Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Matej Ulčar

Razpoznavanje slovenskega govora z metodami globokih nevronskih mrež

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: prof. dr. Marko Robnik Šikonja
Somentor: izr. prof. dr. Simon Dobrišek

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Zahvala

Zahvaljujem se mentorju, prof. dr. Marku Robniku Šikonji in somentorju izr. prof. dr. Simonu Dobrišku za vso pomoč in nasvete pri nastanku magistrskega dela. Zahvaljujem se vsem, ki so prispevali pri izdelavi korpusov Gos, Sofes, Gos VideoLectures, Gigafida in Kres. Brez dobrih učnih virov dela nikakor ne bi mogel opraviti. Na koncu bi se zahvalil še družini in prijatelju Alenu za vso podporo med študijem in izdelavi magistrske naloge.

Matej Ulčar, 2018

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
DNN	deep neural network	globoke nevronske mreže
GMM	Gaussian mixture model	mešanica Gaussovih porazdelitev
HMM	hidden Markov model	prikriti Markovovi modeli
LDA	linear discriminant analysis	linearna diskriminantna analiza
LSTM	long short-term memory	nevronske mreže z dolgim kratkoročnim spominom
MFCC	mel-frequency cepstral coefficients	mel-frekvenčni kepstralni koeficienti
MLLT	maximum likelihood linear transform	linearna transformacija z največjim verjetjem
PLP	perceptual linear prediction	zaznavno linearno napovedovanje
ReLU	rectified linear unit	pragovna linearna funkcija
RNN	recurrent neural network	ponavljajoče nevronske mreže
SAT	speaker adaptive training	učenje s prilagajanjem govorcu
TDNN	time-delayed neural network	časovno zakasnjene nevronske mreže
WER	word error rate	stopnja napačno razpoznanih besed

Povzetek

Naslov: Razpoznavanje slovenskega govora z metodami globokih nevronskih mrež

Ročno zapisovanje govora je počasen proces, ki ga čedalje bolj nadomešča avtomatsko razpoznavanje govora. Slednje se lahko uporablja tudi za glasovno upravljanje programov in naprav. V magistrski nalogi smo kot osnovo za razpoznavanje govorjene slovenščine uporabili uveljavljene metode GMM-HMM za akustični model in n-gramov za jezikovni model. Modela smo nadgradili z uporabo globokih nevronskih mrež, ki so se izkazale za zelo uspešne. Preizkusili smo različne arhitekture časovno zakasnjenih nevronskih mrež in nevronskih mrež z dolgim kratkoročnim spominom na akustičnem in jezikovnem modelu razpoznavalnika govora. Razpoznavalnik smo učili na širokem besednjaku, ki vsebuje približno milijon različnih besed. Najboljše rezultate dosegajo časovno zakasnjene nevronske mreže, kjer smo dosegli $72,84\%$ pravilno prepoznanih besed pri tekočem govoru.

Ključne besede

strojno učenje, globoke nevronske mreže, razpoznavanje govora

Abstract

Title: Automatic Slovene speech recognition using deep neural networks

Manual transcription of speech is slow and is being replaced by automatic speech recognition systems. These systems are also used for voice control of various programs and devices. In this thesis, we used as a baseline for Slovene speech recognition GMM-HMM methods for acoustic model and n-grams for language model. We improved both models with deep neural networks, which have proven to be very successful. We tested several architectures of time-delayed neural networks and neural networks with long short-term memory for both acoustic and language model. We used a large lexicon, containing about a million words. Time-delayed neural networks achieved the best results on continuous speech, with $72,84\%$ of correctly identified words.

Keywords

machine learning, deep neural networks, speech recognition

Poglavje 1

Uvod

Govor večkrat želimo zapisati kot besedilo, na primer zapisnik sestanka, zapiske s predavanj, podnapise na televiziji v pomoč slušno prizadetim, ipd. Za zapis je potrebno govor večkrat poslušati in ga sproti zapisovati, kar je lahko časovno potratno, še posebej, ko je govor hiter in je potrebno posnetek ustavljati in ponovno predvajati.

Problem razpoznavanja govora je sestavljen iz dveh delov: akustičnega modela in jezikovnega modela. Akustični model je prepoznavanje posameznih glasov, oziroma fonemov iz zvočnega posnetka. Jezikovni model predstavlja tvorbo besedila na podlagi prepoznanih fonemov. Za reševanje problemov obdelave naravnih jezikov se v zadnjem času zelo uspešno uporabljajo globoke nevronske mreže. Z uporabo globokih nevronskih mrež smo poizkusili izboljšati rezultate do sedaj uporabljenih metod strojnega učenja pri razpoznavanju govora v slovenščini in izdelati kakovostno odprtokodno rešitev.

1.1 Pregled sorodnih del

Jezikovni modeli pri razpoznavanju govora v slovenščini večinoma uporabljajo Good-Turingovo glajenje [30, 31, 6]. Žgank in sod.[30] so uporabili dve govorni bazi, eno z večjim deležem spontanega govora, drugo z večjim deležem

branega govora. Akustični model so osnovali na zveznih prikritih Markovovih modelih (HMM). Žgank in sod.[30] so ugotavljali predvsem vpliv velikosti uporabljenih besedilnih in govornih korpusov. Njihova analiza je pokazala, da večanje uporabljenih virov izboljša rezultate, vendar je to izboljšanje majhno, za večjo izboljšavo je potrebna tudi uporaba drugih algoritmov.

Žgank in sod.[31] so uporabili razpoznavalnik opisan v prejšnjem odstavku [30] za transkribiranje nove govorne baze SI TEDx-UM. Uporabili so dva jezikovna modela, enega grajenega na govorni bazi BNSI, drugega pa zgolj na besedilnem korpusu FidaPLUS. Njihovi rezultati so pokazali, da je modeliranje govorjene rabe jezika pomemben del jezikovnega modela, po drugi strani pa ima tematika govora velik vpliv na natančnost razpoznavanja besed. Oba jezikovna modela sta dosegla enak rezultat ($50,7$ % napako). Ker se domeni govora pri SI TEDx-UM in BNSI med seboj precej razlikujeta, so rezultati pri razpoznavanju govora predavanj SI TEDx-UM precej slabši od rezultatov pri razpoznavanju govora posnetkov BNSI.

Bolka [2] je v diplomskem delu uporabil zbirko orodij Kaldi za razpoznavanje fonemov v slovenščini. Uporabil je večje število metod učenja akustičnega modela, kjer je model zgrajen z nevronskimi mrežami dosegel najboljše rezultate. Nevronske mreže z dolgim kratkoročnim spominom (angl. Long Short-Term Memory - LSTM) za prevajanje med fonemi in grafemi predlagajo tudi Rao in sod.[24]. V svojem delu izdelajo model, ki napoveduje foneme glede na dane grafeme, z drugimi besedami izgovor besede. Problem je soroden obratnemu, to je določiti zapis besede glede na njen izgovor. Globoke nevronske mreže LSTM vsebujejo posebne enote, imenovane spominske celice. Te so si zmožne podatke zapomniti poljubno dolgo. Pozabna vrata spominske celice skrbijo, da se podatek lahko po potrebi tudi pozabi. Zaradi tega so globoke nevronske mreže LSTM zelo dobre pri prepoznavanju govora, saj pri učenju upoštevajo tudi kontekst (npr. zapis besede je odvisen tudi od predhodnih glasov, ne samo od vsakega posameznega) [2, 24].

Hernandez in sod. [13] so uporabili Kaldi za učenje sistema za razpoznavanja govora v angleščini. Za govorno zbirko so uporabili zbirko predavanj

TED. Akustični model so naučili s hibridnim modelom (DNN-HMM), kjer so najprej uporabili pristop GMM-HMM za učenje in poravnavo zvočnih posnetkov s transkripcijo. Nato so, namesto GMM, uporabili časovno zakasnjene nevronske mreže (TDNN). Naučili so dva jezikovna modela. Prvega z n-grami reda 4 in drugega z uporabo nevronskih mrež, kjer so uporabili tri nivoje TDNN, med njimi pa dva nivoja LSTM.

Microsoft v svojem sistemu za razpoznavanje pogovornega govora [29] uporablja globoke nevronske mreže za akustični in jezikovni model. Akustični model je naučen s kombinacijo konvolucijskih nevronskih mrež in nevronskih mrež z dolgim kratkoročnim spominom (LSTM). Za učenje jezikovnega modela so uporabili ponavljajoče nevronske mreže (RNN).

Googlova aplikacija za pametne telefone, ki uporablja glasovno upravljanje ter omogoča glasovno iskanje, za razpoznavanje govora prav tako uporablja nevronske mreže pri akustičnem modelu [27]. Sak in sod. [26] so uporabili dvosmerne mreže LSTM, za poravnavo zvočnega posnetka s transkripcijo v učni množici pa so namesto modela GMM-HMM uporabili metodo časovne klasifikacije omrežja (angl. connectionist temporal classification - CTC).

1.2 Struktura naloge

V uvodnem poglavju predstavimo motivacijo za delo ter na kratko opišemo sorodna dela. V drugem poglavju opišemo teoretično ozadje razpoznavanja govora s poudarkom na modelih GMM-HMM. V tretjem poglavju opišemo nevronske mreže in na kratko predstavimo različne tipe mrež, ki se uporabljajo pri razpoznavanju govora. V četrtem poglavju opišemo okolje Kaldi, v katerem smo opravili delo. V petem poglavju predstavimo arhitekturo razpoznavalnika, oziroma različne postopke učenja, ki smo jih uporabili. V šestem poglavju sledi predstavitev in analiza rezultatov. V sklepnem delu opišemo opravljeno delo ter predstavimo možnosti za izboljšave.

Poglavje 2

Razpoznavanje govora

Sistem za razpoznavanje govora lahko v grobem razdelimo v dva dela, akustični model in jezikovni model. Akustični model naučimo na značilkah pridobljenih iz zvočnih posnetkov govora in pripadajočih transkripcijah govora. Akustični model zvočnemu signalu pripiše pripadajoč fonem. Posebej naučimo jezikovni model na besedilnem korpusu. Jezikovni model na podlagi predhodnih besed predlaga najbolj verjetno naslednjo besedo. Oba modela povezuje slovar, v katerem vsaki besedi pripišemo njen fonetični zapis.

Pri uporabi naučenega modela za razpoznavanje govora, zvočni signal najprej obdelamo, da dobimo enake značilke, kot smo jih uporabili pri učenju. Akustični model s slovarjem predlaga najverjetnejše besede, ki ustrezajo danemu signalu. Te verjetnosti kombiniramo z jezikovnim modelom, ki predlaga najverjetnejše besede glede na kontekst. Podobno delujejo tudi človeški možgani pri razumevanju govora.

2.1 Določanje značilk

Zvok človeškega govora je odvisen od prisotnosti posameznih frekvenc zvoka. Vsak fonem zastopajo različne frekvence, te se spreminjajo s časom tudi znotraj fonema. Znotraj kratkega časovnega okna je zvočni signal približno konstanten, zato lahko nad njim izvedemo frekvenčno analizo. Zastopanost po-

sameznih frekvenc v vsakem časovnem oknu zvočnega signala nam da osnovo za značilke, na katerih učimo sistem za razpoznavanje govora. Najpogostejša tipa značilk pri razpoznavanju govora sta PLP in MFCC. Obe vrsti značilk oponašata človeški sluh in človeško zaznavo zvoka [1].

2.1.1 Značilke MFCC

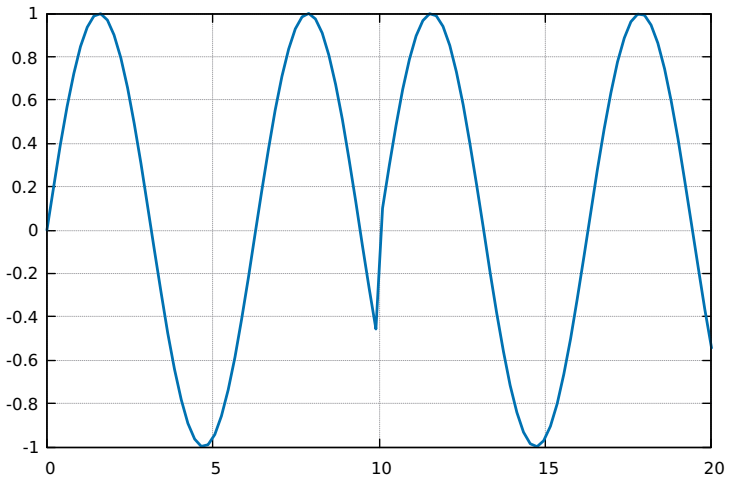
Prvi korak pri izračunu značilk MFCC (mel-frekvenčni kepstralni koeficienti) [6] je predobdelava zvočnega signala. Signalu odstranimo amplitudni zamik in ga centriramo okrog amplitude nič. To pomeni, da signalu odštejemo njegovo povprečje, tako da je nova povprečna amplituda enaka 0. Obenem ojačamo signal pri višjih frekvencah, ker energija govornega signala upada z višanjem frekvence [6, 11]. Tako dobimo pri vseh frekvencah podobno visoke vrhove spektra, v katerih se nahaja informacija o fonemih [6]. Diferenčna enačba ojačitvenega sistema je enaka

$$y[n] = x[n] - \alpha x[n-1], \tag{2.1}$$

kjer je $x[n]$ signal na vhodu, $x[n-1]$ vhod v prejšnjem koraku, $y[n]$ izhodni signal, parameter α pa nekje med 0,9 in 1,0 [25, 6].

