Laboratorij za tehnologije znanja (KTLab)
Zavod za elektroniku, mikrolektroniku, računalne i inteligentne sustave
Fakultet elektrotehnike i računarstva
Sveučilište u Zagrebu

Interni dokument

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 290

Duboki generativni modeli za semantičko grupiranje tekstovnih dokumenata

Paško Pajdek

Zagreb, lipanj 2011.

Zahvaljujem svojim roditeljima koji su mi bili glavna potpora i omogućili da slijedim svoje želje i dosegnem vlastite potencijale. Također, zahvala ide mentoru Janu Šnajderu, kao i cijelom KTLabu na pomoći u vezi izrade ovog rada, na kvalitetnim diskusijama i suradnji tokom studija – Vaša je riječ vrijedila mnogo.

SADRŽAJ

1. Uvod **1**

2. Modeli za prikaz stohastičkih sustava **3**
2.1. Grafički modeli 3
2.1.1. Prikaz grafičkih modela 4
2.2. Statistički modeli 9

3. Duboki generativni modeli **12**
3.1. Ograničeni Boltzmannovi strojevi 12
3.1.1. Gibbsovo uzorkovanje 15
3.1.2. Učenje RBM metodom kontrastne divergencije 17
3.2. Pohlepan algoritam učenja za duboke mreže 18
3.2.1. Modifikacija RBM-a za modeliranje vektora riječi 21

4. Redukcija dimenzionalnosti pomoću dubokih generativnih modela **23**
4.1. Duboke mreže pri nelinearnoj redukciji dimenzionalnosti 23
4.2. Autoenkoderi 24

5. Primjena dubokih mreža za pretraživanje dokumenata **28**
5.1. Mjere sličnosti 28
5.1.1. Mjera TfIdf 29
5.1.2. Latentna semantička analiza 30
5.2. Semantičko grupiranje 32
5.2.1. Učenje modela za semantičko grupiranje 33

6. Implementacija i evaluacija modela **38**
6.1. Opis programskog rješenja 38
6.2. Mjere vrednovanja 39
6.2.1. Preciznost i odziv 39

6.2.2. Graf preciznost-odziv 40
6.2.3. Srednja vrijednost prosjeka preciznosti 42
6.3. Rezultati implementacije 42

7. Zaključak **45**

Literatura **46**

1. Uvod

Danas, kada živimo u vremenu eksplozije informacija, svakim danom sve više i više podataka postaje dostupno za pretragu i analiziranje. Zbog same količine raspoloživih podataka postaje neefikasno i zamorno ručno izvoditi analizu i pretragu te je očito da se javlja potreba za naprednom i automatiziranom pretragom podataka.

Upravo zbog toga, u domeni računarske znanosti javljaju se nove metode pronalaska relevantnih informacija, koje taj zadatak obavljaju brzo no njihova preciznost nije stopostotna, pa se svakog dana javljaju nove, preciznije i brže metode za dohvaćanje određene informacije (ili podatka) iz vrlo velikog skupa podataka. Tim problemima obrade podataka bave se neke podkategorije grane računarske znanosti zvane strojno učenje.

Mnogi postojeći algoritmi strojnog učenja za rješavanje problema koriste plitke arhitekture. To su mreže koje se sastoje od malo, najčešće jednog skrivenog sloja. Rezultati pokazuju da je interni način na koji takve mreže prikazuju ulazne podatke vrlo jednostavan, i nije dovoljan za za prepoznavanje kompleksne strukture i pravilnosti ulaznih podataka. Drugi nedostatak plitkih mreža je taj što im je za učenje potrebno mnogo označenih podataka, tj. podataka s označenom pripadnošću određenoj kategoriji.

Da bismo zadatak što uspješnije riješili, moramo izgraditi sustav koji može iz podataka izvući odgovarajuće reprezentacije na višoj razini i duboke strukture među podacima koje na prvi pogled nisu očite. Teoretska istraživanja kao rješenje problema predlažu duboku arhitekturu modela — model čini mreža sastavljena od više nelinearnih slojeva.

U ovom radu bit će opisani statistički modeli podataka nazvani duboki generativni modeli (engl. *Deep generative models*), njihova teorija i primjena. Konkretno će biti opisan model koji rješava zadatak dohvaćanja sličnih dokumenata te njegovi gradivni elementi. Na temelju tog dubokog generativnog modela, razradit će se postupak za semantičko grupiranje dokumenata na hrvatskom jeziku, izradit će se programska implementacija postupka te će se prikazati rezultati tog postupka.

Rad se sastoji od sedam poglavlja. Nakon uvodnog, u poglavlju nazvanom "Modeli za prikaz stohastičkih sustava" opisuju se temeljni pojmovi iz teorije vjerojatnosti potrebni za razumijevanje daljnjih poglavlja. U poglavlju "Duboki generativni modeli" prikazuje se teoretska pozadina dubokih generativnih modela, konkretno ograničenih Boltzmannovih strojeva (engl. *restricted Boltzmann machines, RBM*), kao i pohlepan algoritam za učenje istih. Nakon toga, u poglavlju "Redukcija dimenzionalnosti pomoću dubokih generativnih modela" pokazat će se kako se ograničeni Boltzmannovi strojevi iskorištavaju pri izgradnji dubokih generativnih modela za redukciju dimenzionalnosti, svojstva i pogodnosti autoenkodera te učinkovitost modela s obzirom na razne arhitekture i algoritme učenja. U poglavlju "Primjena dubokih mreža za pretraživanje dokumenata" će biti opisane mjere sličnosti između tekstovnih dokumenata, kao i algoritam za njihovo semantičko grupiranje. U poglavlju "Implementacija i evaluacija modela" će biti opisana implementacija modela za semantičko grupiranje te rezultati modela nad stvarnim podacima, dok će u zaključku biti dan zaključak rada, kao i preporuke za daljnji razvoj modela.

2. Modeli za prikaz stohastičkih sustava

Da bismo lakše shvatili predmet ovog rada, moramo se najprije upoznati s nekim pojmovima iz teorije vjerojatnosti. U radu biti će opisan stohastički sustav koji će biti dočaran grafom i nizom formula temeljenih na uvjetnoj vjerojatnosti. U ovom poglavlju će biti predstavljeni modeli za olakšavanje prikaza i lakše modeliranje stohastičkih sustava – grafički i statistički modeli.

2.1. Grafički modeli

Grafički modeli (engl. *graphical models*) predstavljaju kombinaciju znanja iz teorije vjerojatnosti i teorije grafova koji nam omogućuju lakše modeliranje stvarnih inženjerskih problema (Murphy, 1998). Konkretno, grafički modeli predstavljaju alat za rješavanje nesigurnosti i složenosti u modelima, što ih čini idealnima za primjenu u oblikovanju i analizi složenih podataka, odnosno za modeliranje ulaznih podataka algoritama za strojno učenje. U osnovi ideje grafičkih modela je modularnost; složen sustav je izgrađen od kombiniranja jednostavnijih dijelova.

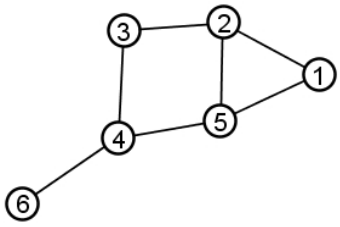
Teorija vjerojatnosti služi se grafičkim modelima tamo gdje je potrebno kombinirati jednostavnije dijelove sustava, osiguravajući konzistentnost cjelokupnog sustava. Također, teorija vjerojatnosti služi pri konstruiranju odgovarajućeg sučelja između modela i podataka.

Teorija grafova, s druge strane, služi se grafičkim modelima da bi se model mogao konstruirati na način koji je intuitivan i ne pretjerano kompliciran ljudima. Na taj način ljudi su sposobni modelirati visoko interaktivne skupove varijabli sustava, kao i strukture podataka, a sve u svrhu dizajniranja snažnih i efikasnih algoritama opće primjene.

2.1.1. Prikaz grafičkih modela

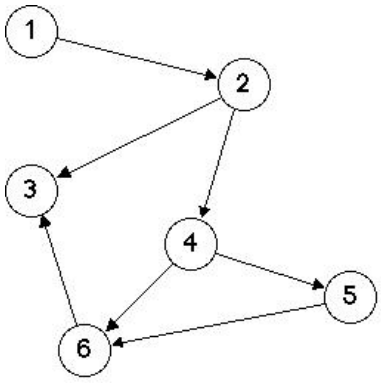
Vjerojatnosni grafički modeli su grafovi u kojima čvorovi predstavljaju slučajne varijable, a bridovi grafa predstavljaju pretpostavke uvjetne nezavisnosti. Dakle, možemo reći kako grafički modeli osiguravaju jednostavan prikaz zajedničke razdiobe vjerojatnosti. Grafički modeli mogu se podijeliti na neusmjerene i usmjerene grafičke modele.

Neusmjereni grafički modeli, u praksi poznati i kao Markovljeva slučajna polja (engl. *Markov Random Fields, MRF*) ili Markovljeve mreže, imaju jednostavnu definiciju nezavisnosti – dva čvora, ili skupa čvorova, A i B su uvjetno nezavisna obzirom na treći čvor C samo onda ako su svi putevi između A i B razdvojeni čvorom C. Takvi grafički modeli popularni su među znanstvenicima na području fizike i računalnog vida, a prikaz jednog jednostavnog neusmjerenog grafičkog modela može see vidjeti na slici 2.1



Slika 2.1: Primjer neusmjerenog grafa

Usmjereni grafički modeli, također zvani i Bayesove mreže ili mreže vjerovanja (engl. *Belief Networks, BN*) imaju nešto kompliciraniji uvjet nezavisnosti od neusmjerenih modela, koji u obzir uzima i smjer bridova. Ovi modeli najčešće se koriste u području umjetne inteligencije i području statističkih analiza podataka, a primjer jednostavnog usmjerenog grafičkog modela može se vidjeti na slici 2.2.

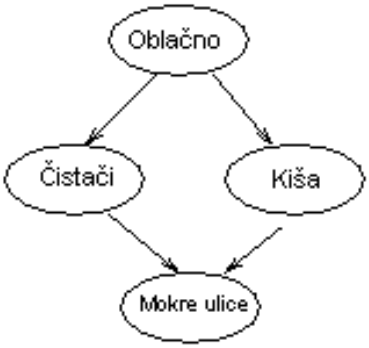


Slika 2.2: Primjer usmjerenog grafa

Iako usmjereni grafički modeli imaju kompliciraniji uvjet nezavisnosti od neusmjerenih modela, ipak takvi modeli imaju i neke prednosti. Najvažnija prednost je u tome što usmjerenost brida označava kauzalnost, koja je ljudima intuitivna. Dakle, ako imamo dva čvora A i B te brid koji ide iz A u B, možemo reći da "A uzrokuje B". Ta intuitivnost uvelike pomaže ljudima konstruirati strukturu grafa. Također, usmjereni modeli mogu predstaviti deterministične relacije između varijabli te su pogodniji za učenje na temelju podataka. Uz neke manje razlike za slučaj neusmjerenog grafa, u nastavku će na primjeru usmjerenog grafičkog modela biti prikazani osnovni pojmovi za razumijevanje modela.

Da bi bolje objasnili određene pojmove, iskoristiti ćemo primjer usmjerenog grafičkog modela. Uz samu grafičku strukturu, potrebno je specificirati parametre modela, točnije vjerojatnosti pojavljivanja određenih događaja. U svakom je čvoru potrebno odrediti distribuciju uvjetne vjerojatnosti obzirom na čvorove koji su povezani bridovima na promatrani čvor. Ako su varijable u čvorovima diskretne, distribucija može biti predstavljena u obliku tablice, koja prikazuje vjerojatnosti da čvor-dijete poprima svaku od mogućih vrijednosti za svaku kombinaciju vrijednosti njegovih roditelja.

Promotrimo sljedeći primjer na slici u kojemu su svi čvorovi binarni; tj. imaju samo dvije vrijednosti koje mogu poprimiti, a označit ćemo ih sa 1 i 0 (istina i laž, *engl. true, false*). Čvorovi, radi sažetosti prikaza, imaju svoje kratice: čvor Oblačno – O, čvor Kiša – K, čvor Čistači ulica – C te čvor Mokre ulice – M.



Slika 2.3: Bayesova mreža – primjer

Sa slike 2.3 možemo iščitati da događaj "ulice su mokre" ima dva moguća uzroka: ili su čistači oprali ceste ili je padala kiša. Vjerojatnost ovih događaja možemo pročitati iz tablice 2.1; vidimo da je $P(M = 1|C = 1, K = 0) = 0.9$ te iz toga slijedi da je $P(M = 0|C = 1, K = 0) = 1 - 0.9 = 0.1$, pošto su ova dva događaja isključiva. Kako čvor O nema roditelja, iz tablice čitamo vjerojatnost koja je unaprijed određena.

Tablica 2.1: Uvjetne vjerojatnosti za mrežu prikazanu slikom 2.3. S lijeve strane tablice vidimo čvorove o kojima je pojedini čvor zavisan, a s desne konkretne vjerojatnosti s obzirom na vrijednost čvorova-roditelja.

P(O=1)	P(O=0)
0.5	0.5

O	P(C=1)	P(C=0)
0	0.5	0.5
1	0.9	0.1

O	P(K=1)	P(K=0)
0	0.2	0.8
1	0.8	0.2

C	K	P(M=1)	P(M=0)
0	0	0	1
0	1	0.9	0.1
1	0	0.9	0.1
1	1	0.99	0.01

Najjednostavnija uvjetno nezavisna veza prikazana usmjerenim grafičkim modelom je sljedeća: čvor je nezavisan u odnosu na sve svoje pretke, osim u odnosu na svoje roditelje; neposredna zavisnost čvora o svojim roditeljima u sebi posredno sadržava i utjecaj predaka na čvor.

Prema lančanom pravilu vjerojatnosti, združena vjerojatnost svih čvorova u grafu je:

$$P(O, C, K, M) = P(O) * P(C|O) * P(K|O, C) * P(M|O, C, K). \tag{2.1}$$

Da bi mogli iskoristiti usmjerenost grafičkog modela, a samim tim i pojednostavniti formule za računanje vjerojatnosti, moramo se upoznati sa pojmom uvjetne nezavisnosti. Dva događaja A i B su uvjetno nezavisna ako postoji neki događaj C takav da su u uvjetnoj razdiobi vjerojatnosti s obzirom na C događaji A i B nezavisni. Ovu tvrdnju možemo prikazati i sljedećom formulom:

$$P(A \cap B \mid C) = P(A \mid C)P(B \mid C). \tag{2.2}$$

Koristeći relacije uvjetne nezavisnosti, gornju formulu možemo napisati na sljedeći način:

$$P(O, C, K, M) = P(O) * P(C|O) * P(K|O) * P(M|C, K), \tag{2.3}$$

gdje nam je dozvoljeno pojednostavniti zadnja dva člana u produktu zato što je čvor K uvjetno nezavisan u odnosu na C, kao što je i čvor M uvjetno nezavisan u odnosu na O.

Vidimo da nam relacije uvjetne vjerojatnosti omogućavaju prikazati veze na efikasniji, više sažet način. U konkretnom primjeru se ne vidi prednost takvog prikaza (zbog malog broja čvorova), no kada bismo imali n binarnih čvorova, potpuna povezanost bi zahtjevala prostornu složenost $O(2^n)$. To je zato što svaki čvor ulazi u formulu s

jednom od svoje dvije vrijednosti pa veličinu formule možemo promatrati kao i broj različitih prikaza binarnog broja s n bitova. U faktoriziranoj formi prostorna složenost se smanjuje na $O(n2^k)$, gdje je k najveći broj bridova koji ulaze u jedan čvor u modelu. Vidimo da faktorizacija ograničava broj mogućnosti koje formula može uzeti u obzir. Sukladno tome, učenje takvih modela je lakše upravo zbog manjeg broja parametara modela.

