Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Krsnik

Napovedovanje naglasa slovenskih besed z metodami strojnega učenja

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Marko Robnik-Šikonja
Somentor: dr. Tomaž Šef

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Zahvala

Najprej bi se zahvalil mentorju, prof. dr. Marku Robniku-Šikonji, za pomoč in svetovanje pri izdelavi magistrske naloge ter za samo idejo. Zahvala gre tudi dr. Tomažu Šefu, ki je prav tako z uporabnimi nasveti in odgovori na vprašanja doprinesel k temu delu in omogočil izdelavo naloge s posredovanjem podatkovne zbirke. Zahvaljujem se Inštitutu Jožefa Stefana in zaposlenim, ki so mi omogočili uporabo strojne opreme. Na koncu bi se zahvalil še družini, ki me je vsa leta študija podpirala in mi stala ob strani ter sošolcem Manci, Primožu, Roku, Ožboltu, Denisu in Sandiju za lažji, krajši in zabavnejši čas študija.

Luka Krsnik, 2017

Vsebina

Povzetek

Naslov: Napovedovanje naglasa slovenskih besed z metodami strojnega učenja

Za naglaševanje slovenskih besed ne obstaja preprost algoritem, naglasa slovenskih besed se namreč govorci naučimo med njihovim spoznavanjem. Metode strojnega učenja so se pri naglaševanju izkazale za uspešne. V magistrski nalogi smo na problemu preizkusili globoke nevronske mreže. Testirali smo različne arhitekture nevronskih mrež, več različnih predstavitev podatkov in ansamble mrež. Najboljše rezultate je vrnil ansambelski pristop, ki je pravilno napovedal 87,62 % besed iz testne množice. S predlaganim pristopom smo za več odstotkov izboljšal rezultate drugih metod strojnega učenja.

Ključne besede

umetna inteligenca, strojno učenje, globoke nevronske mreže, procesiranje naravnega jezika, naglaševanje

Abstract

Title: Prediction of stress of Slovenian words with machine learning methods

There is no simple algorithm for stress assignment of Slovene words. Speakers of Slovene are usually taught accents together with words. Machine learning algorithms give positive results on this problem, therefore we tried deep neural networks. We tested different architectures, data presentations and an ensemble of networks. We achieved the best results using the ensemble method, which correctly predicted 87,62 % of tested words. Our neural network approach improved results of other machine learning methods and proved to be successful in stress assignment.

Keywords

artificial inteligence, data mining, machine learning, deep neural networks, natural language processing, accentuation

Poglavje 1

Uvod

Pretvorba grafema (najmanjša enota pisave) v fonem (najmanjša glasovna enota, s katero razlikujemo pomene besed) je v slovenskem jeziku enostavna, če imajo zapisane besede označen naglas. Večina besedil ni anotiranih z naglasi, kar pomeni, da moramo ob branju sami naglaševati besede. To znamo ljudje, na poznanih in nepoznanih besedah, precej dobro, a pravila, s katerimi bi lahko strojno dobro določili naglase, so slabo definirana.

Če bi imeli slovar vseh naglašenih besed slovenskega jezika, bi naglase lahko določali tako, da bi vsako besedo poiskali v slovarju naglašenih besed in jo naglasili. Žal tak slovar ne obstaja, poleg tega pa se vedno pojavljajo nove besede, ki jih ni v slovarju in jih je potrebno naglasiti.

Nekatere metode strojnega učenja so bile na problemu naglaševanja že preizkušene [23, 32, 15]. Globoke nevronske mreže, ki dajejo odlične rezultate na sorodnih problemih prepoznavanja in generiranja govora, na problemu naglaševanja še niso bile uporabljene, zato smo poiskus naredili v tej nalogi.

1.1 Predstavitev problema in področja

Naglas besede je zlog, na katerem je beseda jakostno ali tonemsko izrazita [30, 24]. Govorci slovenskega jezika se mesto in tip naglasa besed naučimo

hkrati z učenjem jezika, saj je določeno za vsako besedo posebej, preprostih pravil za naglaševanje pa ni. Vsak zlog v slovenščini vsebuje natanko en samoglasnik (črke *a*, *e*, *i*, *o*, *u*) ali črko *r* (samo v primerih, ko črka ni pisana poleg samoglasnika) [32]. Zlogovna znamenja pišemo na teh črkah.

Obstajajo besedne oblike, ki se zapišejo enako, a imajo naglas na različnih mestih. Te besedne oblike se lahko razlikujejo po besedni vrsti, sklonu, številu, spolu, pomenu ipd. Ponavadi se jih da pravilno naglasiti zaradi konteksta, v katerem se nahajajo. Mesta naglasov je težko določiti tudi zato, ker njihovo število v posamezni besedi ni določeno. Večina besed ima en naglas, obstajajo pa tudi besede, ki naglasa nimajo ali pa so naglašene na dveh, ali še redkeje, treh mestih.

Slovenščina pozna dve vrsti naglaševanja - prevladujočo jakostno (ali dinamično) in manj znano tonemsko [31]. Pri jakostnem naglaševanju naglase poudarjamo z večjim izdihom zraka, pri tonemskem pa z visokim ali nizkim tonom. Če se osredotočimo na jakostno naglaševanje, lahko naglasna mesta zaznamujemo z znamenji za kvantiteto in kvaliteto. Po kvantiteti se naglasna znamenja razlikujejo na kratke in dolge, po kvaliteti pa na ozke in široke.

Težavnost določanja naglasa se med različnimi jeziki precej razlikuje [15]. Nekateri jeziki imajo v naprej določen naglas besed - v finščini je naglas vedno na prvem zlogu, v makedonščini pa na tretjem zlogu gledano od zadaj. Drugi jeziki, kot na primer latinščina, imajo besede naglašene na različnih mestih, vendar so ta mesta odvisna od strukture predzadnjega zloga. Angleščina je primer jezika kjer je naglas deloma odvisen od izvora besede. Slovenščina, poleg romunščine, litovščine, ukrajinščine in nekaterih drugih jezikov, spada v skupino, kjer je naglas lahko postavljen na katerikoli zlog.

1.2 Struktura naloge

Naloga je razdeljena na pet poglavij. Uvodno poglavje vsebuje motivacijo za nalogo in kratek opis problema in področja. Drugo poglavje podrobneje

predstavi problem naglaševanja, z njim povezane raziskave in uporabljeno podatkovno množico. Sledi poglavje, ki opiše osnovno idejo nevronskih mrež in različne pristope pri implementaciji mrež. V četrtem poglavju prestavimo in ovrednotimo arhitekture, ki smo jih preizkusili in analiziramo napake. Nalogo zaključimo s pregledom narejenega, predstavitvijo strokovnih prispevkov naloge, njeno kritično analizo in idejami za izboljšave.

Poglavje 2

Problem naglaševanja in povezane raziskave

V tem poglavju so opisani predhodni poskusi z uporabo strojnega učenja na problemu naglaševanja. Sledi predstavitev nevronskih mrež za podobne probleme in opis uporabljene podatkovne množice.

2.1 Povezane raziskave

Ko skušamo določiti naglase besed, moramo vsak jezik obravnavati ločeno. Na litovščini se je kot precej uspešen izkazal pristop za določitev mesta naglasa s pravili [26]. Dobre rezultate so dali tudi poizkusi s skritimi markovskimi modeli in baysovskimi klasifikatorji za francoščino in španščino [2]. Ti poskusi se precej razlikujejo od našega dela, saj so možne variante naglasov znane vnaprej in tako naglas klasificiramo le v pravilni razred izmed nekaj možnih razredov. Podoben problem so reševali tudi Mihalcea in sodelavci [16], ki so v češčini, madžarščini, poljščini in romunščini dodajali manjkajoča diakritična znamenja (znamenja napisana nad ali pod črko, ki so lahko naglasi, ni pa nujno).

Pravila za naglaševanje v slovenščini so strokovnjaki prvič napisali pred

več kot 30 leti [30] in so obsegala približno 10 strani. Šef in Gams [23] sta ta pravila z odločitvenimi pravili zapisala kot program. Pri tem sta naletela na več težav, saj bi v določenih primerih pravila potrebovala podatke, ki jih računalniku ni enostavno pridobiti, predvsem pri prepoznavanju sposojenih besed. Naglase sta skušala določiti tudi s pomočjo algoritmov strojnega učenja, konkretno z odločitvenimi drevesi. Rezultati so bili precej boljši kot pravila strokovnjakov. Za strojno učenje sta problem razdelila na dva dela, iskanje mesta naglasa in iskanje tipa naglasa.

Spoznanje, da se metode strojnega učenja dobro obnesejo na problemu naglaševanja slovenskih besed, je sprožilo testiranje še drugih metod strojnega učenja. Preizkušeni sta bili metodi boosting in markovski modeli na nivoju črk [32]. Prva metoda je pravilno naglasila približno 71 % neznanih besed slovarja, druga pa 82 %. Poleg omenjenih so bile preizkušene še inačice odločitvenih dreves (J48, PART), bagging in ansambelska različica bayesovskega klasifikatorja (AODE) [15]. V omenjenem delu [15] je bila po določitvi verjetnosti naglasov uporabljena še hevristika. Ta kot končni rezultat ni nujno vrnila najbolj verjetne kombinacije mest naglasa, kot jih je predvidel algoritem, ampak tiste, ki so bile najbolj verjetne in hkrati opažene. Da je bila kombinacija naglasnih mest opažena, je moralo v slovarju obstajati vsaj 0.1% besed z istim številom zlogov in isto kombinacijo naglasnih mest.

2.2 Nevronske mreže na podobnih problemih

Za področje naglaševanja v različnih jezikih ne obstaja dosti raziskav. Večinoma problem obravnavajo kot podnalogo pretvorbe grafema v fonem. Naglaševanje se od jezika do jezika razlikuje. Obstajajo pa naglaševanju podobni problemi, ki so jih raziskovalci analizirali s pomočjo metod strojnega učenja.

V zadnjem času so na področju procesiranja naravnega jezika (Natural Language Processing - NLP) [4] zelo uspešne metode globokih nevronskih mrež. Collobert in sodelavci [6, 5] so med prvimi, ki so na NLP problemih

Tabela 2.1: Primeri zapisov v podatkovni množici.

beseda	lema	morfološke informacije	naglašena beseda
brati	brati	Vmn	bráti
berimo	brati	Vmmp1p	berímo
beri	brati	Vmmp2s	bêri

uporabili nevronske mreže. S pomočjo globokih (konvolucijskih) nevronskih mrež so se lotili več jezikovnih nalog v angleškem jeziku. Lai in sodelavci [12] se lotijo problema tekstovne klasifikacije s pomočjo rekurzivne konvolucijske nevronske mreže (Recurrent convolutional neural network). Članek pokaže, da tovrstne nevronske mreže dosegajo dobre rezultate, in obdeluje angleščino ter kitajščino.

Bližja problemu naglaševanju sta izgovorjava [29] in pretvorba grafema v fonem [19]. Delo [19] uporablja nevronske mreže, ki imajo na vhodu dele besed (grafeme) in ne celotnih besed. Avtor uporabi rekurzivne nevronske mreže z dolgoročnim in kratkoročnim pomnilnikom (Long short-term memory recurrent neural network - LSTM). Zhang in sodelavci [33] za vhod nevronske mreže uporabljajo črke. Delo kaže, da za tovrstno klasifikacijo ne potrebujemo sintaktičnega ali semantičnega znanja o besedilih, algoritmi lahko tudi brez tega dosežejo dobre rezultate.

2.3 Podatkovna množica

Podatkovno množico, s pomočjo katere so naučeni algoritmi strojnega učenja v tej magistrski nalogi, sta predstavila Šef in Gams [23]. Vsebuje slabih 540.000 besed, ki so predstavljene s sledečimi informacijami (glej tabelo 2.1): zapisane nenaglašene besede, leme (osnovne oblike) te besede, morfološke informacije o besedi (razloženo v nadaljevanju) in zapisane naglašene besede. Vse besede zapisane v množici imajo skupaj približno 18.000 različnih lem.

