Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Primož Belej

Oblikoskladenjsko označevanje slovenskega jezika z globokimi nevronskimi mrežami

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: prof. dr. Marko Robnik Šikonja
Somentor: dr. Simon Krek

Ljubljana, 2018

[1]V dogovorju z mentorjem lahko kandidat magistrsko delo s pripadajočo izvorno kodo izda tudi pod drugo licenco, ki ponuja določen del pravic vsem: npr. Creative Commons, GNU GPL. V tem primeru na to mesto vstavite opis licence, na primer tekst [**?**].

Zahvala

Na tem mestu zapišite, komu se zahvaljujete za izdelavo magistrske naloge. V zahvali se poleg mentorja spodobi omeniti vse, ki so s svojo pomočjo prispevali k nastanku vašega izdelka.

Primož Belej, 2018

Vsem rožicam tega sveta.

"The only reason for time is so that everything doesn't happen at once."

— Albert Einstein

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
CA	classification accuracy	klasifikacijska točnost
DBMS	database management system	sistem za upravljanje podatkovnih baz
SVM	support vector machine	metoda podpornih vektorjev
...	...	...

Povzetek

Naslov: Oblikoskladenjsko označevanje slovenskega jezika z globokimi nevronskimi mrežami

V magistrskem delu se ukvarjamo z oblikoskladenjskim označevanjem slovenskega jezika. Pri tej nalogi s področja obdelave naravnega jezika povedim priredimo ustrezno zaporedje oznak, ki opisujejo oblikoskladenjske lastnosti besed. Za razliko od tipičnih pristopov, ki vhodne povedi obravnavajo na nivoju besed, naša rešitev obravnava vhodne povedi kot zaporedja znakov. Nalogo označevanja rešujemo s kombinacijo konvolucijskih in rekurentnih nevronskih mrež. Posebnost našega pristopa je tudi v sami naravi označevanja, saj je ne obravnavamo kot problem večrazredne klasifikacije, temveč kot problem dodeljevanja oznak. Z namenom izboljšave rezultatov našo rešitev združimo v ansambel treh označevalnikov skupaj z dvema obstoječima označevalnikoma za slovenski jezik. Ob primerjavi naše rešitve z obstoječimi ugotovimo, da predlagana rešitev dosega najboljše rezultate pri reševanju zadanega problema.

Ključne besede

oblikoskladenjsko označevanje, globoko učenje, konvolucijske nevronske mreže

Abstract

Title: Part of speech tagging of slovene language using deep neural networks

This sample document presents an approach to typesetting your BSc thesis using LaTeX. A proper abstract should contain around 100 words which makes this one way too short. A good abstract contains: (1) a short description of the tackled problem, (2) a short description of your approach to solving the problem, and (3) (the most successful) result or contribution in your thesis.

Keywords

part-of-speech tagging, deep learning, convolutional neural network

Poglavje 1

Uvod

Oblikoskladenjsko označevanje je naloga s področja obdelave naravnega jezika. Cilj te naloge je za vhodne povedi v naravnem jeziku poiskati ustrezno zaporedje oblikoskladenjskih oznak. Oblikoskladenjske oznake vsebujejo informacijo o besedni vrsti in dodatnih morfoloških lastnostih besed.

Ta naloga je pomemben del predpriprave besedil na nadaljnje naloge s področja strojne obdelave naravnega jezika. S pomočjo oznak se lahko računalniški algoritmi naučijo natančneje prevajati besedila, povzemati njihovo vsebino in opravljati druge naloge, pri katerih je koristno poznavanje morfologije besed.

Računskega oblikoskladenjskega označevanja se lotevamo predvsem z algoritmi strojnega učenja. Za označevanje morfološko bogatih jezikov, med katere prištevamo tudi slovenščino, so se za učinkovite izkazali označevalniki temelječi na globokih nevronskih mrežah. Na globokih nevronskih mrežah temelji tudi naš označevalnik, ki ga bomo predstavili v nadaljevanju.

V poglavju REF predstavimo oblikoskladenjsko označevanje in algoritmične pristope k oblikoskladenjskem označevanju. Opišemo probleme, ki se pojavljajo pri tej nalogi, načine reševanja teh problemov in posebnosti pri označevanju besedil v slovenskem jeziku. V tem poglavju naredimo tudi pregled obstoječih označevalnikov za slovenski jezik.

Uporabo nevronskih mrež pri obdelavi besedil opišemo v poglavju REF.

Predstavimo arhitekture nevronskih mrež, ki se uporabljajo pri obdelavi besedil. Opišemo tudi prispevke s tega področja, ki so vplivala na zasnovo naše rešitve.

Sledi poglavje REF z opisom naše rešitve. Na tem mestu opišemo podatke s katerimi delamo, pripravo podatkov na modeliranje in zgradbo našega modela. Pri zgradbi modela opišemo arhitekturo posameznih nevronskih mrež, ki sestavljajo našo rešitev.

Naš nevronski označevalnik ovrednotimo v poglavju REF. V tem poglavju opišemo postopek ocenjevanja napovedi našega označevalnika in naredimo primerjavo našega označevalnika s sorodnimi.

V sklepnem poglavju izpostavimo glavne prispevke našega dela in komentiramo rezultate primerjave označevalnikov. Navedemo tudi razmisleke o možnih izboljšavah naše rešitve.

Poglavje 2

Oblikoskladenjsko označevanje

V uvodu smo povedali, da pri oblikoskladenjskem označevanju pripisujemo besedam oblikoskladenjske oznake in da te oznake vsebujejo informacijo o besedni vrsti in pripadajočih morfoloških in oblikovnih lastnostih.

Besedilo je po naravi nestrukturirana oblika podatkov, zaradi česar je iz njega težko računsko izpeljevati sklepe o pomenu. Oblikoskladenjske oznake vnesejo v besedilo strukturo in tako olajšajo odkrivanje pomena in nadaljnjo delo, ki zahteva semantično razumevanje.

Oblikoskladenjske oznake se uporabljajo kot dodatne značilke pri algoritmičnem reševanju nalog kot sta (REF jurafsky ch8):

Določanje imenskih entitet Poznavanje oblikoskladenjskih oznak olajša določanje entitet (osebe, organizacije, ...). Imenske entitete se uporabljajo pri luščenju informacij iz besedila.

Prepoznava govora in tvorjenje govora Za pravilno naglaševanje besed in poudarke pri izgovorjavi povedi.

V tabeli 2.1 vidimo primer označene povedi skupaj z razlago posameznih oznak. Primer je vzet iz jezikovnega korpusa ssj500k([1]), v katerem so uporabljene oblikoskladenjske oznake iz nabora MULTEXT-East.

Tabela 2.1: Primer označene povedi: "To bi bila pravična vojna."

Beseda	Oznaka	Pomen oznake
To	Zk-sei	Zaimek, kazalni, srednji spol, ednina, imenovalnik
bi	Gp-g	Glagol, pomožni, pogojnik
bila	Gp-d-ez	Glagol, pomožni, deležnik, ednina, ženski spol
pravična	Ppnzei	Pridevnik, splošni, nedoločena stopnja, ženski spol, ednina, imenovalnik
vojna	Sozei	Samostalnik, občno ime, ženski spol, ednina, imenovalnik
.	Z	Ločilo

2.1 Algoritmične rešitve

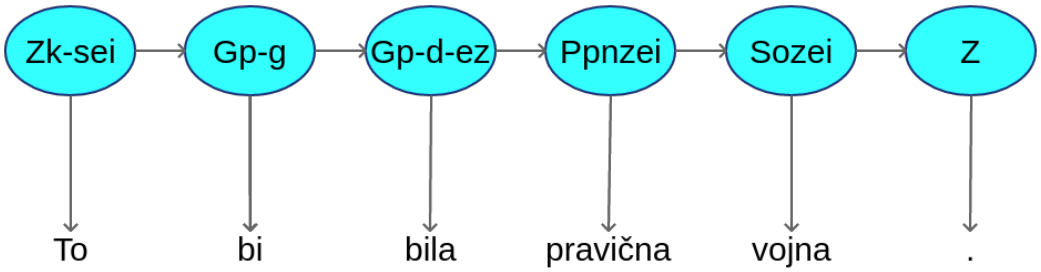
Za določanja oblikoskladenjskih oznak ne zadostujejo enolične preslikave iz besed v oznake, saj za en zapis besede običajno velja mnogo različnih oznak. Za doseganje visoke točnosti oznak, se uporablja predvsem algoritme strojnega učenja.

V tem podpoglavju bomo predstavili pogoste pristope k označevanju z uporabo strojnega učenja. Pristop z nevronskimi mrežami, ki smo ga uporabili v tem diplomskem delu, bomo podrobneje predstavili v naslednjih dveh poglavjih (3, 4).

Skriti markovski modeli

Skriti markovski modeli (Hidden Markov models, HMM) so sekvenčni modeli. Sekvenčni model vsakemu elementu zaporedja dodeli oznako ali razred. Sekvenčni model torej dela preslikave iz zaporedja opažanj v zaporedje oznak, kar tudi ustreza zahtevam oblikoskladenjskega označevanja: poved je zaporedje opažanj, oznake besed v povedi pa so zaporedje razredov.

HMM deluje tako, da izračuna verjetnostno porazdelitev za vsa možna zaporedja oznak pri nekem zaporedju opažanj in izbere najverjetnejše zaporedje. Gre za nadgradnjo Markovskih verig, modelov s katerimi opisujemo



Slika 2.1: Označevalnik implementiran s skritim markovskim modelom. Ker gre za osnovni model prvega reda je oznaka odvisna le od predhodne oznake.

verjetnosti zaporedja slučajnih spremenljivk, ki imajo za zalogo vrednosti množico stanj. Markovske verige niso primerne za označevanje, ker opisujejo samo vidna stanja, ne pa tudi skritih stanj. V primeru oblikoskladenjskega označevanja so vidna stanja besede, skrita stanja pa oznake. Označevanje s skritim markovskim modelom je prikazano na skici (2.1).