Zaključivanje u grafičkim modelima

Jedan od najčešćih zadataka koji se javljaju koristeći grafičke modele jest zaključivanje o događajima na osnovu vjerojatnosti. Na gore pokazanom primjeru možemo promatrati činjenicu da su ulice mokre. Taj događaj ima dva moguća uzroka – ili je padala kiša ili su čistači oprali cestu. Ono što iz grafa možemo iščitati i izračunati jest podatak koji je od uzroka vjerojatniji. Kod ovog primjera usmjerenog grafičkog modela koristimo Bayesovo pravilo da bismo izračunali *a posteriori* vjerojatnost događaja koji objašnjava činjenicu da su ulice mokre. Za neusmjerene modele zaključivanje se vrši na sličnom principu, ali ideja u pozadini ostaje ista.

$$\begin{aligned} P\left(C=1|M=1\right) &= \frac{P\left(C=1,M=1\right)}{P\left(M=1\right)} = \\ &= \frac{\sum_{o,k} P\left(O=o,C=1,K=k,M=1\right)}{P\left(M=1\right)} = \\ &= \frac{\sum_{o,k} P\left(O\right) P\left(C=1|O\right) P\left(K|O\right) P\left(M=1|C=1,K\right)}{P\left(M=1\right)} = \\ &= \frac{0.2781}{0.6471} = 0.43, \end{aligned} \tag{2.4}$$

$$\begin{aligned} P\left(K=1|M=1\right) &= \frac{P\left(K=1,M=1\right)}{P\left(M=1\right)} = \\ &= \frac{\sum_{o,c} P\left(O=o,C=c,K=1,M=1\right)}{P\left(M=1\right)} = \\ &= \frac{\sum_{o,c} P\left(O\right) P\left(C|O\right) P\left(K=1|O\right) P\left(M=1|C,K=1\right)}{P\left(M=1\right)} = \\ &= \frac{0.4581}{0.6471} = 0.708, \end{aligned} \tag{2.5}$$

gdje je

$$P\left(M=1\right)=\sum_{o,c,k}P\left(O=o,C=c,K=k,M=1\right)=$$
$$=\sum_{o,c,k}P\left(O\right)P\left(C|O\right)P\left(K|O\right)P\left(M=1|C,K\right)=0.6471. \quad (2.6)$$

Objašnjavanje uzroka u grafičkim modelima

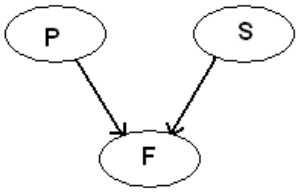
U gornjem primjeru možemo vidjeti da postoje dva moguća uzroka nekog događaja te da se oni, možemo reći, "natječu" da bi "objasnili" uzrok promatranog događaja. Dakle, čvorovi C i K postaju uvjetno zavisni ako imaju zajednički čvor-dijete i ako je događaj koji taj čvor predstavlja poznat, što je u suprotnosti s realnim stanjem stvari koje možemo iščitati iz grafa, a to je da su ti čvorovi nezavisni.

Na primjer, ako pretpostavimo da su ulice mokre, ali isto tako znamo i da je padala kiša, tada posteriorna vjerojatnost da su čistači prali ulice opada:

$$P(C=1|M=1,K=1)=0.1945. \quad (2.7)$$

Ova pojava se naziva objašnjavanje uzroka (engl. *explaining away*). U statistici je ovaj efekt poznatiji kao Berksonov paradoks ili pristranost izbora.

Još jedan primjer za ovaj efekt, preuzet iz (Murphy, 1998), bi bio model fakulteta koji prima ili pametne osobe, ili vrhunske sportaše, ili oboje. Neka F označava događaj da je osoba primljena na fakultet, što je istina ako je ta osoba pametna (P) ili sportaš (S), kao što se može vidjeti na slici 2.4. Isto tako možemo pretpostaviti da su te dvije pojave u populaciji nezavisne. Sada možemo modelirati naše pretpostavke uvjetne nezavisnosti koristeći sljedeću strukturu grafa:



Slika 2.4: Objašnjavanje uzroka

Ako sada pogledamo populaciju studenata na fakultetu pokazat će se da pametni studenti imaju manju vjerojatnost da su ujedno i sportaši, i obratno, zato što je svaka od osobina dovoljna za objašnjavanje uzroka događaja F (tj. $P(P=1|F=1,S=1)\leq P(P=1|F=1)$).

Zaključivanje odozgo i zaključivanje odozdo u grafičkim modelima

U primjeru s mokrim ulicama imali smo događaj – dokaz djelovanja nekog od uzroka (mokre ulice) te smo na temelju toga zaključili najvjerojatniji uzrok događaja. Taj postupak se naziva zaključivanje odozdo (engl. *bottom-up*), ili dijagnostika, jer proces ide od efekata prema uzrocima. Ovaj postupak se najčešće primjenjuje u ekspretnim sustavima.

Usmjereni grafički modeli mogu biti upotrebljeni i za kauzalno zaključivanje, ili zaključivanje odozgo (engl. *top-down*). Na primjer, možemo izračunati vjerojatnost da će ulice biti mokre ako znamo da je oblačan dan. Stoga se Bayesove mreže često nazivaju generativnim modelima; određuju kako uzroci generiraju posljedice.

2.2. Statistički modeli

Statistički model je formalizacija veza između varijabli sustava prikazana u obliku matematičkih jednadžbi. On opisuje načine na koje je jedna ili više slučajnih varijabli povezana s nekom drugom slučajnom varijablom, ili skupom istih. Ono što u samoj osnovi čini neki model statističkim jest to što su varijable modela povezane na stohastički način, a ne deterministički.

Kao što je prikazano u (Field, 2009), matematički gledano, statistički model možemo prikazati kao par (Y, P), gdje je Y skup svih mogućih vidljivih (promatranih) podataka, a P skup mogućih distribucija vjerojatnosti po kojima se ravnaju podaci iz skupa Y. Pretpostavlja se da postoji točno određen element skupa P, točnije, distribucija vjerojatnosti koja generira podatke koji dolaze na ulaz modela.

Formalna definicija statističkog modela

Statistički model $\mathcal{P}$ je skup funkcija distribucija vjerojatnosti ili funkcija gustoće vjerojatnosti. S obzirom na naše prethodne pretpostavke o vjerojatnosnoj distribuciji ulaznih podataka, modele možemo podijeliti na parametarske i neparametarske.

Parametarski model je skup distribucija, od kojih je svaka određena jedinstvenim konačno-dimenzionalnim parametrom:

$$\mathcal{P} = \{\mathbb{P}_\theta : \theta \in \Theta\}, \tag{2.8}$$

gdje je θ taj jedinstveni parametar, a $\Theta \subseteq \mathbb{R}^d$ je skup parametara koji su dohvatljivi, a podskup su d-dimenzionalnog euklidskog prostora.

Statistički model služi kao alat za opis skupa distribucija za koje se pretpostavlja da modeliraju skup ulaznih podataka. Pokažimo to na primjeru. Ako pretpostavimo da se podaci ravnaju po Gaussovoj razdiobi, onda možemo koristiti Gaussov model za opis podataka:

$$\mathcal{P} = \left\{ \mathbb{P}(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{1}{2\sigma^2}(x-\mu)^2 \right\} : \mu \in \mathbb{R}, \sigma > 0 \right\}. \tag{2.9}$$

Neparametarski model, s druge strane, je skup razdioba vjerojatnosti s beskonačno dimenzionalnim parametrima. To znači da nemamo apriorne pretpostavke o razdiobi po kojoj se generiraju ulazni podaci, već takav model možemo označiti kao $\mathcal{P} = \{\text{sve razdiobe}\}$. Uz ove dvije kategorije postoji i poluparametarski modeli; oni također imaju beskonačnu dimenzionalnost parametara, ali za razliku od neparametarskog modela poluparametarski nije gust u prostoru razdioba.

Osim podjele na parametarske i neparametarske metode, statističke modele možemo podijeliti na dvije sljedeće kategorije: generativne modele i diskriminativne modele.

Diskriminativni modeli su jednostavniji, lakši za shvaćanje, ali samim time i ograničeniji u praktičnoj uporabi. Ovi modeli dozvoljavaju komponentama strojnog učenja diskriminaciju ili klasifikaciju između ulaznih podataka, a sadrže podatke koji povezuju uvjetnu vjerojatnost ulaza s nepoznatim parametrima odluke pri klasifikaciji ili diskriminaciji.

Generativni modeli sadrže više informacija o dinamičnosti sustava, što model čini složenijim, ali zato dozvoljava korisniku da bolje iskoristi sve mogućnosti modela. Najveća prednost generativnog modela je u tome što može generirati podatke na temelju pretpostavljenih razdioba, na temelju čega možemo vidjeti kako model uči nad podacima te tu informaciju iskoristiti za što efikasnije učenje nad novim podacima.

Diskriminativni modeli

Diskriminativni modeli su vrsta modela koja se koristi u strojnom učenju za modeliranje ovisnosti skrivene varijable y obzirom na vidljivu varijablu x. To se konkretno radi modeliranjem razdiobe uvjetne zavisnosti $P(y|x)$, što koristimo za predviđanje stanja varijable y na temelju stanja varijable x.

Diskriminativni modeli se razlikuju od generativnih po tome što preko njih nije moguće generirati uzorke podataka samo na temelju združenih razdioba varijabli x i y. Mnogi diskriminativni modeli koji se koriste u strojnom učenju uključuju, uz mnoge druge, i ove modele:

– Linearna diskriminantna analiza (engl. *Linear discriminant analysis, LDA*);
– Strojevi s potpornim vektorima (engl. *Support vector machines, SVM*);
– Boosting;
– Uvjetna slučajna polja (engl. *Conditional Random Fields, CRF*);
– Logistička regresija.

Generativni modeli

U teoriji vjerojatnosti pojam generativnog modela označava model kojim na slučajan način, najčešće uz neke poznate skrivene parametre, generiramo vidljive podatke. Konkretno, model određuje razdiobu vjerojatnosti više varijabli, i to vidljivih podataka na ulazu i niza oznaka pripadnosti (labela) za te podatke te u skladu s tim razdiobama generira podatke slične ulaznima. U području strojnog učenja generativni modeli se koriste za direktno modeliranje podataka, kao i pomoć pri određivanju funkcije gustoće uvjetne vjerojatnosti u sustavu.

Za razliku od diskriminativnih modela, generativni model modelira vjerojatnosnu tablicu svih varijabli sustava, dok diskriminativni model osigurava model vjerojatnosti samo za ciljne varijable uvjetno s obzirom na vidljive (promatrane) varijable. Zbog toga generativni model možemo koristiti za simuliranje (tj. generiranje) vrijednosti bilo koje varijable u modelu, dok diskriminativni model dozvoljava samo računanje vrijednosti ciljnih varijabli, uvjetno s vidljivim vrijednostima.

Neki od generativnih modela koji se koriste pri rješavanju problema regresije, klasifikacije ili redukcije dimenzionalnosti su:

– naivne Bayesove mreže;
– skriveni Markovljevi lanci (engl. *Hidden Markov model, HMM*);
– model Gaussovih mješavina (engl. *Gaussian mixture model*);
– multinomijalna distribucija.

3. Duboki generativni modeli

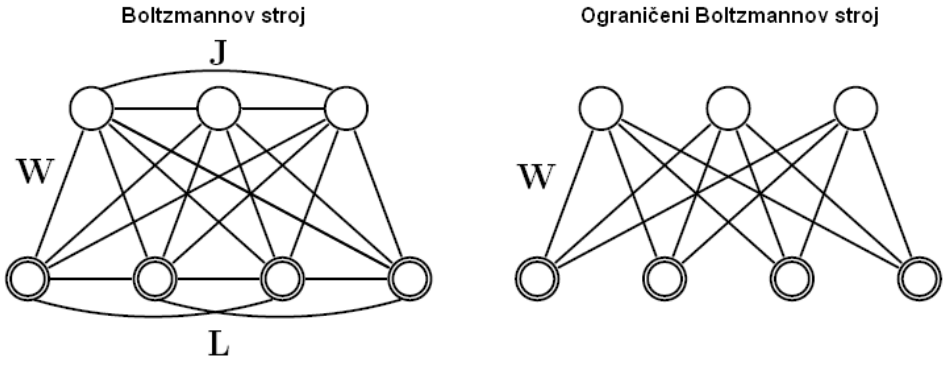
Pojam dubokih generativnih modela označava modele kod kojih povezanost između čistih podataka i značajki koje model iz podataka izvuče nije jasna na prvi pogled, već je povezanost "dublja" – na razini koja je najčešće čovjeku neintuitivna i koju nije moguće vidjeti golim okom. Duboke generativne modele ćemo prikazivati pomoću mreže koja sadrži mnogo slojeva skrivenih varijabli, gdje svaki sloj izvlači skrivene korelacije iz nižeg sloja.

Sustav koji će biti izgrađen u ovom radu je duboki generativni model koji se temelji na podmodelu koji može izlučiti značajke potrebne za uspješno rješavanje problema određivanja sličnosti dokumenata. Taj model se naziva ograničenim Boltzmannovim strojem (engl. *Restricted Boltzmann Machine, RBM*). Sustav čini dubokim to što je temeljen na mreži koja je građena od više ograničenih Boltzmannovih strojeva, koji su i sami bipartitni, neusmjereni grafički modeli. Model ograničenog Boltzimannovog stroja je u domeni strojnog učenja relativno nov model, a koristi se pri filtriranju i pretraživanju slika i informacija.

3.1. Ograničeni Boltzmannovi strojevi

Boltzmannov stroj je mreža simetrično uparenih binarnih jedinica. Građen je od dva sloja — vidljivog i skrivenog — i pripadnih veza između i unutar slojeva. Skriveni sloj Boltzmannovog stroja izvlači složene korelacije višeg reda.

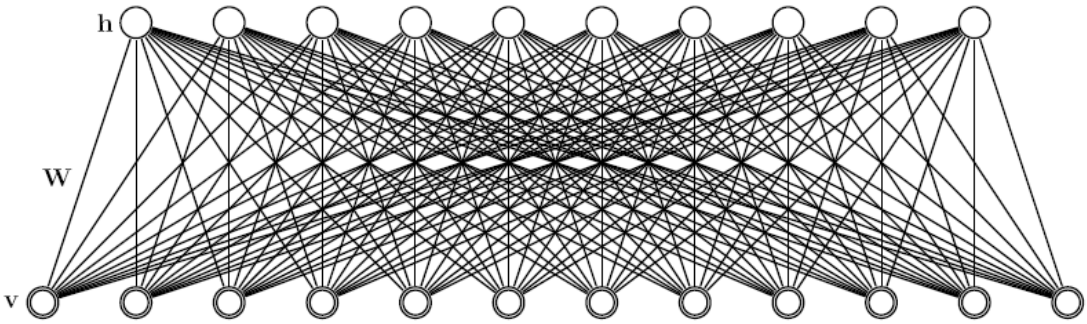
Razlika između Boltzmannovog stroja i ograničenog Boltzmannovog stroja je u tome što ograničeni Boltzmannov stroj nema veze unutar slojeva, već samo one međuslojne, kao što je pokazano na slici 3.1. Samim time ograničeni Boltzmannov stroj je pogodniji za sustave u kojima je potrebno modelirati samo međuslojne veze, a ujedno čini i jednostavniji model za opis čitavog sustava. Upravo zato će se nastavak poglavlja baviti ograničenim Boltzmannovim strojevima.



Slika 3.1: Razlika između Boltzmannovog stroja i ograničenog Boltzmannovog stroja.

Ograničeni Boltzmannovi strojevi

Ograničen Boltzmannov stroj (engl. *Restricted Boltzmann Machine, RBM*) je poseban oblik slučajnog Markovljevog polja koji ima dvoslojnu arhitekturu (Salakhutdinov, 2009). Kod takve arhitekture jedan sloj zovemo vidljivim, a drugi skrivenim te između njih postoje veze kao što je pokazano na slici 3.2. Vidljiv sloj koji se sastoji od D čvorova označavamo sa $\mathbf{v} \in \{0,1\}^D$, dok skriveni sloj od F čvorova označavamo kao $\mathbf{h} \in \{0,1\}^F$.



Slika 3.2: Ograničeni Boltzmannov stroj (RBM).

RBM-ovi pripadaju skupu modela koje nazivamo modelima temeljenim na energiji. Oni dodjeljuju skalarni iznos energije svakoj konfiguraciji varijabli koje su predstavljene čvorovima u mreži. Učenje takvih modela temeljimo na oblikovanju funkcije energije tako da ona ima poželjna svojstva; npr. željeli bismo postaviti izgled funkcije energije tako da poželjne konfiguracije imaju nižu energiju od ostalih.