Morfološke informacije o besedah so informacije o besedni zgradbi in od-

nosu do drugih besed. Povedo nam besedno vrsto, spol, sklon itd. Za naglaševanje neznanih besed so ti podatki lahko ključnega pomena. Primer besede, ki se piše enako, njen pomen in naglas pa se razlikujeta le zaradi morfoloških informacij, je beseda "je". Ko je nenaglašena prihaja iz besede "biti", ko je naglašena pa iz besede "jesti".

V učni množici so morfološke informacije podane v formatu MULTEXT-East [7]. Format v slovenščini in nekaterih drugih jezikih srednje in vzhodne Evrope (bolgarščini, hrvaščini, češčini...) standardizira zapis morfosintaktičnih označb. Konkreten primer zapisa morfološke informacije besede "krožnik" v formatu MULTEXT-East je "Ncmsn". Oznaka poda različne informacij o besedi: "N" - besedna vrsta je samostalnik (Noun), "c" - vrsta samostalnika je občno ime (common), "m" - spol je moški (masculine), "s" - število je ednina (singular) in "n" - sklon je imenovalnik (nominative).

Naglašene besede podatkovne množice imajo na naglasnih mestih zapisane različne vrste naglasov (tabela 2.2). Vsi samoglasniki po kvantiteti spadajo med kratke ali dolge, črki *e* in *o* pa se ločita še po kvaliteti - lahko sta široki ali ozki. Ko je naglašen polglasnik pred črko *r*, je naglas vedno dolg.

Tabela 2.2: Tabela prikazuje vse različne tipe dinamičnih naglasov uporabljenih v podatkovni množici.

Simbol	Pomen
ä	kratek naglašeni *a*
ë	kratek naglašeni široki *e*
î	kratek naglašeni *i*
ö	kratek naglašeni široki *o*
ü	kratek naglašeni *u*
á	dolgi naglašeni *a*
é	dolgi naglašeni ozki *e*
ě	dolgi naglašeni široki *e*
í	dolgi naglašeni *i*
ó	dolgi naglašeni ozki *o*
ô	dolgi naglašeni široki *o*
ú	dolgi naglašeni *u*
ŕ	dolgi naglašeni *r*

Poglavje 3

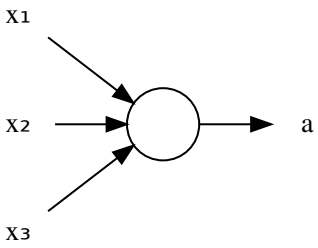
Globoke nevronske mreže

Strojno učenje je proces, katerega namen je odkrivanje vzorcev iz množice podatkov. Algoritmi strojnega učenja delujejo tako, da se najprej naučijo razpoznavati vzorce iz podatkovnih zbirk, nato pa to znanje izkoriščajo na novih podatkih. Eden izmed algoritmov strojnega učenja so tudi nevronske mreže.

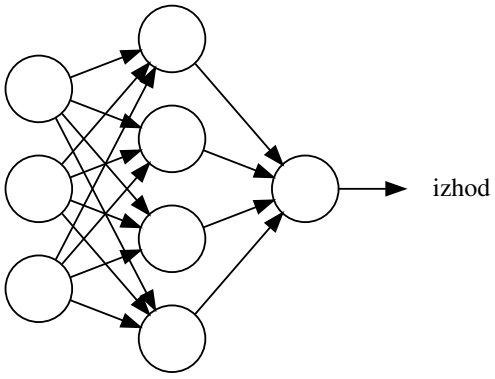
Nevronske mreže so sestavljene iz več povezanih nevronov [22, 17]. Vsak izvede zaporedje aktivacij, glede na dobljene vhodne podatke (slika 3.1). Vrednosti aktivacij nevron izračuna po enačbi $a = f(\sum_j w_j x_j)$, kjer x_j predstavlja posamezen vhod, w_j utež nekega vhoda, $f()$ aktivacijsko funkcijo, a pa vrednost aktivacije (več o aktivacijskih funkcijah v podpoglavju 3.1). Vhodne nevrone aktivirajo podatki iz okolja, ostale pa aktivirajo predhodni nevroni s svojimi rezultati oz. aktivacijami (slika 3.2).

Pri reševanju nekaterih problemov se kot zelo uspešne izkažejo nevronske mreže, katerih nevroni so povezani tako, da je za končni rezultat potrebnih več skritih slojev (to so vsi sloji razen vhodnega in izhodnega), kjer se v vsakem sloju aktivacije spremenijo. Nevronske mreže z več sloji oz. nivoji imenujemo globoke nevronske mreže (slika 3.3).

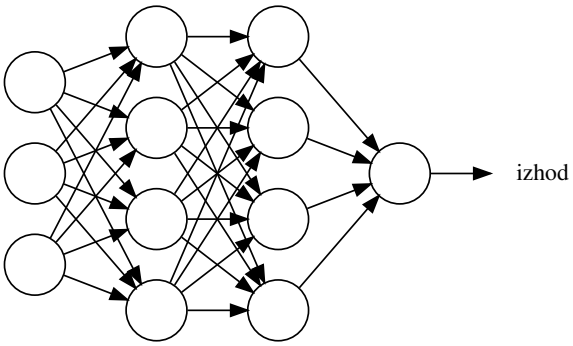
Učenje nevronskih mrež je iskanje uteži na povezavah, uporabljenih v aktivacijskih funkcijah, ki nevronske mreže pripravijo do tega, da na podlagi



Slika 3.1: Primer nevrona, ki dobi podatke iz treh vhodov ter na izhod pošlje aktivacijo.



Slika 3.2: Primer nevronske mreže. Trije najbolj levi nevroni so vhodni nevroni, ki jih aktivirajo podatki iz okolja, ostali pa so aktivirani preko izhodov drugih, prej aktiviranih nevronov. Mreža ima štiri nevrone na skritem nivoju in enega na izhodnem.



Slika 3.3: Primer globoke nevronske mreže z dvema skritima nivojema.

vhodnih podatkov vračajo čim boljše napovedi rezultatov. Za ovrednotenje uspešnosti nevronske mreže se uporabljajo funkcije izgube (glej podpoglavje 3.2). Pri iskanju čim boljših uteži se uporablja metoda vzvratnega razširjanja napake (backpropagation), ki izračuna prispevek napake vsakega nevrona h končni napaki. Ta prispevek je ključen pri optimizacijskem algoritmu gradientnega spusta (glej podpoglavje 3.3), s katerim spreminjamo uteži aktivacijske funkcije tako, da zmanjšamo napako oz. izboljšamo napovedno natančnost.

Delovanje nevronskih mrež še izboljšamo, če uporabimo nekaj postopkov, ki so se kot uspešni izkazali v dosedanjih poskusih [6, 5]. Primeri tovrstnih sprememb so konvolucijski nivoji nevronske mreže (podpoglavje 3.4), združevanje po največjem elementu (maxpooling - podpoglavje 3.5) in izpadanje nevronov (dropout - podpoglavje 3.6).

3.1 Aktivacijske funkcije

Aktivacijska funkcija iz vhodov v nek nevron izračuna izhod [17]. Preprost primer tovrstne funkcije je:

$$z = \sum_j w_j x_j - b \tag{3.1}$$

$$\text{izhod} = \begin{cases} 0, & \text{če velja } z \leq 0 \\ 1, & \text{če velja } z > 0 \end{cases} \tag{3.2}$$

V enačbi x predstavlja vse vhode v nevron, w predstavlja uteži posameznih vhodov, b pa pristranost (bias). Vrednost pristranosti si lahko predstavljamo kot število (prag), ki nam skupaj z obteženo vsoto vhodov pove, če bo izhod funkcije 1 ali 0. Ko je vrednost izhoda funkcije (aktivacije) 1, je nevron aktiven.

Nevronu s tako aktivacijsko funkcijo rečemo perceptron. Ti nevroni sicer niso primerni za dejansko implementacijo, saj mora pri učenju nevronskih mrež veljati, da majhne spremembe uteži le malo vplivajo na rezultat, kar v zgornjem primeru ne drži. Vseeno pa perceptron razloži, kaj se dogaja v nevronu in pomen aktivacijske funkcije.

3.1.1 Sigmoidna funkcija

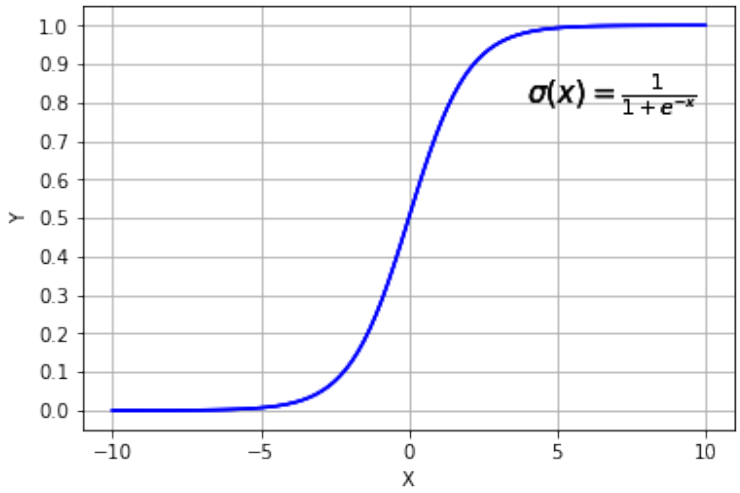
$$\frac{1}{1 + e^{-z}} \tag{3.3}$$

Sigmoidna funkcija (slika 3.4) je primer aktivacijske funkcije, katere rezultat je med vrednostima 0 in 1. Je zvezna in odvedljiva, zaradi česar majhne spremembe uteži le malo vplivajo na rezultat.

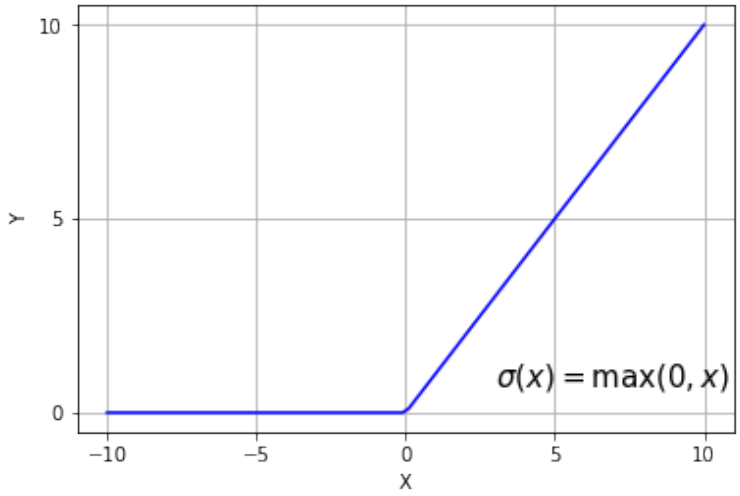
3.1.2 Pragovna linearna funkcija ReLU

$$\max(0, z) \tag{3.4}$$

ReLU funkcija (rectified linear unit - slika 3.5) je aktivacijska funkcija, ki se v zadnjih letih vse več uporablja v nevronskih mrežah [14]. Izkazalo se je namreč, da globoke nevronske mreže, ki uporabljajo to funkcijo, ponavadi vrnejo enake ali boljše rezultate od ostalih [8]. Zanjo je značilno, da je rezultat 0, ko je vhod manjši ali enak 0, sicer pa raste linearno.



Slika 3.4: Primer sigmoidne funkcije.



Slika 3.5: Primer ReLU funkcije.