Zvočni signal razrežemo na kratke odseke (okna) dolge nekaj milisekund (tipična dolžina je 25 ms). Okna se med seboj prekrivajo, običajno sta začetni točki sosednjih oken 10 ms narazen. Za frekvenčno analizo moramo zvočni signal transformirati s Fourierjevo transformacijo. Fourierjeva transformacija je definirana za ponavljajoče signale (če zlepimo konec in začetek signala mora biti signal zvezen). Naš signal temu pogoju ne zadošča, saj širina okna ni točno enaka valovni dolžini signala, nastane preskok (glej sliko 2.1). Zaradi tega preskoka pride pri Fourierjevi transformaciji do šuma pri drugih frekvencah. Ta šum skušamo čimbolj odstraniti z uporabo okenskih funkcij.

Vsako okno pomnožimo z okensko funkcijo, da čimbolj zmanjšamo neželena popačenja, ki nastanejo pri Fourierjevi transformaciji. Množenje dveh signalov v časovni domeni je konvolucija njunih spektrov v frekvenčni domeni,



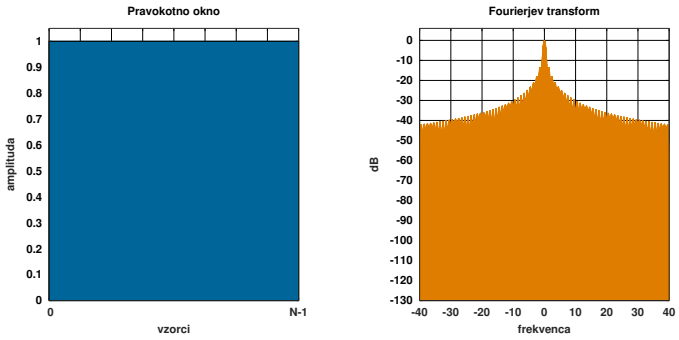
Slika 2.1: Sinusna funkcija $\sin(x)$ za $0 \leq x \leq 10$. Med $x = 10$ in $x = 20$ je funkcija ponovljena. Opazimo nezvezen preskok pri $x = 10$.

zato izbiramo takšne okenske funkcije, da omenjena konvolucija čim manj popači spekter analiziranega odseka signala. Primere različnih okenskih funkcij vidimo na slikah 2.2, 2.3, 2.4, 2.5. Najpogosteje se uporabljata Hannova in Hammingova okenska funkcija [21]. Nad vsakim oknom nato napravimo Fourierjevo transformacijo, da dobimo zastopanost frekvenc.

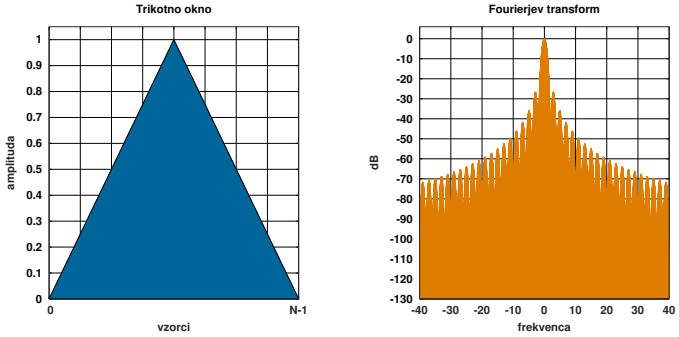
Frekvence (f) preračunamo v mel frekvenčno skalo:

$$m(f) = 1125 \ln\left(1 + \frac{f}{700}\right). \tag{2.2}$$

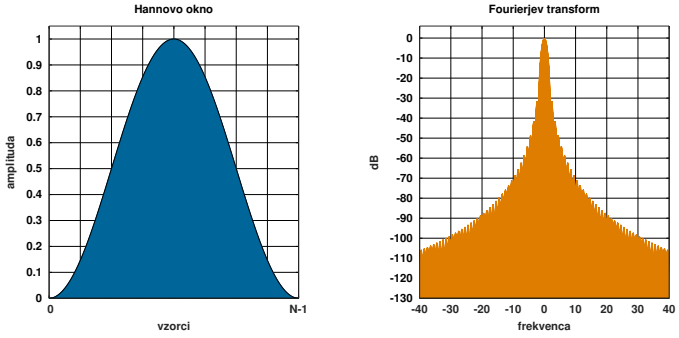
Definiramo mel-frekvenčne filtre, ki so trikotne oblike in centrirani okrog frekvenc, ki so enakomerno oddaljene med seboj v mel-frekvenčni skali (slika 2.6). Filtri se deloma prekrivajo med seboj. Za vsak filter zmnožimo spekter signala s filtrom in izračunamo logaritem spektralne energije signala znotraj filtra, ker je človeško zaznavanje glasnosti sorazmerno logaritmu intenzitete. Nad dobljenim izračunamo kosinusno transformacijo. Rezultat so značilke MFCC. Za potrebe razpoznavanja govora običajno obdržimo le prvih 13 koeficientov [25, 6].



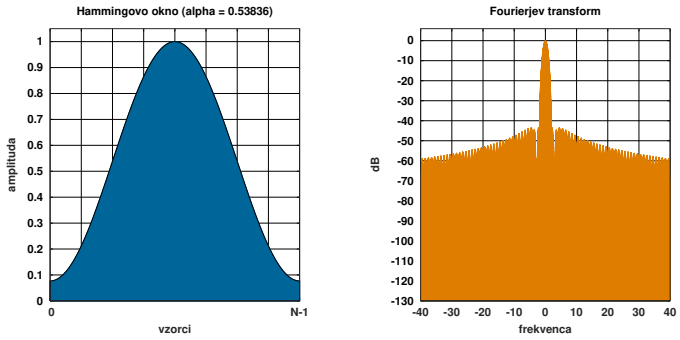
Slika 2.2: Pravokotno okno in njegova Fourierjeva transformacija.



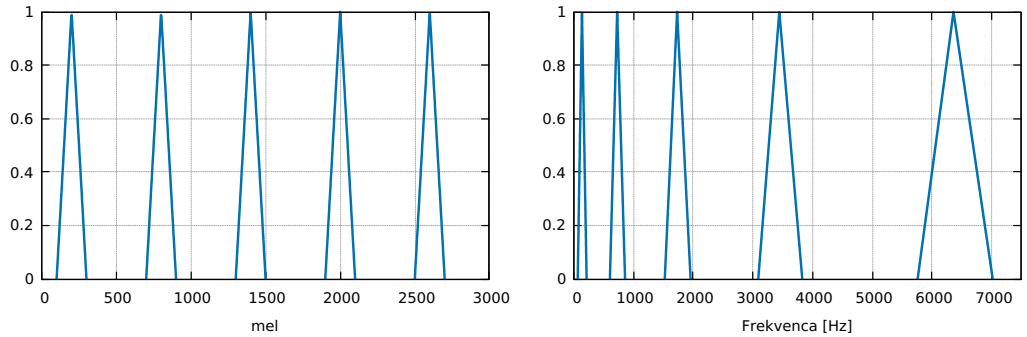
Slika 2.3: Trikotno okno in njegova Fourierjeva transformacija.



Slika 2.4: Hannovo okno in njegova Fourierjeva transformacija.



Slika 2.5: Hammingovo okno in njegova Fourierjeva transformacija.



(a) Filtri v mel frekvenčni skali so enakomerno oddaljeni med seboj.

(b) Filtri prikazani v normalni frekvenčni skali (Hz)

Slika 2.6: Mel-frekvenčni filtri, prikazani v mel skali in normalni frekvenčni skali. Zaradi preglednosti prikazujemo le pet filtrov.

2.1.2 Značilke PLP

Postopek izračuna značilk PLP (zaznavno linearno napovedovanje) [6, 25] je podoben izračunu značilk MFCC. Prvi koraki so identični, namesto mel frekvenčne skale pa uporabimo Barkovo skalo:

$$B(\omega) = 6 \ln \left[\frac{\omega}{1200\pi} + \sqrt{1 + \left(\frac{\omega}{1200\pi} \right)^2} \right]. \tag{2.3}$$

Spekter znova razdelimo v posamezne pasove, ki pa so malo drugače oblikovani kot pri značilkah MFCC. Spektralno energijo ojačamo s funkcijo, ki aproksimira različno občutljivost človeškega sluha pri različnih frekvencah. Nato vzamemo tretji koren rezultata, ker človeško zaznavanje glasnosti ni linearno (podobno kot logaritem pri značilkah MFCC).

Nad rezultatom izvedemo inverzno diskretno Fourierjevo transformacijo. Značilke dobimo s postopkom linearnega napovedovanja nad obdelanim spektrom. V linearnem napovedovanju napovemo vzorec signala $x[n]$ kot linearno kombinacijo preteklih p vzorcev, kjer je p red analize linearnega napovedovanja:

$$x'[n] = \sum_{k} = 1^{p} a_k x[n-k]. \tag{2.4}$$

Pri tem je $x'[n]$ napoved dejanskega vzorca signala $x[n]$. Koeficienti linearnega napovedovanja a_1, ..., a_p so značilke PLP [6].

2.1.3 Značilke delta in delta-delta

Prikriti Markovov model predpostavlja neodvisnost stanj (glej razdelek 2.2.1), zato se spektralnim koeficientom običajno dodajajo značilke Δ in $\Delta - \Delta$. Značilke Δ predstavljajo razliko med posameznimi časovnimi okni. Izračunamo jih po izrazu

$$\Delta y(m,t) = \frac{\sum_{q=1}^{n} q(y(m+q,t) - y(m-q,t))}{2\sum_{q=1}^{n} q^2}, \tag{2.5}$$

kjer je $y(m,t)$ vrednost značilke pri časovnem oknu m in spektralnem koeficientu t, n je širina okna. Podobno dobimo tudi značilke $\Delta - \Delta$, kjer namesto značilk $y(m,t)$ v izrazu 2.5 uporabimo $\Delta y(m,t)$ [1, 7].

2.2 Akustični model

Akustični model razpoznavalnika govora je bil v preteklosti najpogosteje model, osnovan na modelu GMM-HMM (mešanica Gaussovih porazdelitev, angl. Gaussian mixture model, prikriti Markovovi modeli, angl. hidden Markov

Tabela 2.1: Predstavitev besedne zveze „pod skalo“ z monofoni in trifoni, kjer () predstavlja tišino.

grafemi	pod skalo
monofoni	() p o t s k a l o ()
trifoni	() ()-p+o p-o+t o-t+s t-s+k s-k+a k-a+l a-l+o l-o+() ()

model). V zadnjem času se vse pogosteje uporabljajo globoke nevronske mreže.

2.2.1 GMM-HMM

Pomembni predpostavki pri uporabi HMM za razpoznavanje govora sta, da je verjetnost opaženega stanja neodvisna od predhodnega stanja in da lahko zvočni signal razdelimo v stacionarne dele s takojšnjimi prehodi med njimi. Nobena izmed predpostavk ni v celoti izpolnjena. Izgovor fonema je odvisen od fonema pred njim. Prav tako je med fonemoma prehodno obdobje (čas med enim in drugim fonemom). Vseeno se HMM izkaže za uporaben in uspešen model za razpoznavanje govora.

Osnovna enota, ki je zajeta v enem stanju HMM je lahko ena beseda, vendar bi za to potrebovali veliko količino učnih podatkov. Tak pristop je uporaben le za sisteme z majhnim številom besed. Največkrat se za osnovo uporabijo posamezni fonemi (monofonski sistem) ali pa trije fonemi skupaj (trifonski sistemi), ker sosednji fonemi vplivajo na izgovor. Primer zapisa z monofoni in trifoni je predstavljen v tabeli 2.1.

Fonemi predstavljajo skrita stanja v HMM, za njihov zapis pa moramo poznati izgovarjavo vsake besede. Za to uporabimo slovar, v katerem ob vsakem geslu (besedi) opišemo izgovor te besede. Z drugimi besedami, zapišemo besedo in njen fonetični zapis. Opazovana stanja HMM so mešanica Gaussovih porazdelitev (GMM), ki opisujejo spekter posameznega okna.

Pri uporabi trifonskega modela potrebujemo veliko število trifonov. Za jezik s 30 fonemi to pomeni $30^3 = 27.000$ trifonov. Težava je, da v učni množici

morda nimamo dovolj velikega števila vseh trifonov za dobro ponazoritev in učenje ter s tem robusten model. Lahko se zgodi, da katerega izmed trifonov med učnimi podatki sploh ni. Težavo rešujemo z združevanjem podobnih fonemov, oziroma trifonov z odločitvenim drevesom [28, 7].

2.2.2 Linearna diskriminantna analiza

Linearna diskriminantna analiza (LDA) je način, s katerim zmanjšamo število dimenzij v vektorjih značilk, obenem pa ohranimo diskriminantne značilnosti množice značilk. Rezultat so nižje dimenzionalni vektorji značilk, ki so manj korelirani in dobro razlikujejo med posameznimi razredi. Tako je učenje akustičnega modela lažje, oziroma hitrejše.

Poleg linearne diskriminantne analize obstajajo tudi druge metode za zmanjševanje števila dimenzij. Harrag in Mohamadi [12] sta primerjala metodo glavnih komponent (angl. principal component analysis - PCA), linearno diskriminantno analizo in sekvenčno iskanje naprej (angl. sequential forward search - SFS). Pri računanju značilk za razpoznavanje govora najboljše rezultate doseže metoda LDA.

