaPodatkov+1:vhodov);        %pretvorba
podatkov v 2D
    end
else
    xnegativni=negativni(:,1);
    ynegativni=negativni(:,2);
end %if




[steviloNegativnih,dolzina]=size(negativni);

    for i=1:steviloNegativnih
            plot(xnegativni(i),ynegativni(i),'o','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor','r',...
                    'MarkerSize',7);
                hold on
        text(xnegativni(i),ynegativni(i),strcat(['    ',int2str(i)]));
    end %for i
end


hold off

% konec risanja podatkov
```

datoteka IskanjeMejnihParov.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% iskanje sosednjih parov
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[sosedje]=IskanjeMejnihParov(pozitivni,negativni)
sosedje=[];
[steviloPozitivnih,velikost]=size(pozitivni);
[steviloNegativnih,velikost]=size(negativni);
steviloVzorcev=steviloPozitivnih+steviloNegativnih;
vzorci=[pozitivni;negativni];

%[dolzina,vsehVzorcev]=size(Pl);

for i=1:steviloPozitivnih
    for j=1:steviloNegativnih

        soseda=1;
        razdaljaij=sqrt(sum((pozitivni(i,:)-negativni(j,:)).^2));
        for k=1:steviloVzorcev
            if k==i||k==steviloPozitivnih+j
                continue
            end %if
            razdaljaik=sqrt(sum((pozitivni(i,:)-vzorci(k,:)).^2));
            razdaljajk=sqrt(sum((negativni(j,:)-vzorci(k,:)).^2));

            if (razdaljaik<razdaljaij) && (razdaljajk<razdaljaij)
                soseda=0;
                break
            end %if
        end %for k
        if soseda==1
            sosedje=[sosedje;[i,j]];
        end %if soseda

    end %for k

    end % for j

end %for i

% konec iskanja sosednjih parov
```

datoteka RisanjeSosedov.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% risanje sosednjih parov
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
hold on
[stSosedov,velikost]=size(sosedje);

for i=1:stSosedov

        plot(xnegativni(sosedje(i,2)),ynegativni(sosedje(i,2)),'o',...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor','r',...
                'MarkerSize',11)
        hold on
        plot(xpozitivni(sosedje(i,1)),ypozitivni(sosedje(i,1)),'o',...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor','g',...
                'MarkerSize',11)
        %hold off

end

for i=1:stSosedov;


plot([xnegativni(sosedje(i,2)),xpozitivni(sosedje(i,1))],[ynegativni(sos
edje(i,2)),ypozitivni(sosedje(i,1))],'k')


end
%[x,y] = ginput(1);
hold off

% konec risanja sosednjih parov
```

datoteka separacija.m

```
function[pozitivni,negativni]=separacija(podatki)

[steviloPodatkov,velikostPodatkov]=size(podatki);