Energija nekog stanja RBM-a $\{\mathbf{v}, \mathbf{h}\}$ je dana izrazom:

$$E(\mathbf{v}, \mathbf{h}; \theta) = -\mathbf{v}^\top W \mathbf{h} - \mathbf{b}^\top \mathbf{v} - \mathbf{a}^\top \mathbf{h} = \\ = -\sum_{i=1}^{D} \sum_{j=1}^{F} W_{ij} v_i h_j - \sum_{i=1}^{D} b_i v_i - \sum_{j=1}^{F} a_j h_j, \tag{3.1}$$

gdje su $\theta = \{W, \mathbf{b}, \mathbf{a}\}$ parametri modela; W_{ij} predstavlja težinsku vezu između čvora u vidljivom sloju i i čvora u skrivenom sloju j, dok su a_j i b_i pristranosti čvorova. Distribucija vjerojatnosti za čitav RBM dana je izrazom:

$$P(\mathbf{v}, \mathbf{h}; \theta) = \frac{1}{Z(\theta)} exp\left(-E(\mathbf{v}, \mathbf{h}; \theta)\right). \tag{3.2}$$

U gornjoj formuli uvedena je oznaka Z, koja služi za svođenje zbroja svih vjerojatnosti na 1, i naziva se normalizirajuća konstanta:

$$Z(\theta) = \sum_{\mathbf{v}} \sum_{\mathbf{h}} exp\left(-E(\mathbf{v}, \mathbf{h}; \theta)\right). \tag{3.3}$$

Jednostavnom manipulacijom gornje jednadžbe za vjerojatnost, sumacijom po svim mogućim kombinacijama čvorova skrivenog sloja, možemo dobiti vjerojatnost da uz zadane parametre vidljivi sloj poprimi određenu vrijednost:

$$P(\mathbf{v}; \theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} exp\left(-E(\mathbf{v}, \mathbf{h}; \theta)\right). \tag{3.4}$$

Uvjetne vjerojatnosti vidljivog sloja s obzirom na skriveni, i obratno su dani izrazima

$$P(\mathbf{v} \mid \mathbf{h}; \theta) = \prod_i p(v_i \mid \mathbf{h}), \tag{3.5}$$

$$P(\mathbf{h} \mid \mathbf{v}; \theta) = \prod_j p(h_j \mid \mathbf{v}), \tag{3.6}$$

što konkretno računamo pomoću logističke funkcije (S-funkcije), na sličan način kao i u neuronskoj mreži:

$$p(h_j = 1 \mid \mathbf{v}) = g\left(\sum_i W_{ij} v_i + a_j\right), \tag{3.7}$$

$$p(v_i = 1 \mid \mathbf{h}) = g\left(\sum_j W_{ij} h_j + b_i\right). \tag{3.8}$$

gdje je $g(x) = 1/(1 + exp(-x))$ logistička funkcija.

Očito je da je učenje RBM-a ekvivalentno učenju parametara modela, a to su težine W i pristranosti **a** i **b**. Kao što je poznato, u domeni strojnog učenja se ispravni parametri statističkog modela procjenjuju metodom procjene najveće izglednosti (*engl. maximum likelihood estimation, MLE*). U našem modelu to je jednadžba 3.4. U praksi je često pogodnije raditi s logaritmom te vrijednosti – logaritamskom izglednosti. Deriviranjem logaritamske izglednosti modela definiranog RBM-om po parametrima modela dobijamo izraze za procjenu dotičnih parametara:

$$\frac{\partial logP\left(\mathbf{v};\theta\right)}{\partial W} = E_{P_{data}}\left[\mathbf{v}\mathbf{h}^{\top}\right] - E_{P_{Model}}\left[\mathbf{v}\mathbf{h}^{\top}\right], \tag{3.9}$$

$$\frac{\partial logP\left(\mathbf{v};\theta\right)}{\partial \mathbf{a}} = E_{P_{data}}\left[\mathbf{h}\right] - E_{P_{Model}}\left[\mathbf{h}\right], \tag{3.10}$$

$$\frac{\partial logP\left(\mathbf{v};\theta\right)}{\partial \mathbf{b}} = E_{P_{data}}\left[\mathbf{v}\right] - E_{P_{Model}}\left[\mathbf{v}\right]. \tag{3.11}$$

Vidimo da su izrazi koji određuju promjenu u parametrima modela slični; razlika između očekivanja koje je izračunato preko vidljivih, ulaznih podataka, tj. empirijske raspodjele i očekivanja čija je raspodjela definirana modelom, točnije jednadžbom 3.2. Očito je da vrijednost $E_{P_{data}}[]$ možemo izračunati čim imamo ulazne podatke, dok je mnogo veći problem računanje vrijednosti $E_{P_{model}}[]$. Da bi dobili točnu vrijednost, potrebno je eksponencijalno vrijeme $\min\{D, F\}$, što je za praktičnu primjenu jako sporo te se stoga u praksi koristi aproksimacijom koja se naziva Kontrastna divergencija (engl. *Contrastive Divergence, CD*).

3.1.1. Gibbsovo uzorkovanje

Gibbsovo uzorkovanje (Casella and George, 1992) je algoritam pomoću kojeg generiramo niz uzoraka iz združene razdiobe vjerojatnosti dvaju ili više slučajnih varijabli modela. Najčešće se koristi u području statistike i statističke fizike, a služi za aproksimaciju združene razdiobe. Tako, na primjer, Gibbsovo uzorkovanje možemo koristiti za računanje aproksimacije marginalne distribucije neke varijable, ili skupa varijabli (kao na primjer nepoznate parametre skrivenih varijabli) te za računanje integrala u postupku procjene vrijednosti očekivanja neke varijable.

Gibbsovo uzorkovanje koristi se kao alat za statističko zaključivanje, najčešće u Bayesovim mrežama, a pripada Monte Carlo algoritmima za Markovljeve lance. Algoritam je primjenjiv kada nam združena razdioba nije eksplicitno poznata, ili kada je teško dobiti uzorke iz takve razdiobe, ali je poznata uvjetna razdioba svake varijable. U tom slučaju Gibbsovo uzorkovanje olakšava generiranje uzoraka varijabli. Možemo reći da algoritam Gibbsovog uzorkovanja generira instancu za svaku od varijabli, a na

temelju razdiobe po kojoj se ta varijabla ravna. Za generiranje instanci uzima u obzir trenutna stanja ostalih varijabli u sustavu.

Implementacija algoritma Gibbsovog uzorkovanja

Pretpostavimo da želimo dobiti određen broj instanci sustava $\mathbf{X} = \{x_1, \ldots, x_n\}$, a sve na temelju združene razdiobe svih varijabli u sustavu $p(x_1, \ldots, x_n)$. Označimo uzorak, tj. instancu sustava u t-tom koraku sa $\mathbf{X}^{(t)} = \{x_1^{(t)}, \ldots, x_n^{(t)}\}$. Tada provodimo sljedeće korake algoritma:

1. Inicijaliziramo sustav s početnim vrijednostima, i to na način da za svaku varijablu sustava x_i odredimo jednu od mogućih vrijednosti, čime dobijamo $\mathbf{X}^{(0)}$.

2. Za uzorak u koraku t, moramo uzorkovati sve varijable sustava. Tako varijablu $x_i^{(t)}$ uzorkujemo na temelju uvjetne vjerojatnosti:

$$p(x_i^{(t)} | x_1^{(t)}, \ldots, x_{i-1}^{(t)}, x_{i+1}^{(t-1)}, \ldots, x_n^{(t-1)}). \tag{3.12}$$

Možemo vidjeti da algoritam generira niz uzoraka na temelju Markovljevog lanca koji prolazi kroz sva stanja sustava. Kada uzorci dođu u stacionarno stanje, tada imamo združenu vjerojatnost $p(x_1, \ldots, x_n)$ za sve varijable u sustavu. Dakle, ako algoritam provodimo dovoljno dugo do stacionarnog stanja, lako možemo dobiti združenu vjerojatnost.

Naravno, postoje i neke modifikacije algoritma, koje su pogodne jer nam omogućavaju slobodu pri uzorkovanju. Kao što smo rekli, u $t = \infty$ algoritam sigurno dolazi do vrijednosti združene vjerojatnosti. No, u velikom broju slučajeva, nije potrebno jako dugo provoditi algoritam da bi se došlo do vrijednosti koje su jako blizu pravim vrijednostima. Drugo pitanje je kako inicijalno postaviti vrijednost varijabli. Istraživanja su pokazala da početne vrijednosti ne utječu na konačan rezultat, a u nekim slučajevima i na brzinu pronalaženja pravih rezultata, tako da imamo slobodu odabrati inicijalne vrijednosti varijabli kako nam najviše odgovara.

Gibbsovo uzorkovanje na primjeru

Za demonstraciju algoritma Gibbsovog uzorkovanja iskoristit ćemo već prije spomenut primjer sa čistačima ulica i kišom. Dakle, za sustav koji je prikazan na slici 2.3 i određen vjerojatnostima prema tablici 2.1 želimo izvesti zaključak o vjerojatnosti $P(K|C, M)$. Pretpostavimo da su varijable sustava K i C inicijalno postavljeni u 1 (događaji koji su nam otprije poznati), tada uzorkujemo iz razdiobe $P(O, M|K, C)$.

Algoritam Gibbsovog uzorkovanja djeluje na sljedeći način:

1. Inicijaliziramo varijable koje nisu poznate na neke početne vrijednosti, recimo $K = 1$ i $O = 1$. To je stanje sustava $\mathbf{X}^{(0)}$.

2. Odaberemo varijablu K i uzorkujemo pa osvježavamo njenu vjerojatnost po formuli:

   $$P(K|O, C, M),$$

   gdje je vrijednost za varijablu O njena vrijednost u prošlom koraku. Nakon toga uzorkujemo varijablu O na isti način, po formuli:

   $$P(O|K, C, M),$$

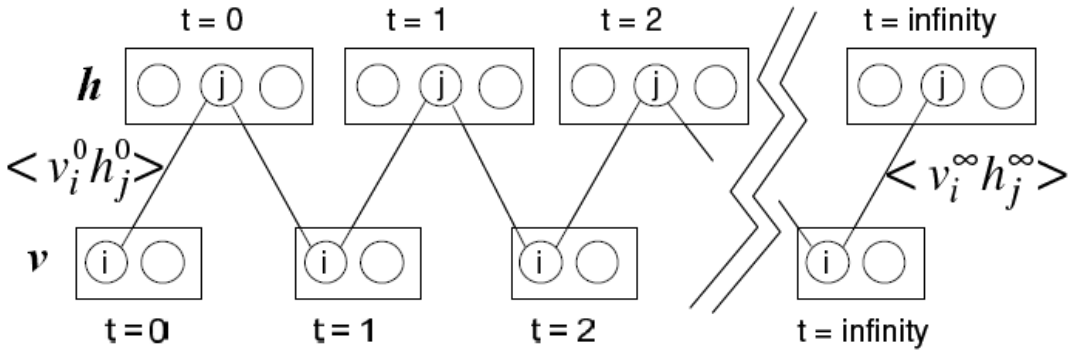
   gdje je vrijednost varijable K izračunata u prošlom koraku. Ovu točku algoritma ponavljamo do stacionarnog stanja.

Vidimo da nam za provođenje algoritma trebaju uvjetne vjerojatnosti za računanje varijable, ako su sve tri ostale vrijednosti poznate. Te se vjerojatnosti izračunaju na temelju tablice vjerojatnosti, izgleda samog grafičkog modela i Bayesove formule, čime dobijamo sve potrebne podatke za provođenje algoritma.

3.1.2. Učenje RBM metodom kontrastne divergencije

Vidjeli smo da se učenje mreže temelji na razlici empirijske raspodjele i one definirane modelom, a da nam je problem računati raspodjelu modela u konstantnom vremenu. Teoretski, potpuno točnu vrijednost možemo dobiti tako da započnemo sa slučajno generiranim stanjem u jednom od slojeva i provedemo Gibbsovo uzorkovanje; na temelju slučajno odabrnih vrijednosti jednog sloja računamo vrijednosti u svim čvorovima drugog sloja. Taj postupak ponavljamo sada sa izračunatim čvorovima drugog sloja kao podacima na temelju kojih računamo opet vrijednosti prvog sloja, kao što se može vidjeti na slic 3.3. Ovaj postupak se ponavlja sve dok sustav ne dođe u ravnotežu, čime smo dobili izraze koji odgovaraju distribuciji samog RBM-a.

Ako pratimo formulu 3.10, vidimo da za svaku težinu w_{ij}, između čvora u vidljivom sloju *i* i čvora u skrivenom sloju *j*, računamo razliku između $v_i^0 h_j^0$, kada vektori **v** i **h** odgovaraju čistim podacima na ulazu i podacima skrivenog sloja dobivenim iz ulaznih podataka, i $v_i^\infty h_j^\infty$ koji je dobiven Gibbsovim uzorkovanjem, i predstavlja distribuciju podataka koju pretpostavlja RBM.



Slika 3.3: Gibbsovo uzorkovanje između vidljivog i skrivenog sloja RBM-a.

Metoda kontrastne divergencije temelji se na prethodnom teoretskom razmatranju, uz modifikaciju da se Gibbsovo uzorkovanje ne provodi do ravnotežnog stanja, već samo za T koraka; točnije rečeno, pri $T = \infty$ imamo učenje s procjenom najveće izglednosti. U ovom slučaju dolazi do kompromisa između preciznosti određivanja distribucije modela i vremenske učinkovitosti računanja istog, a jako je korisna spoznaja da u mnogim domenama koje koriste RBM, CD učenje sa $T = 1$ (ili CD1 učenje) pokazuje izvrsne rezultate (Hinton et al., 2006). Dakle za učenje parametara modela možemo koristiti sljedeću formulu:

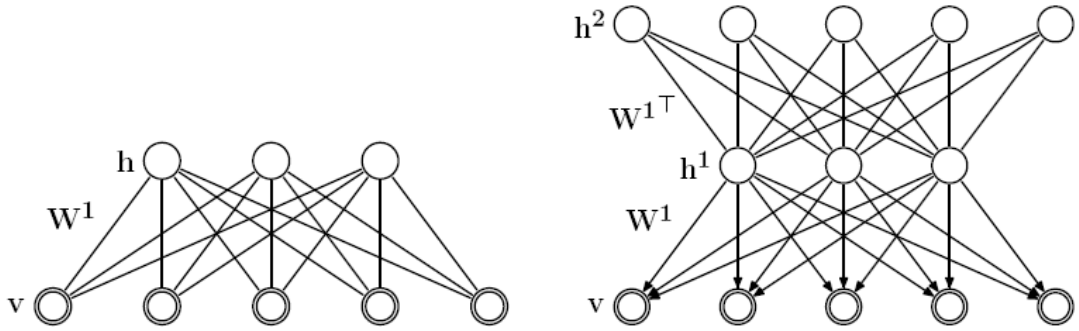
$$\Delta W = \alpha \left(E_{P_{data}} \left[\mathbf{v}\mathbf{h}^\top \right] - E_{P_T} \left[\mathbf{v}\mathbf{h}^\top \right] \right), \tag{3.13}$$

gdje je α stopa učenja, a $T = 1$. Na jednak način učimo i pristranosti **a** i **b**, svaki uz svoju stopu učenja.

3.2. Pohlepan algoritam učenja za duboke mreže

Za zadatak redukcije dimenzionalnosti, koji će biti obrađen u ovom radu, jedan RBM nije dovoljan jer ne može obuhvatiti sve značajke potrebne za efikasnu redukciju. Za kvalitetno izlučivanje značajki treba nam duboka mreža, a ona se postiže tako da "slažemo" RBM-ove jedan na drugi; skriveni sloj RBM-a na $n - 1$ razini je vidljivi sloj RBM-a na n-toj razini, kao što se može vidjeti na slici 3.5. Učenje cijele mreže temelji se na učenju svakog RBM-a posebno, a opravdanje za to će biti dano u sljedećem odlomku.