3.2 Funkcije izgube

Funkcija izgube (loss function) ocenjuje uspešnost nevronske mreže [17]. Njeno natančnost izračunamo tako, da primerjamo vrnjene rezultate nevronske mreže z dejanskimi rezultati, torej rezultati, ki bi jih morali dobiti.

3.2.1 Binarna križna entropija

$$H(y, y') := -\sum_i (y'_i \log(y_i) + (1 - y'_i) \log(1 - y_i)) \tag{3.5}$$

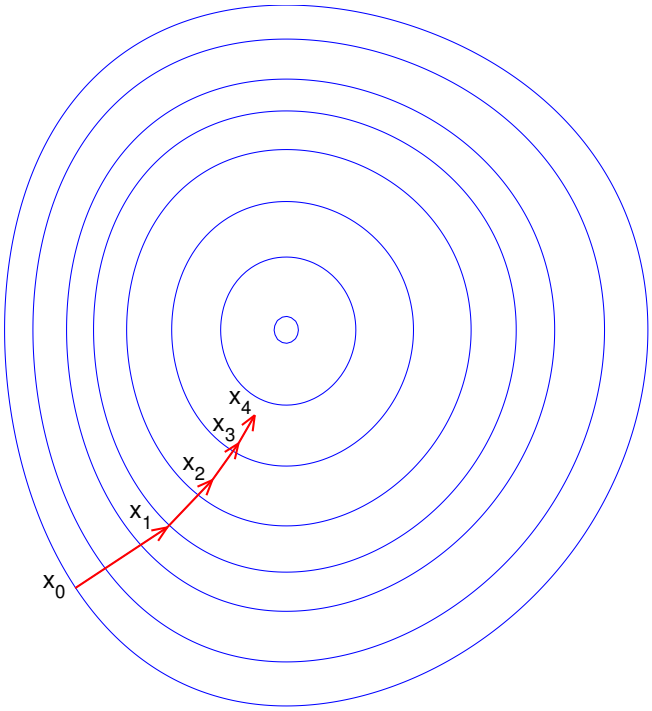
Primer funkcije izgube je binarna križna entropija [25]. V enačbi entropije y predstavlja izračunan izhod iz nevronske mreže, y' pa željen izhod. Ker je izhodov iz nevronske mreže lahko več, funkcija izračuna vsoto napak vseh izhodnih nevronov.

3.3 Gradientni spust

Gradientni spust je optimizacijski algoritem, s pomočjo katerega skušamo najti minimume funkcij [17]. Za nekatere funkcije lahko minimum izračunamo analitično, vendar so funkcije, za katere iščemo minimum v praksi lahko zelo kompleksne. Algoritem deluje tako, da se v majhnih korakih premikamo proti minimumu (slika 3.6). Smer premikanja dobimo s pomočjo gradienta funkcije v točki, ki jo minimiziramo. Dolžino pomika v enem koraku ponavadi določimo s parametrom. Pri nevronskih mrežah uporabljamo gradientni spust za zmanjšanje vrednosti funkcije izgube.

3.3.1 Adam - adaptive moment estimation

Adam je optimizacijski algoritem, ki temelji na gradientnem spustu [11]. Empirične meritve kažejo, da algoritem v veliko primerih prekaša druge, podobne metode. Razlogi za to so v množici manjših izboljšav osnovnega algoritma.



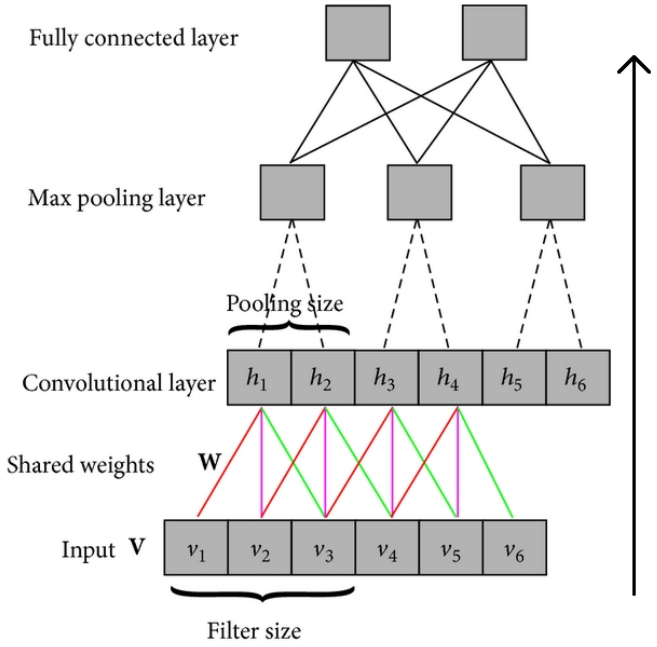
Slika 3.6: Primer delovanja gradientnega spusta [3]. Višina slike, predstavljena z izohipsami ali plastnicam (modre črte), je vrednost funkcije, ki jo izboljšujemo z gradientnim spustom, glede na obe osi, ki predstavljata vrednosti dveh uteži, ki ju v postopku spreminjamo. Lokacija x_i predstavlja vrednosti uteži po i iteracijah gradientnega spusta.

Primer takšne izboljšave je prištevanje momenta h gradientu. Konkretno algoritem pri premiku (koraku) poleg osnovne vrednosti gradienta upošteva tudi del prejšnjega premika, zaradi česar minimum najde hitreje. Druga izboljšava je, da Adam adaptivno izbere učni korak (learning rate) ločeno za vsak parameter posebej. Parametri, ki se ne posodabljajo pogosto, se posledično učijo hitreje. Dodatna prednost te izboljšave je, da so učni koraki prilagojeni avtomatsko, zato ročno prilagajanje parametrov Adama ponavadi ni potrebno.

3.4 Konvolucijski nivoji nevronske mreže

Ponavadi so nivoji nevronskih mrež polno povezani (fully connected), kar pomeni, da vsak nevron na nekem nivoju dobi kot vhod izhode vseh nevronov prejšnjega nivoja [17, 13, 10]. Posledično lahko vsi nevroni prejšnjega nivoja na enak način vplivajo na nevrone trenutnega nivoja. Z drugimi besedami, prostorska struktura vhodov ni pomembna. Pri konvolucijskem nivoju (glej sliko 3.7) so nevroni trenutnega nivoja povezani le z nekaterimi predhodnimi nevroni, ponavadi prostorsko, zaradi česar se nevronska mreža uči iz prostorske strukture podatkov. Pri konvolucijskih nivojih so uteži in pristranost deljeni, torej uporabljeni v vseh nevronih nekega konvolucijskega nivoja. Zato bi lahko rekli, da te uteži razpoznavajo posamezno lastnost dela nevronov prejšnjega nivoja. Ponavadi je teh lastnosti več, zato vsak nivo konvulucije poseduje več skupkov uteži in pristranosti za razpoznavanje večih lastnosti.

Glavni namen konvolucije je razpoznavanje večih vzorcev ali lastnosti iz dela podatkov, ne glede na mesto pojavitve. Če bi nevronski mreži za iskanje mesta naglasa podali besede zapisane kot črke in bi bila velikost filtra enaka tri, bi konvolucijski nivo iskal lastnosti treh zaporednih črk, ki vplivajo na mesto naglasa. Posamezen nevron tega nivoja nevronske mreže vrne isti izhod, če na vhodu dobi isto zaporedje črk, ne glede na to, če nevron išče lastnosti prvih treh črk besede ali zadnjih treh.



Slika 3.7: Primer nevronske mreže [10], ki na drugem nivoju od spodaj navzgor vsebuje konvolucijski nivo (convolutional layer), kjer vsak nevron kot vhod dobi tri podatke (velikost filtra - filter size - je 3). Nad konvolucijskim nivojem je nivo združevanja po največjem elementu (max-pooling), katerega velikost združevanja (pooling size) je 2. Na vrhu je zadnji, polno povezan nivo.

3.5 Združevanje po največjem elementu

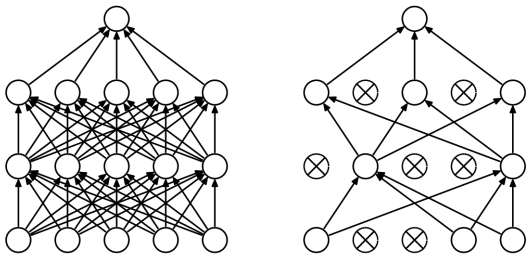
Nivo združevanja po največjem elementu v nevronski mreži ponavadi sledi konvolucijskim nivojem [13, 17] (slika 3.7). Za vsako lastnost, ki so jo razpoznali predhodni nivoji konvolucije, zmanjšamo število povezav, tako da izmed njih izberemo le najbolj pomembne. To naredimo tako, da med povezavami prejšnjega nivoja konvolucije (to število definiramo sami) izberemo največjo aktivacijo.

Z združevanjem po največjem elementu pohitrimo učenje, zaradi zmanjšanja povezav in omogočimo boljšo generalizacijo, saj mreža pri učenju ne uporablja vseh povezav, le pomembnejše.

3.6 Izpadanje

Izpadanje (dropout) je tehnika, ki se uporablja na polno povezanih nivojih nevronske mreže. Njen namen je zmanjšati prilagajanje (overfitting) učnim podatkom [28, 9] (slika 3.8). Deluje tako, da iz nevronske mreže, naključno začasno izpustimo del nevronov skritih nivojev in upoštevamo le preostale. Ko mreža dobi naslednje podatke, se postopek ponovi, le da se izberejo drugi nevroni, ki bodo začasno izpuščeni.

Namen izpadanja je izboljšava generalizacije. Ko tehnike ne uporabimo, se uteži nekaterih nevronov ne naučijo dobro, saj za dobro napoved poskrbijo drugi nevroni. Z izpadanjem nevronske mreže prisilimo k učenju z različnimi kombinacijami aktivnih nevronov in tako k učenju vseh nevronov.



Slika 3.8: Leva slika prikazuje polno povezano nevronsko mrežo, desna pa nevronsko mrežo, ki ima pri učenju na delu učnih podatkov izpuščene nekatere nevrone [28].

Poglavje 4

Arhitektura in evalvacija rešitve

V tem poglavju povemo kako smo iskali arhitekturo rešitve, kako smo primerjali posamezne arhitekture in kako smo preoblikovali podatke za nevronske mreže. Na validacijskih podatkih naredimo analizo napak in preizkusimo ansambelske metode, iz testnih pa pridobimo in predstavimo končne rezultate. Iz vseh podatkov izračunamo uteži nevronske mreže, ki je vračala najboljše rezultate, in jih uporabimo na praktičnih rešitvah.

4.1 Evalvacija

Podatkovna množica je razdeljena na tri dele. Prvi del zajema približno 80 odstotkov vseh podatkov (več o podatkih v poglavju 2.3) in predstavlja učno množico, torej množico iz katere so se učile nevronske mreže (to velja v vseh primerih v magistrskem delu, razen, ko je izrecno omenjeno da je učna množica sestavljena iz drugih podatkov). Tej sledi validacijska množica. To je množica podatkov, s pomočjo katere smo interno primerjali rezultate različnih nevronskih mrež in predstavlja 10 odstotkov podatkovne množice. Zadnjih 10 odstotkov podatkovne množice uporabljamo kot testno množico

za oceno uspešnosti najboljših nevronskih mrež.

Delitev podatkov na tri dele ni povsem naključna. Rezultati takšne delitve bi bili predobri, saj podatkovna množica vsebuje veliko besed, ki se zapišejo in naglasijo enako, minimalno se spremenijo le morfološki podatki o besedi. Zaradi tega smo raje leme (osnovne oblike) besed naključno razdelili v tri skupine in glede na to podatke porazdelili na učno, validacijsko in testno množico.