pozitivni=[];
negativni=[];
for i=1:steviloPodatkov
    if podatki(i,velikostPodatkov)==1
        pozitivni=[pozitivni;podatki(i,1:velikostPodatkov-1)];
    else
        negativni=[negativni;podatki(i,1:velikostPodatkov-1)];
    end %if
end %for i
```

datoteka GrupiranjeMejnihParov.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Grupiranje mejnih parov oziroma
% uèenje nevronov prvega sloja
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function[utezi,pragovi]=GrupiranjeMejnihParov(pozitivni,negativni,sosedj
e)


stNevronov=0;
[stSosedov,dva]=size(sosedje);
urejenSosed=zeros(1,stSosedov);
[vzorcev,dolzina]=size(pozitivni);
center=zeros(stSosedov,dolzina);
utezi=[];
pragovi=[];
%for i=1:stSosedov
 %  center(i,:)=(pozitivni(sosedje(i,1),:)+negativni(sosedje(i,2),:))/2;
%end
%razdalje=dist(center');
nevronovPrag=[];
nevronoveUtezi=[];
P=rand(dolzina,10);
T=rand(1,10);
nevron=newlin(P,T);
nevron.trainParam.Epochs=100+j;
nevron.layers{1}.transferFcn='logsig';
nevron.trainFcn='trainlm';
for i=1:stSosedov
    %i
    if urejenSosed(i)==1
        continue
    end %if
    stNevronov=stNevronov+1
    urejenSosed(i)=1;
    %i;
    clear PM;
    PM(:,1)=pozitivni(sosedje(i,1),:)';      %P(:,sosedje(i,1))
    PM(:,2)=negativni(sosedje(i,2),:)';      %P(:,sosedje(i,2))
    clear TM;
    TM(1)=1;%T(sosedje(i,1));
    TM(2)=0;%T(sosedje(i,2));
    nevron.IW{1}=rand(1,dolzina);% "dolzina" namesto "vhodov"
    nevron.b{1}=rand(1,1);
    [nevron,rezultat]=train(nevron,PM,TM);
    nevronoveUtezi=nevron.IW{1};
    nevronovPrag=nevron.b{1};
    utezi=cat(1,utezi,nevronoveUtezi);
    pragovi=cat(1,pragovi,nevronovPrag);

    %vrsta=razdalje(i,:);
    %vrsta(i)=9999;
    %utezi=[];
    for k=1:stSosedov
        if urejenSosed(k)==1
            continue
        end %if
        %urejenSosed;
        %[i,k]
        %[mali,lega]=min(vrsta);

        %lega;
        %vrsta(lega)=9999;
        PM=[PM,pozitivni(sosedje(k,1),:)',negativni(sosedje(k,2),:)'];
        TM=[TM,1,0]; %TM=[TM,T(sosedje(lega,1)),T(sosedje(lega,2))];
```

```
        %nevron.IW{1}=rand(1,dolzina);% "dolzina" namesto "vhodov"
        %nevron.b{1}=rand(1,1);
        [nevron,rezultat]=train(nevron,PM,TM);
        SrKvNa=rezultat.perf(end);
        if                                                  SrKvNa>2.0e-9
%*********************************************************
            PM(:,end-1:end)=[];
            TM(end-1:end)=[];

            %continue

        else
            %nevronoveUtezi=nevron.IW{1};
            %nevronovPrag=nevron.b{1};
            utezi(stNevronov,:)=nevron.IW{1};
            pragovi(stNevronov)=nevron.b{1};
            urejenSosed(k)=1;

        end %if

    end %for k
    %urejenSosed
    %pause;


end %for i
```

datoteka TvorbaPerceptrona.m

```
[stNevronov,Ena]=size(pragovi);
prvinivo=newlin(rand(vhodov,2),rand(stNevronov,2));
prvinivo.name='prvinivo';
prvinivo.inputs{1}.processFcns={};% izklop pred procesiranja
prvinivo.outputs{1}.processFcns={};% izkolp po procesiranja
prvinivo.layers{1}.transferFcn='logsig';
prvinivo.iW{1,1}=utezi;
prvinivo.b{1}=pragovi;
```

datoteka RisanjeMej.m

```
[nevronov,x]=size(pragovi);
hold on
if dolzina>2
    return
end %if
axis([0 1 0 1])
for i=1:nevronov;

        x1(i)=-pragovi(i,1)/utezi(i,1);
        x2(i)=(-utezi(i,2)-pragovi(i,1))/utezi(i,1);


end

for i=1:nevronov;

        plot([-pragovi(i,1)/utezi(i,1),(-utezi(i,2)-
pragovi(i,1))/utezi(i,1)],[0,1],'k')


end
hold off
```

datoteka Dograjevanje.m

```
dodajanje=1;
while dodajanje
dodajanje=0;

for i=1:stVsehPozitivnih
    pozKoda=round(sim(prvinivo,vsiPozitivni(i,:)'));
    %i
    for j=1:stVsehNegativnih
        negKoda=round(sim(prvinivo,vsiNegativni(j,:)'));
        if negKoda==pozKoda
            dodajanje=1;
            [i,j]
            break
        end %if
    end %for j
    if dodajanje
        break
    end %if
    %delniPodatki

end %for i

if dodajanje
    delniPodatki=[];
    for j=1:steviloVzorcev
        if round(sim(prvinivo,podatki(j,1:end-1)'))==pozKoda
            delniPodatki=[delniPodatki;podatki(j,:)];
        end %if
    end %for j
    [pozitivni,negativni]=separacija(delniPodatki);
    RisanjeVzorcev;
    sosedje=IskanjeMejnihParov(pozitivni,negativni);
    RisanjeSosedov;
    stareUtezi=utezi;
    stariPragovi=pragovi;
    [utezi,pragovi]=GrupiranjeMejnihParov(pozitivni,negativni,sosedje);
    utezi=[utezi;stareUtezi];
    pragovi=[pragovi;stariPragovi];
    TvorbaPerceptrona;
end %if

end %while
```

