Prepoznavanje emocija iz govora temeljem akustičkih značajki primjenom metodologije neuronskih mreža

Fanjek, Ivan

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Ivan Fanjek

Zagreb, 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Mentor:
doc. dr. sc. Tomislav Stipančić

Student:
Ivan Fanjek

Zagreb, 2021.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem mentoru doc. dr. sc. Tomislavu Stipančiću na pomoći i podršci prilikom izrade ovoga rada.

Zahvaljujem se obitelji i prijateljima na velikoj podršci i strpljenju koje su mi pružili za vrijeme izrade ovoga rada i za vrijeme trajanja cijeloga studija.

Ivan Fanjek

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za diplomske radove studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment,
inženjerstvo materijala te mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum:	Prilog:
Klasa: 602 - 04 / 21 - 6 / 1	
Ur. broj: 15 - 1703 - 21 -	

DIPLOMSKI ZADATAK

Student: **IVAN FANJEK** Mat. br.: 0035191813

Naslov rada na hrvatskom jeziku: **Prepoznavanje emocija iz govora temeljem akustičkih značajki primjenom metodologije neuronskih mreža**

Naslov rada na engleskom jeziku: **Speech emotion recognition based on acoustic features using a neural network methodology**

Opis zadatka:

Govor predstavlja prirodan način izražavanja kod ljudi. Emocije pomažu ljudima da bi kontekstualno shvatili jedni druge. Prepoznavanje emocija iz govora objedinjuje skup metodologija koje obrađuju i klasificiraju zvučne signale kako bi se otkrile uključene emocije.
Uz analizu lingvističkih značajki govora (eng. Natural Language Processing – NLP), prepoznavane emocija temeljem akustičkih značajki jedan je od temeljnih pristupa kod analize emocija iz govora. Akustičke značajke u govoru predstavljaju mjeru verbalnog iskaza govornika. Razvoj generalnog modela za prepoznavanje emocija iz glasa danas još uvijek predstavlja izazov zbog mnoštva faktora iz okoline koji unose smetnje u proces prepoznavanja.
U radu je potrebno istražiti te primijeniti dva pristupa kod identifikacije emocija iz akustičkih značajki govora koja su temeljena na različitim arhitekturama neuronskih mreža, uključujući LSTM (eng. Long Short Term Memory) te konvolucijsku neuronsku mrežu. Potom je potrebno analizirati svojstva rada korištenih mreža te usporediti dobivena rješenja koristeći konvencionalne alate za evaluaciju neuronskih mreža. Dobiveno softversko rješenje je potrebno eksperimentalno evaluirati uključivši ljudske subjekte.
U radu je potrebno navesti korištenu literaturu te eventualno dobivenu pomoć.

Zadatak zadan: 12. studenog 2020.

Rok predaje rada: 14. siječnja 2021.

Predviđeni datum obrane: 18. siječnja do 22. siječnja 2021.

Zadatak zadao: doc. dr. sc. Tomislav Stipančić

Predsjednica Povjerenstva: prof. dr. sc. Biserka Runje

SADRŽAJ

SADRŽAJ I
POPIS SLIKA II
POPIS TABLICA IV
POPIS OZNAKA V
POPIS KRATICA VI
SAŽETAK VII
SUMMARY VIII
1. Uvod 1
1.1. Što su emocije 1
1.2. Pokazivanje emocija kod ljudi 2
1.3. Umjetna inteligencija 4
1.4. Akustičke značajke govora 4
2. Zahtjevi Rada 7
2.1. Programski jezik *Python* 8
2.2. PyCharm 8
3. Teorijska podloga 9
3.1. Strojno učenje 9
3.2. Umjetne neuronske mreže 10
3.3. Konvolucijske neuronske mreže 14
3.4. Povratne neuronske mreže 15
3.5. Prenaučenost neuronskih mreža 18
4. Baza podataka za validaciju i učenje neuronskih mreža 21
4.1. AESDD baza podataka 21
4.2. CREMA-D baza podataka 21
4.3. Emo-DB baza podataka 22
4.4. RAVDESS baza podataka 22
4.5. SAVEE baza podataka 22
4.6. TESS baza podataka 23
4.7. Priprema baze podataka 23
4.8. Izdvajanje akustičkih značajki 27
5. Akustički model duboke neuronske mreže 30
6. Akustički model povratne neuronske mreže 37
7. Prepoznavanje emocija iz glasa 41
8. Testiranje programa prepoznavanja emocija 48
9. Zaključak 59
LITERATURA 61
PRILOZI 64

POPIS SLIKA

Slika 1 Vizualizirani tijek rada 7
Slika 2 Građa ljudskog neurona[13] 10
Slika 3 Nelinearni model neurona *k* [14] 11
Slika 4 Nelinearni model neurona *k* s uračunatim pragom [14] 12
Slika 5 Struktura višeslojne unaprijedne mreže [14] 13
Slika 6 Opći oblik konvolucijske neuronske mreže[17] 15
Slika 7 Osnovna povratna neuronska mreža [19] 15
Slika 8 Struktura LSTM jedinice [20] 16
Slika 9 Struktura dvosmjerne LSTM-e mreže [21] 18
Slika 10 Graf točnosti kod ranog zaustavljanja [23] 19
Slika 11 Konceptualni prikaz metode isključivanja [24] 20
Slika 12 Učitavanje biblioteka 23
Slika 13 Učitavanje putanje baza podataka 24
Slika 14 Očitavanje emocija iz imena iz baze podataka AESDD 25
Slika 15 Spajanje baza podataka 26
Slika 16 Raspodjela emocija unutar baze podataka 26
Slika 17 Izvlačenje akustičnih značajki 27
Slika 18 Prikaz MFCC značajki 28
Slika 19 Prikaz Mel spektrograma 28
Slika 20 Spremanje akustičkih značajki 29
Slika 21 Učitavanje biblioteka za CNN mrežu 30
Slika 22 Dijeljenje podataka na skup za učenje i validaciju 31
Slika 23 Kod za *One Hot Encoding*, klase i zapis labela nakon kodiranja 31
Slika 24 Kod za proširivanje dimenzije ulaznih podataka i njihov oblik 32
Slika 25 Model CNN neuronske mreže 33
Slika 26 Softmax aktivacijska funkcija [35] 33
Slika 27 Podešavanje parametara za učenje neuronske mreže 34
Slika 28 Spremanje naučenog CNN modela 34
Slika 29 Graf gubitka kroz učenje CNN mreže 35
Slika 30 Graf točnosti kroz učenje CNN mreže 35
Slika 31 Matrica konfuzije CNN mreže 36
Slika 32 Učitavanje biblioteka za LSTM mrežu 37
Slika 33 Model dvosmjerne LSTM mreže 38
Slika 34 Učenje dvosmjerne LSTM mreže 38
Slika 35 Spremanje dvosmjernog LSTM modela mreže 38
Slika 36 Graf gubitka dvosmjerne LSTM mreže 39
Slika 37 Graf točnosti dvosmjerne LSTM mreže 39
Slika 38 Matrica konfuzije LSTM mreže 40
Slika 39 Učitavanje potrebnih biblioteka 41
Slika 40 Učitavanje CNN modela 42
Slika 41 Učitavanje LSTM modela 42
Slika 42 Stvaranje mape za spremanje audio zapisa 42
Slika 43 Priprema prostora i varijabli za program 43
Slika 44 Postavljanje parametara snimanja 43
Slika 45 Funkcija snimanja audio zapisa 44