Rezultate nevronskih mrež smo primerjali tako, da smo za vsako množico izračunali odstotek pravilno naglašenih celotnih besed. To pomeni, da če ima beseda dva naglasa in smo jo prvič pravilno naglasili drugič pa narobe, se je beseda štela kot napačna.

Nevronske mreže smo učili ločeno s procesnimi enotami in vektorskimi pospeševalniki. Na virtualnem okolju, kjer smo za računanje uporabljali procesno enoto, smo imeli 2,4 GHz procesor, računali pa smo z 8 jedri in 32GB RAM-a. Učenje nevronskih mrež, ki so vhod imele predstavljene s črkami, je na takšnem računalniku, ko smo šli dvajsetkrat čez vse podatke, trajalo približno 3 ure. Računanje mrež z vhodnimi podatki kot zlogi je bilo počasnejše trajalo je okrog 2 dni.

Na računalniku z vektorskim pospeševalnikom (NVidia Tesla Kepler 40) smo imeli 16GB RAM-a in smo uporabljali le eno jedro procesne enote. Za učenje mreže s črkovnim vhodom smo porabili dobro uro in pol, za učenje mrež s približkom zlogov pa približno 4 ure in pol.

4.2 Predprocesiranje podatkov

Nevronske mreže računajo s številskmi vrednostmi, zato smo vhodne podatke (predstavljene v podpoglavju 2.3) predstavili na tak način. Izbrali smo tudi primeren način za predstavitev izhoda mreže.

Tabela 4.1: Način gradnje matrike za besedo *abeceda*. Vodoravno so zapisane črke abecede, navpično pa v prvem stolpcu črke razčlenjene besede, v drugem indeksi črk, od tretjega naprej pa zapis indeksov črk z enicami in ničlami, ki je enak vhodni matriki besede, ki gre v nevronsko mrežo. Vse vhodne matrike besed so enakih velikosti (dolžina abecede × maksimalno število črk v besedi).

črke vhodne besede	*indeksi črk*	brez znaka	*a*	*b*	*c*	*č*	*d*	*e*	...
a	1	0	1	0	0	0	0	0	
b	2	0	0	1	0	0	0	0	
e	6	0	0	0	0	0	0	1	
c	3	0	0	0	1	0	0	0	
e	6	0	0	0	0	0	0	1	
d	5	0	0	0	0	0	1	0	
a	1	0	1	0	0	0	0	0	
	0	1	0	0	0	0	0	0	
...	0	1	0	0	0	0	0	0	

4.2.1 Matrična predstavitev vhodnih podatkov

Vhodne besede smo nevronski mreži predstavili tako, da smo vsako besedo razčlenili na manjše dele, iz katerih nevronske mreže znajo razpoznati ponavljajoče se vzorce. Besede smo razčlenili na črke (glej tabelo 4.1). Vsako črko smo z bitnim indeksom zapisali kot vektor, ki ima na indeksnem mestu enico, drugje pa ničle. Tako je vsaka črka na vhodu neodvisna od druge. Tak zapis je primeren za odkrivanje skupin črk s konvolucijsko mrežo.

4.2.2 Vektorska predstavitev morfoloških podatkov

Tudi morfološki podatki so v podatkovni množici zapisani s črkami (več o tem v poglavju 2.3). Teh podatkov ne moremo predstaviti na enak način kot vhodne besede, saj je pomen črk drugačen. Morfološki podatki se začnejo s črko, ki predstavlja besedno vrsto (samostalnik, pridevnik...). Tej sledijo

črke, ki so specifične za vsako besedno vrsto posebej, npr. ista črkovna oznaka pri samostalniku ali pridevniku ima lahko drugačen pomen. Pomembno je tudi mesto črke v morfološkem zapisu, saj označuje neko lastnost (npr. vrsto pridevnika, stopnjo pridevnika, spol pridevnika ipd.).

Morfološke podatke smo podali v obliki vektorja enic in ničel, kjer lokacije vrednosti definirajo različne lastnosti (glej kodo iz slike 4.1). Najprej smo preiskali, kakšne črke se lahko nahajajo na različnih mestih morfoloških podatkov celotne podatkovne množice in možnosti zapisali v večdimenzionalni seznam.

Seznam je po prvi dimenziji razdeljen na besedne vrste. Druga dimenzija predstavlja lokacije črk (ali vrste lastnosti) posamezne besedne vrste. Tretja dimenzija predstavlja vse možne črke na določenem mestu (ali tip lastnosti neke vrste). Pri gradnji vektorja morfoloških podatkov gremo za vsako besedo čez celoten seznam (po vseh dimenzijah) in se ob vsakem znaku v seznamu vprašamo, če morfološki podatki določene besede vsebujejo ta znak na željenem mestu in hkrati, če dana beseda spada v to besedno vrsto. Ko je odgovor na vprašanje pritrdilen, zapišemo enico sicer ničlo.

4.2.3 Vektorska predstavitev izhodov

Izhode nevronske mreže smo zakodirali na dva načina, saj smo problem naglaševanja prav tako razdelili na dva dela, iskanje mesta naglasa in iskanje vrste naglasa (več o tem v podpoglavju 4.3).

Pri naglaševanju so vedno naglašeni samoglasniki a, e, i, o, u ali polglasnik pred r, ko črka ne meji na noben samoglasnik. S pomočjo tega smo poiskali vsa potencialna mesta naglasa vsake besede in naglašene besede v učni množici zakodirali z vektorjem, ki ima na nenaglašenih mestih naglasa zapisane ničle, na naglašenih pa enice (glej tabelo 4.2). Ta pristop se razlikuje od pristopov uporabljenih v drugih delih [32, 15]. Tam so na vhod v algoritem strojnega učenja podali besedo in mesto potencialnega naglasa. Rezultat je povedal, če je beseda naglašena na tistem mestu ali ni. Z našim

```
[['A',                                  [1,
  ['g', 's'],                            1, 0,
  ['p', 'c', 's'],                       1, 0, 0,
  ['m', 'f', 'n'],                       1, 0, 0,
  ['s', 'd', 'p'],                       1, 0, 0,
  ['n', 'g', 'd', 'a', 'l', 'i'],        1, 0, 0, 0, 0, 0,
  ['-', 'n', 'y']                        0, 1, 0,
 ],
 ['C',                                   0,
  ['c', 's']                             0, 0,
 ],
 ...                                     ...
]                                       ]
```

Slika 4.1: Leva koda prikazuje seznam besednih vrst in možnih črk na različnih mestih. Veliki tiskani črki A in C predstavljata besedni vrsti - pridevnik (adjective) in veznik (conjunction). Začetni črki sledi več seznamov črk. V prvem seznamu za besedno vrsto so zapisane vse črke, ki se lahko nahajajo na drugem mestu morfoloških podatkov. V naslednjem seznamu tiste, ki se lahko nahajajo na tretjem mestu itd. Desna koda je konkreten primer zapisa, ki ga prejme nevronska mreža. Gre za besedo *leden*, katere morfološki podatki so *Agpmsnn*. Za lažje razumevanje pomena vektorja, je slednji zapisan v večih vrsticah, poravnano s črkami, ki jih ponazarja na levi. Celotni vektor vsebuje 140 enic in ničel.

Tabela 4.2: Zapis rezultata iskanja mesta naglasa besede *zamrzniti*, ki je naglašena kot *zamŕzniti*. Beseda je najprej razčlenjena na potencialna mesta naglasa, torej samoglasnike ali r, ki ne meji na drug samoglasnik. Potencialno naglašeni znaki te besede so torej a, r, i in i.

potencialna mesta naglasa	*a*	*r*	*i*	*i*		...
kodiran zapis mesta naglasa	0	1	0	0	0	

Tabela 4.3: Prikaz izhoda mreže, ki išče vrsto naglasa za besedo *zamrzniti*. Nevronska mreža, na vhodu prejme lokacijo naglašenega mesta naglasa (v tem primeru je to število 2, beseda je namreč naglašena na črki r).

tipi naglasov	*ä*	*á*	*ë*	*é*	*ě*	*î*	*í*	*ö*	*ó*	*ô*	*ü*	*ú*	*ŕ*
zapis mesta naglasa	0	0	0	0	0	0	0	0	0	0	0	0	1

pristopom smo v eni iteraciji napovedali vsa mesta naglasa posamezne besede. Ker je število naglašenih mest v besedi lahko različno od ena, smo na zadnjem nivoju nevronske mreže uporabili aktivacijsko funkcijo, ki omogoča klasifikacijo v več razredov (multi-label classification). Uporabili smo sigmoidno funkcijo.

Nevronske mreže za iskanje vrste naglasa iščejo tip vsakega posameznega mesta naglasa posebej (več v podpoglavju 4.3.2). Predvideni rezultati so zato zapisani kot vektor ničel in ene enice, ki ponazarja, kakšen naglas se nahaja na danem mestu (primer v tabeli 4.3).

4.3 Arhitektura rešitve

Iskanje naglasa je v magistrski nalogi razdeljeno na dva problema, iskanje mesta naglasa in iskanje vrste naglasa. Razvili smo dva algoritma za iskanje naglasa. Čeprav vsak izmed njiju naslavlja drug problem, sta arhitekturi globokih nevronskih mrež zelo podobni, saj delata na skoraj identičnih vhodnih podatkih.

Uporabli smo konvolucijske mreže. Čeprav smo načrtovali, da bomo uporabljali tudi rekurenčne, se je izkazalo, da implementacija teh ni smiselna. Pri rekurenčnih mrežah namreč, poleg trenutnih vhodov, tudi predhodni vplivajo na napovedi. Ker učimo nevronske mreže na slovarju, zaporedje vhodnih besed ni pomembno in rekurenčne mreže ne bi imele želenega učinka.