Za dokaz da učenje jednog RBM-a ne utječe na ostale RBM-ove u mreži moramo napraviti neke pretpostavke. Prva je da samo najviša dva sloja imaju izgled čistog RBM-a – neusmjeren bipartitni graf, dok svi niži slojevi čine usmjerenu sigmoidnu mrežu kao što je prikazano na slici 3.4. Promatramo osnovni slučaj od samo dva



Slika 3.4: prikaz običnog ograničenog Boltzmannovog stroja te dva ograničena Boltzmannova stroja povezana u mrežu

RBM-a, sa pripadnim slojevima $\mathbf{v}$, $\mathbf{h}^1$ i $\mathbf{h}^2$. Raspodjela vjerojatnosti za sve slojeve tada ima sljedeći izraz:

$$P\left(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2; \theta\right) = P\left(\mathbf{v} \mid \mathbf{h}^1; W^1\right) P\left(\mathbf{h}^1, \mathbf{h}^2; W^2\right), \tag{3.14}$$

gdje su $\theta = \{W^1, W^2\}$ parametri modela, $P\left(\mathbf{v} \mid \mathbf{h}^1; W^1\right)$ predstavlja usmjereni, sigmoidni dio mreže, a $P\left(\mathbf{h}^1, \mathbf{h}^2; W^2\right)$ predstavlja distribuciju RBM-a na drugoj razini. Svaku od ovih raspodjela možemo dalje raspisati na sljedeći način:

$$P\left(\mathbf{v} \mid \mathbf{h}^1; W^1\right) = \prod_i p\left(v_i \mid \mathbf{h}^1; W^1\right), \tag{3.15}$$

$$P\left(v_i = 1 \mid \mathbf{h}^1; W^1\right) = g\left(\sum_j W_{ij}^1 h_j^1\right), \tag{3.16}$$

$$\begin{aligned} P\left(\mathbf{h}^1, \mathbf{h}^2; W^2\right) &= \frac{1}{Z\left(W^2\right)} exp\left(\mathbf{h}^{1\top} W^2 \mathbf{h}^2\right) = \\ &= \frac{1}{Z\left(W^2\right)} \prod_i \left(1 + exp\left(\sum_j W_{ji}^2 h_j^1\right)\right). \end{aligned} \tag{3.17}$$

Druga pretpostavka da bi dokaz bio ispravan je ta da je broj čvorova u skrivenom sloju drugog RBM-a jednak broju čvorova vidljivog sloja prvog RBM-a, a parametri su vezani, tj. $W^2 = W^{1\top}$. Sada pokušajmo izračunati distribuciju vjerojatnosti samo

po prva dva sloja (koji su činili RBM prve razine):

$$P\left(\mathbf{v}, \mathbf{h}^1; \theta\right) = P\left(\mathbf{v} \mid \mathbf{h}^1; W^1\right) \times \sum_{\mathbf{h}^2} P\left(\mathbf{h}^1, \mathbf{h}^2; W^2\right) =$$

$$\prod_i p\left(v_i \mid \mathbf{h}^1; W^1\right) \times \frac{1}{Z\left(W^2\right)} \prod_i \left(1 + exp\left(\sum_j W^2_{ji} h^1_j\right)\right) =$$

$$\prod_i \frac{exp\left(v_i \sum_j W^1_{ij} h^1_j\right)}{1 + exp\left(v_i \sum_j W^1_{ij} h^1_j\right)} \times \frac{1}{Z\left(W^2\right)} \prod_i \left(1 + exp\left(\sum_j W^2_{ji} h^1_j\right)\right) =$$

$$\mid W^2_{ji} = W^1_{ij}, Z(W^1) = Z(W^2) \mid = \frac{1}{Z(W^1)} \prod_i \left(exp\left(v_i \sum_j W^1_{ij} h^1_j\right)\right) =$$

$$\frac{1}{Z(W^1)} exp\left(\sum_{ij} W^1_{ij} v_i h^1_j\right). \qquad (3.18)$$

Formula koju smo dobili identična je formuli 3.4, što pokazuje da možemo u mrežu uključiti RBM-ova po želji bez straha da će promjena parametara u jednom RBM-u utjecati na ostale RBM-ove. Međutim, vidimo da je druga pretpostavka jako ograničavajuća za našu mrežu, najviše zato jer nas obvezuje da svaki sloj ima jednak broj čvorova kao i njegov drugi prethodnik i drugi sljedbenik. Pokazano je (Salakhutdinov, 2009) da zanemarivanje ove pretpostavke ne utječe bitno na kvalitetu rješenja te da smo slobodni modelirati slojeve bez ograničenja na broj čvorova.

Algorithm 1 Algoritam inicijalnog učenja duboke mreže sastavljene od više RBM-ova

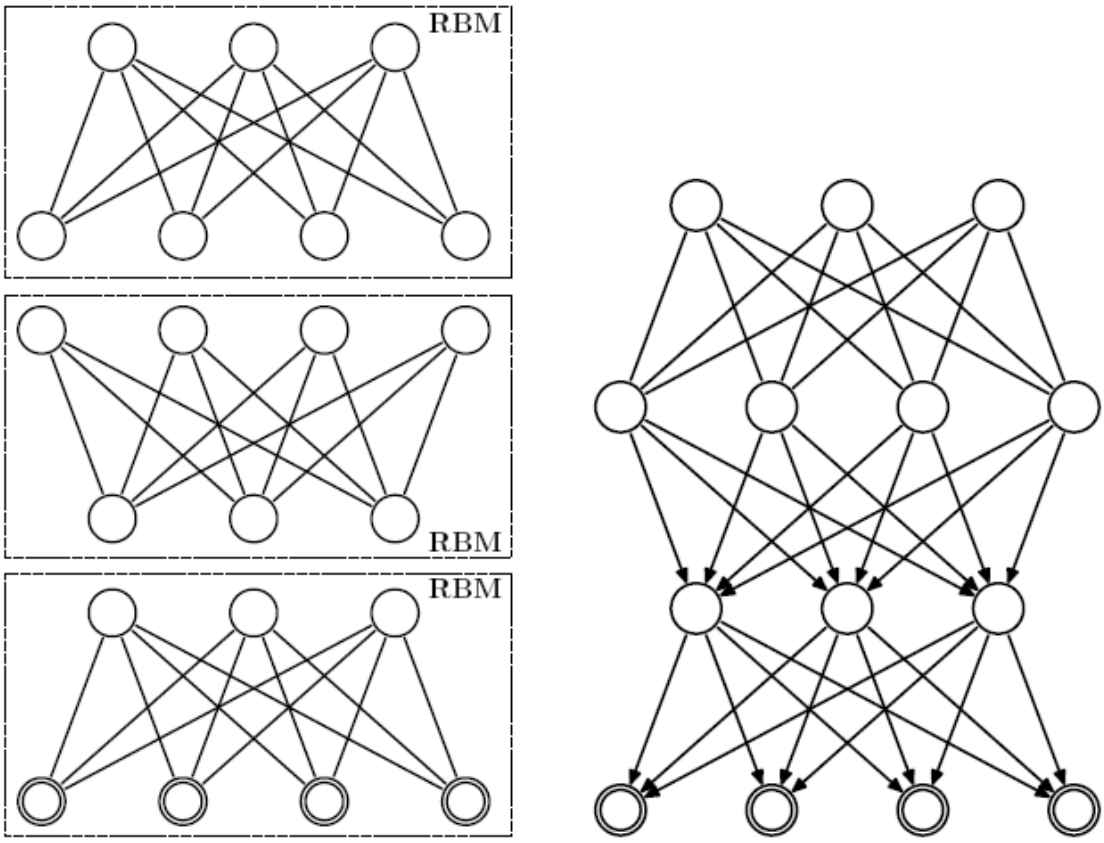
Prilagodi težine W^1 i pristranosti prvog RBM-a po formuli, a na temelju ulaznih podataka.

Fiksiraj parametre prvog RBM-a i koristi uzorke $\mathbf{h}^1$ dobivene iz $P\left(\mathbf{h}^1|\mathbf{v}, W^1\right)$ kao ulazne podatke za učenje drugog RBM-a.

Fiksiraj parametre drugog RBM-a i koristi uzorke $\mathbf{h}^2$ dobivene iz $P\left(\mathbf{h}^2|\mathbf{h}^1, W^2\right)$ kao ulazne podatke za učenje trećeg RBM-a.

Nastavi rekurzivno dok ne naučiš sve RBM-ove.

Pohlepan algoritam učenja funkcionira na dubokim mrežama sačinjenih od RBM-ova tako da se prvo nauči najdonji RBM sa svojim parametrom W^1, kao i pristranostima **a** i **b**. Nakon toga se inicijaliziraju težine druge razine (drugog RBM-a) slučajnim vrijednostima koje se tada treniraju na temelju $\mathbf{h}^1$ – vektora dobivenog kao izlaz iz prvog RBM-a. Proceduru nastavljamo za sve iduće RBM-ove, a za ulazne podatke uvijek uzimamo izlaz iz RBM-a niže razine.



Slika 3.5: Lijevo: model se sastoji od više RBM-ova "naslaganih" jedan na drugi; svaki je učen zasebno. **Desno:** isti model nakon pohlepnog učenja – RBM-ovi se spajaju u jedan model

3.2.1. Modifikacija RBM-a za modeliranje vektora riječi

Kao što će biti opisano u nastavku, dokumente je najpogodnije predstaviti vektorom riječi – to je vektor kojemu je iznos svake dimenzije jednak broju pojavljivanja određene riječi u dokumentu. Riječi koje čine svaku od dimenzija su one koje imaju najveći broj pojavljivanja u čitavom korpusu dokumenata.

Sada promotrimo ograničeni Boltzmannov stroj u kojem se vidljivi sloj, umjesto standardnih binarnih jedinica, sastoji od jedinica koje mogu biti u jednom od više mogućih stanja — takozvanih “softmax” jedinica.

Označimo sa $\mathbf{v} \in \{1, \ldots, K\}^D$ vektor vidljivog sloja, koji je u domeni konačnih diskretnih vrijednosti. Sa $\mathbf{h} \in \{0, 1\}^F$ označimo skriveni sloj, koji se sastoji od binarnih jedinica, a $\mathbf{V}$ je matrica dimenzija $K \times D$, čiji elementi $v_i^k = 1$ ako jedinica i vidljivog sloja poprima vrijednost k.

Energiju stanja $\{\mathbf{V}, \mathbf{h}\}$ možemo definirati na sljedeći način:

$$E(\mathbf{V}, \mathbf{h}) = -\sum_{i=1}^{D}\sum_{j=1}^{F}\sum_{k=1}^{K} W_{ij}^{k} h_j v_i^k - \sum_{i=1}^{D}\sum_{k=1}^{K} v_i^k b_i^k - \sum_{j=1}^{F} h_j b_j, \tag{3.19}$$

gdje W_{ij}^k označava težinu između jedinice i iz vidljivog sloja, koja ima vrijednost k, i skrivene jedinice j. Vrijednost b_i^k je pristranost jedinice i s vrijednosti k, dok je a_j pristranost jedinice j. Na temelju ove formule, na sličan način kao i za obični RBM izvode se sljedeće formule za računanje uvjetne razdiobe:

$$p\left(v_i^k = 1|\mathbf{h}\right) = \frac{exp\left(b_i^k + \sum_{j=1}^{F} h_j W_{ij}^k\right)}{\sum_{q=1}^{K} exp\left(b_i^q + \sum_{j=1}^{F} h_j W_{ij}^q\right)} \tag{3.20}$$

$$p\left(h_j = 1|\mathbf{V}\right) = g\left(a_j + \sum_{i=1}^{D}\sum_{k=1}^{K} v_i^k W_{ij}^k\right). \tag{3.21}$$

4. Redukcija dimenzionalnosti pomoću dubokih generativnih modela

U ovom poglavlju će biti opisano kako iskoristiti duboki generativni model, konkretno RBM, za redukciju dimenzionalnosti visokodimenzionalnih ulaznih podataka.

Učenje koristi dvije glavne značajke RBM-a: ono može biti učinkovito provedeno na neoznačenom skupu podataka, a nakon učenja RBM omogućava fino podešavanje parametara koristeći algoritam s propagacijom pogreške unazad (engl. *backpropagation algorithm*), u daljnjem tekstu označen kao BP-algoritam. Uz pojam finog podešavanja parametara, biti će predstavljen i pojam autoenkodera i njegovog učenja pohlepnim algoritmom, čiji rezultati nadmašuju one dobivene široko primjenjivanim metodama, kao što su analiza glavnih komponenti (engl. *Principal Component Analysis, PCA*) i dekompozicija singularnih vrijednosti (engl. *Singular Value Decomposition, SVD*) (Salakhutdinov, 2009).

4.1. Duboke mreže pri nelinearnoj redukciji dimenzionalnosti

Danas, u vrijeme kada podaci nastaju velikom brzinom, u širokoj je primjeni vrlo velik broj visokodimenzionalnih podataka. Znanstvenici koji rade s takvim podacima suočavaju se s problemom redukcije dimenzionalnosti – kako otkriti niskodimenzionalnu strukturu visokodimenzionalnog podatka, a da se podatak pritom može jednoznačno odrediti. Postoje mnoge tehnike redukcije dimenzionalnosti koje su široko primjenjivane, a mogu se ugrubo podijeliti na linearne metode (PCA), nelinearna preslikavanja (autoenkoderi) te metode zasnovane na udaljenosti (engl. *Localy-Linear Embedding, LLE*).

Mnoge od postojećih metoda imaju značajne nedostatke ovisno o strukturi ulaznih podataka. Ako su podaci pretvoreni u niskodimenzionalnu reprezentaciju nelinearni,

linearne metode ne uspijevaju dobro prikazati takvu strukturu podataka. Prednost linearnih metoda je njihova efikasnost i brzina izračunavanja. Algoritmi nelinearnog preslikavanja su vrlo spori pri učenju te često znaju zapinjati u lokalnim minimumima. Metode zasnovane na udaljenosti izuzetno su jake pri redukciji dimenzionalnosti, ali su računalno vrlo zahtjevne te se stoga ne mogu primjenjivati na velikom broju visokodimenzionalnih podataka.

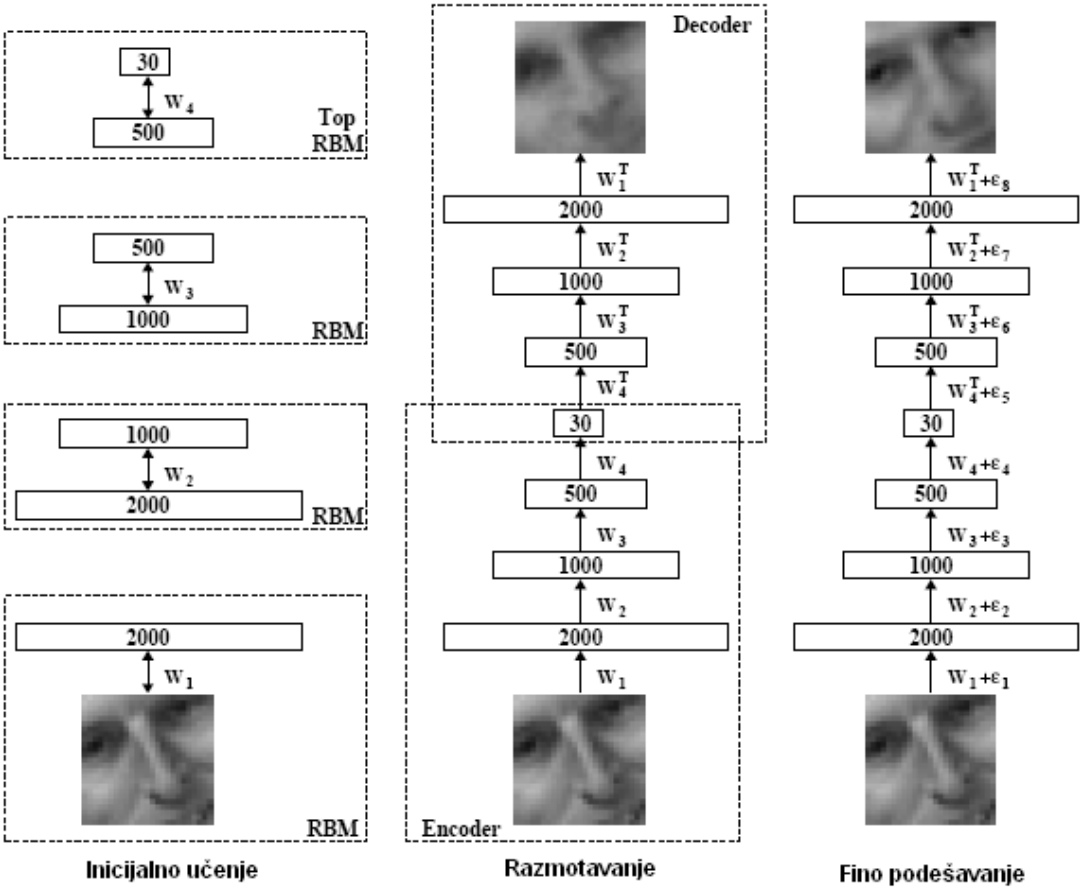
Od gore navedenih metoda, a imajući u vidu znanje o RBM-u, metode nelinearnog preslikavanja mogu najbolje iskoristiti prednosti RBM-a za rješavanje svojih nedostataka; pohlepan algoritam učenja može u kraće vrijeme postaviti težine modela na vrijednosti koje vrlo dobro vrše redukciju dimenzionalnosti, a tada se primjenjuje BP-algoritam, koji je standardan način učenja kod metoda nelinearnog preslikavanja. BP-algoritam se uklapa u koncept finog podešavanja parametara te je očito da bi kombinacija ideje mreže RBM-ova s algoritmima nelinearnog preslikavanja mogla biti pogodna za rješavanje problema redukcije dimenzionalnosti.