Slika 46 Kod While petlje snimanja i obrade audio zapisa 46
Slika 47 Kod ispisa grafa emocija i spremanja tablice kategoriziranih emocija 47
Slika 48 Ženski govornik neutralno CNN 48
Slika 49 Ženski govornik neutralno LSTM 49
Slika 50 Muški govornik neutralno CNN 49
Slika 51 Muški govornik neutralno LSTM 50
Slika 52 Ženski govornik sreća CNN 50
Slika 53 Ženski govornik sreća LSTM 51
Slika 54 Muški govornik sreća CNN 51
Slika 55 Muški govornik sreća LSTM 52
Slika 56 Ženski govornik tuga CNN 52
Slika 57 Ženski govornik tuga LSTM 53
Slika 58 Muški govornik tuga CNN 53
Slika 59 Muški govornik tuga LSTM 54
Slika 60 Ženski govornik ljutnja CNN 54
Slika 61 Ženski govornik ljutnja LSTM 55
Slika 62 Muški govornik ljutnja CNN 55
Slika 63 Muški govornik ljutnja LSTM 56
Slika 64 Ženski govornik strah CNN 56
Slika 65 Ženski govornik strah LSTM 57
Slika 66 Muški govornik strah CNN 57
Slika 67 Muški govornik strah LSTM 58

POPIS TABLICA

Tablica 1 Odnos između emocija i držanja tijela [4] 3
Tablica 2 Izvadak iz tabličnog zapisa AESDD baze podataka 26
Tablica 3 Izvadak iz tabličnog zapisa baze podataka 29

POPIS OZNAKA

Oznaka	Mjerna jedinica	Opis oznake
b_f		Prag od „forget gate“ (f_t)
b_i		Prag od sigmoidnog sloja (i_t)
b_k		Prag neurona k
b_N		Prag od tanh sloja (N_t)
b_o		Prag od sigmoidnih vrata (O_t).
C_t		Memorijsko stanje LSTM ćelije
f_t		Izlazni podatci iz „forget gate“
$\boldsymbol{h_t}$		Izlazni podatci iz LSTM jedinice
$\boldsymbol{i_t}$		izlazni podatci iz sigmoidnog sloja
N_t		Izlazni podatci iz tanh sloja
O_t		Izlazni podatci iz sigmoidnih vrata
u_k		Težinska suma neurona k bez praga
v_k		Težinska suma neurona k
w_{k0}		Težinski faktor praga (b_k)
w_{km}		Težinski faktori neurona k
W_f		Matrica težinskih faktora „forget gate“ (f_t)
W_i		Matrica težinskih faktora sigmoidnog sloja (i_t)
W_N		Matrica težinskih faktora tanh sloja (N_t)
W_o		Matrica težinskih faktora sigmoidnih vrata (O_t).
x_0		Ulazni signal praga (b_k)
x_m		Ulazni signali neurona
X_t		Ulazni podatci u LSTM mrežu
$\boldsymbol{y_k}$		Izlazni signal neurona k
σ		Sigmoidna funkcija
φ		Aktivacijska funkcija

POPIS KRATICA

Kratica		**Opis**
CNN		*Convolutional Neural Networks* – Konvolucijske neuronske mreže
IDE		*Integrated Development Environment* – Integrirano Razvojno Okruženje
LPCC		*Linear Prediction Cepstral Coefficients* – Linearno predviđanje „Cepstral“ koeficijenata
LSTM		*Long Short-Term Memory* – Duga kratkoročna memorija
MFCC		*Mel Frequency Cepstral Coefficients* – „Cepstral“ koeficijenti Mel frekvencije
RNN		*Recurrent neural network* – Povratne neuronske mreže

SAŽETAK

Govor je primarni način komuniciranja između ljudi, a izražene emocije igraju veliku ulogu u razumijevanju konteksta poruke prenesene govorom. Kako se razvija komunikacija između ljudi i računala potrebno je omogućiti računalima prepoznavanje ljudskih emocija. U ovome radu izrađen je softver koji na temelju umjetnih neuronskih mreža iz govora prepoznaje izražene emocije. Umjetne neuronske mreže obrađuju akustičke značajke iz govora kako bi prepoznale emocije. Izrađena su dva različita modela umjetnih neuronskih mreža, te su oba modela trenirana na dostupnim bazama podataka. Dva izrađena modela su eksperimentalno evaluirana na ljudskim subjektima.

Ključne riječi: emocije, akustičke značajke, umjetne neuronske mreže, konvolucijske neuronske mreže, duga kratkoročna memorija.

SUMMARY

Speech is the primary means of communication between people, and expressed emotions play an important role in understanding the context of the message sent through it. As the communication between humans and computers develops it is necessary to enable machines to recognize human emotions. In this paper, a software has been developed which is based on artificial neural networks to recognize expressed emotions from speech. Artificial neural networks process acoustic features of speech to recognize emotions. Two different models of artificial neural networks have been developed, and both models have been trained on available databases. The two developed models were experimentally evaluated by human subjects.

Key words: emotions, acoustic features, artificial neural networks, convolutional neural networks, long short-term memory.

1. Uvod

Svaka osoba na svijetu pokazuje emocije. Kako bi mogli računalno očitati te emocije potrebno je shvatiti njihove podijele i značajke. Ujedno su potrebni računalni alati, kao neuronske mreže, s kojima bi na temelju značajki mogli prepoznati određene ljudske emocije. Ovaj rad je usmjeren na prepoznavanje emocija iz akustičkih značajki iz glasa koji je samo jedan izvor informacija o emocijama koje ljudi u danom trenutku pokazuju.

1.1. Što su emocije

Postoji više definicija emocija te se stručnjaci još danas prepiru oko definicija što su emocije. Prema Merriam-Webster-ovom rječniku definicija emocija glasi: „svjesna mentalna reakcija (kao što je ljutnja ili strah) subjektno doživljena kao snažan osjećaj uobičajeno upućen prema specifičnom objektu i tipično u prisustvu sa psihološkim i biheviorističkim promjenama u tijelu [1]. Glavne teorije o emocijama mogu se podijeliti u tri kategorije [2]:

- **Psihološke teorije** – predlažu da razlog nastanka emocija se nalazi u ljudskome tijelu.
- **Neurološke teorije** – predlažu da akcije unutar mozga dovode do emocionalnih reakcija.
- **Kognitivne teorije** – predlažu da misli i ostale mentalne aktivnosti igraju važnu ulogu pri formiranju emocija.

Emocije ovise o dva kuta gledanja, što ljudi osjećaju i što ljudi misle da osjećaju. Misli i osjećaji su povezani s načinom na koji ljudi žive i o okolišu koji ih okružuje, što su produkti kulture. Iako postoje razlike u pokazivanju emocija između kultura, postoje i sličnosti. Zbog tih sličnosti postoji šest univerzalnih emocija: ljutnja, strah, gađenje, iznenađenost, tuga i sreća, koje dijele sličnosti među kulturama [3].

Kako bi se emocije mogle označiti i prepoznati napravljeni su mnogi modeli koji se mogu podijeliti u dvije skupine: Kategorički modeli i Dimenzijski modeli.

Kategorički modeli prepoznaju emocije pomoću riječi koje označuju emocije ili pomoću označavajućih klasa. Modeli koriste ili šest baznih emocionalnih klasa (ljutnja, gađenje, strah, radost, tuga ili iznenađenost) ili koriste ekspresivne klase iz specifičnih domena kao dosada i zbunjenost. Svaka emocija u modelu ima specifični skup značajki koje izražavaju reakciju ili

provocirajuću okolnost. Kod istraživanja različitih područja koriste se različiti skupovi emocija. Na području edukacije predlažu se pet kategorija umjesto baznih emocija kao što su dosada, zbunjenost, radost, frustriranost i "*flow*", gdje "*flow*" označava emocionalno stanje potpune predanosti i uživanja u izvršavanju aktivnosti. Prednost modela je mogućnost automatskog predstavljanja ljudskih osjećaja jednostavno razumljivih emocionalnih etiketa [2].