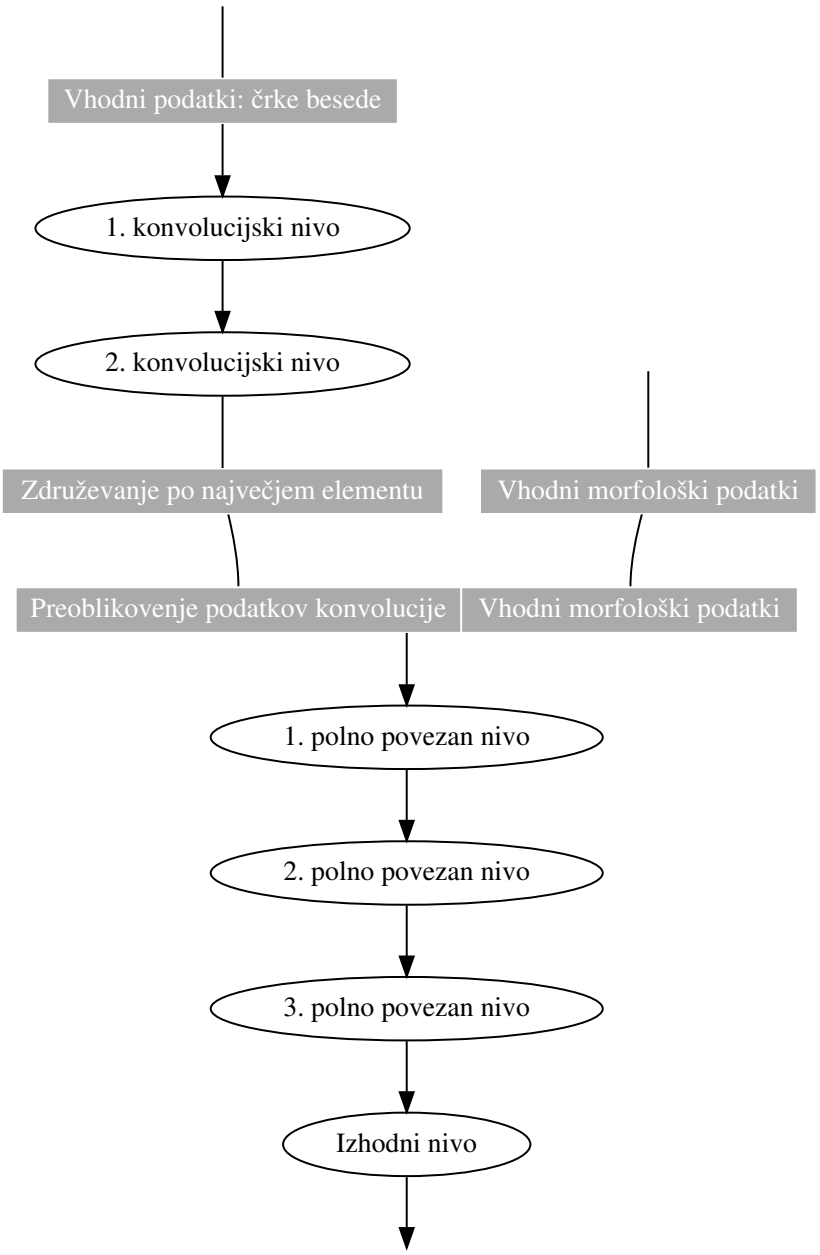
4.3.1 Iskanje mesta naglasa

Preizkusili smo množico različnih arhitektur globokih konvolucijskih nevronskih mrež z različnimi parametri. Za najboljšo se je izkazala mreža, ki se je začela z dvema nivojema konvolucije (slika 4.2). Nevronski mreži smo podali matriko črk besede. Vsak nevron prvega nivoja konvolucije je na vhodu prejel tri zaporedne črke vhodne besede. Ker nivoji konvolucije skušajo iskati vzorce iz podatkov (več o konvoluciji v podpoglavju 3.4), je naš prvi nivo na nek način skušal zajeti podatke o zlogih, ki so približno enako dolgi. Število lastnosti (konvolucijskih filtrov), ki jih je razpoznaval prvi nivo konvolucije je bilo 115. Manjše število je prineslo slabše rezultate, večje pa ni prineslo opaznih razlik.

Naloga drugega nivoja konvolucije je iskati vzorce nad deli besed (lahko bi rekli zlogi) iz prvega nivoja. Najboljše število razpoznanih lastnosti na tem nivoju je bilo 46. Oba nivoja konvolucije sta kot aktivacijsko funkcijo uporabila pragovno linearno funkcijo (ReLU - glej podpoglavje 3.1).

Nivojema konvolucije je sledilo združevanje po največjem elementu (več o tem v podpoglavju 3.5). S tem smo zmanjšali število povezav, tako da smo izbrali le bolj pomembne. Pomembnejše povezave smo izbirali izmed dveh zaporednih.

Nadaljevali smo s polno povezanimi nivoji. Podatke, ki smo jih dobili z združevanjem po največjem elementu iz konvolucijskih nivojev, smo preoblikovali v vektorski zapis. Tem vrednostim smo dodali še morfološke podatke besed, tako da smo jih dodali vektorju konvolucijskih nivojev. Združen vektor smo podali prvemu izmed treh polno povezanih nivojev, ki so vsi vsebovali



Slika 4.2: Shema arhitekture globoke nevronske mreže za napovedovanje mesta naglasa.

Tabela 4.4: Primerjava napovedi globokih nevronskih mrež za iskanje lokacije naglasa z in brez morfoloških podatkov na validacijski množici.

Napovedni model	Klasifikacijska točnost (CA)
Mreža brez morfoloških podatkov	84,28 %
Mreža z morfološkimi podatki	86,62 %

po 256 nevronov. Vsi trije nivoji so uporabljali pragovno linearno funkcijo. Po polno povezanih nivojih smo uporabili tehniko izpadanja, tako da smo pri učenju vsakič izpustili 30 odstotkov nevronov (več o tej tehniki v podpoglavju 3.6).

Polno povezane nivoje smo zaključili z izhodnim nivojem. Ta kot aktivacijsko funkcijo uporablja sigmoidno funkcijo (predstavljena v podpoglavju 3.1), ker kot rezultat vrača vrednosti med 0 in 1. Če je vrnjena vrednost na nekem mestu možnega naglasa manjša od polovice, tisto mesto ni naglašeno, sicer je. Na zadnjem nivoju nevronske mreže je 10 nevronov, saj je to maksimalno število potencialnih mest naglasa ali največje število zlogov katerekoli besede iz naše podatkovne množice.

Ovrednotili smo dve nevronski mreži, ki uporabljata arhitekturo, opisano v tem podpoglavju (tabela 4.4). Nevronski mreži se razlikujeta po tem, da prva med učenjem upošteva morfološke podatke, druga pa ne. Rezultati kažejo, da nevronska mreža z morfološkimi podatki bolje napove mesto naglasa. To potrjuje domneve iz podpoglavja 2.3, da morfološki podatki vplivajo na naglasno mesto.

4.3.2 Iskanje vrste naglasa

Arhitektura za iskanje vrste naglasa je skoraj enaka arhitekturi uporabljeni za iskanje lokacije naglasa. Zato predstavimo le razlike.

Ta globoka nevronska mreža napoveduje vrsto naglasa na vseh mestih, ki jih je mreža za iskanje mesta naglasa označila kot naglašene. To pomeni, da

če je neka beseda naglašena na dveh mestih, dobimo na vhodu v mrežo za določevanje tipa naglasa dve aktivni vrednosti.

Mreži se nekoliko razlikujeta po predstavitvi vhodov v nevronske mreže. Vektorju, ki je prej vseboval le morfološke podatke, dodamo še mesto naglasa, ki nas zanima. Če je maksimalno število potencialno naglašenih mest enako 10 in nas zanima vrsta naglasa drugega zloga, vektorju z morfološkimi podatki dodamo vektor dolžine 10. Ta ima na vseh mestih ničle, razen na drugem mestu, kjer je enica. Če je beseda naglašena na dveh mestih, nevronski mreži dvakrat pošljemo iste podatke, z razlikami le pri tem vektorju.

Od mreže za iskanje lokacije se mreža za iskanje vrste naglasa razlikuje tudi pri izhodu. Podatkovna množica vsebuje 13 različnih naglasov (več o tem v podpoglavju 4.2.3), kar pomeni, da ima izhodni nivo te nevronske mreže 13 nevronov. Proces določanja vrste naglasa besede je nekoliko bolj kompleksen od določanja mesta. Vsaka potencialno naglašena črka lahko postane le ena izmed naglašenih oblik te črke in ne drugih. Na primer, ko je beseda naglašena na črki r, ta vedno postane $ŕ$, črka e lahko postane $ë$, $é$ ali $ě$ itd. To upoštevamo tudi pri napovedovanju vrste naglasa. Napovedni model za neko mesto naglasa v besedi vedno vrne tisto obliko črke, ki ji napovedi pripisujejo največjo verjetnost. Če gledamo vrsto naglasa črke a, bo algoritem preveril le verjetnosti za tipa $ä$ in $á$ in ne vseh.

Čeprav imamo lahko na vhodu več pojavitev iste besede (ko ima beseda več naglasov), evalvacija poteka enako kot pri mrežah za iskanje mesta naglasa (več o tem v podpoglavju 4.1). Če besedo pravilno naglasimo le na enem od dveh mest, celotno besedo štejemo kot napačno. Ko je beseda dvakrat naglašena napačno, pa to štejemo samo enkrat.

Ovrednotili smo dve globoki nevronski mreži za napovedovanje vrste naglasa (tabela 4.5). Tudi tukaj smo primerjali napovedi mrež z in brez morfoloških podatkov. Izkazalo se je, da morfološki podatki vplivajo tudi na uspešnost napovedi mreže za napovedovanje vrste naglasa.

Tabela 4.5: Primerjava napovedi nevronskih mrež za iskanje vrste naglasa z in brez morfoloških podatkov nad validacijsko množico.

Napovedni model	CA
Nevronska mreža brez morfoloških podatkov	93,99 %
Nevronska mreža z morfološkimi podatki	96,03 %

Tabela 4.6: Primerjava napovedi nevronskih mrež za iskanje lokacije naglasa s petimi polno povezanimi nivoji (velikosti 512-1024-2048-1024-512) in tremi polno povezanimi nivoji (z velikostjo 256-512-256) na validacijskih podatkih.

Napovedni model	CA
Mreža s petimi polno povezanimi nivoji	86,26%
Mreža s tremi polno povezanimi nivoji	87,24%

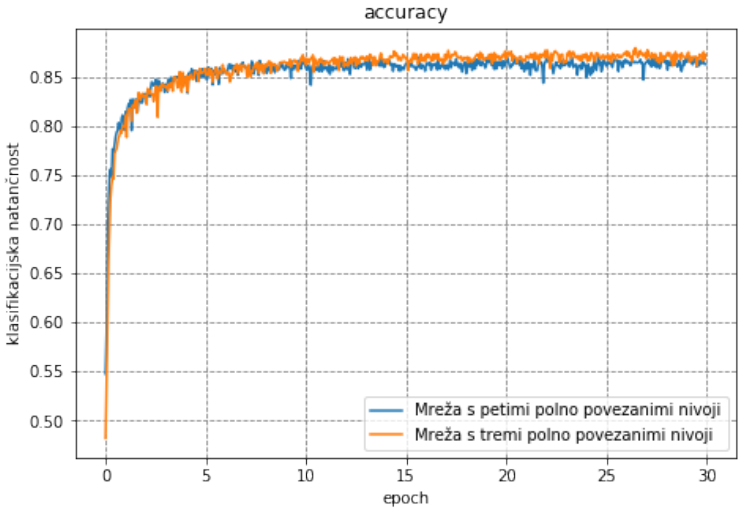
4.4 Iskanje čimboljše arhitekture

Med iskanjem čimboljše arhitekture smo preizkusili več kombinacij velikosti posameznih nivojev, števila nivojev in aktivacijskih funkcij.

Arhitekturo smo testirali tako, da smo spremenili nek parameter nevronske mreže in napovedi na validacijski množici spremenjene mreže primerjali z originalno. Napovedi dveh ali več mrež smo primerjali s klasifikacijsko točnostjo (CA) besed (primer v tabeli 4.6).

Ker so primerjave lahko nepravilne zaradi variance, smo napovedi primerjali tudi tako, da smo jih izračunali večkrat med samim učenjem in primerjali grafa napovedi (slika 4.3)

Primerjava mrež v tabeli in grafu je le prikaz ideje iskanja čimboljše nevronske mreže. Končno arhitekturo smo poiskali tako, da smo proces uporabili nad večimi parametri. Najprej smo spreminjali število polno povezanih nivojev, njihove velikosti in parametre izpadanja, nato pa število in velikost konvolucijskih nivojev.



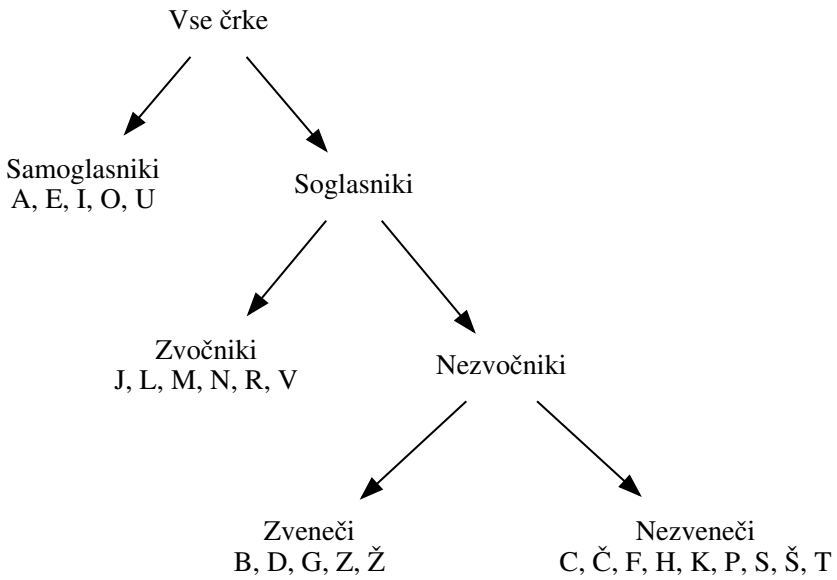
Slika 4.3: Prikaz uspešnosti učenja na validacijski množici za mreži iz tabele 4.6.