Skriti markovski modeli prvega reda imajo dve predpostavki:

Markovska predpostavka Neko skrito stanje je odvisno le od prvega predhodnega skritega stanja, ne pa tudi od drugih predhodnih skritih stanj v zaporedju.

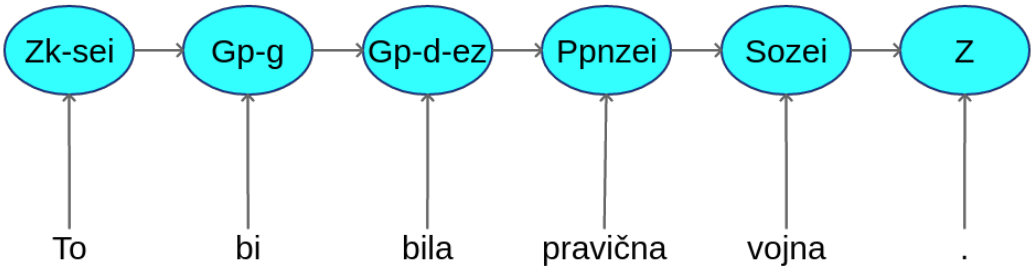
$$P(q_i|q_1...q_{i-1}) = P(q_i|q_{i-1})$$

Neodvisnost opažanj Neko opažanje je odvisno le od skritega stanja, ki je povzročilo to opažanje, ne pa tudi od drugih opažanj ali skritih stanj.

$$P(o_i|q_1...q_i, ..., q_T, o_1, ..., o_i, ..., o_T) = P(o_i|q_i)$$

Postopek prirejanja zaporedja oznak zaporedju opažanj imenujemo dekodiranje. Pri postopku dekodiranja iščemo najverjetnejše zaporedje skritih stanj. Za dekodiranje uporabljamo Viterbijev algoritem, ki s pomočjo dinamičnega programiranja zgradi matriko verjetnosti, s stolpcem za vsako opaženo stanje in vrstico za vsako možno oznako.

Označevalniki osnovani na skritih markovskih modelih v praksi uporabljajo izboljšave osnovnega markovskega modela prvega reda. Prva takšna



Slika 2.2: Označevalnik implementiran z modelom maksimalne entropije.

izboljšava je razširitev kontekstnega okna. Pri računanju verjetnosti za neko oznako pri prejšnjih dveh oznakah dobimo trigramski model. Takšna razširitev navadno prinese izboljšave k točnosti označevanja, a se hkrati poveča tudi kompleksnost dekodiranja.

Bigramski model $P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1})$

Trigramski model $P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1}, t_{i-2})$

Pogosta težava se pojavi pri neznanih besedah. Neznane besede so takšne, ki se niso pojavile v učni množici in jih algoritem še ni videl. To težavo rešujemo z obravnavanjem delov besed kot so pogoste končnice, začetki besed in velike začetnice.

S takšnimi izboljšavami so lahko skriti markovski modeli primerno orodje za označevanje Brants(2000) REF poroča o 96.7% točnosti označevanja, kar je primerljivo z modeli maksimalne entropije in nevronskimi označevalniki.

Modeli maksimalne entropije

Pri markovskih modelih maksimalne entropije (maximum entropy Markov models, MEMM) gre za model logistične regresije uporabljen na poseben način. Logistična regresija je klasifikacijski algoritem, ki pa ni sekvenčen in je tako sam po sebi neprimeren za označevanje zaporedij. Če uporabimo logistično regresijo na zaporednih besedah v povedi in uporabimo poleg ostalih vhodnih parametrov še napoved prejšnjega modela, dobimo model MEMM.

Z uporabo takšnih modelov, lahko preprosteje vključujemo poljubne značilke, kar je pri generativnih modelih, kot je HMM, težje. Pri implementaciji označevalnikov, z uporabo modelov maksimalne entropije vključujemo večje število značilk. Te značilke so običajno sosednje besede in oznake v oknih različnih širin, predpona in končnica besede, prisotnost velikih začetnic, številk, simbolov in podobno.

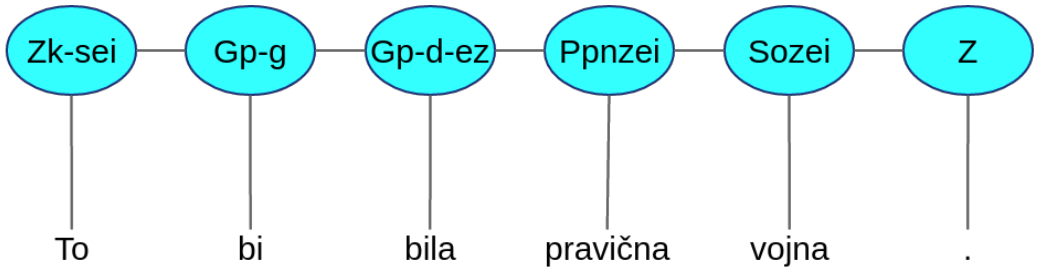
Najpreprostejši način dekodiranja modelov maksimalne entropije je z uporabo požrešnega dekodirnega algoritma. Ta pristop deluje tako, da s klasifikatorjem napovemo oznake vseh besed iz ene smeri v drugo in pri tem napovedi oznak klasifikatorja posredujemo naslednjim stopnjam. Takšen pristop je hiter, a ponavadi njegova točnost ni zadostna, saj so napovedi oznak na vsakem koraku dokončne in jih klasifikator v kasnejših korakih ne more popraviti. Ustreznejši postopek dekodiranja je z uporabo Viterbijevega algoritma, kot pri skritih markovskih modelih. Na skici modela (2.2) opazimo, da je oznaka odvisna od predhodne oznake in trenutne besede.

Dvosmerno označevanje

Pomanjkljivost modelov maksimalne entropije in skritih markovskih modelov pri označevanju besedil je v tem, da označuje le v smeri od začetka do konca povedi. To je težava, ker so besede in njihove oznake med sabo odvisne v obeh smereh. V izogib negativnim posledicam označevanja v eno smer se uporabljajo različni pristopi:

- Vsak sekvenčni model lahko spremenimo v dvosmerni model z uporabo večkratnih prehodov. Ob prvem prehodu model uporablja samo oznake že označenih besed levo od trenutno označevane besede, v drugem prehodu pa lahko uporablja tudi oznake besed, ki se nahajajo desno od trenutne in so bile označene v prejšnjem prehodu.

- Pristop podoben večkratnim prehodom sta dva prehoda, vsak v svoji smeri. Ker imamo tu opravka z verjetnostnimi modeli lahko izberemo oznake, ki jima model pripiše višje verjetnosti.



Slika 2.3: Označevalnik implementiran s pogojno verjetnostnim poljem.

- Označevalnik Stanford Tagger CITE (Toutanova et al., 2003). uporablja dvosmerno različico modela maksimalne entropije, ki se imenuje ciklično odvisnostno omrežje (cyclic dependency network

Pogojno verjetnostna polja

Modeli pogojno verjetnostnih polj (Conditional random fields, CRF) so neusmerjeni verjetnostni grafični modeli in že v svoji osnovi odpravljajo težavo enosmernega označevanja REF. Ti modeli dosegajo najboljše rezultate pri različnih nalogah označevanja zaporedij.

Modeli CRF normalizirajo verjetnosti za celotno zaporedje oznak in ne lokalno, kot modeli MEMM. Prednost tega je, da modeli CRF niso pristranski k nekaterim oznakam. Modeli CRF so kompleksnejši od modelov HMM in MEMM, zato njihovo učenje traja dlje časa.

2.2 Oblikoskladenjsko označevanje slovenščine

Težavnost opisanega problema se med različnimi naravnimi jeziki razlikuje. Na težavnost pomembno vpliva število možnih oznak v jeziku. Teh v slovenskem jeziku poznamo 1900 (REF MULTEXT-East V5), medtem ko nabori oznak za angleški jezik navadno vsebujejo med 45 in 100 različnih oznak (REF). Razlog za veliko število možnih oznak v slovenskem jeziku je veliko število različnih oblik v katerih zapisujemo besede. Dve besedi z isto osnovno

obliko, z eno različno lastnostjo imata običajno v slovenščini različna zapisa.

Jezikom, pri katerih se pojavljajo omenjene lastnosti, pravimo morfološko bogati jeziki (REF jurafsky). Med te jezike, poleg slovenščine, uvrščamo jezike kot so turščina, madžarščina in češčina.

Oblikoskladenjske oznake v morfološko bogatih jezikih nosijo več informacij kot oznake v jezikih kakršen je angleški. V morfološko bogatih jezikih poznamo koncept sklona, časa, števila, osebe in drugih lastnosti, ki so značilne samo za določene besedne vrste. Oblikoskladenjske oznake v takšnih jezikih so zaporedja morfoloških oznak in ne enotne, primitivne oznake.

V okviru pričujočega magistrskega dela smo uporabljali oblikoskladenjske oznake, določene v specifikaciji MULTEXT-East V5. Ta specifikacija uporablja trinajst besednih vrst: samostalnik, glagol, pridevnik, prislov, zaimek, številnik, predlog, veznik, členek, medmet, okrajšava, neuvrščeno in ločilo.

V tabeli 2.2 so za slovenščino po MULTEXT-East V5 navedene besedne vrste, število lastnosti za posamezno besedno vrstno in število vseh veljavnih kombinacij lastnosti.

