Univerza v Ljubljani
Fakulteta za elektrotehniko

Govorne tehnologije

Projektna naloga-Poročilo

Razpoznavanje števk z LSTM omrežjem

Avtor: Iva Eftimska

Ljubljana, 2023

1. Uvod

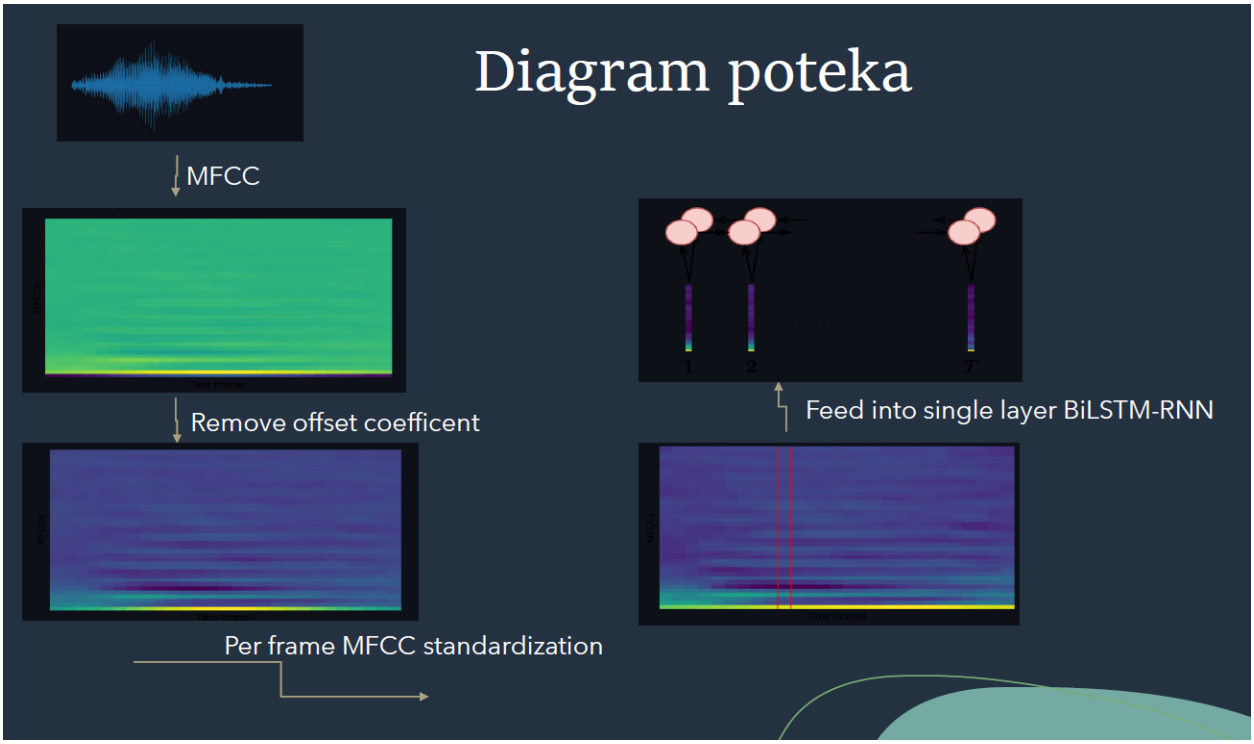
Razpoznavanje ročno napisanih števk z podatkovne baze MNIST je ena od začetnih nalog za ljudje ki se začnejo ukvarjati z globokim učenjem, zaradi tega kjer je baza pripravljena tako da se ne zgublja preveč časa z predoprocesiranjem podatkov. Razviti model za prepoznavanje števk, lahko pomaga ljudje v vsakdanjem zivljenju, kot je npr. avtomatsko branje nekega gesla ki je sestavljeno samo od števk, tako da ni potrebno posameznikov mučiti se razglabljati nekega rokopisa. Zaradi tega kjer je govor najpomemben del komunikacije med ljudji, so znanstveniki začeli razmišljati v to smer, da se lahko prepoznavajo tudi izgovorjenih števk. Cilj seminarske naloge je bila uporabiti LSTM omrežje na Free Spoken Digit Dataset, narediti nekaj eksperimentov tako da se doseže malo boljši rezultat in na koncu podati par zaključkov. Najprej bom opisala uporabljeno podatkovno bazo, potem sam postopek izdelave, arhitekturo modela in na koncu še analiza rezultatov.