Dimenzijski modeli obilježavaju emocionalne utjecaje u dimenzijskoj formi. U modelu, skup dimenzija povezuje različita emocionalna stanja u dvodimenzijskom ili trodimenzijskom prostoru, gdje dimenzije mogu biti: valencija, podražaj i utjecaj. Svaka emocija zauzima položaj u dimenzijskom prostoru modela. Dimenzija valencije definira pozitivnost ili negativnost emocija i ima raspon od neugodnih osjećaja do ugodnih osjećaja u okviru sreće. Dimenzija podražaja obilježava stupanj pobuđenosti koju emocija opisuje i u rasponu je od uspavanosti ili dosade do entuzijazma. Dimenzija utjecaja obilježava stupanj snage kao što je osjećaj kontrole nad emocijama. Dimenzijski modeli ne trebaju povezivati određeno emotivno stanje, nego identificira emociju u dvije ili tri dimenzije. Tako se dobiva finija podjela emocija koje imaju manje stope razlike od kategoričkog modela. Ovaj model je koristan za nalaženje razlika kod sličnih emocija dok kategorički model je bolji kod traženja određenih baznih emocija [2].

1.2. Pokazivanje emocija kod ljudi

Utjecaji emocija mogu se vidjeti na vanjštini osoba, ali postoje i unutrašnje promjene. Većina mjerenja unutrašnjih promjena kod emocija mjere se senzorima koji se spajaju na osobu. Takvim mjerenjem mogu se mjeriti električne promijene mozga, električni parametri kože, promijene brzine otkucaja srca, promijene brzine i dubine udisaja, temperaturna promjena tijela, promijene eklektičnog potencijala mišića i druge. Prednost mjerenja vanjskih promjena od unutrašnjih je što ne treba spajati osobu na senzore. S današnjim razvijenijim vizijskim i audio sustavima može se lakše i bolje obraditi podatke iz vanjskih promjena i značajki kao izrazi lica, držanja tijela i gestikulacija [4].

Izrazi lica, držanje tijela i gestikulacija su u zadnjem desetljeću vidjele povišen interes u prepoznavanju emocija jer se mogu lagano mjeriti. Smatra se da su vanjske promjene tijela također uključene u izražavanje emocija. Skupine mišića se skupljaju, a druge skupine otpuštaju ovisno o emociji. Na temelju toga možemo odrediti izraze lica, geste i držanje tijela

koji se pojavljuju kod sličnih emocija. U tablici 1 mogu se vidjeti neke ovisnosti između emocija te držanja tijela i gesta [4].

Tablica 1 Odnos između emocija i držanja tijela [4]

Emocije	Geste i Držanje
Sreća	Podignuta ramena, podignute ruke u zraku ili odmaknute od tijela, produženo tijelo
Interes	Bočno gibanje ruku i dlana, frontalno ispružene ruke
Iznenađenje	Desna/lijeva ruka su na glavi, dva dlana prekrivaju obraze, doticanje dvaju dlanova dok prekrivaju usta, odmahivanje glavom, pomicanje tijela unazad
Dosada	Podizanje brade (glava se giba unazad), urušeno držanja tijela, glava sagnuta na stranu, prekrivanje lica s obje ruke
Gađenje	Ramena pomaknuta unaprijed, glava gleda prema podu i gornji dio tijela urušen, ruke prekrižene ispred prsa, dlanovi uz tijelo
Bijes	Podizanje ramena, otvaranje i zatvaranje dlana, ruke ispružene ispred tijela, pokazivanje prstom, ramena u obliku kvadrata

Što se tiče vanjskih promjena, izraz lica je veliki pokazatelj emocija. Iako je jedan od najkraćih emocionalnih signala, lako je izričito razaznati po licu sedam baznih univerzalnih emocija: ljutnja, strah, gađenje, iznenađenost, tuga, sreća i prezir. Svaka od gore navedenih emocija stoji za skup relativnih emocija. Tako skup ljutnje može mijenjati jačinu od blage iživciranosti do bijesa i može mijenjati vrstu ljutnje kao smrknuti bijes, ogorčene ljutnje, hladne ljutnje i druge. Varijacije intenziteta unutar iste familije emocija mogu se vidjeti, ali se znanstveno ne mogu odrediti ako različite vrste emocija unutar iste familije imaju jedinstvene izraze lica [5].

Još jedan veliki pokazatelj emocija je govor. Iz govora se mogu izdvojiti lingvističke i akustičke značajke govora. Emocionalno stanje se često pokazuje korištenjem određenih riječi i gramatičkih izmjena. Lingvističke značajke uzimaju se iz izgovorenih riječi ili teksta metodama kao što su: traženje ključnih riječi (engl. K*eyword spotting*), modeliranje temeljem pravila (engl. *Rule-based modelling*), semantička stabla (engl. *Semantic trees*), uočavanje ključnih izraza (engl. *Key-phrase spotting*) i mnoge druge [6]. Akustičke značajke glasa mogu mnogo toga reći o emocijama koje osoba osjeća, te se uzimaju iz izgovorenih riječi.

One mjere varijacije u načinu na koji je nešto izgovoreno. Detaljnije će se proći akustičke značajke u poglavlju 1.4.

1.3. Umjetna inteligencija

Umjetna inteligencija je interdisciplinarna računalna znanost, koja se bavi izradom pametnih uređaja koji su sposobni obavljati zadatke koji uobičajeno zahtijevaju ljudsku inteligenciju. Njena najčešća područja primjene i istraživanja su na: robotici, strojnom učenju, računalnom vidu, obradi govora i automatskom zaključivanju. Umjetna inteligencija može se podijeliti u dvije šire kategorije: Uska umjetna inteligencija (engl. *Narrow Artificial Intelligence*) i Generalna umjetna inteligencija (engl. *Artificial General Intelligence*). Uska umjetna inteligencija obavlja zadatke unutar ograničenog konteksta i simulacija je ljudskoga mozga. Uobičajeno je usmjerena na obavljanje samo jednog zadatka izuzetno dobro i iako se čine inteligentne rade pod strogim ograničenjima. Neki od primjera uskih umjetnih inteligencija su: Google tražilica, algoritmi za prepoznavanje, uređaji pomoćnici kao Siri i Alexa, samoupravljivi automobili i mnogi drugi. Glavni alat za njihovu izradu su neuronske mreže. Generalna umjetna inteligencija cilj je umjetne inteligencije, gdje bi se napravio stroj koji bi mogao misliti i ponašati se kao čovjek. Morala bi moći iskoristiti svoju inteligenciju za rješavanje problema predstavljenih za nju. Trenutno to je umjetna inteligencije koju viđamo u filmovima. Kako je poznato takva inteligencija se i dalje istražuje i smatra se da će još duže vrijeme biti izvan dometa [7].

Kako bi se interakcija između ljudi i strojeva s umjetnom inteligencijom mogla prirodnije odvijati, potrebna je vještina prepoznavanja emocija. Ta vještina će omogućiti strojevima da razumiju ljudske emocija te da ih mogu sintetizirati na što sličniji način. Današnji algoritmi za prepoznavanje emocije pokušavaju što točnije prepoznati emocije iz značajki izraza lica i govora, s time da se većina radova usredotočuju na prepoznavanje emocija ili iz izraza lica ili iz govora. Prepoznavanje emocija iz govora vrši se pomoću lingvističkih ili akustičkih značajki govora ili kombinacijom obiju značajki [8].

1.4. Akustičke značajke govora

Kako bi se dobile akustičke značajke ljudskoga govora za računalnu analizu emocija, potrebno je prikupljanje govornog signala pomoću mikrofona. Prikupljeni signal se preko analogno/digitalnog pretvornika pretvara u digitalni signal. Digitalni signal se obrađuje u skup

vremenskih okvira (engl. *frames*) metodom blokiranja okvira (engl. *Frame blocking*). Vremenski okviri ne smiju biti ni prekratki ni predugi. Kod predugih vremenskih okvira gube se vremenski promjenjive karakteristike audio signala, a kod prekratkih okvira ne mogu se izvući akustičke značajke. Uzima se duljina okvira takva da je pogodna za brzu Fourierovu transformaciju (engl. *Fast Fourier Transformation*). Iz takvoga skupa okvira sada se mogu izvući akustične značajke poput [9]:

Stopa nultog prijelaza (engl. Zero Crossing Rate) – u vremenski diskretnim signalima prijelaz preko nulte točke dogodio se ako dva uzastopna uzorka imaju drugačiji algebarski znak. Mjeri se koliko puta je amplituda govornog signala prošla kroz nulu u zadanom vremenskom intervalu. Kako visoke frekvencije impliciraju visoku stopu nultog prijelaza, a niske frekvenciju nisku stopu, postoji jaka poveznica između raspodjele energije i frekvencije [9].