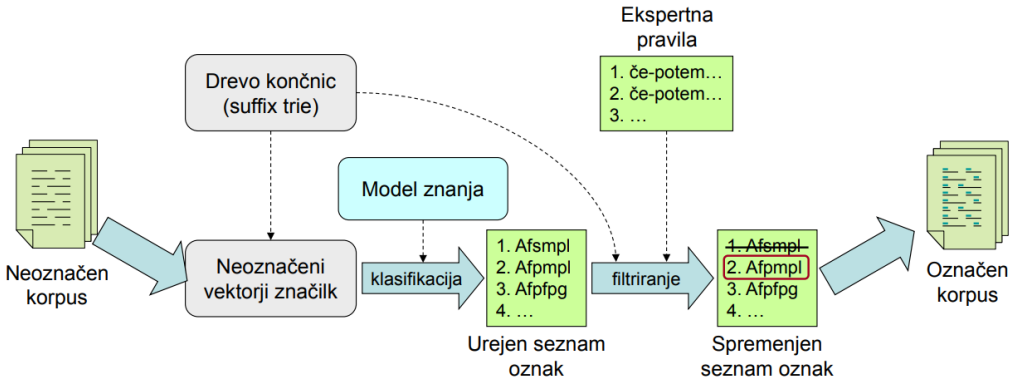
Lastnost slovenščine, ki dodatno vpliva na težavnost označevanja je prost besedni red. Zaradi te lastnosti se lahko besede pojavljajo na različnih mestih v povedi, ne da bi spremenile pomen povedi. Povedi v spodnjem primeru so sestavljene iz istih besed, spremenili smo le njihov vrstni red.

- Vesna na vrtu nabira rože.
- Na vrtu Vesna nabira rože.
- Rože na vrtu nabira Vesna.

Vse povedi so sintaktično pravilne in nosijo isti pomen. Razlika je le v poudarku. Prost vrstni red besed uvede dodatno težavnost za jezikovne modele, saj je zaradi te lastnosti težje prepoznati vzorce, ki se pojavljajo v jeziku.

Tabela 2.2: Besedne vrste po specifikaciji MULTEXT-East V5.

Besedna vrsta	Število lastnosti	Število možnih kombinacij
Samostalnik	5	104
Glagol	7	156
Pridevnik	6	279
Prislov	2	4
Zaimek	8	1122
Števnik	6	215
Predlog	1	6
Veznik	1	2
Členek	0	1
Medmet	0	1
Okrajšava	0	1
Neuvrščeno	1	8
Ločilo	0	1



Slika 2.4: Shema delovanja orodja Obeliks in njegovih komponent.

2.3 Obstoječe rešitve

Najuspešnejši obstoječi izvedbi označevalnika za slovenski jezik sta označevalnika Obeliks in Reldi. Temeljita na algoritmih strojnega učenja, ki smo jih opisali v podpoglavju 2.1.

Grčar, Krek, Dobrovoljc (2012)[2]

V okviru projekta Sporazumevanje v slovenskem jeziku CITE je skupina raziskovalcev razvila orodje imenovano Obeliks. Obeliks ni samo oblikoskladenjski označevalnik. Vključuje tudi orodja za segmentacijo, tokenizacijo in lematizacijo besedila. Segmentacija poskrbi za razdelitev vhodnega besedila na enote za označevanje, ki so običajno povedi. Tokenizacija poišče pojavnice. Pojavnice so največkrat ločila in besede. Lematizacija določa besedam njihovo osnovno obliko.

Za označevanje besed so uporabili model maksimalne entropije (2.1. Model so naučili na jezikovnem korpusu ssj500k [3]. Poleg tipičnih značilk, kot so sosednje besede in oznake prejšnjih besed, so uporabili še potencialne oznake iz drevesa končnic, zgrajenega na podlagi učnega korpusa. Pri napovedovanju besedne vrste je označevalnik dosegel 98,3 % točnost, celotne oznake pa je določal z 91,34 % točnostjo.

Ljubešić, Erjavec (2016)[4]

Avtorji tega dela so razvili nov označevalnik, osnovan na modelu pogojno naključnih polj (2.1). Njihov označevalnik se imenuje Reldi. Za osnovo so uporabili označevalnik Obeliks. Tako kot označevalnik Obeliks, tudi Reldi za dodatne značilke uporablja drevo končnic. Svoj označevalnik so posebej optimizirali za označevanje neznanih in delno neznanih besed. Označevanje neznanih besed je del arhitekture označevalnika in tega ne rešuje dodaten proces, kot je bilo običajno pri označevalnikih za morfološko bogate jezike.

Poročajo o 98,94% točnosti pri napovedovanju besedne vrste, za celotne oznake pa 94,27%.

Poglavje 3

Globoke nevronske mreže za besedila

V prejšnjem poglavju smo predstavili nekatere algoritmične pristope k oblikoskladenjskem označevanju. Za reševanje te naloge, pa tudi drugih s področja obdelave naravnega jezika, postajajo v zadnjih letih vedno pogostejši pristopi z uporabo nevronskih mrež.

V tem poglavju bomo predstavili vrste nevronskih mrež, ki se pogosto uporabljajo na področju obdelave naravnega jezika. Te vrste smo uporabili tudi mi pri implementaciji naše rešitve. Na koncu poglavja predstavimo tudi prispevke s tega področja, ki so vplivali na našo rešitev.

3.1 Konvolucijske nevronske mreže

Konvolucijske nevronske mreže so posebna vrsta nevronskih mrež, namenjene obdelavi podatkov razporejenih v vnaprej znano strukturo. Slike, kjer se konvolucijske nevronske mreže pogosto uporabljajo, imajo slikovne točke razporejene v dvodimenzionalno strukturo. Besedila navadno opisujemo z enodimenzionalo strukturo, saj imamo opravka z enodimenzionalnimi zaporedji besed ali znakov. Konvolucijske nevronske mreže se imenujejo po matematični operaciji konvoluciji, ki jo uporabljajo vsaj na enem nivoju, namesto

običajnega matričnega množenja.

Konvolucija

Konvolucijo opisuje spodnja enačba (3.1).V kontekstu, ko je t časovna komponenta, lahko konvolucijo opišemo kot uteženo povprečje funkcije x, v trenutku t, utež pa dobimo iz funkcije $w(-a)$, z zamikom t. S spremembo parametra t se spreminja tudi vrednost uteži $w(t-a)$.

$$s(t) = \int x(a)w(t-a)da \tag{3.1}$$

Čeprav smo vhodni parameter poimenovali t, ni potrebno, da le ta opisuje čas. Čas bi bil primeren parameter pri modeliranju časovnih vrst, pri besedilih pa je to položaj v besedilu.

Operacijo konvolucije običajno zapisujemo s simbolom $*$ in tako lahko zgornjo enačbo zapišemo kot:

$$s(t) = (x * w)(t) \tag{3.2}$$

Zgornja definicija konvolucije predvideva, da je časovna spremenljivka t zvezna, kar pa ne drži, kadar imamo opravka s podatki v nevronskih mrežah, saj podatki prihajajo v diskretnih enotah, kot so slikovne točke, besede, ali črke. Tudi pri obdelavi časovnih vrst imamo v resnici opravka z diskretnimi podatki. Za modeliranje z nevronskimi mrežami je ustreznejša definicija, ki obravnava spremenljivko t kot diskretno:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{3.3}$$

V kontekstu nevronskih mrež, pravimo argumentu $x(a)$ vhod, drugemu argumentu $w(t-a)$ pa jedro. V praksi so podatki na vhodu običajno v obliki večdimenzionalnih polj, ki jih imenujemo tenzorji. Jedro je prav tako tenzor parametrov, ki jih nastavi algoritem strojnega učenja.

Kadar se konvolucija uporablja na dvo ali večdimenzionalnih podatkih, uporabimo razširjeno definicijo konvolucije, ki deluje na več oseh. Naslednja enačba (3.4) definira konvolucijo na dveh oseh. Vhod I in jedro K sta dvodimenzionalni polji.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{3.4}$$

Lastnosti konvolucijskih nevronskih mrež

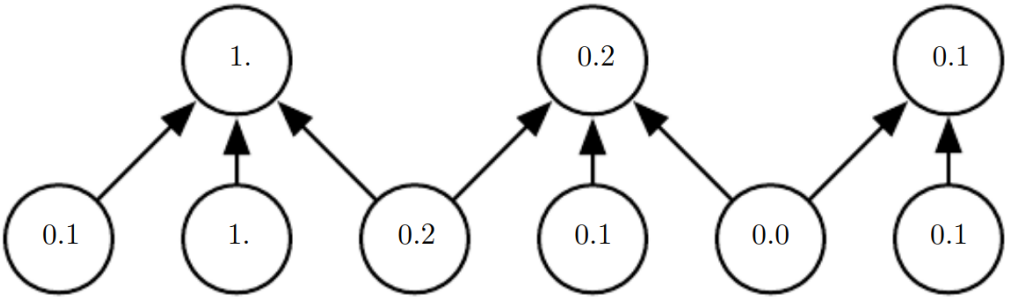
Prehodi med sloji v nevronskih mrežah so običajno predstavljeni z matričnim množenjem. Elementi matrike so uteži povezav med posameznimi vhodnimi in izhodnimi enotami. V takšnih gosto povezanih nevronskih mrežah je vsak izhod povezan z vsakim izhodom. Konvolucijske nevronske mreže se razlikujejo od običajnega pristopa v tem, da imajo redkejše povezave. Ta lastnost izhaja iz tega, da je jedro konvolucije največkrat manjše od vhoda. Takšen pristop ima pri določenih nalogah številne prednosti pred klasičnim:

- Manjša prostorska zahtevnost, ki izvira iz tega, da ni potrebno shranjevati uteži povezav med vsemi vhodi in izhodi.
- Manjša računska zahtevnost učenja. Za izračun vrednosti na izhodih je potrebnih manj računskih operacij.
- Več različnih funkcij modela si lahko deli isti parameter. Pri običajnih nevronskih mrežah se vsak parameter iz matrike uteži uporabi natanko enkrat, pri konvolucijskih nevronskih mrežah pa se vsak element jedra uporabi na vsakem elementu vhoda.