4.5 Izboljšave pri predprocesiranju podatkov

Predstavimo večje spremembe predprocesiranja podatkov za izboljšavo napovedi mrež. Nekatere izboljšave smo izvedli le za problem iskanja mesta naglasa. V nalogi smo se nekoliko bolj osredotočili na iskanje mesta naglasa kot iskanje tipa naglasa. Naš namen je bil namreč izboljšati napovedi obeh algoritmov skupaj, zato je bilo potrebno bolj izboljšati slabšega. Mreža za določanje vrste naglasa je namreč vračala boljše napovedi kot mreža za določevanje mesta naglasa.

4.5.1 Dodane lastnosti črk

Črke so med seboj različne, vseeno pa so si nekatere črke bliže od drugih. Domnevali smo, da se bodo nevronske mreže lažje učile, če jim pri vsaki črki povemo, v katero skupino črk spada. Najbolj osnovna delitev črk, ki smo jo uporabili, je delitev na samoglasnike in soglasnike. Soglasnike smo naprej



Slika 4.4: Delitev črk v slovenščini.

Tabela 4.7: Zapis lastnosti črke *b*.

samoglasnik	*soglasnik*	*zvočnik*	*nezvočnik*	*zveneč*	*nezveneč*
0	1	0	1	1	0

delili na zvočnike in nezvočnike, slednje pa še na zveneče in nezveneče (glej sliko 4.4).

Te lastnosti črk smo zapisali v vektor enic in ničel, katerega dolžina je 6 (primer tabela 4.7). Prva dva elementa vektorja nam povesta, če je črka samoglasnik ali soglasnik, tretji in četrti povesta, če gre za zvočnik ali nezvočnik, zadnja dva pa, če je črka zveneč ali nezveneč nezvočnik. Te lastnosti so bile pripete vsaki vrstici matrične predstavitve vhodnih podatkov (vsaki črki na vhodu).

Rezultati so pokazali, da dodatne lastnosti črk nimajo velikega vpliva na izboljšavo napovedi (tabela 4.8). Sklepamo da se je nevronska mreža sama

Tabela 4.8: Primerjava klasifikacijske točnosti mrež za iskanje lokacije naglasa z in brez dodatnih lastnosti črk.

Napovedni model	CA
Mreža brez dodatnih lastnosti črk	86,62 %
Mreža z dodatnimi lastnostmi črk	87,08 %

sposobna naučiti te lastnosti, brez našega posredovanja.

4.5.2 Obrnjena beseda na vhodu

Nevronske mreže razpoznavajo vzorce. Čeprav pri konvolucijskih nivojih lokacija nekega vzorca ni tako pomembna, to ne velja za polno povezane nivoje. Pri problemu določanja mesta naglasa se prvo potencialno mesto naglasa besede primerja z vzorci istega mesta naglasa, drugo z drugimi itd. Posledično je pomembna oddaljenost naglasa od začetka besede.

Črke besede lahko zapišemo tudi v obratnem vrstnem redu (glej tabelo 4.9). V tem primeru oddaljenost naglasa od začetka besede ni več pomembna, saj postane pomembna oddaljenost naglasa od konca besede. Z obrnjeno besedo, tako gledamo koliko je naglas oddaljen od konca. Ko smo obrnili besede na vhodu, smo enako naredili še za izhode (primer v tabeli 4.10).

Obrnjene vhodne črke so izboljšale rezultat za več odstotkov (glej tabelo 4.11). To nakazuje, da je v slovenščini naglas bolj vezan na oddaljenost zloga od konca besede, kot od začetka.

4.5.3 Predstavitev vhodov s približki zlogov

V prejšnih pristopih smo besede vedno razdelili na črke, te zakodirali in jih v takšni obliki poslali na vhod nevronskim mrežam. Če želimo iskati vzorce v besedah, moramo besedo razdeliti na manjše dele, a ni nujno, da so to črke. Besedo smo poizkusili razdeliti na nekoliko večje enote od črk, na zloge.

Izkaže se, da ne obstajajo pravila, ki bi jih lahko enostavno zapisali kot

Tabela 4.9: Način gradnje matrike za besedo *abeceda*, ko je beseda obrnjena.

črke vhodne besede	brez znaka	*a*	*b*	*c*	*č*	*d*	*e*	...
a	0	1	0	0	0	0	0	
d	0	0	0	0	0	1	0	
e	0	0	0	0	0	0	1	
c	0	0	0	1	0	0	0	
e	0	0	0	0	0	0	1	
b	0	0	1	0	0	0	0	
a	0	1	0	0	0	0	0	
	1	0	0	0	0	0	0	
...	1	0	0	0	0	0	0	

Tabela 4.10: Zapis obrnjenega rezultata iskanja mesta naglasa besede *abecéda*.

potencialna mesta naglasa	*a*	*e*	*e*	*a*		...
kodiran zapis mesta naglasa	0	1	0	0	0	

Tabela 4.11: Primerjava klasifikacijske točnosti nevronskih mrež za iskanje lokacije naglasa z obrnjenimi in neobrnjenimi vhodnimi črkami.

Napovedni model	CA
Mreža z neobrnjenimi vhodnimi črkami	87,08 %
Mreža z obrnjenimi vhodnimi črkami	91,37 %

zlogovalnik [20]. V navedenem delu so za rešitev uporabili strojno učenje. Ker zlogovalnik ni del tega magistrskega dela, manjka pa nam tudi podatkovna množica, s pomočjo katere bi se naučili zlogovati, smo besede razdelili na približke zlogov.

Deljenje besed in zlogovanje sta sicer drugačna, a vseeno podobna problema. Najprej smo preizkusili delilnik besed, ki ga uporabljata latex in open office. Delilnik smo skušali prilagoditi, da bi besede ločil na zloge. Žal je veliko zlogov vsebovalo več kot en samoglasnik ali nobenega. Ta problem bi sicer lahko rešili tako, da bi zloge brez samoglasnika prilepili sosednjim, tiste z večim pa razbili. Tega nismo naredili, ker je bilo število različnih zlogov že samo po sebi veliko, omenjen način pa bi to število še povečal (učenje bi posledično potekalo počasneje).

Namesto tega smo naredili programski zlogovalnik (pravila so zapisana v [31]). Zlogovalnik večkrat napačno loči besede, vseeno pa vsak zlog vsebuje en samoglasnik ali črko r, ko ta ne meji na noben samoglasnik. Kljub mnogim napačnim zlogom, smo preizkusili kako nevronske mreže delujejo na takšnih približkih zlogov. Zloge smo najprej kodirali v obliko primerno za učenje nevronske mreže.

Ideja predstavitve zlogov je podobna predstavitvi črk (glej tabelo 4.12). Razlikujeta se v tem, da seznam približkov zlogov vsebuje 5168 kombinacij črk, kar je precej več od seznama črk. Posledično so vhodi v nevronske mreže precej večji, računanje pa počasnejše.

Arhitektura nevronskih mrež se pri uporabi te predstavitve nekoliko spremeni. Namesto dveh nivojev konvolucije (kjer smo s prvim skušali predstaviti zloge), smo uporabili le en nivo, ki pa je večji od tistih, ki so bili uporabljeni v arhitekturi s črkami (glej podpoglavje 4.3.1). Nivo razpoznava 200 različnih lastnosti zlogov.

Približki zlogov niso izboljšali rezultatov (slika 4.13). Morda bi se rezultati izboljšali, če bi namesto približkov uporabili dejanske zloge. Slaba stran predstavitve z zlogi je, da smo vse zloge delali na enak način in algoritem, za

Tabela 4.12: Način gradnje matrike zlogov za besedo *abeceda*. Besedo najprej razdelimo na približke zlogov, ti pa so zapisani v obratnem vrstnem redu.

približki zlogov vhodne besede	*a*	...	*be*	...	*ce*	...	*da*	...
da	0		0		0		1	
ce	0		0		1		0	
be	0		1		0		0	
a	1		0		0		0	
...	0		0		0		0	

Tabela 4.13: Primerjava klasifikacijske točnosti globokih nevronskih mrež za iskanje lokacije naglasa z vhodnimi črkami in vhodnimi približki zlogov.

Napovedni model	CA
Mreža s črkami na vhodu	91,37 %
Mreža s približki zlogov na vhodu	85,93 %

razliko od pravih zlogov, ni prepoznaval predpon, sestavljenih besed ipd. Ne glede na to, da so rezultati slabši, se pristop precej razlikuje od prvotnega, zato smo ga uporabili pri ansambelskih metodah.

4.5.4 Predstavitev vhodov s črkovanimi približki zlogov

Vhodne besede smo dosedaj razčlenili na črke in na približke zlogov. Preizkusili smo še tretji pristop, s katerim smo skušali združiti prejšnja dva.

Poleg podatkov o zlogih smo nevronski mreži posredovali še podatke o črkah (glej tabelo 4.14). Besede smo najprej razdelili na približke zlogov in jih zapisali v obratnem vrstnem redu. Namesto da bi v vhodno matriko besed zapisali indekse zlogov iz seznama, smo vsak zlog razčlenili na njegove črke in jih podali nevronski mreži.

Rezultati črkovnih približkov zlogov so slabši kot rezultati nad črkovnimi

Tabela 4.14: Način gradnje matrike približkov zlogov, zapisanih s črkami, za besedo *abeceda*. Beseda je najprej razdeljena na približke zlogov, nato so ti zapisani v obratnem vrstnem redu, nazadnje pa še razčlenjeni na črke, ki so dejansko zapisane v matriki. Ta tabela je poenostavljena, saj je v njej maksimalno število črk v enem zlogu enako 2, v dejanski tabeli je to število večje. Skupna dolžina zapisa posameznega zloga je v tem načinu še vedno precej manjša kot tista, kjer smo shranjevali indeks zloga.

	abeceda prve črke zloga							*abeceda druge črke zloga*						
približki zlogov besede	*a*	*b*	*c*	*č*	*d*	*e*	...	*a*	*b*	*c*	*č*	*d*	*e*	...
da	0	0	0	0	1	0		1	0	0	0	0	0	
ce	0	0	1	0	0	0		0	0	0	0	0	1	
be	0	1	0	0	0	0		0	0	0	0	0	1	
a	1	0	0	0	0	0		0	0	0	0	0	0	
...	0	0	0	0	0	0		0	0	0	0	0	0	

vhodnimi podatki in boljši od zlogovnih (tabela 4.15). Morda bi bili boljši, če bi namesto približkov zlogov uporabili prave zloge. Tudi ta pristop smo kasneje uporabili v ansamblih.

Tabela 4.15: Primerjava klasifikacijske točnosti mrež za iskanje lokacije naglasa z vhodnimi črkami, vhodnimi približki zlogov in črkovanimi vhodnimi približki zlogov.

Napovedni model	CA
Mreža s črkami na vhodu	91,37 %
Mreža s približki zlogov na vhodu	85,93 %
Mreža s črkovnimi približki zlogov na vhodu	90,18 %

Tabela 4.16: Primerjava klasifikacijske točnosti nevronskih mrež za iskanje vrste naglasa s črkami na vhodu. V prvem primeru so predstavljeni rezultati brez izboljšav (z morfološkimi podatki), v drugem pa rezultati z obrnjenimi vhodi in dodanimi lastnostmi črk.