Kratkotrajna energija (engl. Short Time Energy) – amplituda govornog signala mijenja se kroz vrijeme. Bezvučni dijelovi govora većinom imaju puno manju amplitudu od zvučnih dijelova. Energija govornog signala koristi se kako bi reprezentirala amplitudne promijene. Kombinacijom stope nultog prijelaza i kratkotrajne energije identificiraju se zvučni signali govora [9].

Osnovna frekvencija (engl. Pitch) – većinom se opisuje je li zvuk visok ili nizak. Definira se kao stopa ponavljanja kompleksnog signala ili stopa kojom se vrhovi autokorelacijske funkcije ponavljaju. Postoje razne metode za izvlačenje osnovne frekvencije iz glasa od kojih je najčešća Autokorelacijska metoda (engl. *Autocorrelation method*). Osnovna frekvencija je niža kod muških glasova nego kod ženskih [9].

Spektralne značajke – koriste se najčešće za izvlačenje točnih emocionalnih stanja iz govornog uzorka. Koristi se za prikazivanje spektra frekvencija zvučnog signala. Najčešće se izvlače MFCC koeficijenti (Mel Frequency Cepstral Coefficients) i LPCC koeficijenti (Linear Prediction Cepstral Coefficients) [9].

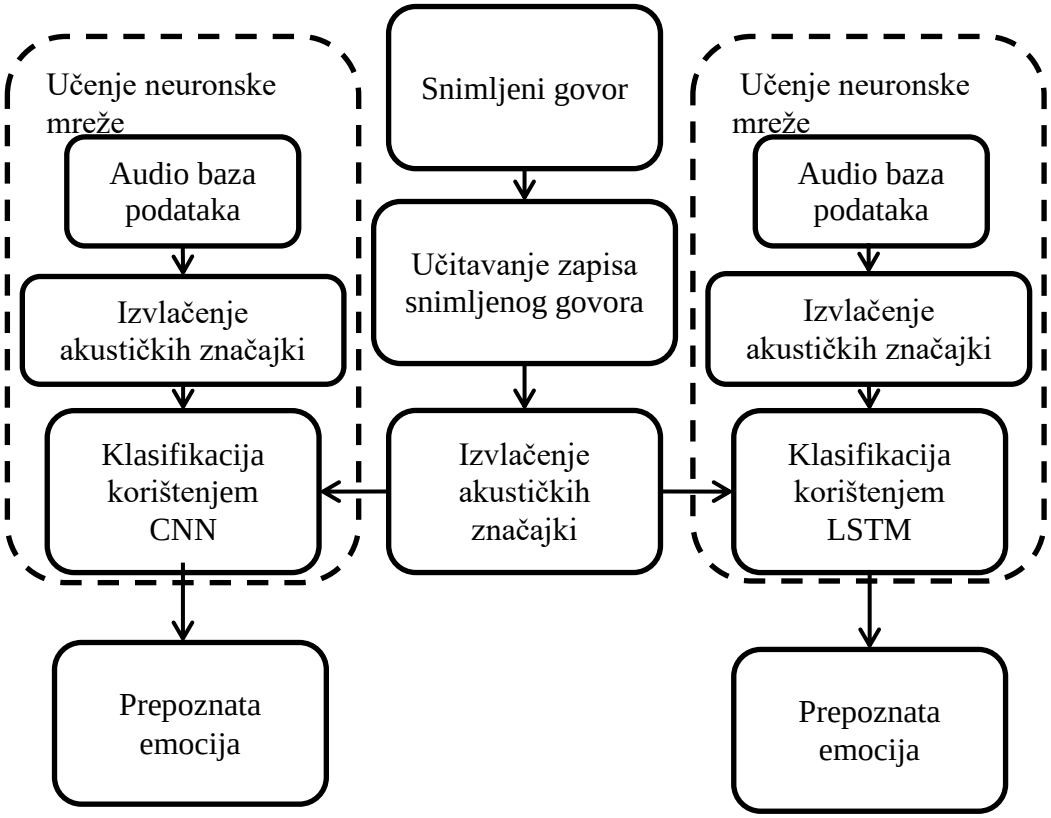
Značajka trajanja – modeliraju se na osnovu vremenskih aspekata, a osnovana jedinica su milisekunde (ms). Značajke se mogu razdvojiti po njihovom načinu izvlačenja na one koje predstavljaju vremenske aspekte drugih akustičnih elemenata i na one koje isključivo predstavljaju parametre trajanja viših fonoloških jedinica poput fonema, slogova, pauza, riječi i drugih. Vrijednosti trajanja često su ovisne o lingvističkim značajkama: na primjer funkcionalne riječi su kraće, dok sadržajne su dužeg trajanja [9].

Značajka energije (intenziteta) – uobičajeno se modelira glasnoća zvuka percipirana ljudskim uhom te se bazira na amplitudi u različitim vremenskim intervalima. Mogu se modelirati po intervalima ili karakterističnim točkama. Kako intenzitet podražaja raste, osjećaj sluha raste logaritmičko po decibel skali. Bitno je napomenuti da percepcija zvuka ovisi o prostornoj raspodijeli i duljini trajanja [8].

Značajke ne lingvističke vokalizacije – identificiraju neverbalne pojave kao disanje i smijanje. Programi za automatsko prepoznavanje govora zahtijevaju da se neverbalne pojave također uključene u vokabular kako bi se iz njih mogle izvući akustičke značajke [8].

2. Zahtjevi Rada

U ovom radu postupak prepoznavanja emocija iz govora zahtjeva da se iz snimljenog govora izvlače akustičke značajke te korištenjem klasifikatora prepoznaju emocije. Korišteni klasifikatori u ovom radu su konvolucijske neuronske mreže (engl. *Convolutional Neural Networks*, CNN) i duga kratkoročna memorija (engl. *Long Short-Term Memory*, LSTM). Neuronske mreže zahtijevaju bazu podataka iz kojih će se izvući akustičke značajke i izvršiti učenje neuronskih mreža. Potom je potrebno napraviti algoritam koji će iz snimljenog zvuka pomoću neuronskih mreža moći prepoznavati emocije. Na slici [Slika 1] prikazano je vizualno pojednostavljene tijek rada.



Slika 1 Vizualizirani tijek rada

Jedan od glavnih problema ovoga rada je izvlačenje akustičkih značajki iz snimljenog govora. Kod većine radova za izvlačenje emocija iz akustičkih značajki koriste se značajke MFCC koeficijenata i LPCC koeficijenata, osnovna frekvencija i energija. Emocije se prepoznaju na temelju jedne ili više značajki, ali u radovima se većinom kombiniraju rješenja fuzijskim

tehnikama (engl. *Fusion Techniques*), povezujući veliku količinu značajki [9]. U ovom radu će se koristiti značajka MFCC i značajka spektrograma po Mel skali. Ostale značajke se neće koristiti kako se ne bi neuronska mreža predimenzionirala.