Združevanje

Sloj v konvolucijski nevronski mreži je običajno sestavljen iz treh nivojev:

1. V prvem nivoju se vzporedno izvede več konvolucij, da dobimo množico linearnih aktivacij.



Slika 3.1: Primer združevanja s korakom $k = 2$ in širino okna 3. [5]

2. Na linearnih aktivacijah se izvede nelinearna aktivacijska funkcija. Ta nivo se imenuje nivo detektorjev.

3. Na zadnjem nivoju uporabimo združavalno funkcijo.

Združevalna funkcija transformira izhode mreže na nekem mestu z opisno statistiko izhodov. Združevanje z maksimumom tako zamenja izhod detektorjev z najvišjo vrednostjo znotraj nekega območja. Druge pogoste funkcije združevanja so še povprečje izhodov, norma L^2 in uteženo povprečje okrog središčnega izhoda. Z združevanjem dosežemo zmanjševanje kompleksnosti modela in tako olajšamo učenje.

Ker funkcija združevanja povzema celotno soseščino izhodov, je včasih smiselno uporabiti manj združevalnih enot, kot je enot v nivoju detektorjev. Pri takšnem pristopu združevalne enote povzemajo območja ki so si narazen več kot eno mest. V primeru na sliki (**??**) je uporabljen korak $k = 2$.

Slabost, ki se pojavi pri uporabi večkratnega združevanja je, da se izgubi informacija o mestu pojavitve prepoznane lastnosti. Prav tako se lahko izgubi informacija o odvisnostih med posameznimi lastnostmi.

3.2 Rekurentne nevronske mreže

Rekurentne nevronske mreže (RNN) so družina nevronskih mrež, ki se uporablja za obdelavo podatkov v zaporedjih. Prednost rekurentnih nevronskih mrež, pomembna za obdelavo zaporedij, je v tem da lahko hrani informacijo

dlje časa. Zaradi te lastnosti lahko obdeluje daljša zaporedje, kot bi bilo praktično pri običajnih nevronskih mrežah. Večina rekurentnih nevronskih mrež lahko obdeluje podatke v zaporedjih različnih dolžin.

V primeru, da bi se obdelave zaporedij lotili z običajnimi nevronskimi mrežami, bi uporabili pristop z drsečim kontekstnim oknom določene širine. Slabost takšnega pristopa je omejen kontekst za zajem informacij, saj pojavnice zunaj kontekstnega okna na odločitve modela nimajo vpliva. Ta slabost je še izrazitejša pri jezikih, kjer so med seboj odvisne besede na različnih koncih povedi. Druga slabost je, da se model težje nauči vzorcev kot so besedne zveze.

Rekurentna nevronska mreža je vsaka nevronska mreža, ki vsebuje usmerjen cikel. V primeru ciklične povezave je enota posredno ali neposredno povezana s svojim izhodom na svoj vhod. Vrednost aktivacije rekurentne enote pri nekem koraku ni odvisna samo od vhoda, ampak tudi od vrednosti aktivacije v prejšnjem koraku. Rekurentne povezave omogočajo modeliranje časovne komponente zaporedij.

Obdelava niza podatkov z mrežami RNN poteka tako, da se elemente niza po vrsti vstavlja v mrežo. Rekurentna povezava daje način pomnjenja z uvajanjem informacije iz prejšnjega koraka. Takšna arhitektura ni omejena samo na prejšnji korak, saj vsi predhodni koraki vplivajo na trenutno stanje.

Glavna razlika med navadnimi nevronskimi mrežami in mrežami RNN je v dodatnih povezavah med skritim slojem v trenutnem korakom in skritim slojem prejšnjega koraka. Te povezave imajo tudi svoje uteži. Kot pri običajnih utežeh, se tudi uteži povezav med skritimi sloji učijo z vzvratnim razširjanjem napak.

Pojemajoči gradient

Pojemajoči gradient je težava, ki se pojavlja pri učenju nevronskih mrež z gradientnimi metodami in vzvratnim razširjanjem napak. Pri gradientnih metodah dobijo uteži na vsakem koraku učenja nove vrednosti, ki so sorazmerne parcialnemu odvodu funkcije napake v odvisnosti od stare vrednosti

uteži. Težava se pojavi, ko se gradient tako zmanjša, da postanejo spremembe vrednosti uteži neznatne. Uporaba aktivacijskih funkcij, pri katerih se pojavljajo odvodi z višjimi vrednostmi, lahko pripelje do obratnega problema, pri katerem gradient narašča prehitro.

Nevronske mreže LSTM

Tipično se pri rekurentnih nevronskih mrežah informacija zakodirana v skritih stanjih najbolj navezuje na zadnji viden del vhodnega niza. Iz tega razloga se pri nalogah, ki zahtevajo uporabo informacij oddaljenih od trenutnega mesta obdelave pojavljajo težave. Pri obdelavi naravnega jezika, se pogosto pojavi potreba po poznavanju daljšega konteksta. Primer, ki se navezuje na temo pričujočega dela so odvisnosti med besedami, ki se nahajajo na različnih koncih dolge povedi.

Eden izmed razlogov za nezmožnost prenosa pomembne informacije je v tem, da sloj v rekurentni nevronski mreži opravlja dve nalogi hkrati. Prva naloga je priprava informacije, ki je koristna v trenutnem kontekstu. Druga naloga sloja pa je posodobitev in prenos informacije naprej, da je uporabna v prihodnjem kontekstu.

Druga težava, ki jo imajo rekurentne nevronske mreže pri učenju, je problem pojemajočega gradienta. Ta problem postene z daljšimi vhodnimi nizi še bolj izrazit.

Z namenom omejevanja teh težav so bili razviti pristopi, ki omogočajo ohranjanje informacije v daljših časovno odvisnih nizih. Ti pristopi obravnavajo kontekst spominsko enoto, ki jo je potrebno upravljati. Spominsko enoto se pri teh pristopih navadno upravlja s pozabljanjem in pomnjenjem.

Nevronske mreže z dolgim kratkoročnim spominom (Long short-term memory, LSTM), razpolagajo s kontekstom z optimizacijo dveh nalog. Prva je pomnjenje informacij, za katere je verjetno, da bodo pomembne pri prihodnjih odločitvah. Druga naloga pa je pozabljanje informacij, ki niso več potrebne. Nadzorovanje konteksta ni zakodirano v nevronske mreže LSTM, ampak se tega naučijo same.

Nevronske mreže LSTM imajo poleg zunanje rekurentnosti, ki je značilna tudi za navadne mreže RNN, tudi notranjo rekurentnost. Na zunaj je enota mreže LSTM ista kot enota navadne mreže RNN, saj ima iste vhode in izhode. V svoji notranjosti ima dodatno kompleksnost, zaradi česar ima tudi več parametrov. Notranja enota stanja ima linearno povezavo na sebe, zaradi katere se pojavi spominski učinek. Utež omenjene rekurentne povezave nadzirajo vrata, ki jim pravimo vrata pozabe. Vrata pozabe nadzirajo količino informacij, ki se na vsakem koraku ohrani. Na vhodu so še vhodna vrata, ki nadzirajo količino informacij, ki v trenutnem koraku vstopijo v celico in preprečujejo pomnjenje nepomembnih informacij. Izhodna vrata, na izhodu celice skrbijo za nadzor količine informacij ki se prenesejo v naslednji korak.

3.3 Hitrocestne nevronske mreže

Družino nevronskih mrež, ki jo zaznamuje uporaba vrat za nadzor pretoka informacij skozi model in omogoča uporabo velikega števila slojev imenujem hitrocestne nevronske mreže [6].

Dejavnik, ki ključno vpliva na moč nevronske mreže je globina modela. Globina pomeni število slojev v nevronski mreži. Večja globina modelov je povezana z boljšo sposobnostjo aproksimacije določenih vrst funkcij.

Težava, ki se začne pojavljati z dodajnajem večjega števila slojev je oslabljeno razširjanje aktivacij in gradientov. Pristopi s katerimi se ta problem rešuje vključujejo boljše optimizacijske algoritme in nastavljanje začetnih uteži.

Drugačen pristop k povečanju števila slojev nevronske mreže brez omenjenih slabosti so hitrocestne nevronske mreže. Ta družina nevronskih mrež uporablja mehanizem vrat za nadzor pretoka informacij skozi enote. Takšen mehanizem omogoča tvorbo poti skozi katere lahko informacija potuje skozi več slojev brez slabljenja. Te poti se imenujejo hitre ceste.

Prednost takšnega pristopa je v tem, da ob velikemu številu slojev omogoča optimizacijo s stohastičnim gradientnim spustom (SGD), za razliko od nava-

dnih nevronskih mrež, kjer ob velikem številu slojev SGD deluje slabo.

Avtorji članka Highway Networks [6] so v svojih eksperimentih uporabili SGD na hitrocestnih nevronskih mrežah do globine 900 slojev. V primerjavi z običajnimi tesno povezanimi nevronskimi mrežami so pokazali, da se slednje tudi ob izboljšavi z normalizirano inicializacijo uteži opazno slabijo, medtem ko na hitrocestne mreže globina ne vpliva.

Pri običajni nevronski mreži lahko izhode nekega sloja opišemo z enačbo 3.5, v kateri je y vektor izhodov sloja, x vektor vhodov, H je nelinearna preslikava, W_H pa so parametri preslikave H.

$$y = H(x, W_H) \tag{3.5}$$

H je navadno afina preslikava, ki ji sledi nelinearna aktivacijska funkcija.