Napovedni model	CA
Mreža brez izboljšav	96,03 %
Mreža z izboljšavami	96,25 %

Tabela 4.17: Primerjava klasifikacijske točnosti mrež za iskanje vrste naglasa z vhodnimi črkami, približki zlogov in črkovnimi približki zlogov.

Napovedni model	CA
Mreža s črkovnim vhodom	96,25 %
Mreža s približkom zlogov	95,57 %
Mreža s črkovnim približkom zlogov	96,41 %

4.5.5 Izboljšave predprocesiranja podatkov za določanje tipa naglasa

Iste izboljšave, ki smo jih uporabili pri iskanju lokacije naglasa smo preizkusili tudi na problemu določanja tipa naglasa na črkah (glej tabelo 4.16). Rezultati so se minimalno izboljšali.

Na problemu določanja tipa naglasa smo preizkusili tudi globoke nevronske mreže z vhodi iz približkov zlogov in kombinacijo teh s črkami (slika 4.17). Mreža, ki ima na vhodu približke zlogov se je izkazala za malce slabšo od tiste s črkami na vhodu, mreža s približki črkovnih zlogov pa za zelo podobno.

4.6 Analiza napak

Pri analizi napak smo ročno pregledali napačne napovedi na validacijski množici in med njimi skušali najti značilne vzorce. Najprej predstavljamo izsledke analize pri napovedovanju mesta naglasa, potem pa še za napovedo-

vanje vrste naglasa.

4.6.1 Napake pri določanju mesta naglasa

Analizo napak pri določanju mesta naglasa smo izvedli na validacijski množici za tri različne algoritme: nad mrežami, ki kot vhod prejemajo črke, približke zlogov in kombinacijo obeh. Najprej predstavljamo napake, ki so se pojavile pri vseh treh pristopih.

Pri določanju mesta naglasa težave povzročajo besede privzete iz drugih jezikov. Takšne napake so bile pričakovane, saj se pravila naglašanja med jeziki razlikujejo. Primeri tovrstnih napak so *belgíjskem* (pravilno *bélgijskem*), *angélček* (*ángelček* - prihaja iz grščine) in *profesórjevo* (*profésorjevo* - iz latinščine) [27].

Opazili smo več napačno naglašenih besed, v katerih se pojavljajo druge besede. Namesto da bi algoritem v takih primerih gledal na naglas celotne besede, naglaša notranjo besedo. Primere teh napak smo opazili predvsem pri sestavljenih besedah - *hládnokŕvna* (pravilno je *hladnokŕvna*) iz *hládno* in *kŕvna*.

Več besed je napačno naglašenih, ker so podobne besede naglašene na drugem mestu. Primera takšnih besed sta *škodíla*, pravilno *škódila* (podobni besedi sta *budíla* ter *rodíla*), in *oceníte*, pravilno *océnite* (podobni besedi sta *tesníte* ter *poceníte*).

Zanimivo je, da je algoritem več besed naglasil, kot jih naglašamo v narečjih ali kot so naglašene starinsko. Primer takšne besede je *jezéru*, ki je pravilno naglašena kot *jézeru*.

Analiza napak, ki so se pojavile samo pri določeni mreži in ne na preostalih dveh, ni razkrila očitnih razlik, saj vse mreže delajo podobne napake. Pokazalo se je, da vse nevronske mreže delajo napake na različnih daljših besedah, ki jih druge uvrstijo pravilno. To nakazuje, da pri teh besedah mreže niso prepričane, kje se nahaja mesto naglasa.

4.6.2 Napake pri določanju tipa naglasa

Pri tej analizi smo prav tako najprej analizirali napake, ki so jih naredile vse tri mreže (tiste ki kot vhod prejemajo črke, približke zlogov in kombinacijo obeh). Pokazalo se je, da je problem iskanja kvalitete naglasov (širok ali ozek *o* in *e*) hujši od iskanja kvantitete naglasov (kratki ali dolgi). Razlog za to je najverjetneje to, da se kratki naglasi ponavadi nahajajo na končnicah besed. Poleg tega je dolgih naglašenih glasov v podatkovni množici, razen pri *e*-ju in *a*-ju, približno 100-krat več kot kratkih. Največ napak smo opazili pri naglaševanju črke *e*, večinoma med širokimi in ozkimi dolgimi naglasi. Sledila je črka *o*, prav tako večinoma s problemom naglaševanja ozkih in širokih dolgih naglasov. Napak pri ostalih črkah skoraj nismo zaznali.

Med analizo posameznih mrež nismo opazili vzorcev napak, po katerih bi se razlikovale.

4.7 Ansamblske metode

Namesto uporabe enega algoritma strojnega učenja za napovedovanje, lahko napovedujemo tudi z večimi, nato pa z ansambelsko metodo združimo njihove napovedi [18].

V delu smo zaradi enostavnosti pristopa uporabljali neobteženo povprečenje (equal weight averaging). Metoda deluje tako, da nad vhodnimi podatki uporabimo več napovednih modelov. Rezultate združimo z izračunom povprečja vsake napovedi iz vseh napovednih modelov.

Pristop smo uporabili ločeno za iskanje lokacije naglasa in iskanje vrste naglasa. Pri obeh problemih smo za izračun rezultata uporabili tri napovedne modele (mreže, ki so imele vhodne podatke predstavljene kot črke, približke zlogov in kombinacijo obeh pristopov).

Rezultati iskanja mesta naglasa so predstavljeni v tabeli 4.18. Uporabljena ansambelska metoda pravilno napove mesta naglasa približno 1,5 % več besed kot najboljša posamezna nevronska mreža.

Tabela 4.18: Delež pravilno določenih mest naglasov besed treh nevronskih mrež in ansambelske metode.

Napovedni model	CA
Mreža s črkovnim vhodom	90,12 %
Mreža s približkom zlogov	87,51 %
Mreža s črkovnim približkom zlogov	89,25 %
Neuteženo povprečenje zgornjih treh mrež	91,71 %

Tabela 4.19: Delež pravilno določenih vrst naglasov treh mrež in ansambelske metode.

Napovedni model	CA
Mreža s črkovnim vhodom	96,25 %
Mreža s približkom zlogov	95,57 %
Mreža s črkovnim približkom zlogov	96,41 %
Neuteženo povprečenje zgornjih treh vhodov	97,15 %

Vrsto naglasa je ansambelska metoda izboljšala pri približno 0,7 % besed (tabela 4.19).

4.8 Končni rezultati nad testnimi podatki

Mreže, s katerimi smo testirali testne podatke, so bile naučene na združenih učnih in validacijskih podatkih. Napovedi mesta naglasa na testnih podatkih so nekoliko slabše kot na validacijskih (tabela 4.20). Rezultati vseh treh napovedi so nekoliko nižji. Največja razlika je pri mreži s črkovnimi vhodi. Tudi napovedi vrste naglasa so se na testnih podatkih poslabšale (tabela 4.21). Najbolj opazno se je spremenila napoved mreže, ki ji vhod predstavimo s približki zlogov.

Končne rezultate naših algoritmov za naglaševanje besed smo dobili tako, da smo izračunali delež besed iz testne množice, ki sta jih pravilno napovedala model za iskanje mesta naglasa in model za iskanje vrste naglasa (tabela

Tabela 4.20: Delež pravilno napovedanih mest naglasov treh nevronskih mrež in ansambelske metode na testnih podatkih.

Napovedni model	CA
Mreža s črkovnim vhodom	88,54 %
Mreža s približkom zlogov	85,67 %
Mreža s črkovnim približkom zlogov	87,98 %
Neuteženo povprečenje zgornjih treh mrež	90,24 %

Tabela 4.21: Delež pravilno napovedanih vrst naglasov treh nevronskih mrež in ansambelske metode na testnih podatkih.

Napovedni model	CA
Mreža s črkovnim vhodom	96,06 %
Mreža s približkom zlogov	94,00 %
Mreža s črkovnim približkom zlogov	95,98 %
Neuteženo povprečenje zgornjih treh mrež	96,37 %

4.22). Rezultati kažejo, da se naglase slovenskih besed da določati s pomočjo globokih nevronskih mrež.

S pomočjo globokih nevronskih mrež smo izboljšali rezultate dosedanjih pristopov in pravilno napovedali 87,65 % besed. Marinčič in sodelavci (leta 2009) so pravilno napovedali 80,34 % besed testne množice z uporabo boostinga in hevristik [15], Tušar in sodelavci pa (leta 2005) 81,9 % z markovskimi verigami [32]. Pri tem je potrebno omeniti še razlike pri testiranju

Tabela 4.22: Delež besed iz testne množice, s pravilno napovedanimi mesti in vrstam naglasov.

Napovedni model	CA
Mreža s črkovnim vhodom	85,65 %
Mreža s približkom zlogov	80,71 %
Mreža s črkovnim približkom zlogov	85,01 %
Neuteženo povprečenje zgornjih treh mrež	87,65 %

med omenjenimi deli in našim. V drugih delih je bila natančnost izračunana na podlagi prečnega preverjanja (cross validation), kjer so bili podatki razdeljeni na tri dele, enega testnega in dva učna v vsaki iteraciji. V našem delu so bili podatki ločeni že v samem začetku. Testni podatki so predstavljali 10 % podatkov in jih je bilo malo čez 50 000. Naše delo se razlikuje tudi v določanju vrste naglasa. Omenjeni deli namreč naglašujeta podatke le po kvaliteti, ne pa tudi po kvantiteti, kar je narejeno v našem delu.

4.9 Praktična uporaba modelov

Arhitekturi nevronskih mrež za iskanje mesta naglasa in iskanje vrste naglasa, ki sta vrnili najboljše napovedi, smo naučili na celotni podatkovni množici in jih uporabili za dva realna problema. Označili smo slovenski oblikoslovni leksikon - Sloleks [1] in modela uporabili v programu za naglaševanje sklopljenih besedil.

4.9.1 Naglasitev Sloleksa

Sloleks je leksikon, ki vsebuje okoli 100.000 različnih lem in skoraj 3 milijone besednih oblik. Za primerjavo, naša podatkovna množica ima okrog 20.000 lem in nekaj čez pol milijona besednih oblik. Eden izmed ciljev tega magistrskega dela je bil naglasiti besede iz te zbirke. Pri realizaciji smo nevronske mreže naučili z vsemi podatki naše podatkovne množice.

Poseben problem je predstavljal morfološki zapis. Podatkovna množica, iz katere sta se nevronski mreži učili je imela morfološke podatke zapisane v angleškem formatu MULTEXT-East verzija 3, morfološki podatki v Sloleksu pa so zapisani v slovenskem formatu MULTEXT-East verzija 4.

Prevod iz slovenskega zapisa v angleškega ni težaven, saj imajo vse črke na vseh mestih slovenskega zapisa točno določene črke v angleščini. Napisali smo program, ki je slovenske črke pretvoril v angleške.

Več težav smo imeli s pretvorbo formata verzije 3 v verzijo 4. Večina lastnosti je sicer ostala enaka ali pa se je le zamenjalo mesto znaka, nekatere pa so se zelo spremenile. Primer spremembe je vrsta pridevnika. Pridevniki so v verziji 3 lahko kakovostni, svojilni ali vrstni. V verziji 4 pridevniki z lastnostjo kakovostni in vrstni postanejo splošni, doda pa se povsem nova lastnost vrste pridevnika - deležniški pridevnik. Vrste pridevnika so v verziji 4 tako lahko splošna, svojilna ali deležniška.

Ker nevronske mreže niso naučene na nekaterih lastnostih in posamezne lastnosti izginejo, so rezultati na tovrstni množici verjetno slabši, kot bi bili, če bi bila formata enaka. Leksikona pred nami ni naglasil nihče, zato ne moremo oceniti napake.