2.1. Programski jezik *Python*

Python je besplatan, objektno orijentiran programski jezik visoke razine s dinamičkom semantikom. Sadrži ugrađene programske strukture, te dinamičko povezivanje koje ga čine povoljnim za razvijanje brzo razvijajućih aplikacija i skripti. Sintaksa je jednostavna i lako čitljiva što omogućuje lakše održavanje programa. Python ima veliku bazu modula i paketa, čime su omogućene modularnosti programa. *Python* tumač zajedno s njegovom standardnom bibliotekom dostupni su u izvornom ili binarnom obliku za sve poznate operativne sustave te se mogu besplatno dijeliti [10]. Danas je Python veoma rasprostranjen zbog toga što je besplatan i jednostavan s vrlo velikim mogućnostima zbog opsežne količine modula. Programski dio rada je rađen u Pythonu izdanja 3.8 pošto ima besplatne biblioteke potrebne za izradu zadatka.

2.2. PyCharm

PyCharm je Integrirano Razvojno Okruženje (engl. *Integrated Development Environment*, IDE) posvećeno programskom jeziku *Python*. Koristi se kako bi mogli napisati algoritme potrebne za izradu rada. Dolazi u tri izdanja a to su: *Community, Professional* i *Edu*. *Community* i *Edu* su besplatni dok se samo *Professional* izdanje plaća. Izabrano izdanje za rad je *Community*, gdje su protokoli svima dostupni [11].

3. Teorijska podloga

3.1. Strojno učenje

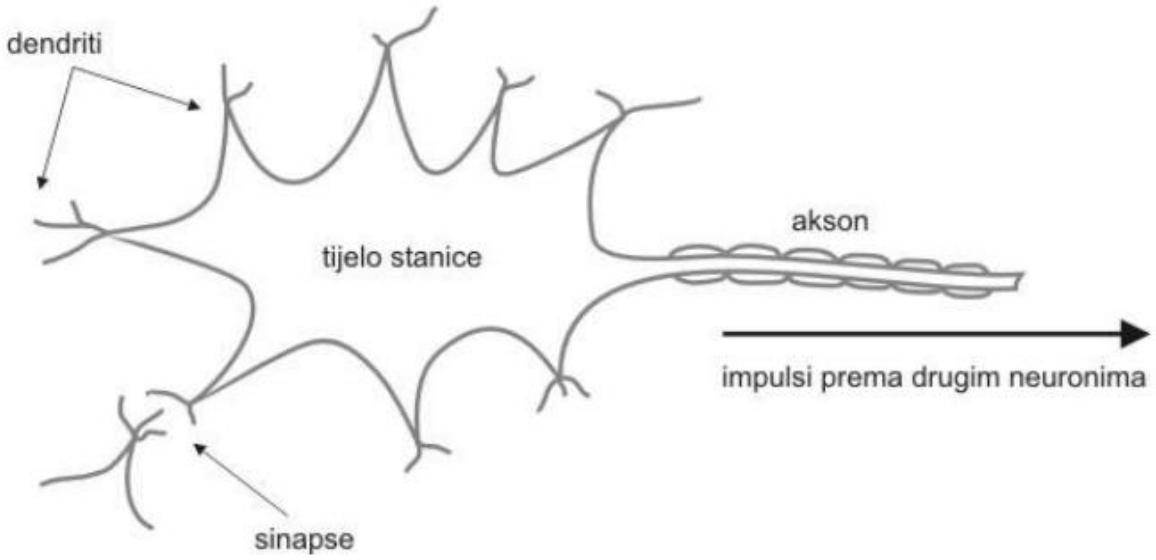
Strojno učenje je jedna od grana umjetne inteligencije koja se bavi izradom aplikacija s mogućnošću učenja iz ulaznih podataka. Velika prednost takvih aplikacija je povećanje točnosti kroz vrijeme bez dodatnog programiranja. Algoritmi strojnog učenja su programirani da nalaze uzorke i značajke iz velikog broja podataka kako bi na temelju novih podataka mogle napraviti odluke i predikcije. Što je bolje izrađen algoritam učenja, odluke i predikcije će biti točnije s obrađenim novim podatcima. Aplikacije koje imaju sposobnost strojnog učenja izrađuju se u četiri koraka [12]:

- Prvi korak – Odabir i priprema skupa podataka za učenje. Skup podataka za učenje mora predstavljati podatke koje će algoritam obrađivati kako bi riješio zadatke namijenjene za njega. Podatci za treniranje mogu biti označeni (engl. *labeled*) ili neoznačeni (engl. *unlabeled*). Označeni podatci su korišteni kada algoritam identificira značajke i odrađuje klasifikaciju, dok se neoznačeni podaci koriste kod izvlačenja značajki i samostalnog izvršavanja klasifikacije.
- Drugi korak – Izbor algoritma koji će biti učeni na odabranom skupu podataka. Vrsta algoritma ovisi o vrsti i količini podataka u skupu podataka za učenje i o problemu koji je potrebno riješiti. Uobičajeni algoritmi su: regresijski algoritmi (engl. *regression algorithms*), stabla odlučivanja (engl. *decision trees*), klastering algoritmi (engl. *clustering algorithms*), neuronske mreže (engl. *neural networks*) i druge.
- Treći korak – Učenje algoritma kako bi se napravio model. Postupak učenja je iterativan, gdje se varijabla provlači kroz algoritam te se uspoređuje izlaz iz njega s rezultatima koje je trebao dobiti. Kroz postupak se podešavaju težinske vrijednosti algoritma prolaskom svake varijable kako bi se dobili točniji rezultati. Postupak se ponavlja provlačeći varijable iz skupa podataka kroz algoritam sve dok algoritam ne proizvodi zadovoljavajuće rezultate.
- Četvrti korak – Korištenje i nadogradnja algoritma. Posljednji korak podrazumijeva korištenje algoritma s novim podacima, gdje je cilj da točnost mreže bude veća s vremenom kako prolazi kroz nove podatke.

3.2. Umjetne neuronske mreže

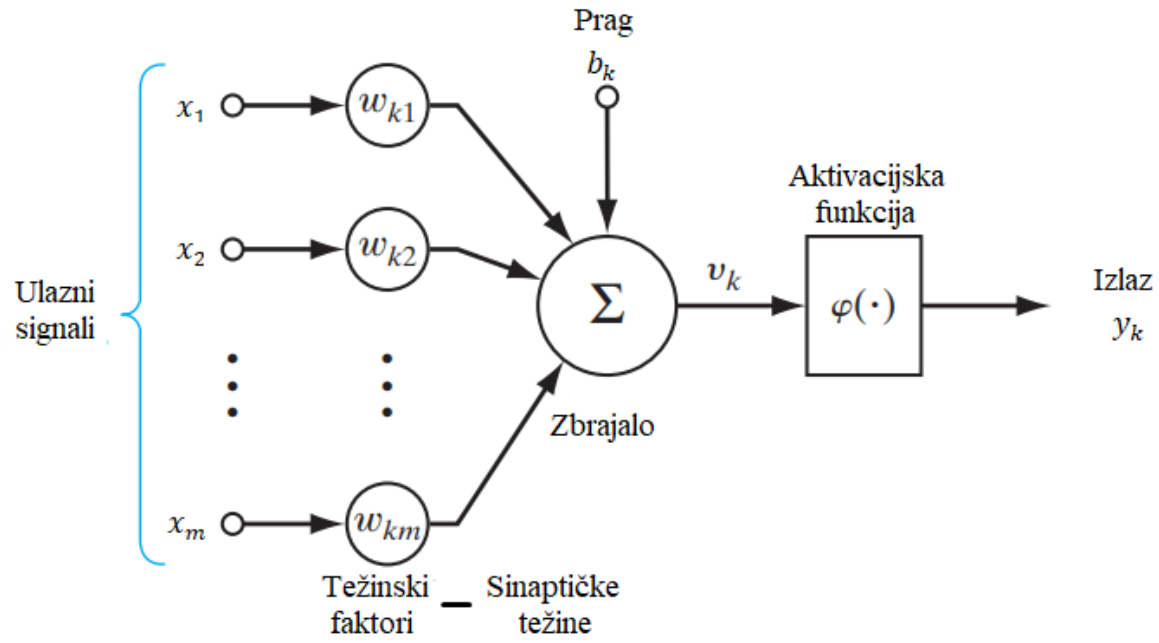
Umjetna neuronska mreža je dizajnirana da simulira metode ljudskoga mozga za rješavanje problema. Kako bi se lakše shvatila struktura neuronske mreže proći će se kratko kroz strukturu neurona.