2. Podatkovna baza

Free Spoken Digit Dataset, je analogija MNIST podatkovne baze. Baza je sestavljena iz 3000 posnetkov, ker govorcev na vsak posnetek izgovarjajo eno števko. Posnetke so ustrezno tudi označene. Števk izgovarjajo skupaj 6 govorcev v angleščini. Imamo 50 posnetkov za vsako števko po govorec. Posnetki so wav file, frekvenco 8 kHz in so izrezani tako da na začetku in konec posnetka ni velikega šuma. Raznolikosti v bazi je dosežena, ker govorcev izgovarjajo števke z različnimi akcenti. Podatkovno bazo sem razdelila 80% posnetkov v učno množico, 10% v validacijsko in ostalih 10% v testno množico. Posnetke pri razdeljevanju so bili naključno izbrani in sem pazila da v učni množici imam približno enako zastopane različne števke, tako da se bo potem model korektno naučil in biti uspešen razpoznavati tudi posnetkov iz testne množice.

3. Potek izdelave

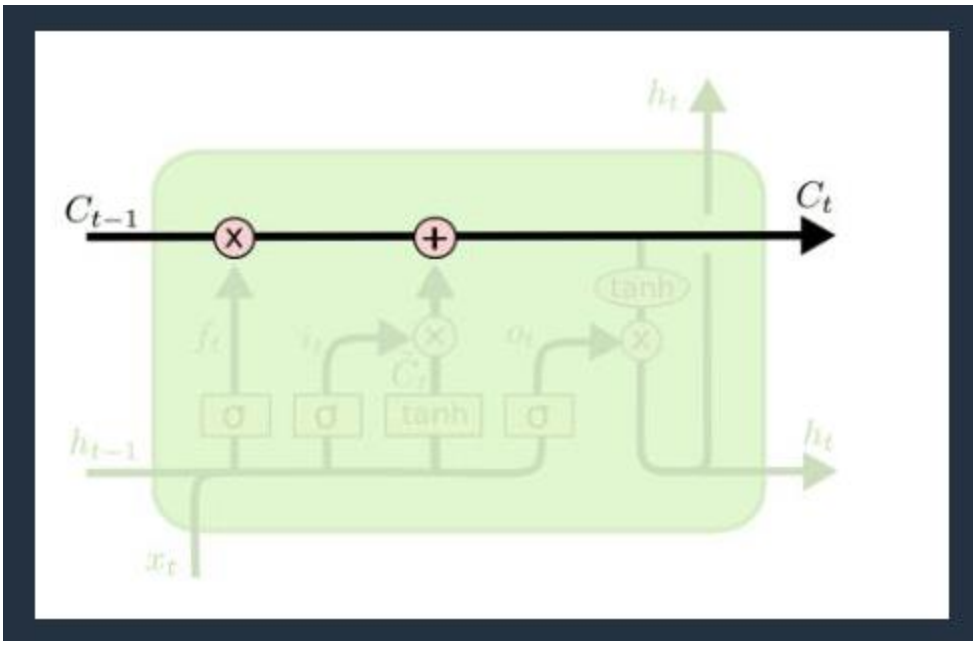
Najprej je bilo potrebno iz posnetkov, izluščiti akustične značilke, v mojem primeru sem uporabila MFCC značilke. MFCC značilke so zelo popularni v področju razpoznavanje govora, ker predvsem reproducirajo slišni sistem ki ga ima človek, to je da človek je precej občutljiv na razlike pri nizkih frekvencah, ni pa občutljiv pri visokih frekvencah, tako da praktično ljudje težko zaznavamo (ali pa sploh ne) razliko med npr. 10000 Hz in 10500 Hz, zaznamo pa lahko razliko med npr. 500 in 1000 Hz. Obstajata skupaj 39 MFCC koeficientov, raziskovalce so tipično pri svojih študijah uporabljali 12-13 ker so se iskazali za zadostni pri dobivanju pomembne informacije iz akustičnega signala. Če kepstralni koeficient ima pozitivno vrednost to pomeni da je večina spektralne energije skoncentrirana v nizko-frekvenčnih območjih, če je pa negativna vrednost, večina spektralne energije je skoncentrirana v visoko-frekvenčnih območjih. V mojem primeru sem uporabila 39 MFCC značilk, uporaba večjih značilk, pomeni samo to da gremo bolj v specifičnih detaljih pri analizi signala. Potem odstranim offset MFCC koeficient ki dejansko ne prinaša nobene koristne informacije od signala, samo kot da je spekter premaknjen za neko konstantno vrednost. Potem se naredi v vsakem frame, predobdelavo MFCC značilk, oz. so predobdelane tako da je povprečje 0 in standardna deviacija 1. Po narejeni predobdelavi, vsak frame iz MFFC-jev koeficentov se pošljejo sekvenčno na vsaki iteraciji v inpute BiDirectional LSTM RNN. Sam postopek je prikazan na sliki 1.