Linearna diskriminantna analiza deluje tako, da maksimizira kovarianco med posameznimi razredi Σ_B in hkrati minimizira kovarianco znotraj razreda Σ_W glede na izbrane značilke. Skupaj lahko zapišemo kriterij, ki ga želimo maksimizirati, kot

$$f = \log\left(\frac{|\boldsymbol{A}\boldsymbol{\Sigma}_{\boldsymbol{B}}\boldsymbol{A}^T|}{|\boldsymbol{A}\boldsymbol{\Sigma}_{\boldsymbol{W}}\boldsymbol{A}^T|}\right), \tag{2.6}$$

kjer je $\boldsymbol{A}$ matrika linearne transformacije med originalnimi vektorji značilk ($\boldsymbol{x}$) in transformiranimi vektorji značilk ($\boldsymbol{y}$). Poiskati moramo tako matriko $\boldsymbol{A}$, da je f največji. Velja tudi zveza

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}. \tag{2.7}$$

2.2.3 STC/MLLT

Linearna transformacija z največjim verjetjem (angl. maximum likelihood linear transform - MLLT), imenovana tudi „delno povezana kovarianca“ (angl.

semi-tied covariance - STC) se pogosto uporablja v kombinaciji z linearno diskriminantno analizo. Če imamo opazovana stanja v HMM predstavljena z GMM, bi morali za vsako komponento, torej za vsako Gaussovo porazdelitev, izračunati celotno kovariančno matriko $\mathbf{\Sigma}_{jm}$. Namesto tega opišemo kovariančno matriko kot matriko, sestavljeno iz dveh delov. Prvi del predstavlja diagonalna matrika, ki je specifična za vsako komponento ($\mathbf{\Sigma}_{jm}^{diag}$). Drugi del je delno povezana matrika $\boldsymbol{H}^{(r)}$, ki ni specifična za vsako komponento, ampak za vsak razred komponent. Delno povezana matrika $\boldsymbol{H}^{(r)}$ lahko predstavlja poljubno število komponent [28]. Kovariančno matriko zapišemo kot:

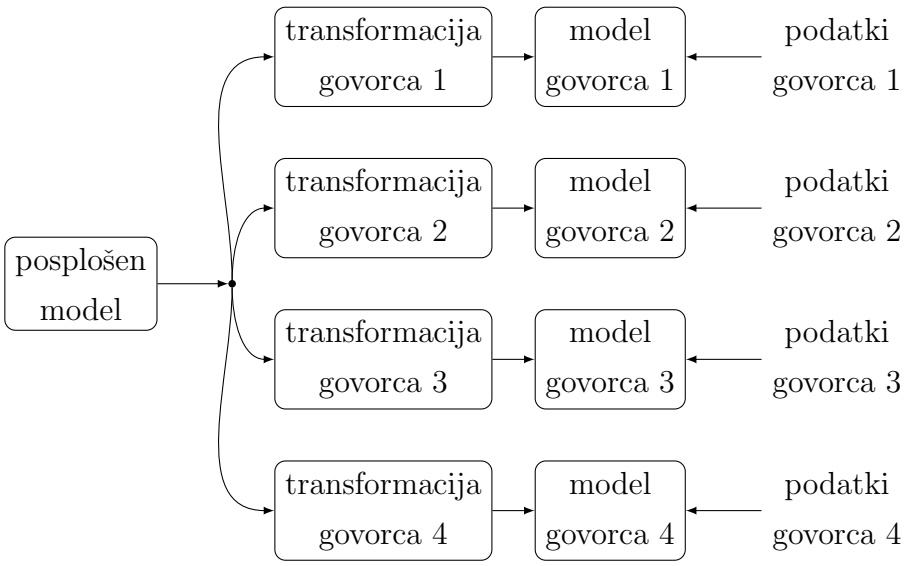
$$\mathbf{\Sigma}_{jm} = \boldsymbol{H}^{(r)} \mathbf{\Sigma}_{jm}^{(diag)} \boldsymbol{H}^{(r)T}. \tag{2.8}$$

2.2.4 Prilagajanje govorcu - SAT

Sistem za razpoznavanje govora, ki je neodvisen od govorca nujno potrebuje veliko število govorcev v učni množici. Akustični modeli naučeni na taki učni množici zato vsebujejo tudi parametre, ki razlikujejo med posameznimi govorci. Naš cilj pa je razlikovati med različnimi besedami, ne glede na govorca. Problem rešujemo z učenjem s prilagajanjem govorcu (angl. speaker adaptive training - SAT). Za vsakega govorca posebej naučimo model z učnimi podatki tega govorca. Nato ocenimo transformacijo, ki splošen model transformira v specifičen model govorca. Končno na podlagi specifičnih modelov posameznih govorcev in njihovih transformacij ocenimo kanoničen akustičen model [7]. Grob opis modela SAT vidimo na sliki 2.7.

2.2.5 DNN-HMM

Hibridni modeli DNN-HMM nadomestijo mešanico Gaussovih porazdelitev v GMM-HMM z globokimi nevronskimi mrežami. Modele GMM-HMM najprej uporabimo, da dobro poravnamo učne podatke. Skrita stanja v HMM, foneme, moramo opisati zgolj s tistimi časovnimi okni, v katerih se nahaja izgovorjen fonem. Ta ujemanja in poravnave določimo tekom učenja GMM-



Slika 2.7: Model učenja s prilagajanjem na govorca pri štirih govorcih, povzeto po [7].

HMM. Sistem nadgradimo z globokimi nevronskimi mrežami, ki se učijo na podanih akustičnih značilkah, kjer je ustrezen fonem želeni rezultat [19]. Globoke nevronske mreže podrobneje predstavimo v poglavju 3, različne vrste nevronskih mrež, ki se uporabljajo pri razpoznavanju govora pa v razdelku 3.4.

2.3 Jezikovni model

Jezikovni model je predstavitev znanja našega razpoznavalnika o jeziku. Jezikovni model uporabimo, da s pomočjo konteksta določimo najverjetnejšo (manjkajočo) besedo v nekem besedilu. Učenje jezikovnega modela poteka na besedilnem korpusu. Pri učenju tako dobimo statistične podatke o besedah v besedilu. Najpreprostejša metoda uporabljena v praksi so bigramski modeli, kjer govorni jezikovni vir modeliramo kot vir s spominom prvega reda. To pomeni, da predpostavimo, da na besedo vpliva le beseda, ki se v besedilu nahaja tik pred njo. Iz besedila izluščimo vse pare sosednjih besed

in pogledamo, kako pogosto vsaki besedi sledi določena beseda. Verjetnost, da se na mestu i nahaja beseda w_i, če je pred njo beseda w_{i-1} je torej

$$P(w_i|w_{i-1}) = \frac{P(w_{i-1}), P(w_i)}{P(w_{i-1})}. \tag{2.9}$$

Tak model lahko izboljšamo tako, da modeliranje razširimo na vir s spominom višjega reda. Trigramski model, na primer, upošteva predhodni dve besedi $P(w_i|w_{i-1}, w_{i-2})$, za n-gramski model torej upoštevamo predhodnih $n-1$ besed $P(w_i|w_{i-1}, ..., w_{i-n+1})$. Pri napovedovanju besede potrebujemo čimveč konteksta (torej predhodnih besed), vendar pa s tem raste kompleksnost in velikost jezikovnega modela. Število n-gramov v našem jezikovnem modelu je enako

$$N = w^n, \tag{2.10}$$

kjer je w število besed v slovarju, n pa dolžina (stopnja) n-grama. Za določitev pogostosti vseh n-gramov v tekstu zato potrebujemo zelo obsežen korpus. Problem nastane, ker tako obsežnih korpusov ni, niti jih ni moč praktično shraniti na disk. Že pri slovarju z 10.000 besedami in pri trigramskem modelu bi potrebovali nekaj terabajtov za zapis vseh trigramov. Za uporaben jezikoven model se mora vsak n-gram ponoviti večkrat, da dobimo dobre statistične podatke. Nekateri trigrami se v jeziku sploh ne pojavijo in jih ni v korpusu, kar potrebno dolžino sicer skrajša, je pa v jeziku precej več kot 10.000 različnih besed. Pri grajenju jezikovnega modela z n-grami zato uporabimo različne tehnike glajenja in sestopanja. Pri sestopanju n-gram nadomestimo z več n-grami nižje stopnje, na primer trigram, ki se v korpusu ne pojavi, nadomestimo z dvema bigramoma:

$$P(w_i|w_{i-1}, w_{i-2}) \rightarrow P(w_i|w_{i-1}) \cdot P(w_{i-1}|w_{i-2}). \tag{2.11}$$

Z glajenjem izboljšamo statistiko manj zastopanih n-gramov. Osnovna ideja je, da številu pojavitev za vsak n-gram prištejemo neko število, denimo ena. Vsi n-grami tako dobijo neničelno verjetnost pojavitve, zato lahko napovemo tudi besede, ki jih sicer ne bi mogli (imajo verjetnost enako nič, ker takega

n-grama ni v korpusu). V praksi se največ uporabljata Good-Turingovo in Kneser-Neyevo glajenje.

Boljši, vendar računsko zahtevnejši, pristop gradnje jezikovnega modela je z globokimi nevronskimi mrežami. Uveljavile so se predvsem ponavljajoče globoke mreže. Nevronska mreža združuje semantično podobne besede, zato lahko napovemo n-grame, ki jih v korpusu ni, če so v korpusu n-grami s semantično podobnimi besedami. Takih podobnosti se jezikovni modeli z n-grami ne morejo naučiti [16, 14].

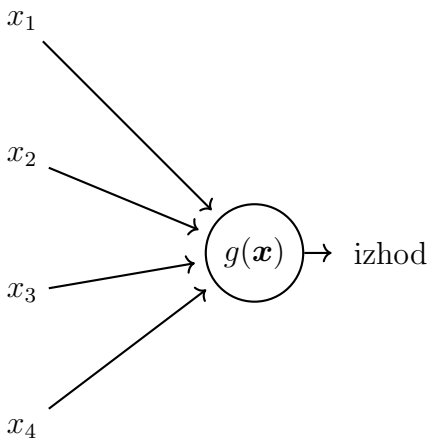
Poglavje 3

Globoke nevronske mreže

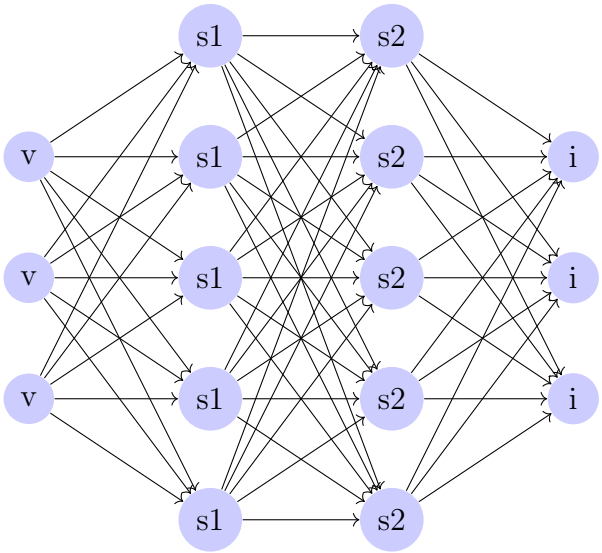
Nevronske mreže so sestavljene iz več preprostih računskih enot, ki jim pravimo nevroni. Nevrone med seboj povežemo v usmerjen graf, kjer je vsaka povezava drugače utežena. Vsak nevron predstavlja neko funkcijo, imenovano aktivacijska funkcija (glej razdelek 3.1). Izhodna vrednost vsakega nevrona je enaka vrednosti aktivacijske funkcije, katere argument je utežena vsota vhodov v nevron (slika 3.1). Cilj učenja nevronske mreže je, da določimo te uteži tako, da s celotno mrežo nevronov predstavimo želeno funkcijo [9].

Osnovna vrsta nevronskih mrež so mreže s povezavami naprej (angl. feed-forward neural network). Graf nevronov v taki mreži je acikličen, saj nimamo povratnih povezav. Nevronske mreže so urejene v nivoje ali sloje (slika 3.2). Na vhodnem nivoju nevroni dobijo na vhod učne podatke. Izhod vhodnega nivoja vodi na vhod prvega skritega nivoja. Sledi poljubno število skritih nivojev. Iz zadnjega skritega nivoja vodijo povezave na izhodni nivo. Na izhodu izhodnega nivoja je končni rezultat, ki ga izračuna nevronska mreža.

Velikost nevronske mreže določata dva parametra, širina in globina. Širina mreže predstavlja število nevronov v vsakem nivoju. Širina se lahko spreminja iz nivoja v nivo. Globina nevronske mreže pove koliko (skritih) nivojev imamo. Nevronske mreže z dvema ali več skritimi nivoji imenujemo globoke nevronske mreže [20].



Slika 3.1: Primer nevrona z aktivacijsko funkcijo $g(x)$, štirimi vhodi in izhodom.



Slika 3.2: Primer nevronske mreže z vhodnim (v), izhodnim (i) in dvema skritima nivojema (s1, s2).

3.1 Aktivacijske funkcije

Globoke nevronske mreže vsebujejo skrite nivoje nevronov. Aktivacijske funkcije izračunajo vrednosti nevronov v skritih nivojih iz vhodov v posamezen nevron. Obstaja veliko različnih aktivacijskih funkcij, v praksi se jih uporablja manjše število [9]. Našteli bomo nekaj aktivacijskih funkcij.

Sigmoidne funkcije (enačba 3.1) se pogosto uporabljajo pri ponavljajočih mrežah, avtoenkoderjih in verjetnostnih modelih. Prednost sigmoidne funkcije je, da je zvezno odvedljiva, in da lahko njen odvod enostavno izračunamo. Glavna slabost je, da je za vhodne vrednosti z, ki so zelo velike po absolutni vrednosti, odvod sigmoidne funkcije blizu nič, saj ima funkcija omejeno zalogo vrednosti. Učenje je zato v tem območju težavno in počasno. Najpogosteje sta uporabljeni sorodni funkciji, logistična sigmoidna funkcija (slika 3.3)

$$g(z) = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.1}$$

in hiperbolični tangens (slika 3.4)

$$g(z) = \tanh(z). \tag{3.2}$$

Med njima velja zveza $\tanh(z) = 2\sigma(2z) - 1$ [9].