Ljudski mozak se sastoji od 10^{11} neurona kojih ima više od 100 vrsti. Svaki neuron je povezan s drugih 10^{4} neurona. Neuroni se sastoje od četiri osnovna dijela: tijela stanice, aksona, dendrita i završnih članaka. Na slici [Slika 2] je prikazana građa ljudskoga neurona. Informacija sadržana u tijelu neuronske stanice je u obliku električnog potencijala između vanjskog i unutrašnjeg dijela stanice. Potencijal stanice se povećava ili smanjuje ovisno o informacijama koje prima od drugih neurona putem sinapsi, što su spojna sredstva neurona. Informacije su primljene u obliku post-sinaptičkog potencijala te se zbrajaju svi post-sinaptički potencijali iz susjednih neurona. Tako zbrojeni potencijal stvara napon u stanici te ako se prođe određeni prag zvan aktivacijski potencijal neuron otpušta neurotransmitere koji pokreću isti niz opisanih događaja [13].



Slika 2 Građa ljudskog neurona[13]

Neuroni su informacijsko-procesna jedinica koja je neophodna za rad neuronskih mreža. Na slici [Slika 3] je prikazan blok diagram modela neurona koja je baza za većinu neuronskih mreža [14].



Slika 3 Nelinearni model neurona *k* [14]

Iz modela [Slika 3] se mogu prepoznati tri osnovna člana modela neurona: težinski faktori, zbrajalo i aktivacijska funkcija. Skup težinskih faktora predstavlja skup sinapsi na neuronu, gdje svaka sinapsa je karakterizirana sa svojim vlastitim težinskim faktorom. Ulazni signal u neuron x_j sa sinapse *j* spojen na neuron *k* se množi s težinskim faktorom w_{kj}. Razlika između težinskih faktora u ljudskom neuronu i umjetnom neuronu je u tome što u umjetnom neuronu težinski faktori mogu sadržavati i negativne vrijednosti. Zbrajalo zbraja signale obrađene težinskim faktorima. Aktivacijska funkcija ograničava amplitudu neurona te je kod umjetnih neurona je normalizirana na vrijednosti između [0,1] ili, ako sadržava negativne vrijednosti na [-1,1]. Na slici [Slika 3] je prikazan model neurona koji prikazuje i prag b_k koji snižava ili povisuje težinski zbrojeni ulaz aktivacijske funkcije [14]. Matematički se može zapisati neuron *k*:

$$u_k = \sum_{j=1}^{m} w_{kj}\, x_j \qquad \textbf{(1)}$$

i

$$y_k = \varphi(u_k + b_k) \qquad \textbf{(2)}$$

Ulazni signali modela su x_1, x_2,..., x_m; Težinski faktori neurona k odgovarajućih ulaznih signala su w_{k1}, w_{k2},..., w_{km}; Težinska suma bez uključenog praga je označena sa u_k, a aktivacijska funkcija $\varphi(.)$; b_k označuje prag i y_k označuje izlazni signal iz neurona k.

Težinska suma neurona dana je izrazom:

$$v_k = u_k + b_k \tag{3}$$

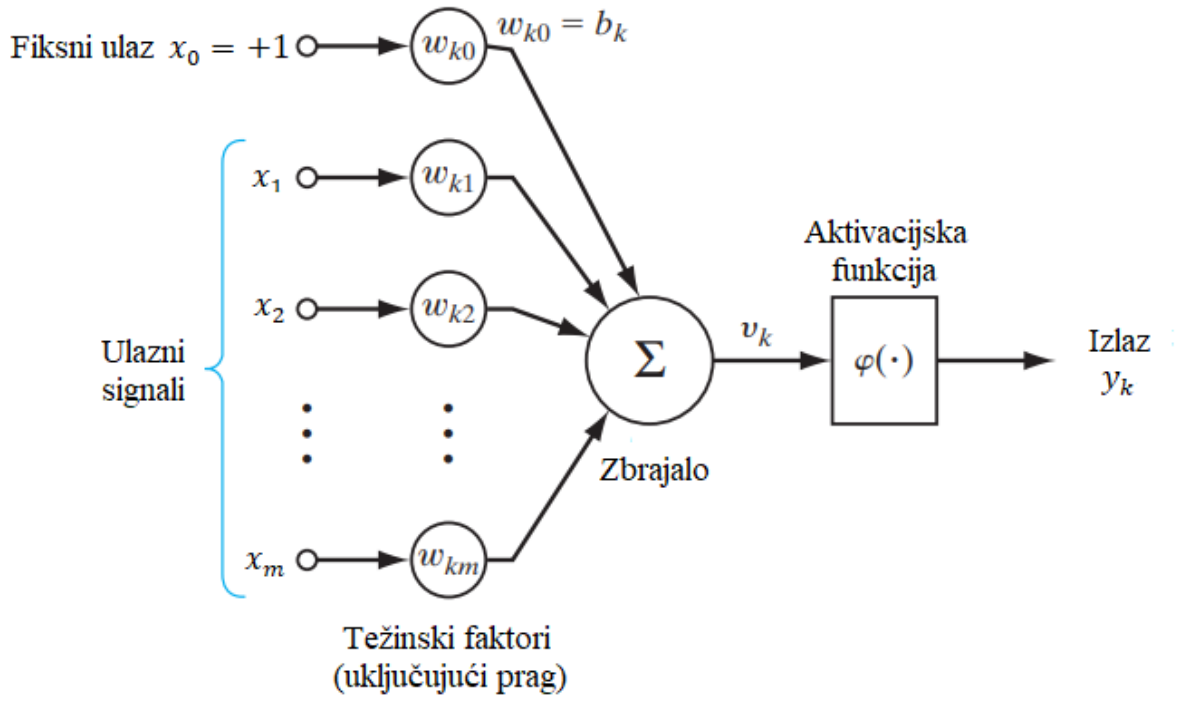
Jednadžbe (1), (2) i (3) mogu se zapisati tako da se prag b_k uračuna u izraz u_k. Dodaje se novi težinski faktor $w_{k0} = b_k$ koji je spojen na konstantan ulazni signal $x_0 = +1$ te se matematički tada može zapisati jednadžbe (1) i (3):

$$v_k = \sum_{j=}^{m} w_{kj}\, x_j \tag{4}$$

i jednadžba (2):

$$y_k = \varphi(v_k) \tag{5}$$

Blok diagram neurona k s uračunatim pragom u težinsku sumu je dan na slici [Slika 4].

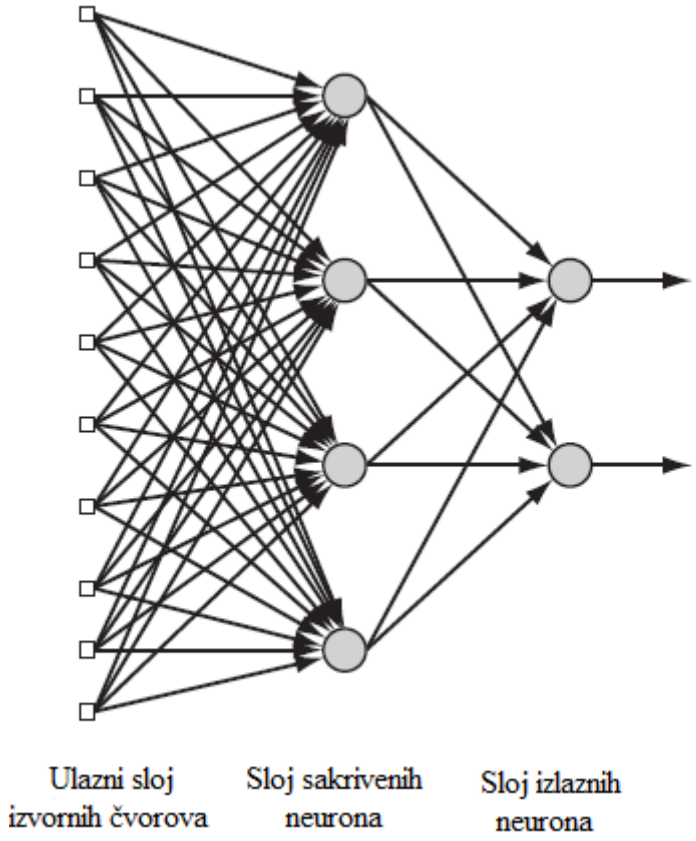


Slika 4 Nelinearni model neurona *k* s uračunatim pragom [14]