Sloji v hitrocestni nevronski mreži uvedejo še dve nelinearni preslikavi $T(x, W_T)$ in $C(x, W_C)$:

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot C(x, W_C) \tag{3.6}$$

Preslikavo T imenujemo vrata preslikave, C pa vrata prenosa. Poimenovanje izhaja iz dejstva, da ta vrata uravnavajo delež informacije, ki se nelinearno preslika in delež, ki je prenešen naprej.

3.4 Rešitve sorodnih problemov z uporabo nevronskih mrež

V preostanku poglavja predstavimo nekatere prispevke s področja modeliranja naravnega jezika z globokimi nevronskimi mrežami. Nekatere od rešitev namesto predstavitev besedila na nivoju besed uporabljajo predstavitve na nivoju znakov, ali pa združujejo oba pristopa. Glavni prednosti pristopa s predstavitvijo besedila na nivoju znakov sta:

Preprostejša priprava podatkov. Edini vhodni podatki v takšen nevronski model so povedi, ki jih model vidi kot zaporedja znakov. Ta podatek

dobimo neposredno iz jezikovnega korpusa in ne zahteva dodatne obdelave. Sorodne rešitve vpeljujejo dodatne značilke, ki jih tvorijo iz delov besed in besednih zvez ali pa jih vzamejo iz zunanjih virov, kot so leksikoni.

Možnost boljših predstavitev podatkov. Model predstavljen v [7] lahko razdelimo na dva glavna dela. Jezikovni model, implementiran z rekurentnimi nevronskimi mrežami, je del, ki se iz vzorcev uči zakonitosti jezika. Drugi del, implementiran s konvolucijskimi nevronskimi mrežami, se uči vhodne podatke pretvoriti v jezikovnemu modelu uporabne vektorje značilk. Zasnova predstavitvenega modela je takšna, da spodbuja prepoznavo vzorcev kot so pogoste končnice in predpone besed ter druga pogosta zaporedja znakov.

Collobert, Weston, Bottou, Karlen, Kavukcuoglu, Kuksa (2011)[8]

Avtorji predstavijo nevronsko mrežo z arhitekturo, ki je uporabna pri označevanju različnih zaporedij s področja obdelave naravnega jezika. Te naloge vključujejo oblikoskladenjsko označevanje, prepoznavo imenskih entitet in določanje semantičnih vlog.

To vsestransko uporabnost modela so dosegli s tem, da niso ročno tvorili značilk namenjenih točno določenim nalogam, ampak so pripravili model, ki se sam nauči iz besedila izluščiti uporabne lastnosti.

Model, ki so ga implementirali, ima na vhodu konvolucijski sloj, ki mu sledi združevalni sloj, nazadnje pa še več tesno povezanih slojev. Njihova nevronska mreža deluje na nivoju besed, ki vstopajo v model kot vektorske vložitve. Model so učili na korpusu, ki so ga zgradili iz celotne angleške Wikipedije in na korpusu Reuters RCV1 REF.

Svoj cilj, da bi razvili večnamenski model za različne vrste označevanja besedil so uresničili, saj so ocene njihovega označevalnika le rahlo zaostajale za ocenami najboljših, namenskih označevalikov.

Dos Santos, Zadrozny (2014)[9]

V tem članku je predstavljen model oblikoskladenjskega označevalnika z globokimi nevronskimi mrežami, ki združuje predstavitve na nivoju besed in na nivoju znakov. Globoka nevronska mreža, ki so jo zgradili ima na vhodu konvolucijski sloj, ki omogoča iskanje značilk iz besed poljubne dolžine. Tekom označevanja konvolucijski sloj gradi vektorske vložitve na nivoju znakov za vsako besedo, tudi za takšne, ki jih še ni videl v učni množici. Prednost pristopa k obdelavi besedila na nivoju znakov je v tem, da ni potrebno vnašati značilk s podatki o besedah in njihovih lastnosti.

Za demonstracijo učinkovitosti takšnega pristopa so razvili označevalnika za portugalski in angleški jezik. Oba označevalnika delujeta brez vpeljave dodatnih značilk. Angleški je dosegel 97,32% točnost, portugalski pa 97,42%.

V članku pokažejo tudi, da je za doseganje primerljivih rezultatov brez uporabe vložitev na nivoju znakov potrebno vpeljati dodatne, ročno pripravljene značilke. Kot slabost pristopa brez uvajanja dodatnih značilk navajajo dejstvo, da je pri takšnem modelu potrebno nastaviti več parametrov modela, kar pa po njihovem zahteva manj ročnega dela kot tvorjenje značilk.

Labeau, Allauzen (2016)[10]

Avtorji članka so uporabili različne arhitekture nevronskih mrež za implementacijo oblikoskladenskega označevalnika, ki deluje na nivoju znakov.

Deluje na dveh nivojih modeliranja, ki se učita hkrati. Ta nivoja sta konvolucijska nevronska mreža in tesno povezan nivo za napovedovanje oznak.

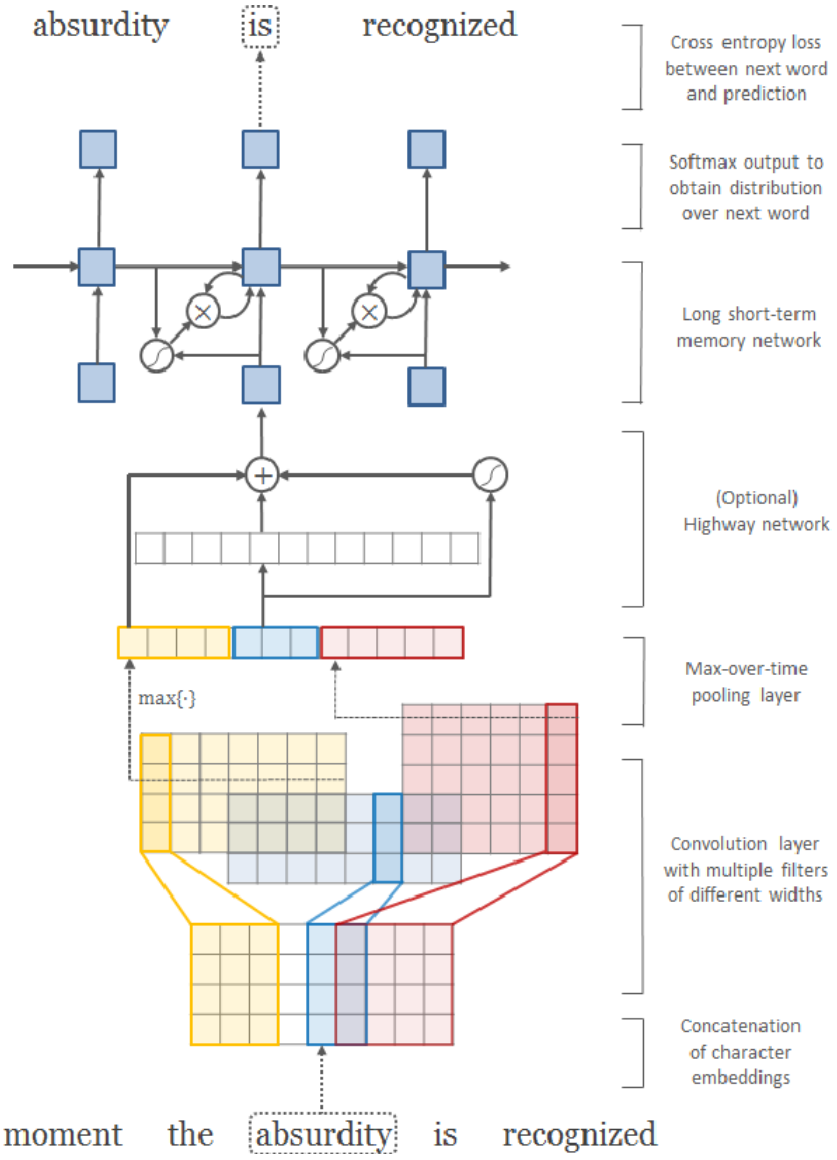
Konvolucijska nevronska mreža se uči predstavitev posameznih besed na nivoju znakov, napovedni nivoji pa se učijo iz teh predstavitev izpeljevati oblikoskladenjske oznake. Napovedni nivo so implementirali z uporabo tesno povezanih nevronskih mrež in z dvosmerno rekurentno mrežo. Model so učili na nemškem korpusu TIGER Treebank in svoje rezultate primerjali z označevalnikom ki temelji na algoritmu CRF (REF Mueller et al., 2013). V primerjavi rezultatov so pokazali, da njihov model na nivoju znakov vedno

deluje bolje kot njihova implementacija nevronskega označevalnika na nivoju besed. Najboljšim rezultatom označevalnika z algoritmom CRF se je najbolj približal model, ki združuje predstavitvi na nivoju znakov in na nivoju besed.

Kim, Jernite, Sontag, Rush (2015) [7]

Avtorji predstavijo jezikovni model osnovan na globokih nevronskih mrežah. Kot pri zgornjih dveh rešitvah (3.4, 3.4), je bil tudi tukaj uporabljen pristop z vhodi na nivoju znakov in ne na nivoju besed, oziroma pojavnic, kot je to pri jezikovnih modelih običajno.

Njihov jezikovni model je opravljal nalogo napovedovanja naslednje besede. Struktura nevronske mreže je podobna prejšnjim rešitvam opisanim v tem podpoglavju. Zgradba modela s primerom uporabe je vidna na sliki **??**. V svojem modelu so uporabil tudi hitrocestno nevronsko mrežo med konvolucijskimi in rekurentnimi sloji. V primerjavi rezultatov so opazili, da njihov jezikovni model ob uporabi hitrocestne mreže daje boljšo oceno čudenja, kakor model, ki vmesne hitrocestne mreže ne uporablja.