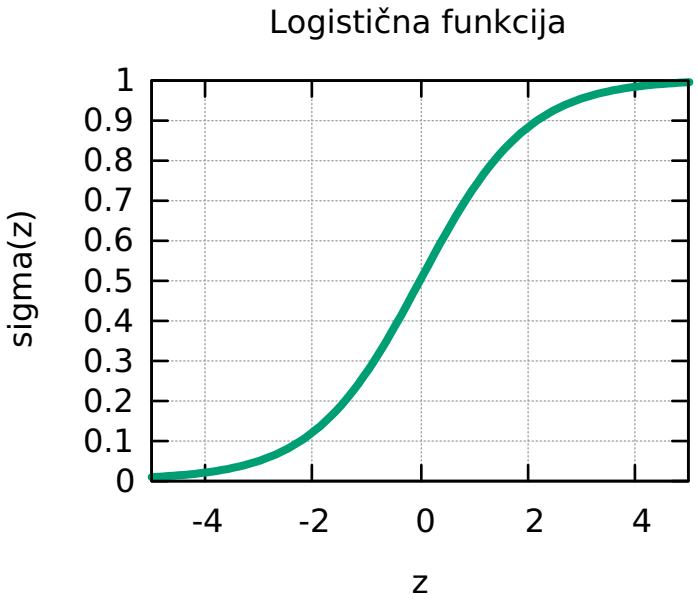
Slabost sigmoidnih funkcij rešujejo funkcije, ki nimajo omejene zaloge vrednosti. Uporaba takih funkcij pospeši učenje nevronske mreže. Ena izmed takih funkcij je p-norma, ki upošteva večje število vhodov[33, 32]:

$$g(\boldsymbol{z}) = \left(\sum_{k=1}^{K} z_k^p \right)^{1/p}. \tag{3.3}$$

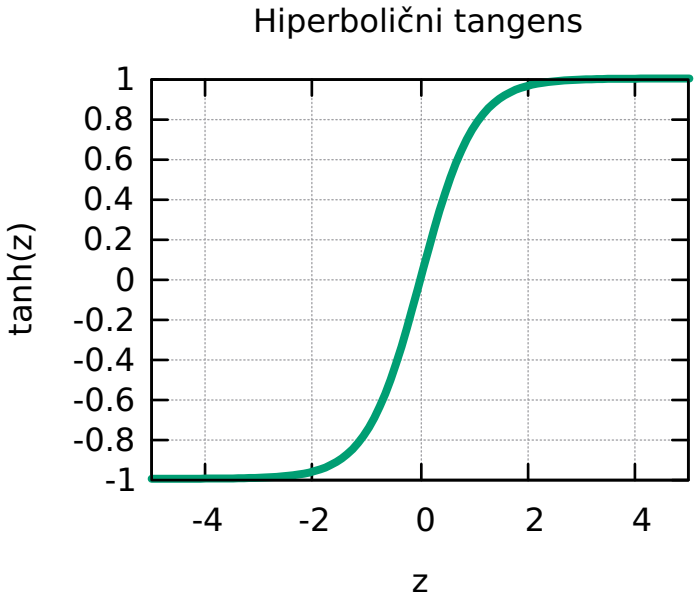
V zadnjem času se uporabljajo pragovne linearne funkcije (ReLU, slika 3.5)

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

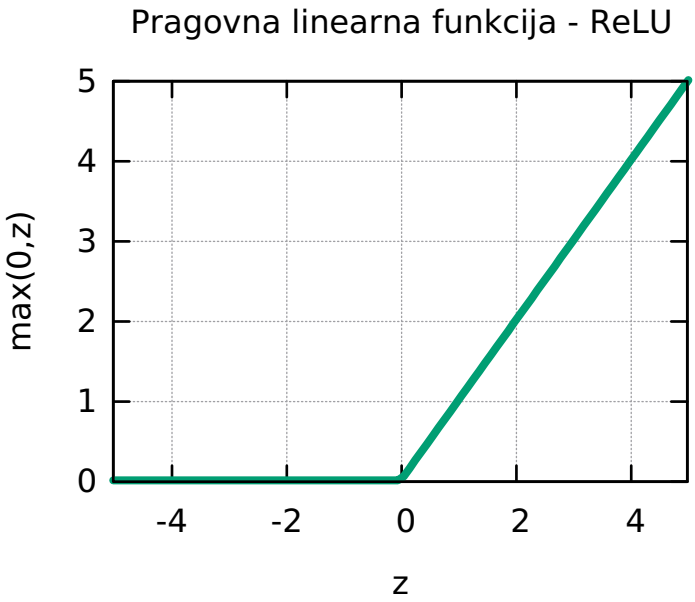
Prednost teh funkcij je lahka optimizacija, ker so podobne linearnim funkcijam. Njihov odvod je trivialno izračunati, saj je za vse nenegativne vrednosti konstanten. Obenem je odvod dovolj strm, da je učenje lahko. Za negativne



Slika 3.3: Primer aktivacijske funkcije: logistična sigmoida.



Slika 3.4: Primer aktivacijske funkcije: hiperbolični tangens.



Slika 3.5: Primer aktivacijske funkcije: ReLU.

vhodne vrednosti je vrednost ReLU enaka nič, aktivacije nevrona tako ni, odvodi nas zato ne zanimajo [9].

3.2 Gradientni spust in funkcije izgube

Zaradi nelinearnosti nevronskih mrež je večina funkcij izgube nekonveksnih. To pomeni, da za učenje nevronske mreže uporabljamo iterativne postopke, kot je gradientni spust. Gradientni spust je algoritem, s katerim optimiziramo uteži povezav med nevroni tako, da minimiziramo funkcijo izgube. Zaradi nekonveksnosti konvergenca rešitve ni zagotovljena, prav tako je algoritem občutljiv na začetne vrednosti uteži. Utežem določimo majhne naključne začetne vrednosti.

Funkcije izgube ocenjujejo kakovost nevronske mreže s primerjavo izhoda nevronske mreže z rezultatom, ki ga želimo. Slednji je del učnih podatkov. Običajno uporabljena funkcija izgube je križna entropija med učnimi podatki

in rezultati modela:

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

kjer so y_i rezultati modela, y'_i pa želen rezultat za vsak izhodni nevron i [15].

3.3 Izpuščanje

Izpuščanje (angl. dropout) je tehnika, ki se uporablja pri učenju nevronskih mrež z namenom izogibanja prenaučenosti (angl. overfitting). V vsaki iteraciji učenja določen delež nevronov v vsakem (skritem) nivoju izpustimo, tako da pri učenju niso uporabljeni. V naslednji iteraciji postopek ponovimo, le da naključno izberemo druge nevrone, ki jih izpustimo. S tem dosežemo, da so v končnem modelu vsi nevroni približno enako upoštevani, noben izmed njih nima občutno večje teže od ostalih [9].

Delež izpuščenih nevronov ni nujno enak tekom učenja, ampak se lahko spreminja. Cheng in sod. [4] pri razpoznavanju zvoka z mrežami LSTM uporabijo režim, kjer je delež izpuščenih nevronov v začetku enak nič, nato linearno raste do neke točke ter spet linearno pada proti nič na koncu učenja.

3.4 Vrste globokih nevronskih mrež

Nevroni v nevronskih mrežah so med seboj lahko povezani na več različnih načinov. Tako ločimo več različnih vrst nevronskih mrež. Če vsi nevroni v nivoju dobijo na vhod izhode vseh nevronov prejšnjega nivoja, govorimo o polno povezanem nivoju. Alternativno lahko sprejmejo na vhod le nekaj izmed nevronov prejšnjega nivoja. V tem primeru govorimo o konvolucijskih nivojih. V tem poglavju smo govorili le o nevronskih mrežah s povezavami naprej. Nevronske mreže imajo lahko tudi povratne povezave. Mreže s povratnimi povezavami imenujemo ponavljajoče nevronske mreže (angl. recurrent neural networks - RNN). V nadaljevanju predstavimo tipe nevronskih mrež, ki se pri razpoznavanju govora najpogosteje uporabljajo.

3.4.1 Ponavljajoče nevronske mreže

Ponavljajoče nevronske mreže (RNN) se od nevronskih mrež s povezavami naprej razlikujejo tako, da vsebujejo tudi povratne povezave. Ta lastnost jim omogoča, da ima mreža neke vrste spomin. Pri učenju na nevronsko mrežo tako ne vplivajo le trenutni vhodni podatki, ampak tudi predhodni podatki. To je zelo uporabno pri učenju jezikovnega modela razpoznavalnika govora, ko želimo napovedati naslednjo besedo glede na podane predhodne besede [18].

V preprosti ponavljajoči nevronski mreži z enim skritim nivojem s izračunamo vrednosti nevronov z naslednjimi enačbami [17, 18]:

$$x(t) = w(t) + s(t-1), \tag{3.6}$$

$$s_j(t) = g_1\left(\sum_i x_i(t) u_{ji}\right), \tag{3.7}$$

$$y_k(t) = g_2\left(\sum_j s_j(t) v_{kj}\right). \tag{3.8}$$

x predstavlja vhodni nivo, y izhodni nivo, w besedo na vhodu, u_{ji} in v_{kj} pa uteži povezav med nivoji. Funkcija g_1 je aktivacijska funkcija skritega nivoja, kjer uporabimo logistično sigmoido. Funkcija g_2 je funkcija softmax, kjer vrednosti izhodnega nivoja pretvorimo v verjetnostno distribucijo:

$$g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}. \tag{3.9}$$

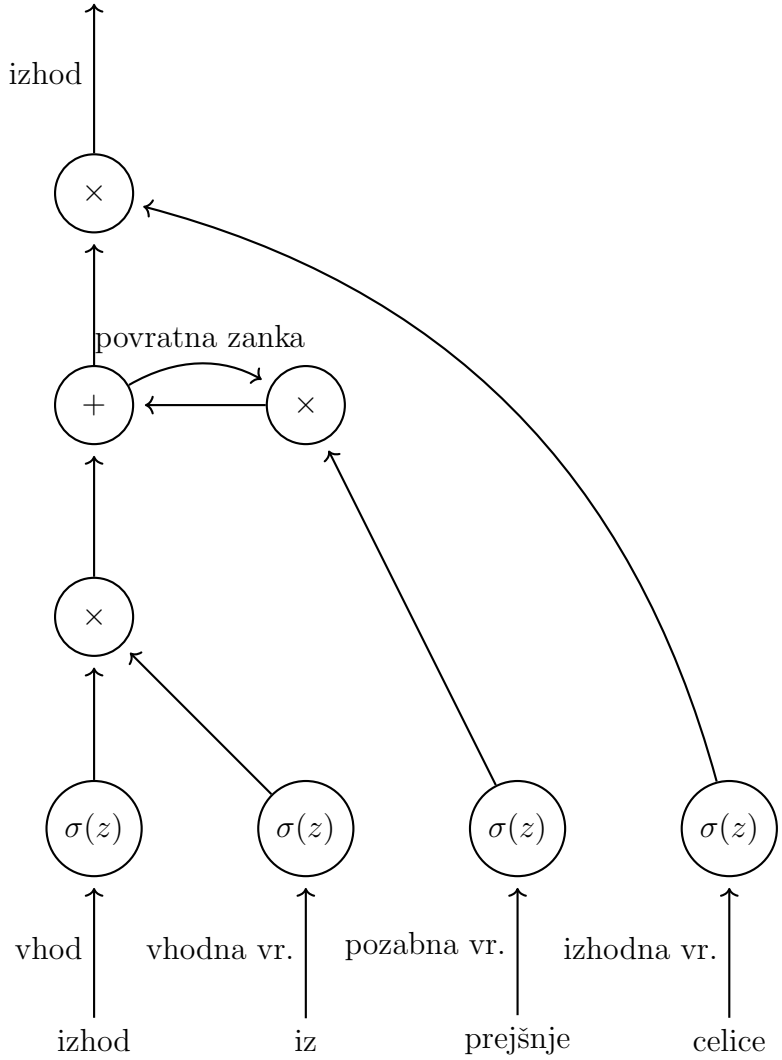
3.4.2 LSTM

Vrsta ponavljajočih mrež, ki se je v zadnjem času v praksi uveljavila z najboljšimi rezultati, so mreže z dolgim kratkoročnim spominom (angl. long short-term memory - LSTM). Težava RNN je izginjajoč gradient, saj se pri učenju dolgotrajne odvisnosti porazgubijo. Mreže LSTM so bile oblikovane z namenom, da rešijo ta problem [8]. Poleg spomina, kot ga imajo ponavljajoče

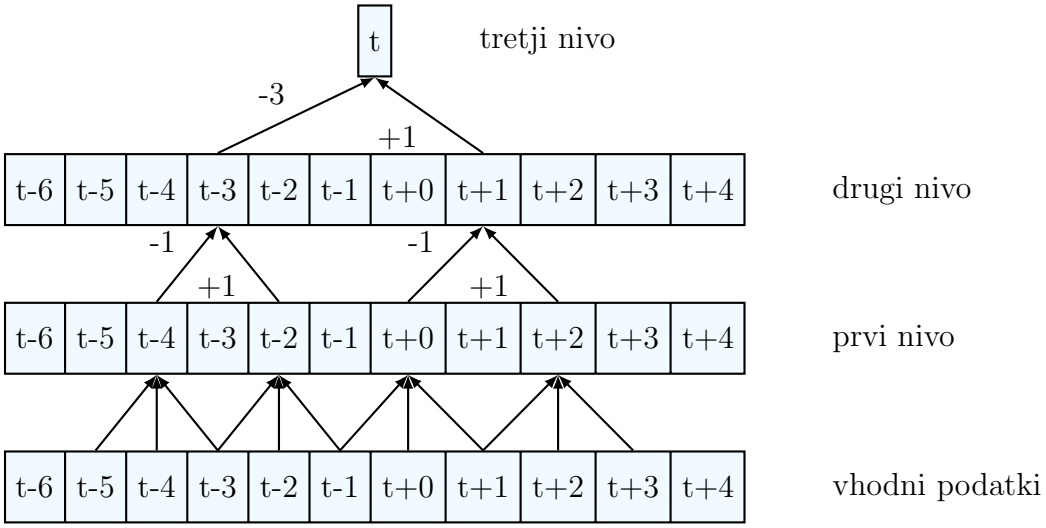
mreže, imajo mreže LSTM možnost podatke tudi pozabiti. Mreža LSTM je sestavljena iz posameznih celic, ki so med seboj povezane s povezavami naprej in s povratnimi povezavami. Vsaka posamezna celica je sestavljena iz več enot (slika 3.6). Poleg vhoda, aktivacijske funkcije in izhoda ima celica tudi povratno zanko znotraj celice ter troje vrat, ki utežujejo posamezne dele celice. Vhodna vrata dajo utež vhodnim podatkom celice. Izhodna vrata dajo utež izhodu iz celice. Pozabna vrata utežijo povratno zanko znotraj celice. Vsa vrata imajo sigmoidno aktivacijsko funkcijo, podatki na vhodu pa imajo poljubno (nelinearno) aktivacijsko funkcijo [9].