Postoji više aktivacijskih funkcija koje su korištene kod neuronskih mreža kao što su funkcija praga (engl. *Threshold function*) i sigmoidalna funkcija (engl. *Sigmoid function*). Jedna od vrsta funkcija praga je Heavisideova funkcija (engl. *Heaviside function*) kod koje je izlazni

signal jednak 1 ako je težinska suma veća ili jednaka nuli, a 0 ako je manja od nule[14]. Funkcija praga ReLU (engl. *Rectified Linear Units*) ima izlazni signal jednak 0 dok je težinska suma manja od nula, a ako je težinska suma v_k veća od nule izlazni signal jednak je v_k[15]. Aktivacijska funkcija ReLU korištena je u ovome radu.

Umjetna neuronska mreža je skup međusobno povezanih procesnih jedinica umjetnih neurona. Strukture neurona u umjetnoj neuronskoj mreži ovise o algoritmu učenja korištenog za učenje mreže[14]. Osnovne tri različite strukturne klase neuronskih mreža su: Jednoslojna unaprijedna mreža (engl. *Single-Layer Feedforward Network*), Višeslojna unaprijedna mreža (engl. *Multilayer Feedforward Network*) i Povratne mreže (engl. *Recurrent Network*). Na slici [Slika 5] prikazana je struktura višeslojne unaprijedne mreže [14].



Slika 5 Struktura višeslojne unaprijedne mreže [14]

Neuronske mreže imaju računalnu moć iz masivne paralelno raspoređene strukture i mogućnosti da uči zbog čega može generalizirati podatke. Generalizirani podaci znače da mreža može proizvesti razumne rezultate izlaza za ulazne podatke na koje nisu naišle za vrijeme treniranja mreže. Može vrlo dobro zbog navedenog aproksimirati rješenja kompliciranih problema koje je vrlo teško pratiti[14].

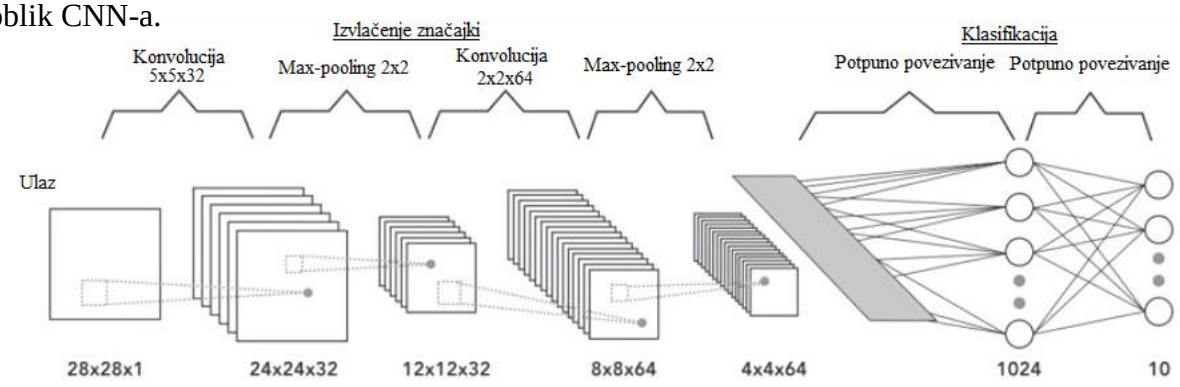
Korisna svojstva i mogućnosti neuronskih mreža su sljedeći [13]:

- Nelinearnost neuronskih mreža za procjenu nelinearnih odnosa uzorka
- Mogućnost raspoznavanja uzorka iz podataka koji su nejasni ili kojih je premalo
- Robusne na pogreške u podacima
- Stvaraju se vlastiti odnosi između podataka, a da nisu zadani na eksplicitan simbolički način
- Mogućnost učenja iz iskustva
- Prilagodljive okolini i mogu raditi s velikim brojem informacija

Neuronske mreže se najčešće koriste za rješavanje mnogih zadatka kao: raspoznavanje uzoraka, obrada slike, obrada govora, problemi optimizacije, nelinearno upravljanje, obrada nepreciznih i nepotpunih podataka, simulacije i druge. Proces učenja neuronskih mreža se može podijeliti u dvije kategorije učenje s učiteljem (engl. *supervised learning*) i učenje bez učitelja (engl. *unsupervised learning*). Kod učenja s učiteljem mreža poznaje za koje ulaze koje rezultate treba dobiti dok kod učenja bez učitelja mreža ne poznaje što treba dobiti na izlazu [13].

3.3. Konvolucijske neuronske mreže

Konvolucijske neuronske mreže se najviše koriste kod prepoznavanja slika, govora i uzoraka. Prednost CNN-a je što se ne trebaju ručno izraditi algoritmi izvlačenja značajki već se koriste težinski faktori unutar konvolucijskih slojeva mreže, dok težinski faktori potpuno povezanih slojeva mreže se koriste za klasifikaciju. Težinski faktori se određuju za vrijeme treniranja mreže. Spadaju pod unaprijedne neuronske mreže što znači da nema povratnih informacija rezultata koji bi utjecao na iduću klasifikaciju[16]. Na slici [Slika 6] se može vidjeti opći oblik CNN-a.

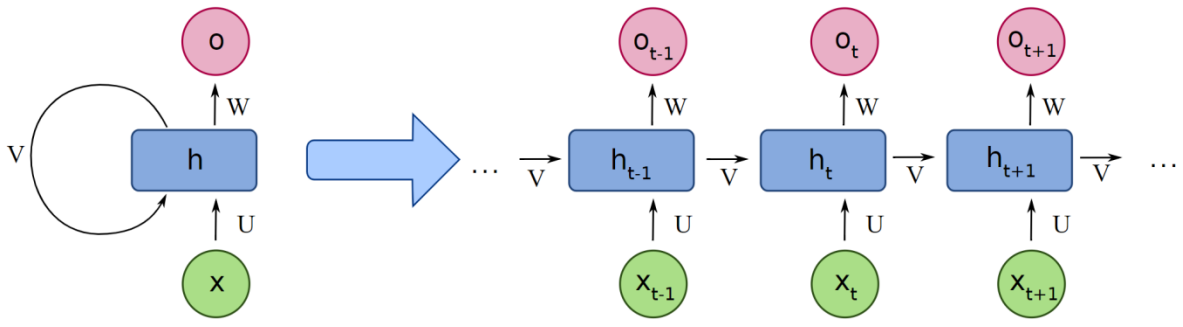


Slika 6 Opći oblik konvolucijske neuronske mreže[17]

Konvolucijske neuronske mreže se sastoje od više različitih slojeva. Uobičajeno se koriste tri vrste slojeva: konvolucijski sloj (engl. *convolutional layer*), sloj sažimanja (engl. *pooling layer*) i potpuno povezani sloj (engl. *fully connected layer*). Konvolucijski sloj izvršava operaciju konvolucije koja izvlači različite značajke iz ulaznih signala. Ulazni podaci veličine N x N x D se obrađuju matricom H k x k x D koja izvodi konvoluciju. Ovisno o vrsti matrice H izvlače se različite značajke iz podataka. Pri završetku konvolucijskog sloja nalazi se aktivacijska funkcija koja prilagođava podatke. Sloj sažimanja smanjuje razlučivost značajki te ih čini robusnim na smetnje i iskrivljenje. Smanjuje količinu informacija koju mreža razmatra, a zadržava dominantna svojstva. Sažimanje se radi dvjema metodama: "max pooling" i " average pooling". Potpuno povezan sloj se većinom koristi kao zadnji sloj u mreži. Uzima težinske vrijednosti značajki iz prethodnog sloja kako bi se odredio specifični traženi rezultat [16].