Slika 3.2: Arhitektura modela za napovedovanje naslednje besede z vhodi na nivoju znakov. Takšen jezikovni model je bil predstavljen v [7]. Vhod na nivoju znakov je na spodnji strani skice. Od spodaj navzgor vhodu sledijo preslikave v vektorske vložitve znakov, konvolucijski filtri različnih širin, združevalni sloji s časovnim maksimumom, hitrocestna nevronska mreža, mreža LSTM, in izhodni sloj enot z aktivacijo softmax.

Poglavje 4

Arhitektura rešitve

V tem poglavju predstavimo zasnovo in implementacijo naše rešitve. Pri zasnovi rešitve zastavljenega problema smo se zgledovali po jezikovnem modelu predstavljenem v [7]. Omenjeno rešitev smo vzeli za zgled, ker zadostuje našemu cilju, da bi se model nauči dodeljevati pravilne oznake, brez uvajanja dodatnih značilk, kot v [2] in [4].

Glavni deli naše rešitve so priprava podatkov, zgradba označevalnika z nevronskimi mrežami. Za izboljšavo kvalitete napovedi našega označevalnika smo nazadnje implementirali še preprost ansambel, ki združuje naš označevalnik skupaj z označevalnikoma Obeliks [2] in Reldi [4].

4.1 Priprava podatkov

Naš model smo učili na podatkih iz jezikovnega korpusa ssj500k, različice 2.0 [1]. V času nastajanja pričujočega dela je bil to največji ročno označen korpus za slovenski jezik. Vključuje 586248 pojavnic v 27829 povedih, vsaki pojavnici pa je prirejena tudi oblikoskladenjska oznaka pa specifikacijah MULTEXT East REF.

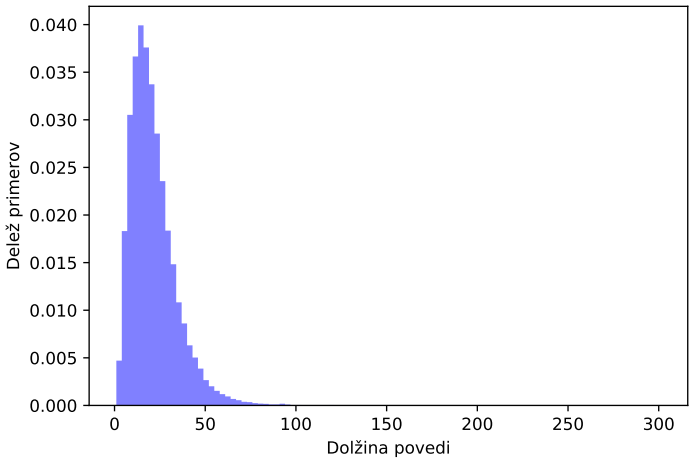
Korpus je na voljo v dveh formatih: XML-TEI in vert. Format XML-TEI boljše predstavlja hierarhijo korpusa, prednost formata vert pa je, da je enostavnejši za razčlenjevanje. Korpus v formatu vert je besedilna zbirka,

v kateri ima vsaka vrstica pojavnico skupaj s pripadajočimi oznakami. Vrednosti v vrstici so med sabo ločene s tabulatorjem. Ker smo iz korpusa potrebovali samo pare pojavnic in njihovih oznak, smo se odločili za format vert.

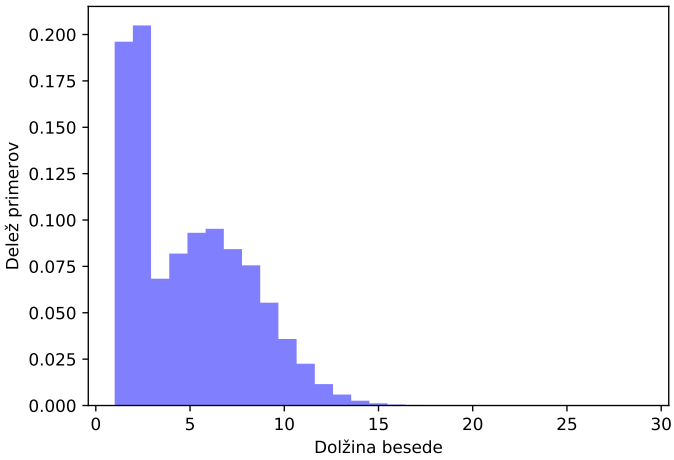
Izbor povedi

Pri zgradbi nevronske mreže našega označevalnika smo se morali odločiti za določeno dolžino vhodnih povedi in pojavnic, ki jih bo naš model dovoljeval. Ta zahteva izhaja iz dejstva, da morajo imeti vhodni primeri v model enotne dimenzije.

Z večanjem dimenzij vhodnih podatkov narašča zahtevnost problema, s tem pa tudi čas učenja nevronskega modela. Iz tega razloga nismo želeli določiti našemu modelu prevelikih vhodnih dimenzij. Po drugi strani smo želeli ohraniti kar se da veliko primerov iz korpusa. Za lažji izbor omejitev dolžin, smo preverili porazdelitvi dolžin povedi in pojavnic. Porazdelitvi sta prikazani na slikah **??** in **??**.



Slika 4.1: Porazdelitev dolžin povedi v korpusu ssj500k. Dolžino povedi merimo s številom pojavnic.



Slika 4.2: Porazdelitev dolžin pojavnic v korpusu ssj500k.

Za zgornjo mejo dolžin pojavnic smo izbrali dolžino dvajset znakov, za povedi pa 80 pojavnic. Po odstranitvi predolgih povedi iz korpusa, jih je od začetnih 27829 ostalo še 27702. Tako smo odstranili manj kot pol odstotka vseh primerov iz korpusa in si s tem omogočili omejitev dimenzionalnost vhodnih podatkov na spremenljivo velikost.

Pretvorba povedi v vektorske vložitve

Podatke smo za obdelavo v nevronskem modelu pretvorili v vektorsko obliko. Vsako poved je bilo potrebno pretvoriti v matriko z dimenzijami, ki ustrezajo izbranim omejitvam dolžine. Ker smo povedi omejili na dolžino 80 pojavnic in pojavnice na 20 znakov, smo na tem koraku povedi pretvorili v matrike z 80 vrsticami in 20 stolpci.

Vrednosti v matriki smo zapolnili z indeksi znakov, prazna mesta pa zapolnili z ničlami. Spodnji primer ponazarja preslikavo povedi ”To bi bila pravična vojna.”v ustrezno matriko.

Dana poved je zaporedje pojavnic:

$$\begin{pmatrix} \text{To} \\ \text{bi} \\ \text{bila} \\ \text{pravična} \\ \text{vojna} \\ . \end{pmatrix}$$

Vsaki pojavnici preslikamo znake v indekse in jih vstavimo v vektor dolžine 20. Vektorje pojavnic zložimo skupaj v vektor povedi:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ \vdots \\ 80 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & \dots & 20 \end{array} \\ \begin{pmatrix} 51 & 75 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 62 & 69 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 62 & 69 & 72 & 61 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 76 & 78 & 61 & 82 & 69 & 122 & 74 & 61 & 0 & \dots & 0 \\ 82 & 75 & 70 & 74 & 61 & 0 & 0 & 0 & 0 & \dots & 0 \\ 14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \end{pmatrix} \end{array}$$

Pretvorba oblikoskladenjskih oznak v vektorske vložitve

Oblikoskladenjske oznake smo za ustrezno predstavitev v računalniškem pomnilniku preslikali v vektorske vložitve.

Pri tipičnem pristopu s področja strojnega učenja bi oznake predstavili s kodiranjem one-hot. Takšno kodiranje ima v našem primeru dve pomanjkljivosti:

Velika prostorska zahtevnost Vseh možnih oznak po specifikaciji MULTEXT East je 1900. To pomeni, da bi morali vsaki izmed pol milijona pojavnic v korpusu prirediti 1900 dimenzionalni vektor.

Izguba informacij o odvisnostih med oznakami Pri kodiranju one-hot so vsi razredi med sabo linearno neodvisni in si zato ne morejo deliti nekaterih istih ponavljajočih lastnosti. V našem primeru so takšne lastnosti sklon, spol, število in podobne, ki se pojavljajo pri različnih besednih vrstah.

Zaradi navedenih pomanjkljivosti smo uvedli novo kodiranje oblikoskladenjskih oznak. To kodiranje tvori gostejše vektorske vložitve oznak in omogoča deljenje posameznih lastnosti med različnimi vrstami oznak.

Za potrebo implementacije kodiranja smo iz specifikacij oblikoskladenjskih oznak zbrali vse možne lastnosti in njihove vrednosti. Posamezne lastnosti smo s kodiranjem one-hot preslikali v vektorske vložitve in jih zaporedno zložili v daljši vektor. Seznam lastnosti oblikoskladenjskih oznak je skupaj z njihovimi mesti v vektorskih vložitvah prikazan v tabeli 4.1.

S takšnim kodiranjem smo vektorje oznak v primerjavi s kodiranjem one-hot dosegli zmanjšanje dolžin vektorjev iz 1900 na 118. Poleg zmanjšanja dimenzionalnosti smo dosegli tudi deljenje lastnosti med različnimi oznakami. Tako se lahko naš model o pogostih besednih lastnostih v slovenskem jeziku uči preko različnih besednih vrst.

Takšna predstavitev spremeni pristop k problemu. V primeru, da bi razrede predstavili s kodiranjem one-hot, bi oblikoskladenjsko označevanje reševali z večrazredno klasifikacijo, kjer bi bila vsaka možna oznaka en razred. V našem primeru imamo drugačen cilj, saj ne dodeljujemo enega razreda, temveč več različnih oznak. Sprememba pristopa ima to slabost, da lahko model napoveduje kombinacije oblikoskladenjskih lastnosti, ki ne sodijo skupaj in tako tvori neveljavne oblikoskladenjske oznake. Primer neveljavne oznake je glagol, ki mu model pripiše sklon.