3.4.3 TDNN

Časovno zakasnjene nevronske mreže (angl. time delay neural network - TDNN) so vrsta nevronskih mrež s povezavami naprej, vendar se učijo na časovno širšem kontekstu. Skriti nivoji v globoki nevronski mreži združujejo informacijo iz prejšnjega nivoja, tako da nevroni v vsakem naslednjem nivoju upoštevajo večji časovni razpon. Na primer, če v vhodnem nivoju vsak nevron predstavlja eno časovno okno, bo nevron v prvem skritem nivoju predstavljal pet časovnih oken. Združil bo informacijo enega okna s po dvema predhodnima in dvema naslednjima oknoma (glej sliko 3.7). Nevron v drugem skritem nivoju na primer združi informacijo štirih nevronov iz prvega skritega nivoja: enega pri istem časovnem indeksu ter še enega z večjim in dva z manjšim časovnim indeksom (ali obratno) [22].



Slika 3.6: Sestava ene celice mreže LSTM [9].



Slika 3.7: Časovno združevanje v TDNN [22]. Na prvem nivoju se združi informacija pri časovnih indeksih t-1, t in t+1, glede na vhod. Na drugem nivoju se združi informacija pri časovnih indeksih t-1 in t+1. Na tretjem nivoju pri indeksih t-3 in t+1. Vsak nevron v tretjem nivoju vsebuje informacije devetih časovnih okvirjev, pri indeksih med t-5 in t+3.

Poglavje 4

Orodje Kaldi

Kaldi je odprtokodna zbirka orodij za gradnjo samodejnih razpoznavalnikov govora [23]. Kaldi je napisan v jeziku C++ in izdan pod licenco Apache 2.0. Sestavljen je iz več manjših programov, kjer ima vsak točno določeno, ozko vlogo. Te manjše programe, ki jih lahko obravnavamo kar kot funkcije knjižnice, povezujemo v delujoč sistem za razpoznavanje govora s skriptami napisanimi v Bashu, Pythonu in Perlu. Nekatere skripte vsebuje že okolje Kaldi, predvsem tiste, ki smiselno povezujejo delovanje sorodnih programov na nekem podproblemu. Ostale skripte, ki obravnavajo konkreten problem razpoznavanja govora, napišemo sami.

Kaldi se pri svojem delovanju naslanja na nekatere zunanje knjižnice in orodja. Nekatere se namestijo med nameščanjem Kaldija na naš operacijski sistem in jih ni potrebno nameščati ločeno. To so OpenFst; SRILM ali alternativa IRSTLM za izdelavo jezikovnega modela; knjižnice za linearno algebro ATLAS, CLAPACK ali OpenBLAS ter nekatera druga orodja, kot na primer sph2pipe za pretvorbo datotek formata sph v format wav.

Kaldi podpira paralelizacijo, učenje lahko poteka na več računalnikih hkrati. Najboljši način za uravnavanje paralelizacije na med seboj povezanih računalnikih je z uporabo programa Sun GridEngine. Kaldi podpira tudi druga podobna orodja, na primer Slurm. Uporaba paralelizacijskega orodja je sicer koristna tudi pri učenju na enem samem računalniku, saj tako

lažje uravnavamo število nalog, ki se izvajajo hkrati ter s tem poskrbimo za smotrno uporabo sistemskih virov. Brez uporabe teh orodij nam lahko zmanjka delovnega pomnilnika na računalniku, saj ga nekateri postopki zahtevajo ogromno, sploh če se vse naloge izvajajo hkrati.

Kaldi omogoča učenje akustičnega modela na značilkah MFCC in PLP. Učenje poteka na modelih GMM-HMM (GMM = mešanica Gaussovih porazdelitev, HMM = prikriti Markovovi modeli). Na podlagi značilk, pridobljenih z učenjem na modelih GMM-HMM, lahko naučimo globoko nevronsko mrežo. Tipično se uporabi 40-dimenzijske značilke sestavljene iz značilk MFCC, LDA, MLLT in fMLLR. Učenje poteka na procesorju ali grafični kartici s CUDA podporo. Možno je učenje tudi na več grafičnih karticah hkrati.

Kaldi uporablja jezikovni model v formatu FST. Tega je možno naučiti direktno na podlagi podanega korpusa z uporabo zunanjih orodij SRILM ali IRSTLM. Če že imamo jezikovni model v pogosto uporabljenem formatu ARPA, Kaldi ponuja orodje za pretvorbo jezikovnega modela ARPA v format FST. Omenjeni postopki se navezujejo na jezikovne modele zgrajene z n-grami. Projektu Kaldi je bil priključen tudi projekt RNNLM Toolkit [18], ki omogoča učenje jezikovnega modela z ponavljajočimi globokimi nevronskimi mrežami (RNNLM).

Poglavje 5

Arhitektura in učenje razpoznavalnika

V tem poglavju predstavimo učne podatke, ki smo jih uporabili za učenje razpoznavalnika govora in povemo, kako smo te podatke obdelali. Opišemo uporabljene postopke pri učenju akustičnega in jezikovnega modela.

5.1 Učni podatki

Učenje dobrega razpoznavalnika govora zahteva mnogo učnih podatkov. Potrebujemo posnetke govora, njihove prepise, korpus in slovar besed z izgovorjavami. Za govorne posnetke smo uporabili korpuse Gos, Gos VideoLectures 2.0 (GosVL) in Sofes. Lastnosti govornih korpusov so opisane v tabeli 5.1.

Tabela 5.1: Lastnosti govornih korpusov.

Lastnost	Gos	GosVL	Sofes(mik)
Dolžina posnetkov	120h	9h48min	9h52min
Število vseh govorcev	1526	44	134
Število ženskih govork	681	17	32
Število moških govorcev	845	27	102
Število stavkov	96334	4073	12536

Korpus Sofes ima nekatere stavke podvojene, vendar so le-ti posneti v različnih kvalitetah. Posnetki v visoki kvaliteti imajo datoteke oblike `*m.wav`, posnetki v nizki kvaliteti pa `*t.wav`. Posnetke nizke kvalitete smo pri učenju izločili. Korpusa Gos in Sofes smo razdelili na učno in validacijsko množico. Posnetki iz obeh korpusov so prisotni tako v učni, kot v validacijski množici. Razdelili smo ju tako, da se nihče izmed govorcev ne pojavi v obeh množicah, vedno le v eni. Korpus GosVL smo uporabili za testno množico.

Korpusa Gos in Sofes imata govor transkribiran na dva načina. V prvem so stavki in besede zapisani v zbornem knjižnem jeziku. V drugem pa je zapis fonetičen. Ta dva zapisa smo želeli uporabiti za izdelavo slovarja izgovarjav, vendar smo pri tem naleteli na težave. Korpusa imata različna fonetična zapisa, ki nista kompatibilna med sabo. Morali bi prevesti enega v drugega ali uporabiti le enega. Korpus Sofes je sam premajhen, vsebuje premalo različnih besed. Pri obdelavi podatkov v korpusu Gos pa smo našli neujemanja med obema transkripcijama. Ti nimata enakega števila besed, morda tudi ne vedno enakega vrstnega reda. Rezultati so bili neuporabni, zato smo se odločili, da uporabimo leksikon Sloleks. Prednost je, da vsebuje več besed, oziroma besednih oblik kot korpusa Gos in Sofes. Leksikon Sloleks vsebuje 1.129.141 različnih besed, korpus Gos pa je dolg 1.035.101 besed in vsebuje približno 83.000 različnih besed. Slabost pa je, da smo morali sami določiti fonetični zapis.

Besedilni korpus smo oblikovali na podlagi korpusa ccGigafida. Iz korpusa ccGigafida smo izločili vse prazne vrstice, odstranili večkratne presledke, oziroma jih nadomestili z enojnimi ter odstranili vsa ločila. Tako je učenje jezikovnega modela lažje. V nasprotnem primeru model upošteva ločila kot del besede, teh besed pa ni v slovarju. Prav tako se poveča število besed, ki bi jih morali upoštevati, korpus bi moral biti mnogo večji, učenje zahtevnejše in počasnejše.

5.2 Obdelava podatkov

Govorni korpusi imajo transkripcijo zapisano v datotekah `*.trs`, ki so oblika formata XML. Za učenje z uporabo orodij Kaldi, moramo najprej iz teh datotek izluščiti potrebne podatke. V ta namen smo napisali skripte, ki preberejo datoteke `*.trs` in vsakemu izreku pripišejo unikaten identifikator (ID), ID govorca, spol govorca ter datoteko zvočnega posnetka. Ti podatki se shranijo v različne datoteke. Datoteka `text` vsebuje v vsaki vrstici najprej identifikator izreka, nato sam izrek. Datoteka `utt2spk` vsebuje v vsaki vrstici identifikator izreka in identifikator govorca. Datoteka `spk2gender` vsebuje identifikator govorca in spol govorca (m za moški spol, f za ženski). Datoteka `wav.scp` vsebuje identifikator izreka in polno pot do zvočne datoteke. V določenih primerih je lahko v isti zvočni datoteki več izrekov. Vsi uporabljeni govorni korpusi imajo sicer zvočne posnetke razdeljene glede na posamezne izreke, vendar se pri korpusu GosVL ti ne ujemajo popolnoma. Število izrekov ni enako številu zvočnih posnetkov. Ročno ugotavljanje, kje pride do neujemanj, pa je zelo zamudno. Korpus GosVL ima zvočne posnetke razdeljene tudi na posamezna predavanja. Datoteke `.trs` vsebujejo podatke o časovni poziciji vsakega izreka znotraj zvočnega posnetka celotnega predavanja. Zato smo uporabili zvočne posnetke celotnih predavanj ter v datoteko `segments` za vsak izrek zapisali začetno in končno mesto (v sekundah) v zvočnem posnetku.

Korpus Gos ima pri nekaterih izrekih spol govorca označen kot „nedoločen“. V teh primerih smo najprej preverili ID govorca. Zadnja črka ID govorca namreč označuje spol govorca („m“ za moškega, „f“ za žensko). V večini primerov, ko je spol označen kot nedoločen, je zadnja črka ID govorca „n“. Takrat smo se odločili, da govorcu pripišemo ženski spol „f“, ker so to večinoma posnetki otrok, ki so po višini glasu bolj podobni ženskam, kot moškim. Nekateri izreki imajo več govorcev. Na primer, en govorec začne stavek, drugi ga dokonča. Ker ne vemo točno, kdaj govori kateri izmed govorcev, lahko pa se zgodi celo, da govorita hkrati, smo cel izrek pripisali enemu govorcu, tistemu, ki je v transkripcijski datoteki zapisan prvi.

Tabela 5.2: Pravila izgovarjav. Znak ! označuje poljuben samoglasnik, znak $ pa poljuben soglasnik. Oznaka -xy pomeni, da se pravilo upošteva le, če se beseda konča z xy. Oznaka xy- pomeni, da se pravilo upošteva le, če se beseda začne z xy. Kjer je več možnih izgovarjav, so te ločene s presledki.

Zapis	Izgovarjave
x	ks
q	k
ë	e
ï	i
sch	š
nj	n nj
lj	lj l
šč	šč š
w	v w
ch-	č- š- k-
-ch	-h
-il	-iw -iu -u -il
-el	-ew -el -u
-ol	-ou -ow -ol
-al	-aw -au -al
-rl	-rw
-!v	-!u -!f
!v$	!u$
v	w u
-v$	-w$ -u$
-$v	-$w -$u
y!	j!
y	i
!y	!j
-$y	-$i

Sloleks vsebuje nekaj več kot 100.000 lem, oziroma skupaj 2.791.919 besednih oblik. Odstranili smo podvojene vnose, na primer samostalnik „miza“ ima enako obliko v rodilniku ednine ter imenovalniku in tožilniku množine („mize“). V Sloleksu bi to bili trije vnosi, potrebujemo pa le enega, saj ne uporabljamo podatkov o spolu, številu, sklonu in podobno. Končno imamo 1.129.141 različnih besed. Izgovarjave teh besed smo tvorili s pomočjo pravil zapisa in pravil izgovarjave v Slovenskem pravopisu. Dodali pa smo še nekaj svojih pravil, ki bolje opisujejo pogovorni jezik, ne zgolj zborni knjižni jezik. Primer takega pravila je izgovarjava končnice „-el“ z glasom „-u“ (ne „-ew“). Z znakom „w“ smo tu označili vse oblike dvoustničnega u. Pravila so predstavljena v tabeli 5.2.