4.9.2 Naglaševanje sklenjenih besedil

Drugi cilj magistrske naloge je bil naglaševalnik sklenjenih besedil. Največ težav smo imeli s tem, kako označiti sklenjen tekst z morfološkimi informacijami. Uporabili smo obstoječ označevalnik ReLDI [21]. Deluje tako, da na vhod damo poved, označevalnik pa določi morfološke podatke vsake posamezne besede (v formatu MULTEXT-East verzija 4 - imeli smo enake težave kot so opisane v prejšnjem podpoglavju).

Naglaševalnik deluje tako, da s pomočjo označevalnika najprej pridobimo morfološke oznake vsake besede, nato vse velike tiskane črke spremenimo v male in izpustimo znake, kot so vejice, pike, oklepaji, zaklepaji ipd. Besede z njihovimi morfološkimi podatki pošljemo nevronski mreži za iskanje mest naglasov in mreži za iskanje vrst naglasov. Vrnjene besede spremenimo v prvotno stanje (besedam vrnemo velike tiskane črke) in ponovno vstavimo ločila. Rezultat je naglašeno sklenjeno besedilo.

Tudi pri tej nalogi nismo mogli kvantitativno oceniti uspešnosti algoritma, saj nismo imel na voljo nobenega predhodno naglašenega besedila. Rezultati so pokazali, da je napačno naglašenih nekaj besed, ki so večkrat uporabljene v vezanih besedilih (rezultat na sliki 4.5). Takšne rezultate smo dobili, ker smo

Izbrúhi na sóncu só žé vëčkrat pokazáli zóbe nášim satelítom, poslédično nášim mobílnim telefónom, navigáciji, célo eléktričnemu omréžju. Á vesóljskega vreména šë në móremo napovédati – kakó bî ga láhko, se tá téden na Blédu pogovárja okóli 70 znánstvenikov Evrópske vesóljske agéncije, ki jé sebój pripeljála svôjo näjvéčjo ikóno, británca Mátta Taylorja.

Slika 4.5: Primer besedila, ki ima določena mesta naglasa.

modele naučili s podatki iz slovarja in nismo upoštevali dejanskega števila pojavitev besed v besedilih.

Če bi želeli izboljšati napovedni model za naglaševanje povezanega teksta, bi med učenjem lahko upoštevali pogostost pojavitve besed. Dobili bi model, ki bi bil slabši za naglaševanje slovarja, a boljši za naglaševanje vezanega besedila.

Poglavje 5

Zaključek

V delu smo poizkusili napovedati mesta in vrsto naglasov slovenskih besed s pomočjo globokih nevronskih mrež. Rezultati so pokazali, da so konvolucijske nevronske mreže primerne za reševanje teh problemov, saj so na testnih podatkih pravilno napovedale velik del naglasov besed.

Preizkusili smo več različnih arhitektur nevronskih mrež z različnimi parametri. Hkrati smo skušali izboljšati napovedi tudi s spremembami zapisa vhodnih podatkov. Te smo nevronskim mrežam podali v obliki črk, približkov zlogov in kombinaciji obeh pristopov. Rezultate smo izboljšali tudi s tem, da smo obrnili vhode v nevronsko mrežo in črkovnim vhodom dodali podatke o lastnostih črk. Nad nevronskimi mrežami smo uporabili ansambelski pristop, ki je izboljšal napovedi.

S pomočjo napovednih modelov za mesto in vrsto naglasa smo naglasili slovar besednih oblik Sloleks in implementirali program za naglaševanje sklopljenih besedil.

5.1 Strokovni prispevki

Glavni prispevek dela je ugotovitev, da so globoke nevronske mreže sposobne napovedati mesto in vrsto naglasa. Združene v ansambel so pravilno napo-

vedale 87,65 % besed, kar je bolje od dosedanjih pristopov [15, 32].

Strokovna prispevka predstavljata tudi naglašena množica podatkov Sloleks (do sedaj ni obstajala odprtodostopna množica naglašenih podatkov) in program za naglaševanje besedil.

5.2 Kritična analiza

Podatkovna zbirka je bila v nalogi razdeljena na tri množice, učno, validacijsko in testno. Vse besede z enako lemo so se nahajale v isti množici, da bi zmanjšali pretirano prilagajanje, a se temu nismo v celoti izognili. V podatkovni zbirki se nahaja precej podobnih besed, katerih leme se le malo razlikujejo (primer podobnih lem so *carinikov*, *carinik* in *carinski*). Ker so takšne besede velikokrat naglašene na podobnih mestih in s podobnim naglasom, je mogoče, da so prikazani rezultati nekoliko boljši kot so v resnici.

Ker je iskanje dobrih parametrov precej počasno, v nalogi vsem problemom nismo posvetili dovolj časa. Pri globokih nevronskih mrežah ne moremo trditi, da smo našli najboljšo arhitekturo in parametre, saj je kombinacij, ki bi jih morali preizkusiti, preveč. Največ časa smo porabili, da smo sestavili čim boljšo mrežo za iskanje mesta naglasa s pomočjo črk. Posledično smo za ta problem bolj izboljšali rezultate kot za ostale.

5.3 Ideje za izboljšave

Za izboljšanje celotne rešitve naglaševanja, lahko izboljšamo iskanje mesta naglasa ali določanje tipa naglasa. Pri določanju vrste naglasa bi lahko, namesto ene nevronske mreže za določanje tipa poiskusili problem razdeliti na več mrež. Lahko bi zgradili 5 različnih mrež, po eno za vsak samoglasnik (če je naglašena črka r, je vrsta naglasa vedno ista). Na ta način bi se vsaka mreža bolj osredotočila na razpoznavanje vrste naglasa posameznega samoglasnika, kar bi morda izboljšalo rezultat.

Pri določanju mesta naglasa bi lahko preverjali, če se napovedana kombinacija mest naglasov pojavi tudi v drugih besedah. Če je kombinacija zelo redka, bi lahko kot končni rezultat vrnili podobno kombinacijo, ki je bolj verjetna.

Namesto približkov zlogov, ki smo jih uporabili v tem delu, bi lahko uporabili zloge. Za njihovo določanje bi lahko uporabili algoritme strojnega učenja, a bi morali dobiti podatkovno množico za zlogovanje. Že implementirane nevronske mreže bi lahko naknadno izboljšali z večjo podatkovno množico.

Lahko bi preskusili še druge ansambelske metode, kot sta bagging ali stacking. Z njimi bi lahko združili še več metod strojnega učenja in ne le nevronskih mrež.

Literatura

[1] Š. Arhar, "Učni korpus SSJ in leksikon besednih oblik za slovenščino," v *Jezik in slovstvo*, zv. 1(3–4), str. 43–56, 2009.

[2] S. Armstrong, K. Church, P. Isabelle, S. Manzi, E. Tzoukermann in D. Yarowsky, Eds., *A Comparison of Corpus-Based Techniques for Restoring Accents in Spanish and French Text*, str. 99–120. Dordrecht: Springer Netherlands, 1999.

[3] S. Chennupati, "Hierarchical decomposition of large deep networks," Master's thesis, Rochester Institute of Technology, 2016.

[4] G. G. Chowdhury, "Natural language processing," v *Annual review of information science and technology*, zv. 37(1), str. 51–89, 2003.

[5] R. Collobert in J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," v *Proceedings of the 25th international conference on Machine learning*, str. 160–167, ACM, 2008.

[6] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu in P. Kuksa, "Natural language processing (almost) from scratch," v *Journal of Machine Learning Research*, zv. 12(Aug), str. 2493–2537, 2011.

[7] T. Erjavec, "MULTEXT-East version 3: Multilingual morphosyntactic specifications, lexicons and corpora.," v *Proc. of the Fourth Intl. Conf. on Language Resources and Evaluation*, 2004.

[8] X. Glorot, A. Bordes in Y. Bengio, "Deep sparse rectifier neural networks," v *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, zv. 15, str. 315–323, 2011.

[9] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever in R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," v *arXiv preprint arXiv:1207.0580v1*, 2012.

[10] W. Hu, Y. Huang, L. Wei, F. Zhang in H. Li, "Deep convolutional neural networks for hyperspectral image classification," v *Journal of Sensors*, zv. 2015, str. 1–12, 07 2015.

[11] D. P. Kingma in J. Ba, "Adam: A method for stochastic optimization," v *arXiv preprint arXiv:1412.6980v9*, 2017.

[12] S. Lai, L. Xu, K. Liu in J. Zhao, "Recurrent convolutional neural networks for text classification.," v *Proceedings of the 29th American Association for Artificial Intelligence conference on Artificial Intelligence*, str. 2267–2273, 2015.

[13] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," v *The handbook of brain theory and neural networks*, zv. 3361(10), str. 1995, 1995.

[14] Y. LeCun, Y. Bengio in G. Hinton, "Deep learning," v *Nature*, zv. 521(7553), str. 436–444, 2015.

[15] D. Marinčič, T. Tušar, M. Gams in T. Šef, "Analysis of automatic stress assignment in Slovene," v *Informatica*, zv. 20(1), str. 35–50, 2009.

[16] R. Mihalcea in V. Nastase, "Letter level learning for language independent diacritics restoration," v *Proceedings of the 6th Conference on Natural Language Learning - Volume 20*, (Stroudsburg, PA, USA), str. 1–7, 2002.

[17] M. A. Nielsen, *Neural networks and deep learning*. Determination Press, 2015.

[18] M. Panda in M. R. Patra, "Ensemble voting system for anomaly based network intrusion detection," v *International journal of recent trends in engineering*, zv. 2(5), 2009.

[19] K. Rao, F. Peng, H. Sak in F. Beaufays, "Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks," v *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, str. 4225–4229, IEEE, 2015.

[20] M. Romih, "Amebis in jezikovne tehnologije," v *Zbornik konference Jezikovne tehnologije za slovenski jezik. Ljubljana: Institut Jožef Stefan*, str. 29–34, 1998.

[21] T. Samardzic, N. Ljubešic in M. Milicevic, "Regional linguistic data initiative (reldi)," v *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, str. 40, 2015.

[22] J. Schmidhuber, "Deep learning in neural networks: An overview," v *Neural networks*, zv. 61, str. 85–117, 2015.

[23] T. Šef in M. Gams, "Data mining for creating accentuation rules," v *Applied Artificial Intelligence*, zv. 18(5), str. 395–410, 2004.

[24] T. Šef, M. Gams in M. Škrjanc, "Naglaševanje nepoznanih besed pri sintezi slovenskega govora," v *Informacijska družba IS'2002: Jezikovne tehnologije*, 2002.

[25] R. Setiono in H. Liu, "Neural-network feature selector," v *IEEE Transactions on Neural Networks*, zv. 8(3), str. 654–662, 1997.

[26] M. Skripkauskas in L. Telksnys, "Automatic transcription of Lithuanian text using dictionary," v *Informatica*, zv. 17(4), str. 587–600, 2006.

[27] M. Snoj, *Slovenski etimoloski slovar*. Modrijan, 2003.

[28] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever in R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting.," v *Journal of machine learning research*, zv. 15(1), str. 1929–1958, 2014.

[29] N. Takahashi, T. Naghibi in B. Pfister, "Automatic pronunciation generation by utilizing a semi-supervised deep neural networks," v *arXiv preprint:1606.05007v1*, 2016.

[30] J. Toporišič, *Slovenska slovnica*. Obzorja, 1984.

[31] J. Toporišić, *Slovenski pravopis*. Slovenska akademija znanosti in umetnosti, 2001.

[32] T. Tušar, A. Bratko, M. Gams in T. Šef, "Comparison between humans and machines on the task of accentuation of Slovene words," v *Proc. 8th International Multiconference Information Society, IS*, str. 353–356, 2005.

[33] X. Zhang, J. Zhao in Y. LeCun, "Character-level convolutional networks for text classification," v *Advances in Neural Information Processing Systems*, str. 649–657, 2015.