Slika 1. Postopki izdelave seminarja

4. Model

LSTM omrežje je precej uporabljeno v NLP (Natural Language Processing) naloge, zaradi možnosti da lahko zapomnijo nek podatek dalj časa. LSTM je sicer bazirano na RNN omrežje, samo da je izboljšana verzija RNN-ja v tem da je bolj robusten na gradient vanishing problema, pa da deluje dobro z dolgimi stavki. LSTM in RNN se večinoma uporabljata kadar imamo podatke podani kot neki sekvenci npr. tekst. Z uporabo LSTM-ja lahko predvidemo katera bo naslednja beseda v nekem dolgem tekstu, glede na neke prejšnje besede ki se nahajajo daleč iz trenutne besede. Ključno pri LSTM-ja, je to da obstajata gates, ‘vrata ’, ki lahko odločijo katero informacijo bo prepuščeno naprej ali ne bo in to vbistvu omočoga pomnenje informacije dalj časa. Glavni element LSTM omrežja z gates in cell state je prikazan na sliki 2.



Slika 2. Osrednji element LSTM

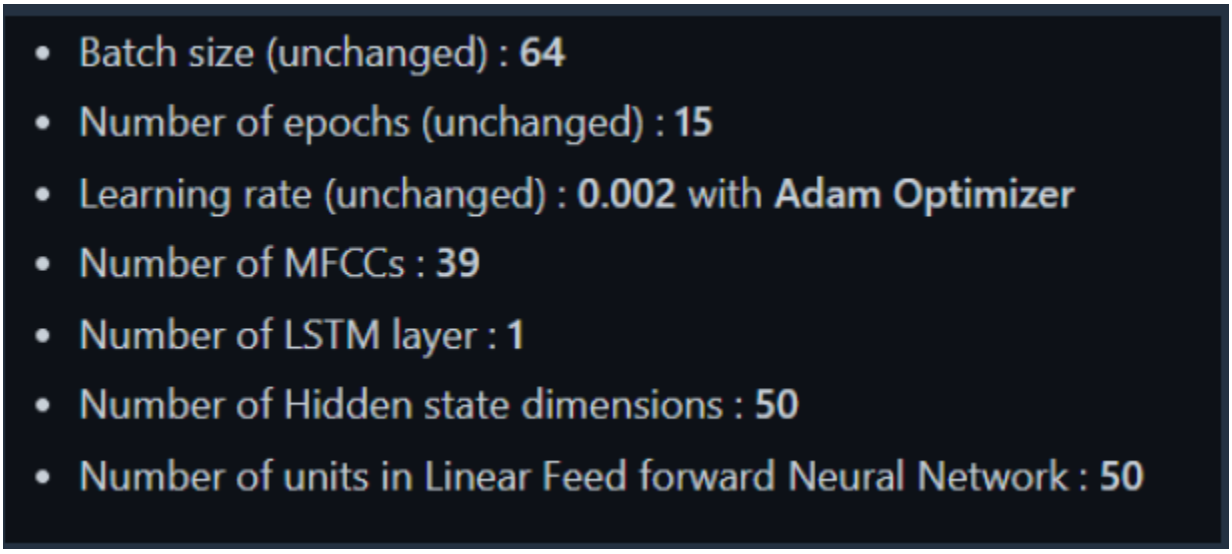
V moji seminarski sem uporabila biDirectional LSTM RNN, biDirectional pomeni, da so povezani dve hidden layers v nasprotnih smerah v enem outputu, to omogoča da izhodni layer pridobiva podatke, tako da se podatke obdelujejo naprej in nazaj. Sam model implementiran v Pytorchu je prikazan na sliki 3.

```
def __init__(self, n_mfcc, n_label, h, d, n_lstm):
    super().__init__()
    self.lstm_layer = nn.LSTM(input_size=n_mfcc, hidden_size=h, num_layers=n_lstm, batch_first=True, bidirectional=True)
    self.lstm_layer_dropout = nn.Dropout()
    self.linear_layer = nn.Linear(in_features=h*2, out_features=d)
    self.linear_layer_relu = nn.ReLU()
    self.linear_layer_dropout = nn.Dropout()
    self.output_layer = nn.Linear(in_features=d, out_features=n_label)
    self.output_layer_logsoftmax = nn.LogSoftmax(dim=1)

def forward(self, x, lengths):
    batch_size = len(x)
    x = pack_padded_sequence(x, lengths.to('cpu'), batch_first=True)
    x, (hn, cn) = self.lstm_layer(x)
    hn = self.lstm_layer_dropout(hn)
    hn = hn.transpose(1, 2).reshape(-1, batch_size).transpose(1, 0)
    hn = self.linear_layer_relu(self.linear_layer(hn))
    hn = self.linear_layer_dropout(hn)
    return self.output_layer_logsoftmax(self.output_layer(hn))
```