5.3 Učenje

Učenje akustičnega modela razpoznavalnika govora poteka v več fazah, ki se navezujejo ena na drugo (rezultat ene uporabimo pri učenju druge). V vsaki fazi smo uporabili drug model učenja, ki je kompleksnejši od prejšnjega. Iz zvočnih posnetkov smo najprej izračunali značilke MFCC in jih normalizirali z uporabo CMVN. Na teh značilkah smo učili naš sistem. Najprej smo uporabili različne modele GMM-HMM. Uporabili smo monofonski model učenja (mono), trifonski model z delta in delta-delta značilkami (tri1), trifonski model z LDA in MLLT značilkami (tri2b), trifonski model z LDA, MLLT in dodanim prilagajanjem govorcu (SAT) (tri3b). Rezultate modela GMM-HMM smo uporabili za osnovo učenja hibridnega modela DNN-HMM, kjer smo uporabili globoke nevronske mreže namesto GMM za napovedovanje fonema. Uporabili smo več različnih konfiguracij globokih nevronskih mrež.

Za učenje jezikovnega modela smo uporabili n-gramski model, kjer smo uporabili 3-grame, ter modele z ponavljajočimi nevronskimi mrežami.

Tabela 5.3: Parametri odločitvenega drevesa pri različnih akustičnih modelih. GPL je razmerje med številom komponent v GMM in številom listov v odločitvenem drevesu. Pri monofonskem modelu je odločitveno drevo trivialno, navedemo le število Gaussovih porazdelitev za primerjavo.

Učni model	število komponent v GMM	število listov	GPL
mono	1000		
tri1	12000	2000	6,00
tri2b	20000	3000	6,67
tri3b	30000	3500	8,57

5.3.1 GMM-HMM

Monofonski model smo učili le na podmnožici učnih podatkov. Vzeli smo le 20.000 najkrajših izrekov. Razlog za učenje na krajših izrekih je, da smo zagotovili boljše ujemanje med zapisom besed in zvočnim posnetkom le-teh. V učnih podatkih nimamo zapisa, kje natančno znotraj zvočnega posnetka se nahajajo posamezne besede, niti kje natančno so posamezni fonemi znotraj besede. Za dober model moramo vsak fonem opisati zgolj s tistim delom posnetka, kjer je fonem izrečen. To skušamo ugotoviti med samim učenjem, kar je lažje, ko so posnetki krajši. Po vsaki fazi učenja smo poskusili čimbolje poravnati zvočne posnetke s fonemi, oziroma v naslednjih fazah s trifoni. Te poravnave uporabimo kot osnovo pri učenju naslednje faze.

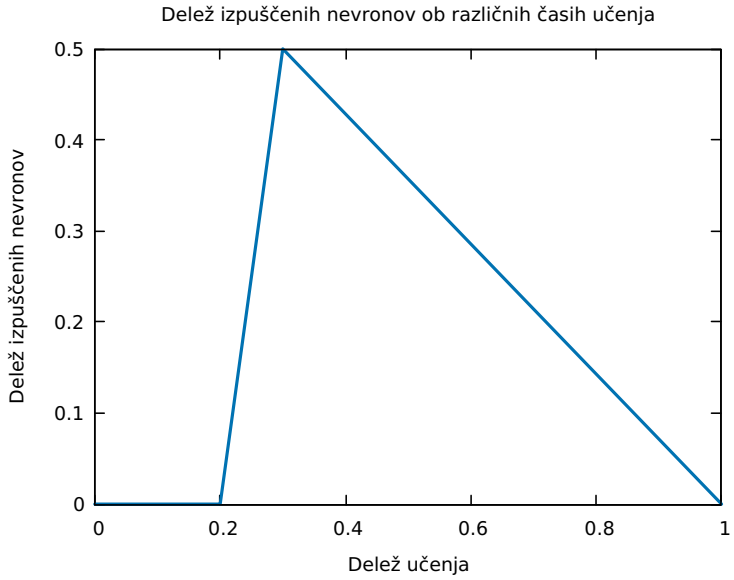
Pri učenju ostalih modelov smo uporabili celotno učno množico. Z vsako naslednjo fazo smo povečali število Gaussovih porazdelitev (komponent v GMM) in število stanj, oziroma listov v našem odločitvenem drevesu (glej poglavje 2), saj so modeli čedalje bolj kompleksni. Poizkusili smo več različnih parametrov in primerjali rezultate na validacijski množici. Rezultati so bili zelo podobni, občutno izboljšanje smo dosegli le pri močno povečanem številu stanj, predvsem pa številu komponent v GMM. Zaradi bojazni, da bi s tem model prenaučili, smo se odločili za manjše število Gaussovih porazdelitev in števila listov v drevesu. Uporabljeni parametri so navedeni v tabeli 5.3.

5.3.2 DNN-HMM

Pred učenjem globoke nevronske mreže, smo umetno povečali količino učnih podatkov tako, da smo spremenili hitrost posnetkov. Originalno hitrost smo pomnožili s faktorji 0,9, 1,0 in 1,1 ter dobili trikrat toliko podatkov. Dodatno smo zmanjšali izhodno vzorčno frekvenco modela na tretjino, tako da je vsak vzorec sestavljen iz treh podvzorcev. Efektivno izhod nevronske mreže ovrednotimo na vsakem tretjem (originalnem) vzorcu oziroma časovnem oknu. Nato zamaknemo podatke za en podvzorec in učimo ponovno. Končno zamaknemo podatke za še en podvzorec (skupno za dva podvzorca) in spet učimo. Nevronske mreže smo učili 5 dob, kar z zamikanjem vzorcev skupno nanese 15 dob. Za učenje nevronskih mrež smo povečali spektralno resolucijo. Število značilk MFCC smo dvignili s 13 na 40. Želeli smo uporabiti tudi značilke iVector, vendar smo pri določanju teh značilk naleteli na napako, ki je nismo uspeli odpraviti. Značilke iVector so vektorji dimenzije 100, ki vsebujejo podatke o lastnostih govorca. Vse mreže imajo tako vhodni nivo dimenzije 40.

Preizkusili smo več različnih konfiguracij mrež z različnim številom skritih nivojev in z različno povezanimi nivoji. Osredotočili smo se na mreži TDNN (opisana v razdelku 3.4.3) in LSTM (opisana v razdelku 3.4.2), saj naj bi ti dosegali najboljše rezultate.

Prva časovno zakasnjena mreža (`tdnn_1a`) na prvem skritem nivoju (nivo lda) združi časovni kontekst petih okvirjev pri časovnih indeksih, ki se razlikujejo za -2, -1, 0, 1 in 2 glede na opazovan časovni indeks. Nivo je polno povezan in je dimenzije 40. Sledi osem polno povezanih nivojev (nivoji tdnn1-8) dimenzije 512, kjer uporabimo izpuščanje. Delež nevronov, ki jih izpustimo se spreminja tekom učenja in je prikazan na sliki 5.1. Za aktivacijsko funkcijo uporabimo ReLU. V petih nivojih TDNN združimo časovni kontekst treh različnih okvirjev iz prejšnjega nivoja, kot je prikazano v tabeli 5.4. Sledi še en polno povezan nivo dimenzije 512 z aktivacijsko funkcijo ReLU, brez izpuščanja in nato izhodni nivo. Konfiguracijo smo povzeli po Kaldijevem projektu `vystadial_cz` za češki jezik[5].



Slika 5.1: Izpuščanje: delež izpuščenih nevronov skozi iteracije (0 - začetek učenja, 1 - konec učenja).

Tabela 5.4: Združevanje po času pri mreži `tdnn_1a` - upoštevani so našteti časovni indeksi prejšnjega nivoja, glede na trenuten časovni indeks t.

Nivo	časovni indeksi
lda	t-2, t-1, t+0, t+1, t+2
tdnn2	t-1, t+0, t+1
tdnn4	t-1, t+0, t+1
tdnn6	t-3, t+0, t+3
tdnn7	t-3, t+0, t+3
tdnn8	t-6, t-3, t+0

Tabela 5.5: Združevanje po času pri mrežah `tdnn_1d` in `tdnn_1e` - upoštevani so našteti časovni indeksi prejšnjega nivoja, glede na trenuten časovni indeks t.

Nivo	časovni indeksi
lda	t-1, t+0, t+1
tdnn2	t-2, t+0, t+1
tdnn4	t-1, t+0, t+2
tdnn6	t-3, t+0, t+3
tdnn8	t-3, t+0, t+3
tdnn10	t-6, t-3, t+0

Časovno zakasnjena mreža `tdnn_1c` je zelo podobna mreži `tdnn_1a`, le da vsebuje šest nivojev tdnn. Časovno združevanje je enako, le da pri tej konfiguraciji poteka pri nivojih lda, tdnn2, tdnn3, tdnn4, tdnn5 in tdnn6. Časovno zakasnjena mreža `tdnn_1b` je identična mreži `tdnn_1c`, le da ne uporabljamo izpuščanja nevronov. Časovno zakasnjeni mreži `tdnn_1d` in `tdnn_1e` vsebujeta 10 nivojev TDNN, kjer uporabljamo izpuščanje nevronov. Združevanje časovnih indeksov je opisano v tabeli 5.5. Nevronske mreže `tdnn_1a`, `tdnn_1b`, `tdnn_1c` in `tdnn_1d` smo učili 5 dob. Parameter število začetnih nalog smo nastavili na 2, število končnih nalog pa na 12. Pri nevronski mreži `tdnn_1e` smo oba parametra števila nalog nastavili na 1, kar naj bi bilo bolj optimalno, saj smo za učenje uporabljali eno grafično kartico. Število dob smo zmanjšali na 3, tako da smo približno ohranili število iteracij in čas učenja. Število iteracij je bilo pri konfiguraciji `tdnn_1e` nekoliko večje kot pri konfiguraciji `tdnn_1d`, čas učenja pa daljši, kljub manjšemu številu dob in enakim skritim nivojem.

Prva mreža LSTM (`lstm_1b`) ima enak vhodni nivo kot mreže TDNN, prav tako ima enak prvi skriti nivo. Sledijo štirje nivoji tipa LSTMP, dimenzije 1024. Nato imamo izhodni nivo. Nivo LSTMP se od LSTM razlikuje v tem, da izhod nivoja ne pelje nazaj na vhod istega nivoja, ampak sta vmes dva vzporedna projekcijska nivoja. Izhod enega projekcijskega nivoja pe-

lje na vhod nivoja LSTM, izhod drugega projekcijskega nivoja pa na vhod naslednjega skritega nivoja. Dimenzija projekcijskih nivojev je 256. Druga mreža LSTM (`lstm_1c`) ima po prvem skritem nivoju šest nivojev LSTM (ne LSTMP) dimenzije 512. Obe mreži LSTM smo učili 4 dobe.

5.3.3 Jezikovni modeli

Jezikovni modeli so v obliki končnih pretvornikov (angl. finite-state transducer - FST). Uporabili smo 3-grame z Witten-Bellovim glajenjem. Uteži končnih pretvornikov ponovno ocenimo z uporabo globokih nevronskih mrež. Preizkusili smo več različnih konfiguracij časovno zakasnjene mreže in mreže LSTM. Model s 3-grami smo naučili na besedilnem korpusu ccGigafida, modele z nevronskimi mrežami pa na besedilnem korpusu ccKres. Za korpus ccKres smo se odločili, ker so različni viri besedil (internet, revije, časopisi, leposlovje, ...) bolj enakomerno zastopani kot pri korpusu ccGigafida. Pravtako je korpus ccKres manj obširen, kar je pohitrilo učenje, ki je bilo pri korpusu ccGigafida dolgotrajno.

V poglavju 6 ovrednotimo tri konfiguracije nevronskih mrež, ki smo jih uporabili za učenje jezikovnega modela. Konfiguracijo `rnnlm_1a` sestavlja pet skritih nivojev, od tega trije nivoji TDNN in dva nivoja LSTM. Dimenzija vsakega nivoja je 800. Prvi skriti nivo je TDNN, kjer združimo časovna indeksa t in t-1. Sledi nivo LSTMP. Dimenzija projekcijskih nivojev je 200. Tretji skriti nivo je TDNN, kjer združimo časovna indeksa t in t-2, sledi še en enak nivo LSTMP. Zadnji skriti nivo je TDNN z združevanjem časovnih indeksov t in t-1. Opisano nevronsko mrežo smo učili 10 dob.

Konfiguracijo `rnnlm_1b` sestavljajo trije skriti nivoji tipa TDNN. V prvem in drugem skritem nivoju združimo časovna indeksa t in t-1, v tretjem skritem nivoju pa časovna indeksa t in t-2. Vsi nivoji so dimenzije 800. Nevronsko mrežo smo učili 15 dob.

Konfiguracijo `rnnlm_1c` sestavljajo trije skriti nivoji tipa LSTM. Vsi nivoji so dimenzije 512. Nevronsko mrežo smo učili 15 dob.

Poglavje 6

Rezultati in analiza

Uspešnost modelov pri razpoznavanja govora smo primerjali z rezultati na validacijski množici. Uspešnost podamo kot delež besed, ki jih moramo dejanskemu besedilu dodati (vriniti), izbrisati in zamenjati, da bi dobili razpoznano besedilo. Končna mera za uspešnost je WER (angl. word error rate), ki jo izračunamo kot:

$$WER = \frac{V + I + Z}{B}, \tag{6.1}$$

kjer V predstavlja število vrivanj, I število izbrisov, Z število zamenjav in B število vseh izgovorjenih besed v dejanskem besedilu.