4.2. Autoenkoderi

Autoenkoder je umjetna neuronska mreža koja služi za učenje efikasnog kodiranja ulaznih podataka (Hinton and Salakhutdinov, 2006). Cilj autoenkodera je naučiti niskodimenzionalni prikaz (sažetu reprezentaciju) skupa podataka (enkodiranje), dakle, služi pri redukciji dimenzionalnosti. Autoenkoderi koriste tri ili više slojeva – ulazni sloj, jedan ili više manjih skrivenih slojeva čija je zadaća provesti enkodiranje ulaznih podataka te izlazni sloj koji ima isto značenje kao i ulazni sloj.

Standardni način učenja autoenkodera je BP-algoritam kako bi se smanjila pogreška rekonstrukcije. Kao što je pokazano u radovima (DeMers and Cottrell, 1993) i (Hecht-Nielsen, 1995), općenito je vrlo teško i vremenski neučinkovito optimizirati nelinearne autoenkodere koji imaju veći broj skrivenih slojeva i velik broj parametara. To je jedan od glavnih razloga zašto se ova tehnika ne koristi u praksi, iako je potencijalno vrlo moćna. Da bi nadvladali taj problem, koristimo niz RBM-ova koje najprije učimo svakog posebno pohlepnim algoritmom. Takvim pristupom možemo brzo pronaći parametre koji daju dobar model za rekonstrukciju ulaznih podataka.

Nakon inicijalnog učenja RBM-ova, spajamo ih u jednu mrežu te "razmotamo" model tako da imamo enkoderski i dekoderski dio kao što je pokazano na slici 4.1. U sredini se nalazi izlazni sloj duboke mreže, koji pri razmotavanju modela postaje središnji sloj. Kako on ne sudjeluje u izlučivanju značajki duboke mreže, u autoenkoderu se ponaša kao linearni sloj — samo prenosi ulaznu vrijednost na izlaz. Inicijalno,

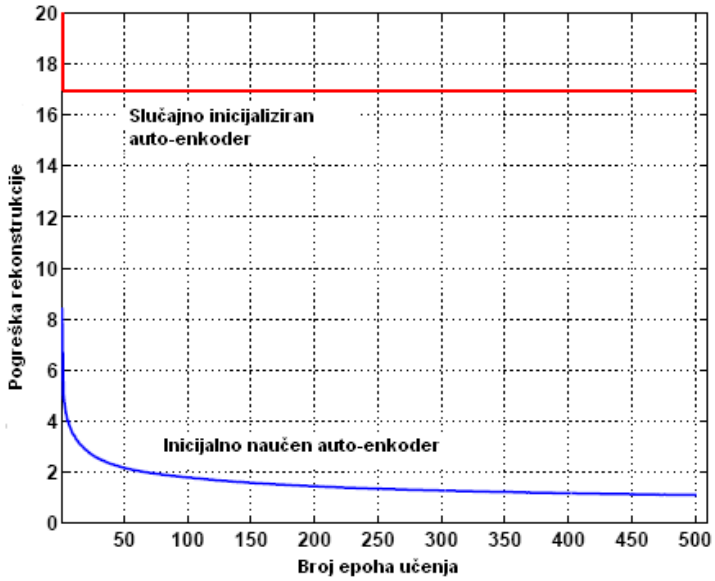


Slika 4.1: Inicijalno učenje se sastoji od učenja niza RBM-ova. Nakon toga, RBM-ovi su složeni u sustav te "razmotani" u autoenkoder, koji se tada fino podešava BP-algoritmom.

dekoder je inverzno preslikan enkoder, s jednakim težinama. Nakon što dobijemo takvu mrežu, na ulaz mreže dovodimo podatke za fino podešavanje; aktiviramo sloj po sloj enkodera i dekodera te na izlazu iz dekodera promatramo koliki je gubitak informacije uspoređujući s podatkom na ulazu. Samo fino podešavanje je ustvari običan BP-algoritam kojim mijenjamo težine čitavog modela da dobijemo optimalnu rekonstrukciju podataka.

Možemo se upitati koliko inicijalno učenje koristi modelu, ako ga ionako ponovno učimo BP-algoritmom. Također, može se postaviti pitanje o uspješnosti sustava ovisno o njegovoj dubini; je li mreža od više slojeva sposobnija pri učenju izlučivanja značajki?

Istraživanja provedena u (Salakhutdinov, 2009) pokazuju kako se performanse dubokih modela nad kojim je izvršeno inicijalno učenje mogu uvelike razlikovati od performansi modela koji je prošao samo kroz fino podešavanje parametara (inicijalno su mu parametri postavljeni na slučajan način). Na slici 4.2 je vidljivo kako bez inicijal-



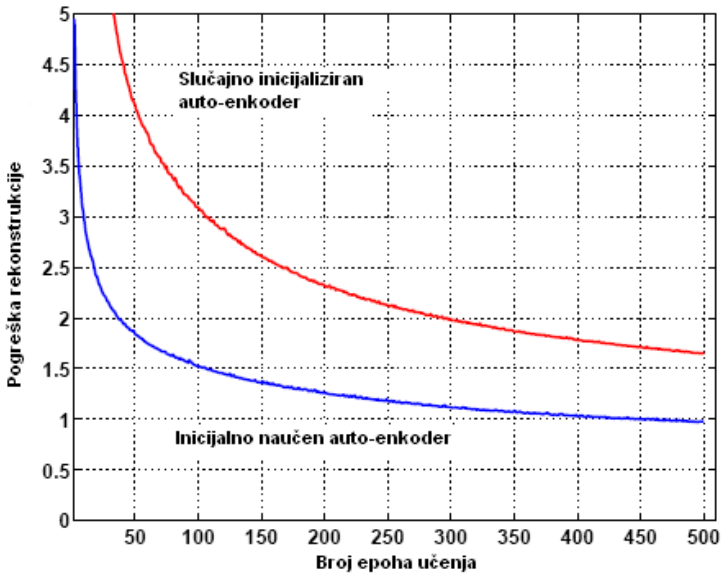
Slika 4.2: Duboki autoenkoder (broj jedinica po slojevima: 784-400-200-100-5) brzo uči nakon inicijalnog učenja, dok uopće ne napreduje ako nije inicijalno naučen, (Salakhutdinov, 2009)

nog učenja autoenkoder zapinje u lokalnom minimumu; rekonstruira ulazne podatke kao srednju vrijednost svih podataka koji su korišteni za učenje.

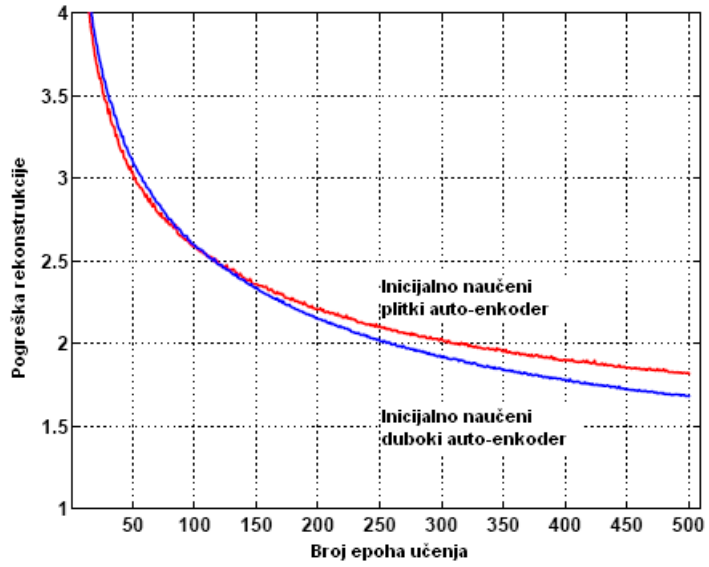
Plitki autoenkoderi, za razliku od dubokih, mogu imati dobre performanse i bez inicijalnog učenja, međutim, kao što je pokazano na slici 4.3 , vidi se da inicijalno učenje u velikoj mjeri smanjuje vrijeme ukupnog učenja.

U slučaju kada imamo plitki i duboki autoenkoder, koji imaju približno sličan broj parametara, duboki autoenkoder daje nešto bolje rezultate od plitkoga. Rezultate te usporedbe vidimo na slici 4.4.

Istraživanja u području redukcije dimenzionalnosti podataka (Salakhutdinov, 2009) pokazuju kako metoda nelinearnih autoenkodera sastavljenih od RBM-ova nadmašuje performanse najčešće korištenih metoda, kao što su LSA, PCA, LLE ili ISOMAP.



Slika 4.3: Plitki autoenkoder (784-532-6) može učiti finim podešavanjem i bez inicijalnog učenja, ali mnogo sporije dolazi do optimalnih parametara, (Salakhutdinov, 2009)



Slika 4.4: Plitki autoenkoder (704-108-6) ima malo lošije performanse od dubokog autoenkodera (784-100-50-25-6). Oba modela imaju približno jednak broj parametara, (Salakhutdinov, 2009).

5. Primjena dubokih mreža za pretraživanje dokumenata

U ovom će se poglavlju pokazati kako mreža sa mnogo skrivenih slojeva i i stotinama tisuća parametara može biti naučena da otkriva semantičke binarne kodove ulaznih podataka, i to tako da slični ulazni vektori imaju slične binarne kodne riječi. To nam omogućuje veoma brz i točan način za dohvaćanje dokumenata koji su slični s dokumentom upita.

5.1. Mjere sličnosti

Problem pretraživanja dokumenata (engl. *document retrieval*) možemo opisati na način da korisnik odabere jedan dokument iz velikog skupa dokumenata, i želi da računalo vrati kao rezultat one dokumente koji su semantički povezani s odabranim dokumentom. Rezultantni dokumenti su prikazani korisniku rangirani po opadajućoj sličnosti s dokumentom upita.

Pitanje kod svih algoritama za dohvaćanje dokumenata jest rješenje kako izračunati koliko su dokumenti slični, odnosno mjeru sličnosti dokumenata. Za računanje mjere sličnosti potrebno je imati uvid u sve riječi dokumenta, pošto je to jedini način na temelju kojeg možemo vidjeti koreliranost dokumenata, što je i intuitivno jasno – dok ne pročitamo dva dokumenta, nikako ne možemo znati da li su oni slični.

Jedan od načina za predstavljanje dokumenta jest onaj pomoću vektorskog prostora – dokument je prikazan pomoću n-dimenzionalnog vektora, gdje svaka dimenzija odgovara jednoj riječi, ključnoj riječi ili čitavoj frazi. Ako dokument sadrži takvu riječ, ključnu riječ ili frazu, vrijednost te dimenzije vektora je veća od 0. Postoje razni načini za računanje takvog vektora – jedan od načina, ako gledamo riječi kao dimenzije, je pronalaženje svih riječi u korpusu (skupu svih dokumenata) čime se dobija dimenzionalnost vektora riječi za dokumente. Pošto broj riječi može biti jako velik za velike korpuse, jedna od modifikacija je računanje određenog broja najčešće pojavlji-

vanih riječi u korpusu. Nakon što smo dobili n najčešćih riječi, svaki od dokumenata predstavljamo n-dimenzionalnim vektorom.

Nakon što je svaki od dokumenata korpusa predstavljen vlastitim vektorom, sličnost možemo izračunati na razne načine. Neki od načina su semantička sličnost i prekrivanje. Semantička sličnost gleda koliko je pojedina riječ iz dokumenta upita semantički slična sa svim riječima iz pojedinog dokumenta iz ostatka korpusa. Za informaciju o semantičkoj sličnosti potrebno je izgraditi adekvatan sustav koji može prepoznati semantičku sličnost riječi (kao npr. semantički graf). Mjera prekrivanja utvrđuje za koji se postotak riječi iz dokumenta upita pojavljuje semantički slična riječ u dokumentu koji rangiramo.

Uz spomenute mjere sličnosti koje ne iskorištavaju reprezentaciju dokumenta kao vektora, imamo mjere sličnosti koje koriste vektorske operacije kako bi na što bolji način izračunale sličnost dvaju dokumenata. Jedna od takvih je i kosinusna mjera sličnosti, kojom računamo kosinus kuta između vektora koji predstavlja dokument upita i vektora dokumenta koji rangiramo. Formula mjere sličnosti je:

$$\cos(q, d) = \frac{\mathbf{v}_q \cdot \mathbf{v}_d}{\|\mathbf{v}_q\| \|\mathbf{v}_d\|}, \tag{5.1}$$

gdje je $\mathbf{v}_q$ vektorska reprezentacija dokumenta upita q, a $\mathbf{v}_d$ vektorska reprezentacija dokumenta d koji trenutno rangiramo. Očito je da je mjera sličnosti dva potpuno jednaka dokumenta jednaka 1, dok je kod dokumenata čiji vektor riječi nema nikakvih podudaranja mjera sličnosti 0.

5.1.1. Mjera TfIdf

Jedna od statističkih metoda za prikaz vektora riječi je TfIdf. Metoda se sastoji od dvije komponente – frekvencije pojma (engl. *term frequency, Tf*) i inverzne frekvencije pojma u dokumentima (engl. *inverse document frequency, Idf*). Frekvencija pojma je omjer između pojavljivanja konkretnog pojma u dokumentu i pojavljivanja svih pojmova u dokumentu, kao što je prikazano formulom:

$$Tf(w, d) = \frac{n_{wd}}{\sum_{k \in d} n_{kd}}, \tag{5.2}$$

gdje je n_{wd} broj pojavljivanja pojma w u dokumentu d, a $\sum_{k \in d} n_{kd}$ zbroj pojavljivanja svih pojmova u dokumentu d.

Inverzna frekvencija pojma w se računa preko formule:

$$Idf(w) = \log\left(\frac{N}{n_{wz}}\right), \tag{5.3}$$

gdje je N broj dokumenata u korpusu, a n_{wz} broj dokumenata koji u sebi sadrže pojam w.

Nakon što su izračunate ove dvije komponente, vrijednost TF-IDF dobijamo njihovim umnoškom:

$$TfIdf\,(w,d) = Tf\,(w,d) \times Idf\,(w)\,. \tag{5.4}$$

Svaka dimenzija vektora pri vektorskom prikazu dokumenta se računa tako da se za svaki od pojmova izračuna TfIdf vrijednost, kao što je prikazano sljedećom formulom:

$$\mathbf{v}_d = (\,TfIdf\,(w_1,d)\,,\,TfIdf\,(w_2,d)\,,\ldots,\,TfIdf\,(w_n,d))\,, \tag{5.5}$$

gdje je $\mathbf{v}_d$ vektorski prikaz dokumenta d, a $w_1, w_2, \ldots, w_n$ pojmovi odabrani za prikaz dimenzija vektora.