Ta problem smo poskusili omejiti s tem, da smo vse vektorje lastnosti razširili z dodatnim poljem ki označuje, da ta lastnost ni veljavna. S temi dodatnimi polji smo želeli nevronskemu modelu eksplicitno označiti, kdaj določena lastnost ni veljavna.

Tabela 4.1: Lokacije lastnosti v vektorski vložitvi oznak.

Lastnost	Začetek	Konec
Besedna vrsta	1	13
Določnost	14	16
Naslonskost	17	19
Nikalnost	20	22
Glagolska oblika	23	30
Oseba	31	35
Sklon	36	43
Spol	44	48
Spol svojine	49	53
Število	54	58
Število svojine	59	63
Stopnja pridevnika	64	67
Glagolski vid	68	72
Vrsta glagola	73	75
Vrsta neuvrščene pojavnice	76	83
Vrsta pridevnika	84	87
Vrsta prislova	88	90
Vrsta samostalnika	91	93
Vrsta števnika	94	98
Vrsta veznika	99	101
Vrsta zaimka	102	111
Zapis števnika	112	115
Živost samostalnika	116	118

4.2 Nevronske mreže

Pri strukturi nevronskega modela smo se zgledovali po modelu predstavljenem v [7]. Nevronsko mrežo smo zgradili iz več delov, ki so med sabo funkcionalno ločeni.

Uporabljena orodja

Za implementacijo nevronske mreže smo uporabili odprtokodno knjižnico Keras za programski jezik Python. Knjižnica Keras nudi enoten vmesnik za delo z implementacijami nevronskih mrež Tensorflow, Theano in Microsoft cognitive toolkit.

Keras omogoča enostavno gradnjo različnih vrst nevronskih mrež. Nudi gradnike, in ovojnice za gradnike, ki se jih da preprosto povezati v modele. Kljub preprosti uporabi, uporabniku dopušča svobodo pri nastavljanju parametrov.

Vhod

Za začetni sloj v nevronski model smo določili Kerasov gradnik Input. Ta gradnik iz vhodnih podatkov zgradi vektorske vložitve.

V našem primeru so vhodni podatki vektorji besed, dolžine 20 znakov. Po zgledu iz [7] smo gradniku Input določili dimenzijo 60, kar pomeni, da se vsak znak preslika v gosto vektorsko vložitev dolžine 60. Sloj Input tako preslika vhodne vektorje besed v matrike dimenzije 20×60.

Kot alternativa gostim vektorskim vložitvam znakov, bi lahko znake predstavili z redkimi vektorji one-hot, vendar bi v tem primeru imeli opravka z matrikami besed dimenzije 20×131, saj je v korpusu 131 različnih znakov, mi pa smo v našem modelu želeli obravnavati vse znake.

Tabela 4.2: Parametri konvolucijskih slojev

Širina filtrov	Število filtrov	Aktivacijska funkcija	Dimenzija izhoda
1	50	*tanh*	20×50
2	100	*tanh*	20×100
3	150	*tanh*	20×150
4	200	*tanh*	20×200
5	200	*tanh*	20×200
6	200	*tanh*	20×200
7	200	*tanh*	20×200

Predstavitveni model

Vhodnemu sloju, ki pripravi vektorske vložitve znakov sledi več vzporednih konvolucijskih slojev. V [7] so preizkusili dve konfiguraciji konvolucijskih slojev, ki se razlikujeta v številu slojev in številu enot v slojih. Ker poročajo o boljših rezultatih pri uporabi konfiguracije z več parametri, smo si tudi mi izbrali takšno.

Konvolucijska nevronska mreža, ki smo jo implementirali ima sedem vzporednih slojev. Vsi sloji uporabljajo aktivacijsko funkcijo hiperbolični tangens, razlikujejo pa se v številu filtrov in njihovi širini. Uprabljena konfiguracija je predstavljena v tabeli 4.2.

Vsakemu izmed konvolucijskih slojev sledi sloj globalnega združevanja po maksimumu 3.1. V teh slojih se izhodi konvolucijskih slojev strnejo v gostejši zapis, saj združevalna funkcija podatke v vsaki časovni enoti povzame z njihovim maksimumom. Tako dobljene združitve zložimo v skupni vektor.

Združevalnim slojem sledi še hitrocestna nevronska mreža 3.3, s katero omogočimo lažji pretok informacij.

Do sedaj opisane sloje smo ovili z ovojnico TimeDistributed, ki omogoči modeliranje časovno odvisnih podatkov. V našem primeru so časovne enote besede v povedi.

Jezikovni model

Del implementacije naše rešitve, ki je najbolj prilagojen za učenje zakonitosti jezika je implementiran z rekurentnimi nevronskimi mrežami z dolgim kratkoročnim spominom LSTM. Kot pri večjem izmed modelov, ki sta predstavljena v [7], smo tudi mi sloju LSTM dodelili 650 enot.

Jezikovni model iz [7] smo prilagodili tako, da nismo imeli dveh slojev LSTM za obdelavo besedila v eno smer, ampak smo uporabili dvosmerni pristop. Pri našem pristopu smo sloju, ki podatke obdeluje v smeri iz leve proti desni dodali še sloj, ki deluje v obratno smer. S tem smo želeli naš model spodbuditi k odkrivanju odvisnoti med besedami v obeh smereh.

Jezikovni model smo implementirali z gradnikoma LSTM in Bidirectional, ki sta sestavni del knjižnice Keras.

Odločitveni model

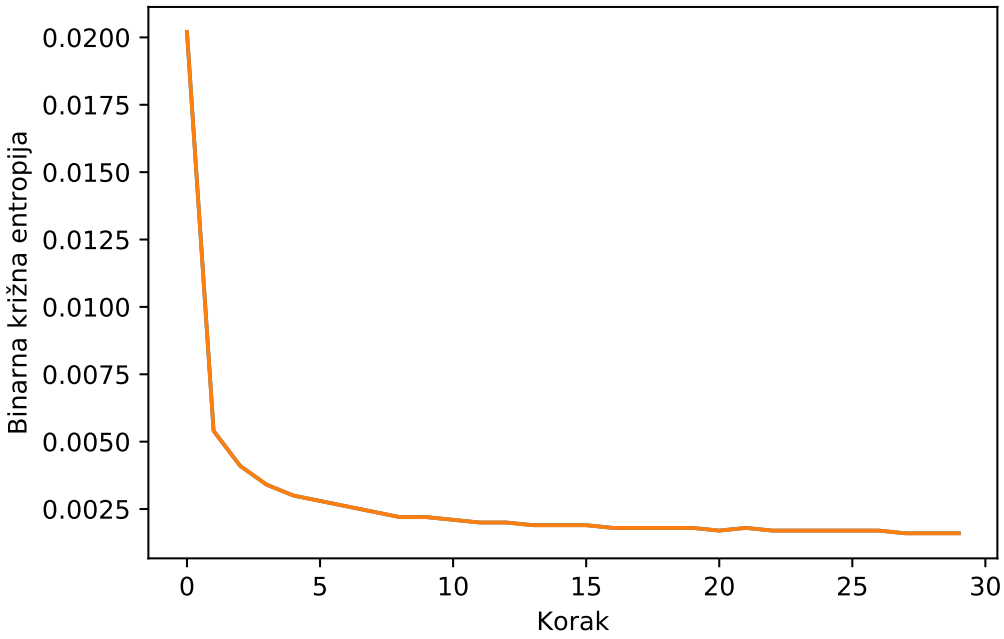
Za določanje oznak smo kot zadnji gradnik našega modela dodali sloj tesno povezanih enot. Teh enot je 118; toliko, kolikor je dolžina naših predstavitev oznak. Ker smo želeli v vsaki izmed enot dobiti oceno verjetnosti smo uporabili za aktivacijsko funkcijo uporabili logistično funkcijo.

4.3 Učenje modela

Našo nevronsko mrežo smo med razvojem učili na 90 odstotkih celotne učne množice. Za funkcijo izgube smo določili binarno prečno entropijo. Binarna prečna entropija je primerna funkcija izgube pri problemih klasifikacije z več oznakami. Ta funkcija je že implementirana v knjižnici Tensorflow in podprta v Kerasu.

Dokončno sestavljen označevalnik smo učili z optimizacijskim algoritmom Adam v 30 sprehodih skozi učno množico. Na sliki **??** je prikazano gibanje vrednosti funkcije izgube med učenjem. Pojemajoče padanje nakazuje, da se je algoritem približal optimumu, kar smo tudi potrdili s preizkusi na testni

množici in s prečnim preverjanjem.



Slika 4.3: Prikaz izgube pri učenju modela v 30 korakih.

4.4 Ansambel označevalnikov

Z namenom izboljšanja našega modela, smo implementirali preprost ansambel označevalnikov. V ansambel smo poleg naše rešitve vključili še označevalnika Obeliks [2] in Reldi [4].

Implementirali smo preprost algoritem, ki oznake za vsako besedo vhodne povedi dodeljuje na podlagi glasovanja vključenih modelov. Algoritem v primeru treh različnih, veljavnih oznak vzame tisto, ki jo predlaga naš nevronski označevalnik. V primeru da nevronski označevalnik ustvari neveljavno kombinacijo oblikoskladenjskih lastnosti, algoritem vrne oznako, ki jo predlaga označevalnik Reldi. V ostalih primerih, ko vsaj dva označevalnika predlagata isto oznako, vrne algoritem predlog večine.

Pri implementaciji tega algoritma smo dali največjo težo našemu označevalniku, ker je v primerjavi označevalnikov dosegel najboljše ocene (podpoglavje 5.1).