Validacijska množica je sestavljena iz dveh korpusov, Gos (`dev_gos`) in Sofes (`dev_sofes`). Zaradi posebnosti obeh validacijskih množic, rezultate ločeno predstavljamo za vsak korpus posebej. Množica `dev_sofes` vsebuje veliko tujih lastnih imen, predvsem imen letališč in mest, zaradi česar pričakujemo slabše rezultate. Množica `dev_gos` vsebuje nekaj izrekov, kjer nastopata dva govorca hkrati. Transkripcija korpusa Gos zapiše dva ločena stavka, ki pa se navezujeta na isti zvočni posnetek. Ločena stavka moramo združiti v en izrek, pri čemer ne vemo, v katerem vrstnem redu sta stavka izrečena. Možno je tudi, da govorca govorita hkrati. V teh primerih se transkripcija ne ujema z zvočnim posnetkom. Primer takega izreka je v tabeli 6.1. Rezultati v teh primerih prikažejo slabše stanje od dejanskega. Če ovrednotimo transkripcijo T3 iz tabele 6.1 glede na transkripcijo T1 dobimo napako WER= $85,0$ %,

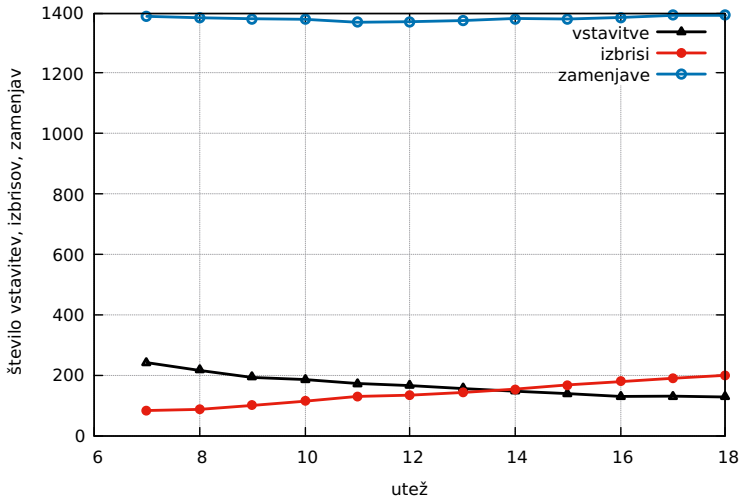
Tabela 6.1: Primerjava treh transkripcij izreka iz korpusa Gos, brez ločil. Primerjamo transkripcijo zapisano v korpusu Gos (T1), našo ročno transkripcijo (T2) in transkripcijo modela `tdnn_1a` (T3).

T1	„to je to je še pred volitvami bilo ja in sem takrat deloval kot državni sekretar na Ministrstvu za notranje zadeve kar v bistvu ni bila politična funkcija“
T2	„in sem takrat deloval kot državni sekretar na ministrstvu za notranje zadeve kar TO TO JE ŠE PRED VOLITVAMI JA politična funkcija“
T3	„in sem takrat deloval kot državni sekretar na ministrstvu za notranje zadeve kako je biti še pred volitvami je policija“

glede na transkripcijo T2 pa WER= $35,0$ %. Pri tem nismo upoštevali malih in velikih začetnic.

Metode opisane v razdelku 5.3 smo ovrednotili na omenjenih validacijskih množicah in na testni množici. Primerjali smo rezultate pri različnih utežeh med jezikovnim in akustičnim modelom. Z večanjem uteži jezikovnega modela, narašča število vrivanj, vendar pada število izbrisov (glej sliko 6.1). Število zamenjav se pri množici `dev_sofes` ne spreminja dosti v odvisnosti od uteži, pri množici `dev_gos` pa rahlo pada. Rezultate navajamo pri uteži, ki da najboljše rezultate na posamezni validacijski množici, rezultate na testni množici pa pri uteži, ki da najboljše rezultate na uniji obeh validacijskih množic. Rezultati najboljših modelov so prikazani v tabeli 6.2. Najboljši rezultat je dosegla kombinacija akustičnega modela `tdnn_1a` in jezikovnega modela `rnnlm_1a`, kjer na testni množici dosežemo WER=$27,16$ %.

V primerjavi z GMM-HMM, so vsi akustični modeli, naučeni z nevronskimi mrežami, dosegli boljše rezultate, razen konfiguracije `tdnn_1e`, kjer je rezultat na validacijski množici `dev_sofes` slabši. Glede na to, da je zgradba nevronske mreže v `tdnn_1e` enaka tisti v `tdnn_1d`, sklepamo, da je nismo dovolj dolgo učili, čeprav je bil čas učenja daljši. Mreža `tdnn_1d` doseže za $8,21$ % boljši rezultat na množici `dev_gos`, kot mreža `tdnn_1e`. Najboljša časovno zakasnjena mreža je mreža `tdnn_1a`, ki na validacijskih množicah pravilno razpozna $7,42$ %, oziroma $17,39$ % več besed kot model GMM-HMM. Na testni množici je izboljšanje $14,41$ %. Mreže LSTM so prinesle



Slika 6.1: Število vrivanj (črno), izbrisov (rdeče) in zamenjav (modro) v odvisnosti od uteži med jezikovnim in akustičnim modelom za validacijsko množico `dev_sofes` pri akustičnem modelu `tdnn_1a` in jezikovnem modelu `rnnlm_1a`.

manjše izboljšanje od časovno zakasnjenih mrež. Najboljša mreža LSTM, v primerjavi z najboljšo mrežo TDNN, pravilno razpozna $2,95$ % manj besed na množici `dev_gos` in $4,16$ % manj besed na testni množici. Pravilno pa razpozna $0,46$ % več besed na množici `dev_sofes`. Tu je razlika zelo majhna, zato ne moremo smiselno sklepati o prednosti mreže LSTM.

Razlike v uspešnosti med jezikovnimi modeli naučenimi z nevronskimi mrežami so zelo majhne. Pri obeh validacijskih množicah in pri testni množici so razlike v deležu pravilno razpoznanih besed med nevronskimi jezikovnimi modeli manjše od $0,5$ %. Najboljši nevronski jezikovni model `rnnlm_1a` v primerjavi s 3-gramskim jezikovnim modelom pravilno razpozna $1,96$ % več besed na množici `dev_gos`, $4,71$ % več besed na množici `dev_sofes` in $2,61$ % več besed na testni množici. Celotni rezultati so predstavljeni v dodatku A.

Rezultate našega sistema smo primerjali z Googlovim vmesnikom Google Cloud speech-to-text [10]. Naključno smo izbrali pet izrekov iz testne množice, jih transkribirali z Googlovim vmesnikom in transkripcije primer-

Tabela 6.2: Rezultati akustičnih modelov GMM-HMM (tri1), najboljše mreže TDNN in najboljše mreže LSTM pri 3-gramskem jezikovnem modelu ter akustičnega modela TDNN pri najboljšem nevronskem jezikovnem modelu (rnnlm_1a) .

model	dev_gos	dev_sofes	test
tri3b	70,69 %	36,72 %	44,18 %
tdnn_1a	53,30 %	29,30 %	29,77 %
lstm_1b	56,25 %	28,84 %	33,93 %
tdnn_1a + rnnlm_1a	51,34 %	24,59 %	27,16 %

jali z transkripcijami našega razpoznavalnika. Napaka pri izbranih izrekih je WER= $33,33$ % za Googlov sistem in WER= $21,28$ % za naš sistem, kjer smo uporabili akustični model `tdnn_1a` in jezikovni model `rnnlm_1a`. Googlov sistem ima manj vrivanj kot naš sistem, kar je prednost pri obotavljanju govorca. Mašila, kot sta „hmm“ ali „eee“ in prekinjene besede Googlov razpoznavalnik izpusti, medtem ko jih naš razpoznavalnik napačno zazna kot neke druge besede. Prednost našega razpoznavalnika v primerjavi z Googlovim je, da ima manjše število izbrisov in zamenjav. Torej zazna večje število izgovorjenih besed in te tudi bolj pravilno prepozna. Transkripcije Googlovega razpoznavalnika, našega razpoznavalnika in ročna transkripcija izbranih petih izrekov je prikazana v dodatku B v tabeli B.1 in tabeli B.2.

Podrobneje je Googlov vmesnik preizkusil David Čefarin v svojem diplomskem delu [3], kjer sistem dosega $61,72$ % točnost, oziroma WER=$38,28$ % pri prostem govoru. Naš razpoznavalnik ima napako WER=$27,16$ %, vendar omenjenih točnosti, oziroma napak Googlovega in našega sistema ni moč neposredno primerjati med seboj, saj smo našega ovrednotili na drugi testni množici, kot je David Čefarin Googlovega.

Naš razpoznavalnik je bil izdelan z namenom razpoznavanja splošnega, vsakdanjega govora. Pri učenju smo uporabili široko besedišče in veliko število govorcev, tako da razpoznavalnik ni specializiran za specifično področje uporabe (npr. zgolj v medicini ali zgolj za glasovno upravljanje nekega

programa) niti ni prilagojen na enega posameznega govorca. Prednost tega je, da razpoznavalnik dosega podobne rezultate ne glede na vsebino govora ali na to, čigav govor razpoznavamo. Slabost pa je, da dosega slabše rezultate od specializiranih razpoznavalnikov, ki pa so omejeni zgolj na ozko besedišče ali enega govorca. Pri učenju našega razpoznavalnika nam ni uspelo vključiti značilk iVector, kar pomeni, da je razpoznavalnik počasnejši pri sprotnem razpoznavanju govora, oziroma to ni mogoče. Pred razpoznavanjem moramo govor shraniti v zvočno datoteko, obdelava posameznih datotek pa je daljša, kot če bi sproti brali zvočni signal.

Poglavje 7

Sklepne ugotovitve

V magistrskem delu smo izdelali sistem za razpoznavanje slovenskega govora z metodami globokih nevronskih mrež. Za akustični model smo uporabili hibridni sistem DNN-HMM, kjer uporabimo globoke nevronske mreže za napovedovanje skritih stanj v HMM. Časovno zakasnjene nevronske mreže so se izkazale za bolj uporabne od mrež z dolgim kratkoročnim spominom, saj dosegajo nekoliko boljše rezultate, obenem pa sta učenje in transkribiranje precej hitrejša. Jezikovni model smo naučili z metodo n-gramov in z različnimi globokimi nevronskimi mrežami. Model z n-grami dosega slabšo natančnost, vendar je mnogo hitrejši pri transkribiranju. Med modeli z globokimi nevronskimi mrežami ni velikih razlik v uspešnosti. Naš sistem dosega boljše rezultate pri razpoznavanju tekočega govora od primerljivih razpoznavalnikov za slovenščino.

Naš razpoznavalnik je uporaben na vseh področjih, kjer visoka pravilnost razpoznavanja ni ključnega pomena. Pri učenju smo uporabili široko besedišče, zato je uporaben pri različnih tematikah, tako strokovnih kot pri vsakdanjem govoru. Slabost razpoznavalnika je njegova hitrost, saj nima možnosti sprotnega razpoznavanja govora.

Kljub uspešnemu razpoznavanju, je možnosti za izboljšave še precej. Na točnost razpoznavanja vplivata velikost učnih podatkov in arhitektura razpoznavalnika. Lahko bi vključili več govornih in besedilnih korpusov, prav

tako bi lahko drugače obravnavali težave korpusa Gos. Problematične izreke bi lahko popolnoma izpustili iz učne množice ali jih transkribirali na novo. Pri učenju jezikovnega modela bi uspešnost lahko izboljšali z upoštevanjem morfoloških podatkov in z uporabo večjega besedilnega korpusa. Obstaja veliko različnih možnih konfiguracij nevronskih mrež. V našem delu gotovo nismo našli najbolj optimalne, saj je možnosti preveč, učenje pa dolgotrajno.

Naš sistem bi lahko izboljšali z uporabo značilk iVector, ki nekoliko izboljšajo točnost razpoznavanja. Največja prednost teh značilk je, da se uporabljajo pri tekočem razpoznavanju govora. To pomeni, da ni potrebno vnaprej posneti celotnega zvočnega posnetka, ampak govor razpoznavamo sproti, med snemanjem.

Razpoznavalnik bi lahko razširili s sistemom, ki v besedilu avtomatsko postavi ločila. Tako besedilo bi bilo lažje berljivo in manj dvoumno, če bi tak sistem dobro naučili. Transkribiran govor bi potreboval manj ročnega urejanja, saj moramo pri uporabi našega razpoznavalnika ročno dodati tudi ločila.

Uporabniško izkušnjo bi lahko izboljšali z aplikacijo, ki z mikrofonom snema govor, sproti shranjuje posnetke in jih obdela za uporabo razpoznavalnika govora. Aplikacijo bi lahko uporabili tudi za že obstoječe posnetke govora.

Programska koda našega razpoznavalnika je dostopna na spletnem naslovu `https://github.com/MatejUlcar/kaldi/tree/slovenscina/egs/slovenscina`.

Dodatek A

Podrobni rezultati

V tem dodatku podajamo podrobne rezultate vseh modelov, opisanih v razdelku 5.3. Rezultati so podani za obe validacijski množici in za testno množico pri dveh utežeh med jezikovnim in akustičnim modelom. Prva utež je utež, pri kateri ima validacijska množica `dev_gos` najboljši rezultat, druga utež pa kjer ima validacijska množica `dev_sofes` najboljši rezultat.