Datoteka OnlineUcenje.m

```
'zacetek online piflanja'

for i=1:steviloOnline
    podatki=[podatki;online(i,:)];
    steviloVzorcev=steviloVzorcev+1;
    napaka=abs(round(sim(ostanek,sim(prvinivo,online(i,1:end-1)'))-
online(i,end)'));
    if napaka
        %'napaka pri',i
        delniPodatki=[];
        Koda=round(sim(prvinivo,podatki(end,1:end-1)'));
        for j=1:steviloVzorcev
            if round(sim(prvinivo,podatki(j,1:end-1)'))==Koda
                delniPodatki=[delniPodatki;podatki(j,:)];
            end %if
        end %for j
        [pozitivni,negativni]=separacija(delniPodatki);
        RisanjeVzorcev;
        sosedje=IskanjeMejnihParov(pozitivni,negativni);
        RisanjeSosedov;
        stareUtezi=utezi;
        stariPragovi=pragovi;

[utezi,pragovi]=GrupiranjeMejnihParov(pozitivni,negativni,sosedje);
        utezi=[utezi;stareUtezi];
        pragovi=[pragovi;stariPragovi];
        TvorbaPerceptrona;
        TvorbaOstalihNivojev;
    end %if
end %for i

%%%%%%%%%%%%%%%%%%
```

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

FERI

IZJAVA KANDIDATOVEGA MENTORJA O USTREZNOSTI DOKTORSKE DISERTACIJE

Podpisani **Milan Zorman**, mentor doktorskemu kandidatu, izjavljam, da je doktorska disertacija z naslovom **Metoda mejnih parov za učenje umetnih nevronskih mrež**, ki jo je izdelal doktorski kandidat **Bojan Ploj**, v skladu z odobreno temo, Pravilnikom o pripravi in zagovoru doktorske disertacije ter mojimi navodili in predstavlja izviren prispevek k razvoju znanstvene discipline.

Datum in kraj: **Maribor 15. 7. 2013**

Podpis mentorja-ice:

Obrazec RŠZ

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

IZJAVA DOKTORSKEGA KANDIDATA

Podpisani **Bojan Ploj**,

vpisna številka **95033324**.

izjavljam,

da je doktorska disertacija z naslovom

METODA MEJNIH PAROV ZA UČENJE UMETNIH NEVRONSKIH MREŽ

- rezultat lastnega raziskovalnega dela,
- da predložena disertacija v celoti ali v delih ni bila predložena za pridobitev kakršnekoli izobrazbe po študijskem programu druge fakultete ali univerze,
- da so rezultati korektno navedeni in
- da nisem kršil-a avtorskih pravic in intelektualne lastnine drugih.

Podpis doktorskega-e kandidata-ke:

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova ulica 17
2000 Maribor, Slovenija

FERI

IZJAVA O OBJAVI ELEKTRONSKE VERZIJE DOKTORSKE DISERTACIJE IN OSEBNIH PODATKOV, VEZANIH NA ZAKLJUČEK ŠTUDIJA

Ime in priimek doktoranda: **Bojan Ploj**
Vpisna številka: **95033324**
Študijski program: **Računalništvo in informatika**
Naslov doktorskega dela:

Metoda mejnih parov za učenje umetnih nevronskih mrež

Mentor: **Milan Zorman**
Somentor-ica: ----------

Podpisani soglašam z objavo doktorske disertacije v Digitalni knjižnici Univerze v Mariboru.

Tiskana verzija doktorske disertacije je istovetna elektronski verziji, ki sem jo oddal v Digitalno knjižnico Univerze v Mariboru.

Podpisani hkrati izjavljam, da dovoljujem objavo osebnih podatkov, vezanih na zaključek študija (ime, priimek, leto in kraj rojstva, datum diplomiranja, naslov diplomskega dela) na spletnih straneh in v publikacijah Univerze v Mariboru.

Datum in kraj: **15. 7. 2013, Maribor**

Podpis doktoranda: 

Obrazec RŠZ