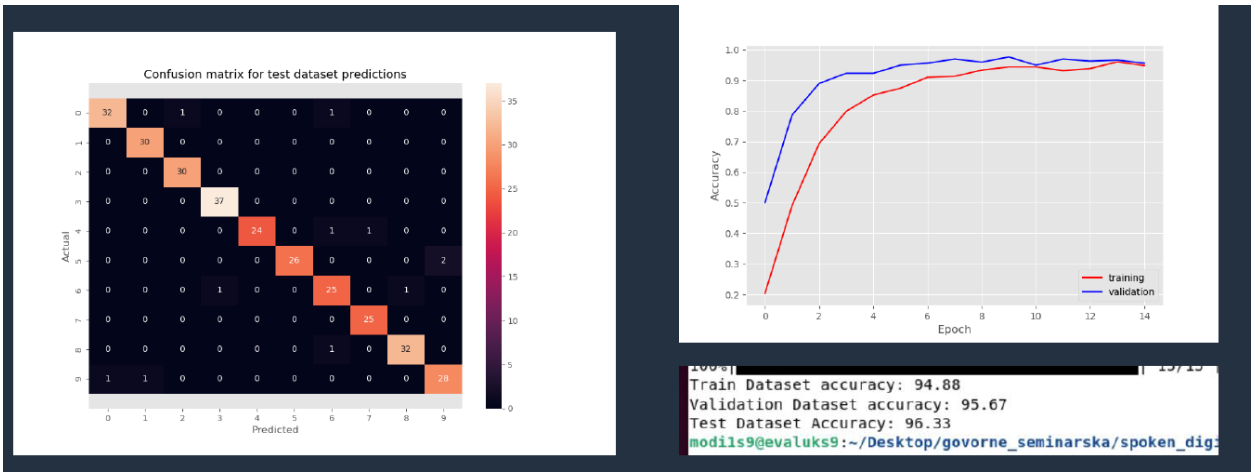
Slika 3. BiDirectional enoslojni LSTM RNN v Pytorch

Dropout layerje pomagajo pri zaščiti overfittanju modela, tako da se nekaj vhodnih nevronov lahko vržejo stran. Hiperparametre ki so prikazani na sliki 4 sem jih določila eksperimentalno.



Slika 4. Hiperparametrov pri LSTM modelu

Rezultate ki jih dobim so zadovoljive.



Slika 5. Rezultate modela

5. Zaključke

Izbiro 39 MFCC značilk vidimo da je upravičljiva zaradi dobljenih dobrih rezultatov, ki sem se sicer pri samem številu MFCC značilk, sklicevala po enem članku. Ker govorcev pri nekaterih posnetkih izgovarjajo številke tako da se foneme bolj vlečejo, bolj počasno izgovarjajo številke, pri drugih posnetkov pa bolj na hitro izgovarjajo številke, izbiro 39 MFCC značilk omogoča to da gremo v bolj specifičnih detaljov pri analizi akustičnega signala, še posebej to da tistih bolj razvlečenih izgovorjenih številk da jih model prepozna kot iste številke ki so pa izgovorjene hitro. Iz konfuziske matrike npr. vidimo da je 5, bila prepoznana dvakrat kot 9, odvisno od govorcev kako izgovarjajo številke, lahko se zgodi da se pet sliši kot 9, tako da so MFCC značilk prišli podobni številki 9 in da jih je model razpoznal kot 9. Poiskušanjem z različnim MFCC značilk nisem naredila, ker sem ze pri sami prvi izbiri, 39 značilk, dobila dobre rezultate. Vidimo da z ustrezno izbrano število MFCC značilk, ustrezna izbrana arhitektura modela, dobimo dobre rezultate na precej enostavno bazo kot je Free Spoken Digit Dataset.