Metoda TfIdf ima nekoliko loših strana; računa sličnost dokumenata na temelju svih riječi u korpusu, što je izrazito sporo za velike korpuse i jezike s velikim vokabularom. Nadalje, pretpostavlja da dokumenti s različitim riječima nemaju nikakve povezanosti, što ne mora biti slučaj, jer ne pretpostavlja semantičku sličnost između riječi. Da bi se riješili ovi problemi, predstavljeni su novi modeli koji mogu prepoznati neke niskodimenzionalne i skrivene reprezentacije podataka, koji mogu izlučiti manji broj odgovarajućih značajki za rekonstrukciju podataka, a samim tim smanjiti i dimenzionalnost ulaznih podataka. Jednostavna i široko primjenjivana takva metoda je latentna semantička analiza (engl. *Latent Semantic Analysis, LSA*).

5.1.2. Latentna semantička analiza

Latentna semantička analiza je metoda koja analizira veze između skupa dokumenata i riječi koje oni sadrže (Deerwester et al., 1990). Glavna pretpostavka latentne semantičke analize je da se povezane riječi često u dokumentima nalaze u neposrednoj blizini, što nam govori da ova metoda koristi i prepoznavanje dokumenata sa semantičke strane. Također, slične riječi se nalaze u sličnim okruženjima ostalih riječi, što se također uzima u obzir. U središtu metode je matrica koja prikazuje riječi koje se pojavljuju u velikim paragrafima, pa čak i cijelim korpusima – retci matrice označavaju pojedinu riječ, dok stupci matrice označavaju pojedini paragraf ili dokument. Možemo reći da ta matrica prikazuje učestalost i strukturu pojavljivanja pojmova u dokumentu. Kao što znamo, broj različitih pojmova kao i broj dokumenata u korpusu može biti vrlo velik, što za posljedicu ima rijetkost matrice pojam-dokument. Takva rijetka matrica je nepogodna za daljnju obradu zbog svoje veličine te se matematičkom obradom,

točnije, pomoću dekompozicije singularnih vrijednosti (engl. *Singular Value Decomposition, SVD*) dobije matrica koja je aproksimacija originalne i ima niži rang, ali je zato pogodnija za obradu. Na temelju takve aproksimirane matrice moguće je izlučiti niskodimenzionalnu semantičku strukturu te je pri dohvaćanju dokumenata iskorišten i semantički sadržaj, a ne samo čisto brojanje riječi koje se u dokumentima pojavljuju.

LSA je linearna metoda te je zato vrlo ograničena pri prepoznavanju semantičkih struktura koje su nelinearne. Kao rješenje tog problema je uvedena probabilistička verzija LSA (pLSA) (Hofmann, 1999) te latentna Dirichletova alokacija (engl. *Latent Dirichlet Allocation, LDA*) (Blei and Jordan, 2003). Kao i druge metode koje pripadaju u ovu skupinu, ove metode koriste skrivene (latentne) varijable kako bi lakše predstavili "višu razinu" odakle mogu na jednostavniji način klasificirati ili usporediti dokumente.

Ovi probabilistički modeli mogu se promatrati kao grafički modeli u kojima su skrivene varijable povezane izravno na varijable koje predstavljaju ulazni vektor pojmova. Njihov glavni nedostatak je što je nemoguće provesti točno zaključivanje o vrijednostima vjerojatnosti varijabli sustava preko objašnjavanja uzroka, tako da je za računanje posteriorne distribucije varijabli modela potrebno ili mnogo vremena, ili uvođenje velikih nepreciznosti, što otežava prilagođavanje modela podacima. Također, brzo zaključivanje je nužno za praktičnu uporabu modela pri problemu dohvaćanja informacija. (Welling et al., 2005) predložili su dvoslojni neusmjereni grafički model koji je generalizacija ograničenih Boltzmannovih strojeva. Samo učenje preko procjene najveće izglednosti nije moguće u ovakvim modelima, ali se učenje može provesti metodom kontrastne divergencije. Razvoj sličnih, neusmjerenih modela u (Gehler et al., 2006) i (Xing et al., 2005) pokazuju da su ovakvi modeli pogodni za dohvaćanje dokumenata te da su performanse slične kao i kod široko primjenjivanih metoda (LSA, TfIdf).

Međutim, postoje i neka ograničenja s obzirom na konfiguraciju i performanse modela. Mnoge strukture neće moći biti najbolje predstavljene pomoću samo jednog skrivenog sloja; istraživanja su pokazla (Salakhutdinov, 2009) da mreže s mnogo skrivenih slojeva i stotinama tisuća parametara mogu otkriti skrivene reprezentacije koje mnogo bolje funkcioniraju pri dohvaćanju dokumenata. Nadalje, mnogi od postojećih algoritama temelje se na računanju mjere sličnosti između dokumenta upita i ostalih dokumenata u korpusu. Sličnost se uglavnom računa uz pomoć čistih riječi (TfIdf) ili niskodimenzionalnih reprezentacija ulaznih vektora (LSA), a njihovo računanje traje proporcionalno s brojem dokumenata u korpusu. Jasno je da je većina korpusa jako

velika i da računanje mjera sličnosti na prikazan način nije naročito učinkovit. Postoje neka vremenska poboljšanja ovih modela, ali samo uz uporabu posebnih podatkovnih struktura, kao što su KD stabla. KD stablo (ili k-dimenzionalno stablo) je struktura podataka za organizaciju točaka u k-dimenzionalnom prostoru.

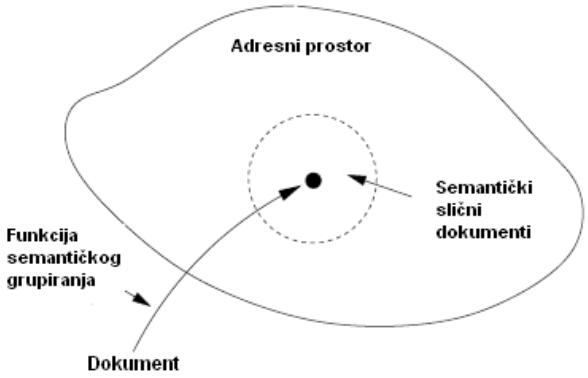
5.2. Semantičko grupiranje

U ovom radu će biti opisana metoda za učinkovito dohvaćanje dokumenata nazvana semantičko grupiranje. Prednost ove metode je u tome što iskorištava mogućnosti višestrukih skrivenih slojeva za učinkovitu redukciju dimenzionalnosti. Također, važno je napomenuti da metoda vraća listu sličnih dokumenata u vremenu koje je nezavisno o broju dokumenata u korpusu, već samo linearno zavisno o broju dokumenata koje želimo u listi sličnih dokumenata.

Osnovna ideja semantičkog grupiranja je u tome da se mreža nauči na takav način da za ulazni vektor na izlaz daje binarne kodne vrijednosti. Ako su dokumenti slični, kodne vrijednosti dvaju dokumenata imaju malu Hammingovu udaljenost. To svojstvo modela koristi se za vrlo brzo dohvaćanje sličnih dokumenata: odredimo radijus Hammingove kugle i dohvatimo sve dokumente čija je kodna vrijednost unutar kugle. Na taj način dobijamo listu sličnih dokumenata, a određivanjem radijusa kontroliramo broj dohvaćenih dokumenata.

Jedno od proširenja koje pridonosi brzini dohvaćanja sličnih dokumenata jest spremanje dokumenata na fizičke adrese jednake kodnim vrijednostima dobivenim na izlazu iz mreže. Možemo reći da ovim postupkom generiramo semantički adresni prostor, gdje se slični dokumenti nalaze na sličnim adresama. Pošto su kodne vrijednosti binarne, računanje adresa odakle dohvaćamo slične dokumente se svodi na jednostavne i procesorski brze bitovne operacije.

Konkretno, model mreže koja izvodi semantičko grupiranje koristi prethodno opisana saznanja o RBM-u, dubokim mrežama i njihovom učenju. Duboki generativni model sastoji se od više slojeva: najniži sloj predstavlja vektor pojmova dokumenta, dok najviši (točnije, najdublji) sloj reprezentira naučenu binarnu kodnu riječ za taj dokument. Najviša dva sloja čine neusmjereni bipartitni graf, dok ostali slojevi tvore mrežu vjerovanja, tj. veze između slojeva su usmjerene od viših prema nižim, kao što je pokazano na slici 3.5.



Slika 5.1: Shematski prikaz rada semantičkog grupiranja (preuzeto iz (Salakhutdinov, 2009))

5.2.1. Učenje modela za semantičko grupiranje

Učinkovito učenje ovakvog modela može se postići tako da svaka dva sloja promatramo kao nezavisni RBM, pri čemu najviši i najniži sloj sudjeluju u samo jednom RBM-u, dok svi ostali slojevi sudjeluju u dva odvojena RBM-a – u jednom kao vidljivi, a u jednom kao skriveni sloj. Inicijalno učenje težina između slojeva i pristranosti skrivenih varijabli se provodi za svaki RBM posebno, prema pohlepnom algoritmu.

Nakon pohlepnog algoritma parametri su postavljeni na dobre inicijalne vrijednosti, ali one ne daju bolje rezultate od mreže koja ima samo jedan skriveni sloj. Da bi se potpuno iskoristile mogućnosti koju nam pruža višeslojna arhitektura mreže potrebno je dodatno naučiti parametre. Pošto su parametri već postavljeni na vrijednosti koje su vrlo bliske konačnim, dodatno učenje nije pretjerano vremenski zahtjevno. Ovaj proces učenja naziva se finim podešavanjem parametara modela i koristi BP-algoritam, unutar kojeg metodu gradijentnog spusta, čime dobijamo mnogo bolji model.

Modeliranje vidljivih podataka i učenje najnižeg RBM-a

Na ulazu u mrežu nalazi se vidljivi sloj koji predstavlja vektor pojmova samog dokumenta. Pitanje koje se nameće je kako modelirati takav vektor. Postoji nekoliko tehnika koje modeliraju vektor riječi, a samim tim i najdonji RBM u mreži.

Vidljivi sloj prvog RBM-a sastoji se od elemenata koji mogu imati bilo koju cjelobrojnu vrijednost, tj. iz skupa $\mathbf{N}_0$, dok su elementi skrivenog sloja binarni.

Prvi način za modeliranje vidljivih podataka je korištenje uvjetne Poissonove distribucije, čime dobijamo izraz za vjerojatnost elementa vidljivog sloja ako su poznati

svi elementi skrivenog sloja:

$$p\left(v_i = n|\mathbf{h}\right) = Ps\left(n, \frac{exp\left(\lambda_i + \sum_j h_j w_{ij}\right)}{\sum_k exp\left(\lambda_k + \sum_j h_j w_{kj}\right)} N\right), \tag{5.6}$$

gdje je w_{ij} težina između vidljive riječi i i elementa j skrivenog sloja. Broj riječi koji se uzima u obzir ovisi o implementaciji – jedan od načina je taj da se prije učenja pronađe određen broj najčešćih riječi i tada je svaka od tih riječi jedna dimenzija vektora. N je ukupna duljina čitavog dokumenta, tj. zbroj riječi u dokumentu, a koje su odabrane kao najčešće za cijeli korpus – $N = \sum_i v_i$.

Vrijednost λ_i označava parametar pristranosti uvjetnog Poissonovog modela za riječ i, dok funkcija Ps označava formulu za računanje vjerojatnosti podataka pri Poissonovoj razdiobi:

$$Ps\left(n, \lambda\right) = \frac{e^{-\lambda}\lambda^n}{n!}. \tag{5.7}$$

Skriveni sloj se ravna po uvjetnoj Bernoullijevoj razdiobi:

$$p\left(h_j = 1|\mathbf{v}\right) = \sigma\left(b_j + \sum_i w_{ij} v_i\right), \tag{5.8}$$

gdje je b_j pristranost elementa j skrivenog sloja, a $\sigma()$ označava sigmoidalnu funkciju:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{5.9}$$

Drugi način za modeliranje najnižeg RBM-a je po modelu *Softmax*, koji je već ranije opisan u odjeljku 3.2.1.

Učenje ostalih RBM-ova

Nakon prvog RBM-a, čiji je vidljiv sloj opisan jednom od tehnika (Poissonova razdioba ili model Softmax), a skriveni sloj Bernoullijevom distribucijom i sigmoidalnom funkcijom, ostali RBM-ovi se uče jednakim algoritmom. Kao ulazne podatke vidljivog sloja, RBM-ovi dobijaju aktivacijske vjerojatnosti skrivenog sloja "nižeg" RBM-a. Jedina razlika između početnog i ostalih RBM-ova je u tome što su u ostalim strojevima oba sloja predstavljena binarnim elementima koji se ravnaju po Bernoullijevoj distribuciji. Formule po kojima se provodi osvježavanje elemenata dane su izrazima:

$$p\left(h_j = 1|\mathbf{v}\right) = \sigma\left(b_j + \sum_i w_{ij} v_i\right), \tag{5.10}$$

$$p\left(v_j = 1|\mathbf{h}\right) = \sigma\left(b_i + \sum_j w_{ij} h_j\right). \tag{5.11}$$

Pohlepni algoritam može se izvoditi i više puta kako bi se dobila naučena duboka hijerarhijska mreža, u kojoj svaki sloj značajki može prepoznati korelaciju između aktivnosti značajki u nižem sloju na jednoj višoj razini. Algoritam je prikazan u algoritmu 1 . Sam algoritam pohlepnog učenja je tek prvi korak u učenju mreže, pošto on pronalazi točku u prostoru parametara modela koja je globalno dobra, ali ne i najbolja. Algoritmom finog podešavanja parametara iskorištavamo gradijentni spust kako bismo našli točku u blizini trenutne, koja rezultira značajno boljim rezultatima.

Učenje čitavog modela – fino podešavanje

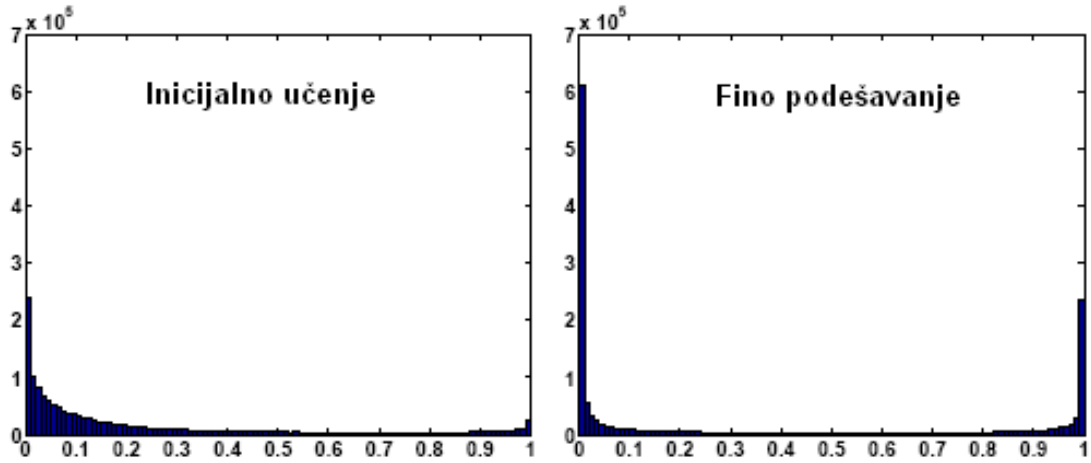
Postupak finog podešavanja kreće "razmotavanjem" RBM-ova u duboki autoenkoder kao što je prikazano na slici 4.1. Težine dodane između novih slojeva su simetrične obzirom na već postojeće težine. Da bismo mogli izvesti BP-algoritam, potrebno je malo modificirati prolazak kroz mrežu na način da skriveni slojevi više ne poprimaju samo binarne vrijednosti već da imaju vrijednost konkretnih vjerojatnosti aktivacije. Dodatno, moguće je modificirati ulazne podatke tako da svaku dimenziju vektora podijelimo s ukupnim brojem riječi u vektoru, čime dobijamo distribuciju vjerojatnosti svih riječi u dokumentu, čime rješavamo problem različitog broja riječi u dokumentima.