Tabela A.1: Rezultati akustičnega modela `tri3b` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	3,08	27,93	39,67	70,69
	13	1,68	34,47	35,91	72,05
dev_sofes	8	6,93	2,68	29,31	38,92
	13	4,35	3,51	28,87	36,72
test	8	4,23	8,44	31,52	44,18
	13	2,38	11,68	30,37	44,43

Tabela A.2: Rezultati mreže `tdnn_1a` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,96	22,66	27,69	53,30
	11	1,85	27,99	25,09	54,93
dev_sofes	8	4,49	1,50	24,09	30,09
	11	2,90	2,31	24,08	29,30
test	8	3,17	6,26	20,34	29,77
	11	2,18	8,71	19,79	30,68

Tabela A.3: Rezultati mreže `tdnn_1b` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,97	22,86	31,54	57,37
	11	1,91	28,10	28,68	58,69
dev_sofes	8	4,72	1,77	25,51	31,99
	11	3,39	2,42	25,69	31,49
test	8	3,37	6,60	23,01	32,98
	11	2,39	9,09	22,18	33,66

Tabela A.4: Rezultati mreže `tdnn_1c` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,90	22,88	31,50	57,28
	12	1,73	29,73	28,09	59,54
dev_sofes	8	4,95	1,65	25,57	32,17
	12	3,05	2,64	25,64	31,33
test	8	3,37	6,55	22,78	32,70
	12	2,15	9,84	21,92	33,90

Tabela A.5: Rezultati mreže `tdnn_1d` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,96	22,71	31,64	57,32
	12	1,76	29,29	28,29	59,34
dev_sofes	8	4,88	1,72	25,41	32,01
	12	3,01	2,68	25,70	31,39
test	8	3,48	6,56	23,07	33,11
	12	2,23	9,80	22,14	34,16

Tabela A.6: Rezultati mreže `tdnn_1e` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,62	25,10	37,81	65,53
	9	2,21	26,90	36,59	65,71
dev_sofes	8	5,07	2,89	31,82	39,77
	9	4,70	3,11	31,67	39,48
test	8	3,82	7,84	30,20	41,86
	9	3,40	8,70	29,86	41,96

Tabela A.7: Rezultati mreže `lstm_1c` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,86	24,83	28,69	56,58
	10	1,95	29,41	25,96	57,32
dev_sofes	8	4,04	1,61	23,56	29,21
	10	2,90	2,23	23,62	28,75
test	8	3,47	8,21	22,35	34,03
	10	2,65	9,89	21,87	34,41

Tabela A.8: Rezultati mreže `lstm_1b` in 3-gramskega jezikovnega modela. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,77	24,79	28,70	56,25
	10	1,99	29,17	26,05	57,21
dev_sofes	8	3,61	1,74	23,55	28,90
	10	2,64	2,64	23,56	28,84
test	8	3,66	7,94	22,33	33,93
	10	2,75	9,65	21,73	34,12

Tabela A.9: Rezultati akustičnega modela `tdnn_1a` in jezikovnega modela `rnnlm_1a`. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,20	24,55	24,59	51,34
	12	1,83	27,26	22,97	52,06
dev_sofes	8	3,17	1,27	20,39	24,83
	12	2,43	1,96	20,20	24,59
test	8	2,18	7,41	17,57	27,16
	12	1,85	8,64	16,94	27,43

Tabela A.10: Rezultati akustičnega modela `tdnn_1a` in jezikovnega modela `rnnlm_1b`. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	8	2,26	24,45	24,87	51,58
	11	1,90	26,66	23,42	51,99
dev_sofes	8	3,11	1,34	20,47	24,92
	11	2,53	1,84	20,38	24,76
test	8	2,30	7,31	17,82	27,42
	11	2,01	8,34	17,25	27,60

Tabela A.11: Rezultati akustičnega modela `tdnn_1a` in jezikovnega modela `rnnlm_1c`. Rezultat je podan pri dveh različnih utežeh razmerja med jezikovnim in akustičnim modelom (najboljša za množico `dev_gos` in najboljša za množico `dev_sofes`).

množica	utež	vrivanja (%)	izbrisi (%)	zamenjave (%)	WER(%)
dev_gos	9	2,09	25,23	24,17	51,49
	11	1,90	26,64	23,32	51,86
dev_sofes	9	2,76	1,53	20,44	24,73
	11	2,37	1,86	20,28	24,51
test	9	2,15	7,63	17,47	27,25
	11	1,94	8,30	17,13	27,37

Dodatek B

Primerjava z Google Cloud speech-to-text

V poglavju 6 smo opisali primerjavo našega razpoznavalnika z Googlovim Cloud speech-to-text. V tabeli B.1 in tabeli B.2 so zapisane celotne transkripcije Googlovega razpoznavalnika, našega razpoznavalnika in ročne transkripcije, ki so del korpusa Gos VideoLectures. V tabelah je pet naključno izbranih izrekov, na katerih smo naredili primerjavo.

Tabela B.1: Primerjava z Google Cloud speech-to-text.

metoda	izrek
Google Cloud	dojemamo je pod neko metafizično kategorijo pa ne čeprav nikakor ne kot pojav ki bi se v času in prostoru spreminjajo in garantiram da se v času in prostor spreminja in tu drastično kar bomo dali v nadaljevanju tudi pokazal
naš sistem	dojemamo kot neko metafizično kategorijo pa le še prav nikakor ne kot pojav ki bi se v času in prostoru spreminjal in garantiram vam da se v času in prostoru spreminja in to drastično ker bomo dali moden v nadaljevanju tudi pokazal
ročno (GosVL)	dojemamo jo kot neko metafizično kategorijo a ne eem se pravi nikakor ne kot pojav ki bi se v času in prostoru spreminjal in garantiram vam da se v času in prostoru spreminja in to drastično kar bom v ndal() v na() v nadaljevanju tudi pokazal
Google Cloud	letno zdravstven skupaj menges organizatorju najprej zahvalil za povabilo na seminar
naš sistem	lep pozdrav vsem skupaj jaz bi se organizatorju najprej zahvalil za povabilo na seminar
ročno (GosVL)	lep pozdrav vsem skupaj jaz bi se organizatorju najprej zahvalil za povabilo na seminar
Google Cloud	karkoli je sledilo indukcijski valjar samo obsevanja lp akumulacije radio kemoterapije ni bilo važno
naš sistem	karkoli je sledilo indukcijski fazi ali samo obsevanje ali pa kombinacije rado kemoterapije ni bilo važno
ročno (GosVL)	karkoli je sledilo indukcijski fazi ali samo obsevanje ali pa kombinacija radiokemoterapije ni bilo važno

Tabela B.2: Primerjava z Google Cloud speech-to-text, nadaljevanje.

metoda	izrek
Google Cloud	dzs vedež predavanje mi je zelo hitro šlo skos
naš sistem	zdaj seveda prejšnjo predavanje mi je zelo hitro šlo skozi
ročno (GosVL)	zdaj seveda prejšno predavanje mi je zelo hitro šlo skozi
Google Cloud	njihov namen je bil čim bolj zmanjšati čim bolj zmanjšati tumorsko mass s tem namenom da v zda med obsevanjem ki je sledilo indukcijski fazi čim bolj zanesljivo obsevalno polje za javno mousse tumorske celice pa tudi če je gostota tumorskih celic manjša večja verjetnost da jih seveda z obsevanji tu tu ni češ
naš sistem	njihov namen je bil čim bolj zmanjšati čim bolj zmanjšati tumor s kom maso s tem vrata za namenom da v da med obsevanjem ki je sledilo indukcijski fazi čim bolj zanesljiva v obsevalno polje za Jamama se tumorske celice pa tudi če je gostota tumorskih celic zmanjša večjo verjetnost da jih seveda z obsevanjem tudi uničiš
ročno (GosVL)	njihov namen je bil čim bolj zmanjšati čim bolj zmanjšati tumorsko maso s teml() eee z namenom da v eee da med obsevanjem ki je sledilo indukcijski fazi čim bolj zanesljivo v obsevalno polje zajamemo vse tumorske celice pa tudi če je gostota tumorskih celic manjša je večja verjetnost da jih seveda z obsevanjem tudi uničiš

Literatura

[1] Alam, J., Kinnunen, T., Kenny, P., Ouellet, P., and O'Shaughnessy, D. (2013). Multitaper MFCC and PLP features for speaker verification using i-vectors. *Speech Communications*, 55:237–251.

[2] Bolka, A. (2016). Samodejno razpoznavanje fonemov slovenskega govora z uporabo zbirke orodij Kaldi. diplomsko delo, Univerza v Ljubljani, Fakulteta za elektrotehniko.

[3] Čefarin, D. (2016). Preizkus Googlovega govornega programskega vmesnika za slovenski govorni jezik. diplomsko delo, Univerza v Ljubljani, Fakulteta za elektrotehniko.

[4] Cheng, G., Peddinti, V., Povey, D., Manohar, V., Khudanpur, S., and Y., Y. (2017). An exploration of dropout with LSTMs. 18th Annual Conference of the International Speech Communication Association (INTERSPEECH 2017).

[5] Denisov, P. (2018). `https://github.com/kaldi-asr/kaldi/blob/master/egs/vystadial_cz/s5b/local/chain/tuning/run_tdnn_1a.sh`, dostopano 6. 8. 2018.

[6] Donaj, G. (2015). Avtomatsko razpoznavanje govora za pregibni jezik z uporabo morfoloških jezikovnih modelov s kontekstno odvisno strukturo. doktorska disertacija, Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko.

[7] Gales, M. and Young, S. (2007). The Application of Hidden Markov Models in Speech Recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304.

[8] Goldberg, Y. (2016). A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*, 57:345 – 420.

[9] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[10] Google. Google cloud text-to-speech. `https://cloud.google.com/speech-to-text/`, dostopano 18. 9. 2018.

[11] Harjani, D., Jethwani, M., and Roja, M. (2013). Speaker Recognition System using MFCC and Vector Quantization Approach. *International Journal for Scientific Research & Development*, 1(9).

[12] Harrag, A. and Mohamadi, T. (2011). PCA, SFS or LDA: What is the Best Choice for Extracting Speaker Features? *International Journal of Computer Applications*, 15(3).

[13] Hernandez, F., Nguyen, V., Ghannay, S., Tomashenko, N., and Estève, Y. (2018). TED-LIUM 3: twice as much data and corpus repartition for experiments on speaker adaptation.

[14] Kombrink, S., Mikolov, T., Karafiát, M., and Burget, L. (2011). Recurrent Neural Network based Language Modeling in Meeting Recognition. Interspeech 2011, 12th Annual Conference of the International Speech Communication Association.

[15] Krsnik, L. (2017). Napovedovanje naglasa slovenskih besed z metodami strojnega učenja. magistrsko delo, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko.

[16] Mikolov, T., Deoras, A., Kombrink, S., Burget, L., and Černocký, J. (2011a). Empirical Evaluation and Combination of Advanced Language Modeling Techniques. Interspeech 2011, 12th Annual Conference of the International Speech Communication Association.

[17] Mikolov, T., Karafiát, M., Burget, L., Černocký, J., and Khudanpur, S. (2010). Recurrent neural network based language model. 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010).

[18] Mikolov, T., Kombrink, S., Deoras, A., Burget, L., and Černocký, J. (2011b). RNNLM - Recurrent Neural Network Language Modeling Toolkit. ASRU 2011 Demo Session.

[19] Mirsamadi, S. and H L Hansen, J. (2015). A Study on Deep Neural Network Acoustic Model Adaptation for Robust Far-field Speech Recognition. 16th Annual Conference of the International Speech Communication Association (INTERSPEECH 2015).

[20] Nielsen, M. A. (2015). Neural Networks and Deep Learning. Determination Press.

[21] Oppenheim, A. V. and Shafer, R. W. (1999). *Discrete time signal processing, 2nd edition.* Prentice-Hall.

[22] Peddinti, V., Povey, D., and Khudanpur, S. (2015). A time delay neural network architecture for efficient modeling of long temporal contexts. Sixteenth Annual Conference of the International Speech Communication Association.

[23] Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The Kaldi Speech Recognition Toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Under-*

standing. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.

[24] Rao, K., Peng, F., Sak, H., and Beaufays, F. (2015). Grapheme-to-Phoneme Conversion Using Long Short-Term Memory Recurrent Neural Networks. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4225 – 4229.

[25] Rishi, C., Manisha, A., Karthik, R., and Rajesh Kumar, M. (2017). A text-independent speaker verification model: a comparative analysis. International Conference on Intelligent Computing and Control.

[26] Sak, H., Senior, A., Rao, K., and Beaufays (2015a). Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition. 16th Annual Conference of the International Speech Communication Association (INTERSPEECH 2015).

[27] Sak, H., Senior, A., Rao, K., Beaufays, F., and Schalkwyk, J. (2015b). Google voice search: faster and more accurate. Google AI Blog. `https://ai.googleblog.com/2015/09/google-voice-search-faster-and-more.html`, dostopano 10. 9. 2018.

[28] Stuttle, M. N. (2003). A Gaussian Mixture Model Spectral Representation for Speech Recognition. doktorska disertacija, Hughes Hall and Cambridge University Engineering Department.

[29] Xiong, W., Droppo, J., Huang, X., Seide, F., Seltzer, M., Stolcke, A., Yu, D., and Zweig, G. (2017). The microsoft 2016 conversational speech recognition system. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5255–5259.

[30] Žgank, A., Donaj, G., and Sepesy Maučec, M. (2014). Razpoznavalnik tekočega govora UMB Broadcast News 2014: kakšno vlogo igra velikost učnih virov? v: Zbornik 9. konference Jezikovne tehnologije, Informacijska družba - IS 2014.

[31] Žgank, A., Verdonik, D., and Sepesy Maučec, M. (2016). Razpoznavanje tekočega govora v slovenščini z bazo predavanj SI TEDx-UM. v: Zbornik konference Jezikovne tehnologije in digitalna humanistika.

[32] Zhang, X., Trmal, J., Povey, D., and Khudanpur, S. (2014). Deep Neural Network Accoustic Models Using Generalized Maxout Networks. Center for Language and Speech Processing & Human Language Technology Center of Excellence, The Johns Hopkins University.

[33] Zorrilla, A. L., Dugan, N., Torres, M. I., Glackin, C., Chollet, G., and Cannings, N. (2016). Some ASR Experiments using Deep Neural Networks on Spanish Databases.