Izjemo za primer neveljavnih oznak smo uvedli, ker naš označevalnik takšne oznake lahko tvori, za razliko od ostalih dveh označevalnikov, ki vedno vračata veljavne oznake.

Poglavje 5

Evalvacija

5.1 Primerjava označevalnikov

Metodologija

Prečno preverjanje

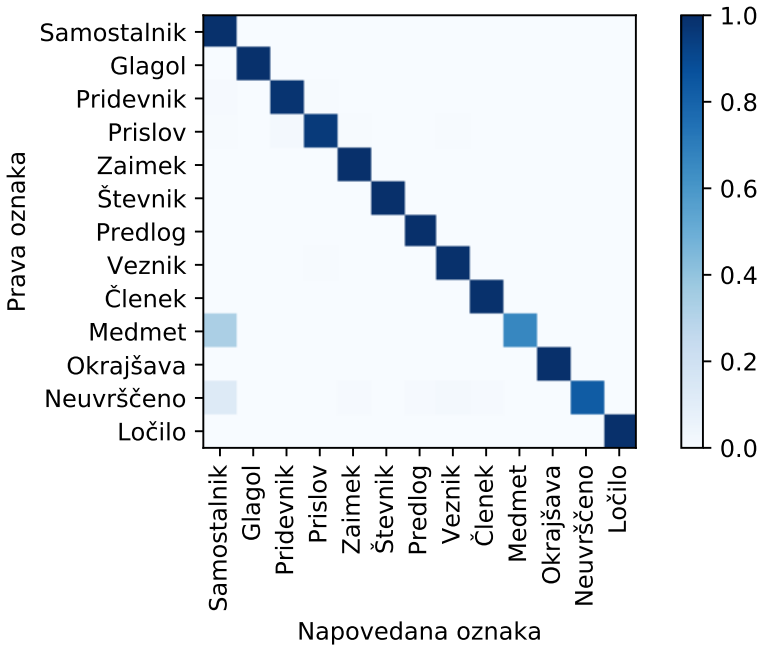
Rezultati

5.2 Analiza napak nevronskega označevalnika

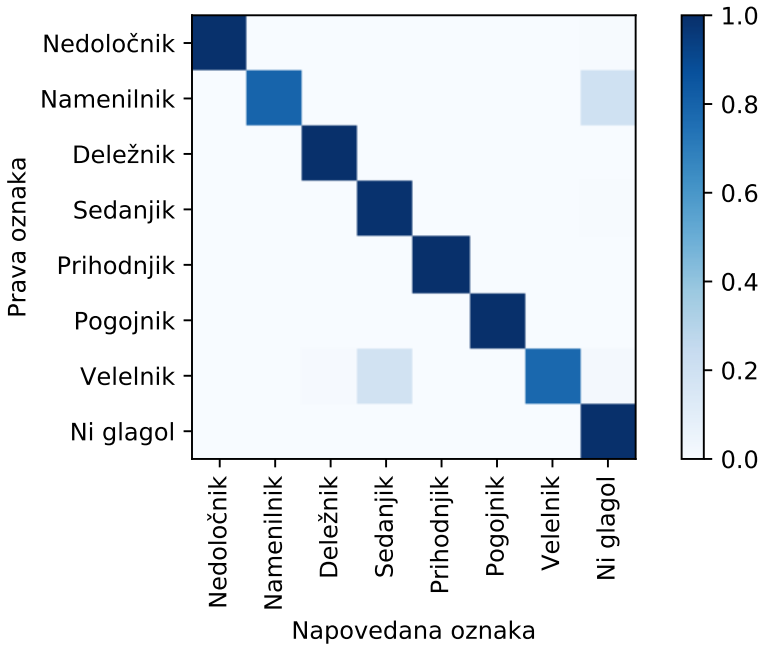
Tabela 5.1: ocene

Označevalnik	Točnost besednih vrst	Točnost oznak
Obeliks	0.923	0.810
Reldi	0.987	0.943
Nevronski označevalnik	0.988	0.956
Ansambel z večino	0.991	0.955

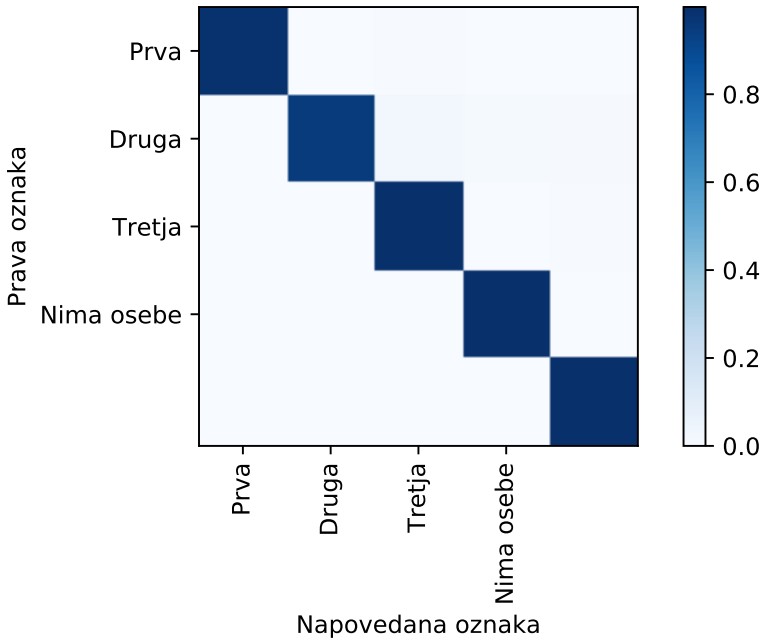
Slika 5.1: Matrika napačnih klasifikacij za besedne vrste. Izstopata razreda za medmet in neuvrščeno, ki ju model največkrat označi kot samostalnik.



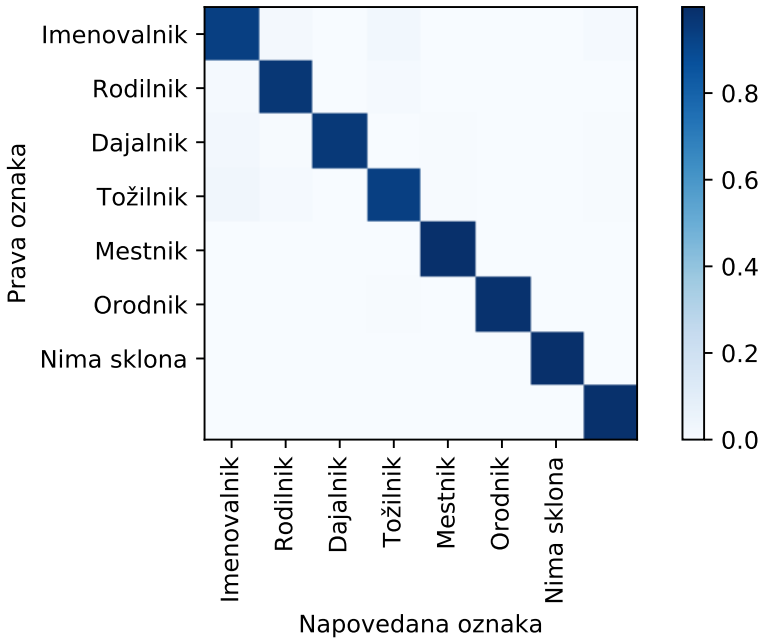
Slika 5.2: Napačne klasifikacije za glagolsko obliko. Model največkrat napačno označi glagole v namenilniku in velelniku.



Slika 5.3


Prva
Druga
Tretja
Nima osebe
Prava oznaka
Prva
Druga
Tretja
Nima osebe
Napovedana oznaka
0.8
0.6
0.4
0.2
0.0

Slika 5.4


Imenovalnik
Rodilnik
Dajalnik
Tožilnik
Mestnik
Orodnik
Nima sklona
Prava oznaka
Napovedana oznaka
0.8
0.6
0.4
0.2
0.0

Poglavje 6

Zaključki

Literatura

[1] S. Krek, K. Dobrovoljc, T. Erjavec, S. Može, N. Ledinek, N. Holz, K. Zupan, P. Gantar, T. Kuzman, Training corpus ssj500k 2.0 (2017) [navedeno 2017-11-29].
URL http://hdl.handle.net/11356/1165

[2] M. Grčar, S. Krek, K. Dobrovoljc, Obeliks: statistični oblikoskladenjski označevalnik in lematizator za slovenski jezik, Proceedings of the 8th Language Technologies Conference (2012) 89–94.

[3] S. Krek, T. Erjavec, K. Dobrovoljc, S. Može, N. Ledinek, N. Holz, Training corpus ssj500k 1.3 (2013) [navedeno 29.11.2017].
URL http://hdl.handle.net/11356/1029

[4] N. Ljubešić, T. Erjavec, Corpus vs. lexicon supervision in morphosyntactic tagging: the case of Slovene, LREC 2016 (2016) 1527–1531.

[5] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org.

[6] R. K. Srivastava, K. Greff, J. Schmidhuber, Highway NetworksarXiv:1505.00387.
URL http://arxiv.org/abs/1505.00387

[7] Y. Kim, Y. Jernite, D. Sontag, A. M. Rush, Character-aware neural language models, Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16) (2015) 2741–2749.

[8] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, Natural language processing (almost) from scratch, Journal of Machine Learning Research 12 (2011) 2493–2537. arXiv:1103.0398, doi:10.1.1.231.4614.

[9] C. Dos Santos, B. Zadrozny, Learning Character-level Representations for Part-of-Speech Tagging, Proceedings of the 31st International Conference on Machine Learning ICML-14 (2011) (2014) 1818–1826.

[10] M. Labeau, A. Allauzen, Non-lexical neural architecture for fine-grained POS tagging, Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015) (September) (2015) 232–237.