Nakon što je mreža naučena i fino podešena, za svaki ulaz u mrežu dobijemo odgovarajući izlaz u obliku vektora realnih brojeva u rasponu [0, 1]. Međutim, za semantičko grupiranje potrebno je kao izlaz imati kodove u binarnom obliku, stoga želimo BP-algoritam modificirati na način da mreža bude dobra pri rekonstrukciji podataka, ali da izlazni kodovi budu što je više moguće blizu binarnima. Na ovom mjestu potrebno je napomenuti da se izlazni sloj mreže ustvari nalazi u sredini "razmotanog" autoenkodera te da izlaz autoenkodera nije izlaz mreže koju dobijemo nakon učenja.

Postizanje binarnog oblika kodova

Kako bismo dobili binarne kodove, na ulaz u izlazni sloj mreže dodajemo Gaussov šum. Najbolji način za prenošenje informacije sa ulaznog sloja autoenkodera na izlazni sloj autoenkodera uz prisutnost šuma je da vrijednosti koje dolaze u izlazni sloj mreže učinimo ili jako velikima ili jako malima, kao što se može vidjeti na slici 5.2. To je rezultat adaptacije mreže na prisutan šum; u slučaju da su izlazne vrijednosti ostale iste, s malom razlikom, rekonstrukcija ulaznih podataka bi rezultirala velikom pogreškom.

Problem koji se javlja pri dodavanju Gaussovog šuma je u tome što on svojim nedeterminizmom može poremetiti BP-algoritam, točnije gradijentni spust. Zato je



Slika 5.2: Raspodjela vrijednosti aktivacijskih vjerojatnosti izlaznog sloja mreže (Salakhutdinov, 2009). Vidljivo je kako se dodavanjem šuma postižu ekstremi aktivacijskih vjerojatnosti.

potrebno koristiti tzv. deterministički šum, sa točno određenim očekivanjem i varijancom. Također, za svaku iteraciju učenja potrebno je uzorkovane vrijednosti šuma zapamtiti, i ne mijenjati tokom učenja, iz razloga da šum ne bi utjecao na BP-algoritam, tj. da ga algoritam može uključiti u svoj prolaz kroz mrežu i ispravljanje težina mreže.

Pitanje koje se može postaviti je hoće li mreža biti naučena za točno određene vrijednosti Gaussovog šuma, pošto je on deterministički? Odgovor je potvrdan, ali samo u slučajevima kad imamo vrlo malo podataka za učenje. Kad je broj iteracija učenja veći (veći od broja parametara mreže), ukupan broj različitih fiksiranih šumova predstavlja dobru aproksimaciju nedeterminističkog šuma.

Naravno, nije moguće postići da su sve vrijednosti čvorova u izlaznom sloju ili 0 ili 1, zato uzimamo neki prag kao granicu, tako da svi čvorovi čija je vrijednost veća od tog praga imaju vrijednost 1, dok se čvorovima sa manjom vrijednošću od praga nova vrijednost postavlja na 0. Konačan rezultat ne ovisi uvelike o tom pragu, kao što je pokazano u (Salakhutdinov, 2009).

Sam izgled algoritma za učenje duboke mreže koja vrši semantičko grupiranje može se vidjeti u pseudokodu.

Grafički izgled cijelog sustava je prikazan na slici 5.3.

Algorithm 2 Algoritam učenja duboke mreže za semantičko grupiranje

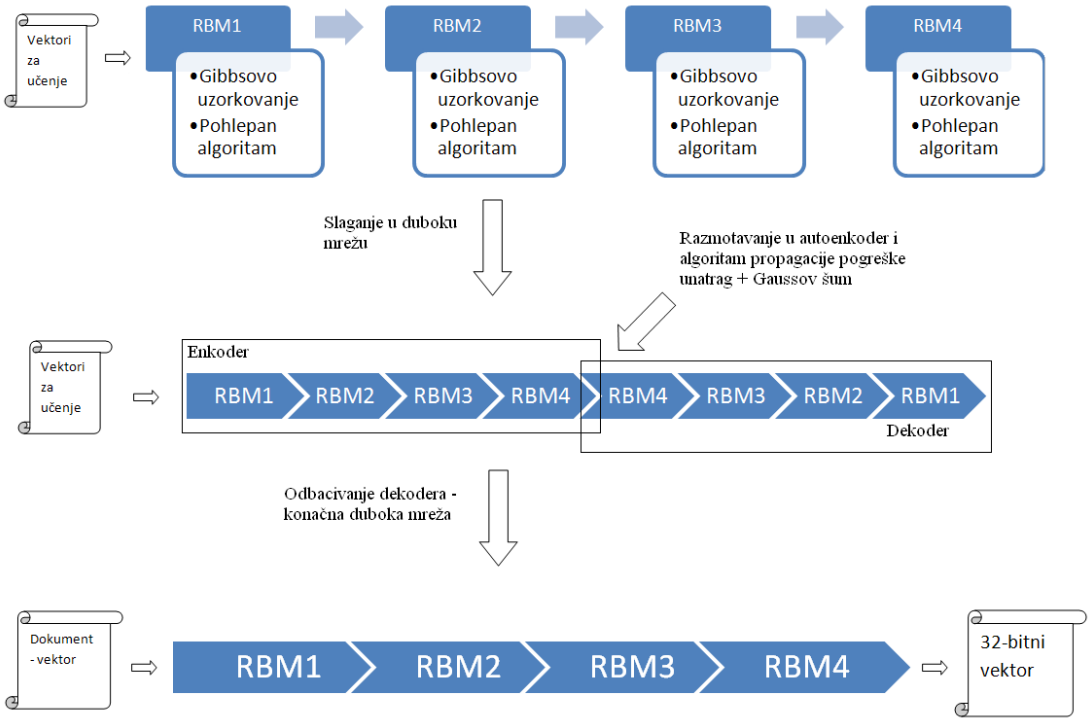
Na temelju ulaznih podataka pohlepnim algoritmom nauči početni RBM u mreži.

Sve ostale RBM-ove uči pohlepnim algoritmom na temelju podataka iz izlaznog sloja prošlog RBM-a.

Razmotaj tako naučenu mrežu u autoenkoder.

Upotrijebi backpropagation algoritam za fino podešavanje svih parametara mreže, uz dodatak Gaussovog šuma na izlaznom sloju.

Odbaci dekoderski dio autoenkodera, čime se dobiva duboka mreža za prikaz dokumenata pomoću binarnih vektora.



Slika 5.3: Konačan izgled sustava za semantičko grupiranje tekstovnih dokumenata.

6. Implementacija i evaluacija modela

6.1. Opis programskog rješenja

Programska implementacija sustava za semantičko grupiranje se sastoji od dva dijela. Prvi dio priprema ulazne podatke u oblik pogodan za obradu pomoću duboke mreže. Nad svim riječima u korpusu program vrši lematizaciju – svođenje riječi iz raznih oblika (roda, broja, lica, padeža i sl.) na osnovni oblik te od tako lematiziranih riječi odabire 2000 najčešćih riječi u korpusu. Nakon toga svaki članak predstavlja pomoću vektora na način da prebraja pojavljivanja svake od 2000 odabranih riječi te broj riječi zapiše na odgovarajuću dimenziju 2000-dimenzionalnog vektora. Kako bi se zaobišao problem različitog broja riječi u člancima, svaka dimenzija vektora je podijeljena s ukupnim brojem riječi članka koje su ušle u vektor.

Taj dio programskog rješenja je izveden u programskom alatu Visual Studio 2010, jeziku C#, uz korištenje biblioteke MLTools za lematizaciju riječi, koja je stvorena u KTLabu na Fakultetu elektrotehnike i računarstva u Zagrebu.

Izlaz tog dijela programa su datoteke u kojima su definirani vektori svih dokumenata korpusa na već opisan način. Stvara se više datoteka – jedna u sebi sadrži skup za učenje mreže, druga skup za testiranje mreže, a posljednje dvije su datoteke koje će služiti u vrednovanju naučene mreže. Te dvije datoteke sadrže vektore i pripadnosti svakog vektora određenoj kategoriji.

Ove datoteke su generirane u formatu pogodnom za čitanje u programskom okruženju Matlab (ekstenzija .ascii), kako bi ih drugi dio programa, koji je rađen u programskom okruženju Matlab mogao pročitati.

Drugi dio programa je konkretna implementacija duboke mreže na temelju algoritma opisanog u prethodnim poglavljima ovog rada. Konkretno, program prima vektore koji su prethodno stvoreni u C#, grupira vektore u skupine za učenje i testiranje (*engl. batch data*) te izvodi algoritam.

Mreža se sastoji od 4 RBM-a, a broj varijabli u slojevima je 2000 - 1000 - 500 - 250 - 32, od ulaza prema izlazu. Program na temelju ulaznih podataka za učenje provodi

inicijalno učenje svakog od 4 RBM-a (kroz 10 epoha), a nakon toga stvara duboku mrežu uz pripadni autoenkoder te ulazi u BP-algoritam finog podešavanja težina (kroz 20 epoha). Taj algoritam koristi metodu trostrukog linijskog pretraživanja u ravnini varijabli sustava, koja u temelju ima metodu konjugiranih gradijenata. Pri tom zadatku, koriste se i vektori za testiranje, kako bi mogli pratiti napredak učenja mreže. Također, pri prolasku kroz mrežu, na izlazni se sloj dodaje prethodno određen Gaussov šum, koji utječe na učinkovitiju binarizaciju izlaznih kodova, kako je pokazano u prethodnom poglavlju.

Kao rezultat programa na izlaz dobijamo mrežu, tj. težine između slojeva mreže, a na temelju toga možemo iz ulaznih vektora dobiti binarne kodove koji reprezentiraju ulazne podatke. Uspješnost mreže pri semantičkom grupiranju će se testirati na skupu za vrednovanje koji se sastoji od vektora koje mreža nikad prije nije vidjela.

6.2. Mjere vrednovanja

Da bi mogli prikazati učinkovitost algoritma, moramo imati objektivnu mjeru za vrednovanje. Među mnogim mjerama izdvajaju se najčešće korištene u vrednovanju skupa dokumenata – preciznost i odziv.

6.2.1. Preciznost i odziv

Preciznost i odziv su dvije najraširenije metode za vrednovanje i brojčani prikaz ispravnosti algoritma za pretraživanje dokumenata (ili bilo kojeg algoritma strojnog učenja). Pri tom zadatku algoritam na izlaz producira skup dokumenata, gdje se tada na semantičkoj razini utvrđuje njihova sličnost s dokumentom upita. Za potrebe ove metode jasno definiramo granicu sličnosti; dohvaćeni dokument je ili sličan, ili nije sličan s dokumentom upita.

Na najvišoj razini bismo mogli definirati preciznost kao mjeru točnosti dohvaćenog skupa podataka, dok bi odziv bila mjera potpunosti algoritma. Drugim riječima, visoka preciznost algoritma znači da su svi dohvaćeni dokumenti slični dokumentu upita, ali to ne garantira da je pronađen potpun skup sličnih dokumenata. S druge strane, visoka vrijednost odziva nam govori da je algoritam dohvatio najveći dio skupa sličnih dokumenata, ali je moguće da se uz slične dokumente, u skupu dohvaćenih nalazi mnogo dokumenata koji nisu slični.

Za primjer uzmimo algoritam koji bi u idealnom slučaju za dokument upita vratio skup od 20 sličnih dokumenata, dok je ukupan broj dokumenata iz kojih algoritam bira

slične 100. Primjer gdje bi preciznost bila maksimalna je upit koji dohvati samo jedan dokument, a on je sličan s dokumentom upita. No, u ovom slučaju je odziv jako malen. Primjer za maksimalan odziv bi bio da algoritam vrati svih 100 dokumenata, čime je pokriven cijeli skup sličnih dokumenata, ali je zato preciznost mala.

Formalne definicije ovih vrijednosti su dane sljedećim formulama:

$$\text{Preciznost} = \frac{\text{Broj dohvacenih relevantnih dokumenata}}{\text{Broj svih dohvacenih dokumenata}},$$

$$\text{Odziv} = \frac{\text{Broj dohvacenih relevantnih dokumenata}}{\text{Broj svih relevantnih dokumenata u korpusu}}.$$

Vidimo da je maksimalna vrijednost ovih metoda je 1.0, dok je minimalna 0.0.

F-mjera

U najvećem broju slučajeva, vrijednosti preciznosti i odziva ne razmatraju se zasebno, već se vrijednosti jedne mjere uspoređuju tako da se fiksira vrijednost druge mjere. Primjer takve usporedbe bi bila usporedba preciznosti pri fiksiranoj vrijednosti odziva 0.68. Također, postoji način kako obe mjere uključiti u izračun vrijednosti jedne druge mjere, kao što je F-mjera.

F-mjera se može definirati kao težinska harmonička sredina preciznosti i odziva, a računa se po formuli:

$$F_1 = 2 \cdot \frac{\text{Preciznost} \cdot \text{Odziv}}{\text{Preciznost} + \text{Odziv}}. \tag{6.1}$$

Ovaj izraz je poznat i kao F_1-mjera, gdje su preciznost i odziv u jednakom odnosu pri računanju konačne vrijednosti. U općem slučaju bi formula za F-mjeru, ili točnije F_β-mjeru bila sljedeća:

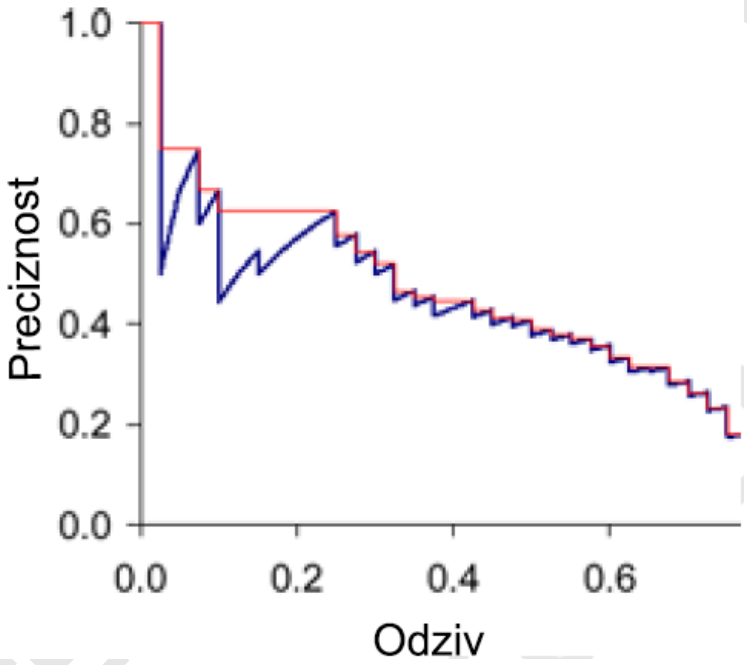
$$F_\beta = (1 + \beta^2) \cdot \frac{\text{Preciznost} \cdot \text{Odziv}}{\beta^2 \cdot \text{Preciznost} + \text{Odziv}}. \tag{6.2}$$

Iz ovog izraza možemo zaključiti kako faktor β označava korisnikovu pristranost prema odzivu, naspram preciznosti. Tako F_2-mjera daje duplo veći značaj odzivu, dok $F_{0.5}$-mjera daje duplo veći značaj preciznosti.

6.2.2. Graf preciznost-odziv

Nakon što smo izračunali vrijednosti preciznosti i odziva, možemo iscrtati graf preciznost-odziv, koji nam na jednostavan način predočava kako se te vrijednosti kreću s obzirom na broj dohvaćenih dokumenata.

Graf preciznost-odziv najčešće ima pilasti oblik, kao što je prikazano na slici 6.1. Nakon dohvaćenih k dokumenata, i računanja pripadnih vrijednosti preciznosti i odziva, dohvaćamo k+1 dokument. Ako taj dokument nije relevantan, preciznost se smanjuje, dok odziv ostaje isti. Ako k+1 dokument jest relevantan, rastu i preciznost i odziv. Ovo ponašanje je bitno jer se može pretpostaviti da je korisniku važnije prikazati nešto više ukupnih dokumenata s većom preciznošću umjesto manje dokumenata uz manju preciznost.



Slika 6.1: Graf preciznost-odziv

Točke grafa preciznost-odziv izračunavaju se za svaki upit pri vrednovanju te se potom skupni graf preciznost-odziv računa kao srednja vrijednost preciznosti za pojedine upite.

Na slici 6.1 se mogu vidjeti i vodoravne crte koje predstavljaju interpoliranu srednju preciznost. Motivacija u računanju interpolirane srednje preciznosti se nalazi u tome što informacije iz grafa preciznost-odziv možemo skupiti u nekoliko brojeva. Uobičajen način da se to napravi jest izračunati interpoliranu srednju preciznost u 11 točaka (Manning et al., 2007); od 0.0 do 1.0, s korakom od 0.1. Interpolirana preciznost u nekoj točki grafa je najviša vrijednost preciznosti za odziv veći ili jednak od odziva u toj točki grafa. Potom za te vrijednosti izmjerimo srednju vrijednost preko svih upita i dobijemo interpoliranu srednju preciznost u 11 točaka.

6.2.3. Srednja vrijednost prosjeka preciznosti

Srednja vrijednost prosjeka preciznosti (engl. *mean average precision, MAP*) je mjera koja opisuje kvalitetu sustava preko svih vrijednosti odziva. MAP se računa na način da se izračuna preciznost na svim (diskretnim) razinama odziva. Potom računamo prosjek tako izračunatih preciznosti za svaki upit te izračunamo srednju vrijednost dobivenih prosjeka preciznosti. Motivacija računanja srednje vrijednosti prosjeka preciznosti je u tome što dobivamo samo jedan broj koji nam govori o kvaliteti sustava za pretraživanje dokumenata, a koji se tada može uspoređivati s ostalim sustavima.

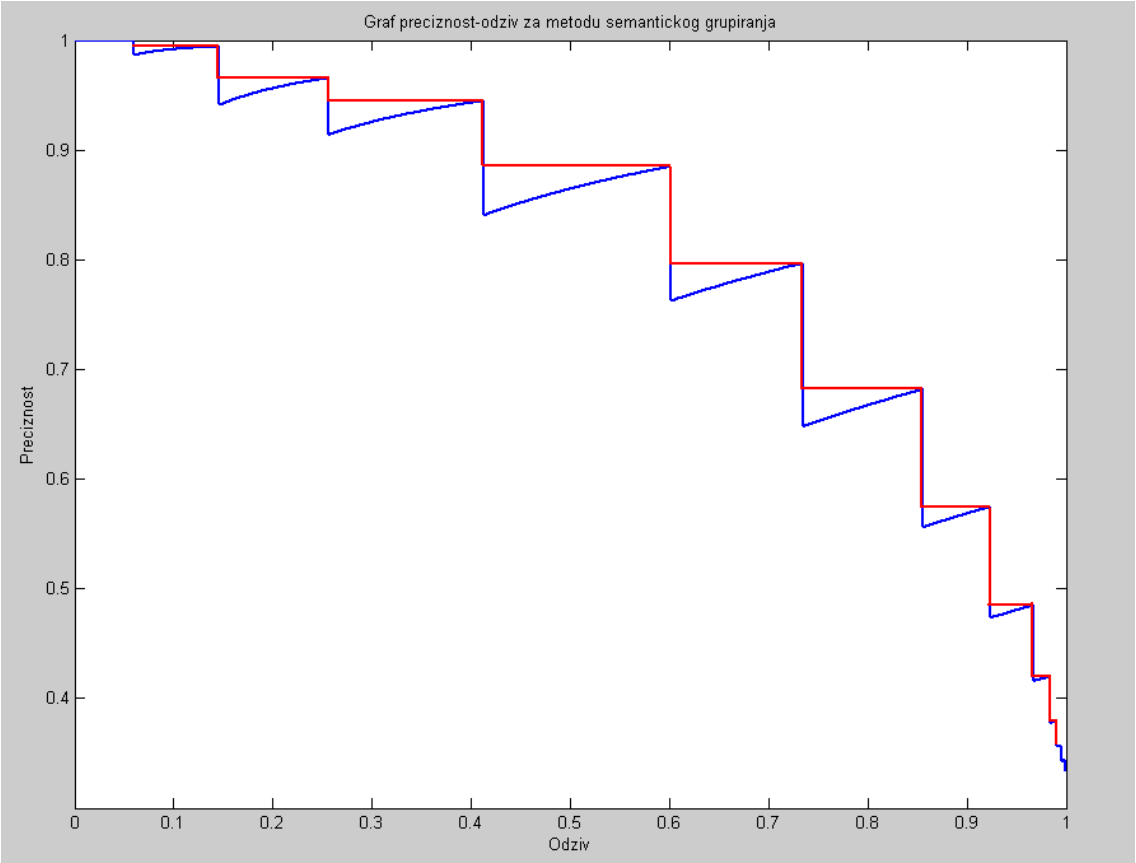
6.3. Rezultati implementacije

Nakon učenja mreže potrebno je taj sustav vrednovati, kako bi odredili kvalitetu rješenja. Mjera koja će se koristiti za vrednovanje je graf preciznost - odziv. Podaci korišteni pri učenju mreže su uzeti iz Vjesnikovog korpusa koji sadrži više od 500000 članaka, podijeljenih u 13 kategorija. Zbog vremenske složenosti koja je ovisna o broju članaka, ulazni podaci određeni su između tri kategorije korpusa u prvom dijelu programske implementacije – Crna kronika, Sport i Gospodarstvo. Skup za učenje čini 34900 članaka iz tih kategorija, dok skup za testiranje sadrži 6000 članaka (po 2000 iz svake kategorije). Same podatke za vrednovanje sustava čini 3600 dokumenata ravnomjerno raspoređenih između tri određene kategorije. Vektore koji predstavljaju te podatke provučemo kroz mrežu kako bi dobili 32 bitni prikaz tih podataka. Nakon toga se odabere jedan 32 bitni vektor te se među ostalim vektorima traže oni koji imaju najmanju Hammingovu udaljenost s obzirom na početni vektor. Kad se odabere jedan od takvih vektora, gleda se njegova pripadnost nekoj od kategorija te da li spada u istu kategoriju kao i početni vektor. Na temelju toga se izgrađuje graf preciznosti i odziva, zajedno s interpoliranom srednjom vrijednosti, koji je prikazan na slici 6.2 za samo jedan upit, dok je na slici 6.3 prikazan graf sa srednjom vrijednosti preciznosti s obzirom na sve validacijske dokumente.

Srednja vrijednost prosjeka preciznosti (MAP) za skup validacijskih dokumenata je 0.468.

Od ostalih rezultata, može se spomenuti prosječna Hammingova udaljenost između vektora koji pripadaju istoj kategoriji, kao i između vektora koji ne pripadaju istoj kategoriji, a koja je dana u tablici 6.1.

Možemo vidjeti kako je Hammingova udaljenost kod vektora koji pripadaju istoj kategoriji u prosjeku manja nego kod vektora iz drugih kategorija, što daje bolje re-

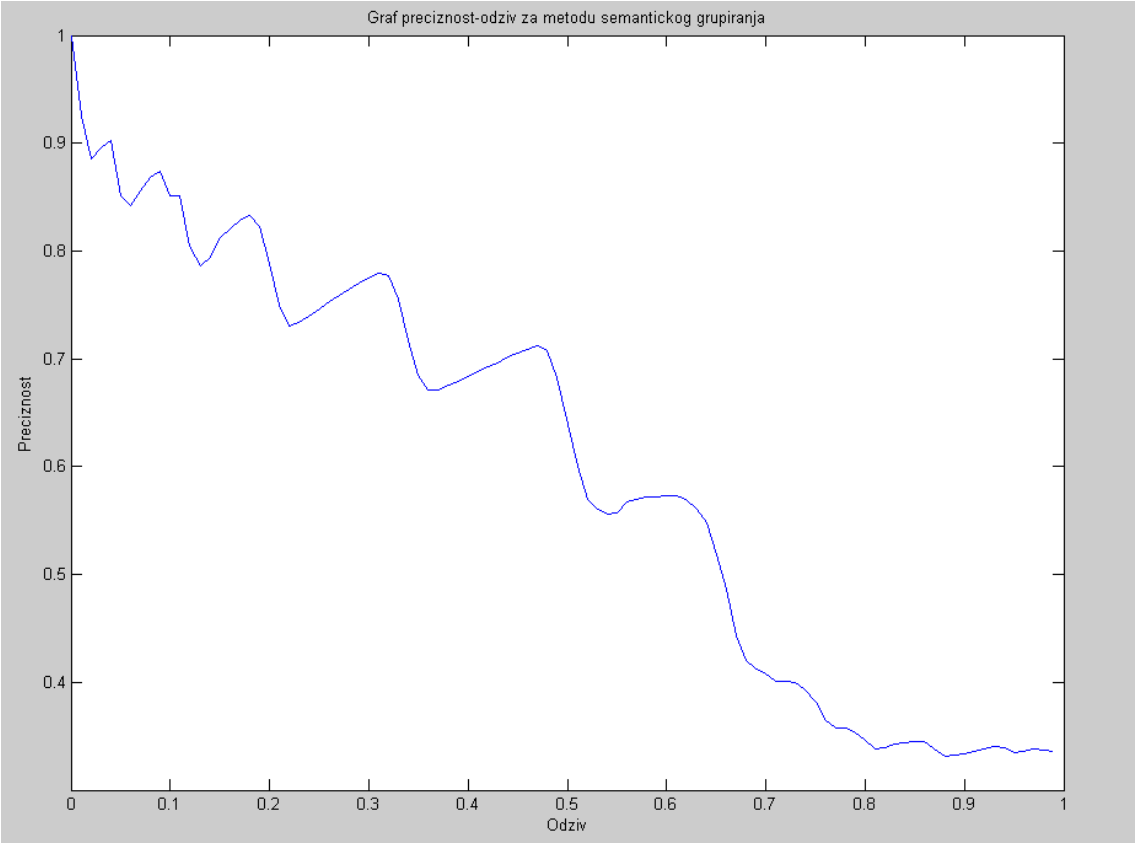


Slika 6.2: Graf preciznost-odziv za metodu semantičkog grupiranja – graf za jedan upit

zultate pri dohvaćanju sličnih dokumenata. Algoritam je uspješniji što je veća razlika između ta dva prosjeka za svaku kategoriju, što je i logično – ako su vektori iz iste kategorije bliži nego ostali vektori, lakše ih je prepoznati kao slične i vratiti kao rezultat na upit. Konkretno, implementirani algoritam daje rezultate koji su dobri za tri kategorije, ali bi učinkovitost bila upitna kada bi se broj kategorija povećao. Rješenje tog problema bi se moglo nalaziti u više mogućnosti. Prva mogućnost bi bila u smanjivanju vremenske složenosti algoritma finog podešavanja, kako bi se moglo izvesti više epoha učenja. Druga mogućnost za poboljšanje je istraživanje utjecaja Gaussovog šuma na konačan iznos težina mreže, a kao treća mogućnost je modeliranje ulaznog

Tablica 6.1: Hammingove udaljenosti među kategorijama

Kategorija	Udaljenost unutar kategorije	Udaljenost van kategorije
Crna kronika	11.50	14.52
Gospodarstvo	8.49	15.95
Sport	10.12	12.83



Slika 6.3: Graf preciznost-odziv za metodu semantičkog grupiranja – graf sa srednjom vrijednosti preciznosti za sve validacijske dokumente

sloja podataka na prikladniji način.

7. Zaključak

Cilj ovog rada bio je pokazati učinkovitost učenja dubokih generativnih modela koji sadrže mnogo slojeva skrivenih varijabli i parametara. Takvi modeli su u stanju izvući iz ulaznih podataka svojstva višeg reda, na razini neintuitivnoj čovjeku te na temelju njih vršiti zadaće klasifikacije, dohvaćanja informacija i sl.

U radu se čitatelja upoznaje s teorijom dubokih generativnih modela, kao i gradivnim elementom takvih modela – ograničenim Boltzmannovim strojem. Predstavljeni su algoritmi za učenje ograničenog Boltzmannovog stroja i cijele duboke mreže, kao i primjena dubokih mreža za pretraživanje dokumenata, točnije metoda semantičkog grupiranja.

U radu je pokazano kako uspješnost dubokih generativnih modela počiva na tri ključne značajke: svaki od slojeva mreže se može naučiti posebno, na pohlepan način. Drugo, taj pohlepan način inicijalnog učenja je moguće provesti nenadziranim učenjem. Treće, moguće je izvesti algoritam finog podešavanja nad mrežom kako bi se mreža još bolje prilagodila ulaznim podacima i pronašla još bolji prostor parametara.

Vrednovanjem sustava pokazalo se da sustav uspješno obavlja zadatak na malom broju kategorija i vektora. Za veći broj primjena je upitna. Stoga bi daljni rad na sustavu uključivao poboljšanje samog pohlepnog algoritma ili algoritma finog podešavanja u smislu brzine i potrošnje memorije računala, čime bi bilo moguće koristiti mnogo veće skupove za učenje. Drugo poboljšanje je intuitivno povezano s binarizacijom reduciranog vektorskog prikaza dokumenata, a to je spremanje svakog dokumenta na odgovarajuću memorijsku lokaciju računala. Tim postupkom bi pristupanje dokumentima bilo još brže, jer odmah imamo informaciju o adresi na kojoj se slični dokumenti nalaze. Treće poboljšanje je provjeriti rad sustava kao metode koja filtrira dokumente za neku već razrađenu metodu (TfIdf). Naime, postoje naznake da algoritam semantičkog grupiranja uspješno eliminira one dokumente koje metoda TfIdf označava kao slične dokumentu upita, a ustvari nisu (engl. *false positive, FP*). Stoga bi se semantičkim grupiranjem mogao stvoriti skup dokumenata nad kojim bi TfIdf postizao bolje rezultate.

LITERATURA

D.M. Blei and M.I. Jordan. Modeling annotated data. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 127–134. ACM, 2003.

G. Casella and E.I. George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

D. DeMers and G. Cottrell. Non-linear dimensionality reduction. *Advances in neural information processing systems*, pages 580–580, 1993.

A.P. Field. *Discovering statistics using SPSS*. SAGE publications Ltd, 2009.

P.V. Gehler, A.D. Holub, and M. Welling. The rate adapting poisson model for information retrieval and object recognition. In *Proceedings of the 23rd international conference on Machine learning*, pages 337–344. ACM, 2006.

R. Hecht-Nielsen. Replicator neural networks for universal optimal source coding. *Science*, 269(5232):1860, 1995.

G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504, 2006.

G.E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

K. Murphy. A brief introduction to graphical models and bayesian networks. 1998.

R. Salakhutdinov. *Learning deep generative models*. PhD thesis, Citeseer, 2009.

M. Welling, M. Rosen-Zvi, and G. Hinton. Exponential family harmoniums with an application to information retrieval. *Advances in neural information processing systems*, 17:1481–1488, 2005.

E. Xing, R. Yan, and A.G. Hauptmann. Mining associated text and images with dual-wing harmoniums. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI-2005)*. Citeseer, 2005.

Duboki generativni modeli za semantičko grupiranje tekstovnih dokumenata

Sažetak

Generativni modeli s latentnim varijablama statistički su modeli podataka koji podatke opisuju temeljem njihovih skrivenih odnosno latentnih svojstava. Takvim opisom je omogućeno kvalitetnije modeliranje podataka, odnosno u njima sadržane duboke strukture. Predstavljen je teoretski uvod u problematiku dubokih generativnih modela te algoritmi učenja takvih modela. Implementirana je metoda za semantičko grupiranje dokumenata na hrvatskom jeziku. Rezultati pokazuju da metoda pridonosi kvaliteti pretraživanja dokumenata.

Ključne riječi: Duboki generativni modeli, ograničen Boltzmannov stroj, Gibbsovo uzorkovanje, autoenkoder, semantičko grupiranje.

Deep Generative Models for Semantic Document Clustering

Abstract

Generative models with latent variables are statistical data models which model the data based on their hidden attributes. With data modeling like that, we can model the input data and it's deep structure more precisely. Theoretical introduction to generative models and algorithms for learning these models are given in the paper.Also, method for semantic hashing od text documents on Croatian language is implemented. Results show that method is suitable for document retrieval.

Keywords: Deep generative models, restricted Boltzmann machine, Gibbs sampling, autoencoder, semantic hashing.