3.4. Povratne neuronske mreže

Povratne neuronske mreže (engl. *Recurrent neural network, RNN*) se razlikuju od unaprijednih neuronskih mreža po tome što imaju barem jednu povratnu vezu. Aktivacije neurona prethodnog diskretnog vremenskog koraka šalju se u ulaz trenutačnog vremenskog koraka kako bi utjecao na njega. Zbog povratne veze RNN poboljšava modeliranje sekvencijskih zadataka kao što su predviđanja vremenskih nizova [18]. Na slici [Slika 7] se može vidjeti osnovna povratna neuronska mreža „odmotana" po vremenu.

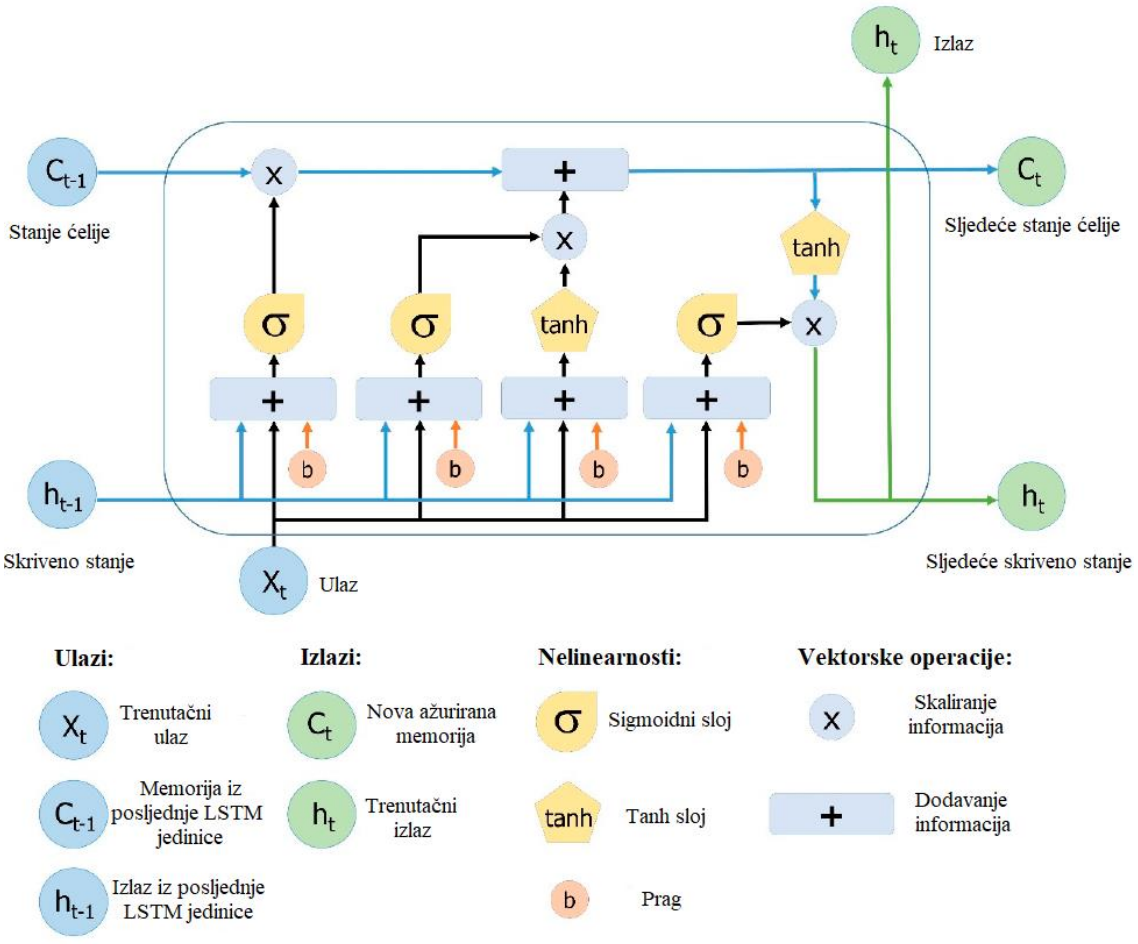


Slika 7 Osnovna povratna neuronska mreža [19]

Učinkovito učenje RNN je otežano zbog nestajućeg gradijenta (engl. *vanishing gradient*) i eksplozivnog gradijenta (engl. *explosive gradient*). Nestajući gradijent označava utjecaj

povratne veze na ulaze u skrivene neurone, stoga i na izlaz mreža te se taj utjecaj prebrzo smanjuje kroz vrijeme dok eksplozivni gradijent prikazuje eksponencijalni rast utjecaja povratne veze na izlaz mreže. Zbog problema navedenih gradijenata u klasičnim RNN broj mogućih vremenski diskretnih uzoraka je od 5 do 10. Kako bi se moglo uzimati više uzoraka koristi se LSTM što je poseban oblik RNN. LSTM je korišten kod prepoznavanja rukopisa, modeliranja jezika, označavanje fonema kod akustičkih vremenskih okvira[18].

LSTM ima mogućnosti učenja dugoročnih ovisnosti i pamćenja informacija na duže vremenske periode. Struktura LSTM ćelije je prikazana na slici [Slika 8]. Uobičajeno LSTM mreže su sačinjene od memorijskih blokova ćelija (engl. *cells*) na mjestima gdje su inače standardni slojevi neuronskih mreža. Iz ćelije se prosljeđuju dva stanja, stanje ćelije i skriveno stanje. Stanje ćelije je glavni lanac protoka podataka što omogućuje da podaci putuju unaprijed nepromijenjeni. Podaci se mogu dodati ili izbaciti koristeći sigmoidna vrata (engl. *sigmoid gate*). Sigmoidna vrata su slična nizu matričnih operacija koje sadrže vlastite težinske faktore te služe za upravljanje procesa pamćenja [20].



Slika 8 Struktura LSTM jedinice [20]

Kako bi se napravila LSTM mreža prvo je potrebno odrediti koje će se informacije izbaciti u koraku *t* ćelije i koje informacije će biti izbačene iz prethodnog izlaza ćelije u vremenskom trenutku $t - 1$. Identificiranje i izbacivanje podataka se vrši sigmoidnom funkcijom, koja uzima za ulaz zadnji izlaz LSTM jedinice (h_{t-1}) iz vremenskog koraka $t - 1$ i trenutačnog ulaza u mrežu (X_t) u vremenskom koraku *t*. Sigmoidna funkcija sa svojim ulazima se naziva „forget gate" ili f_t, gdje je f_t vektor s vrijednostima u rasponu od 0 do 1 koji odgovaraju brojevima u ćeliji C_{t-1} [20].

$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \quad \textbf{(6)}$$

Gdje je σ sigmoidna funkcija, W_f težinske matrice i b_f prag, vezani za „forget gate".

Sljedeći korak je izbor informacija koje će se spremati iz novoga ulaza (X_t) u stanje ćelije i ažuriranje stanja ćelije. Ovaj koraka se sastoji od dva dijela, sigmoidnog sloja (i_t) i tanh sloja (N_t). Sigmoidni sloj je zadužen za izbor informacija koja će se ažurirati ili ignorirati dajući im vrijednosti 0 ili 1. Tanh funkcija pridonosi težinske faktore vrijednostima te vrijednosti su u rasponu od –1 do 1 ovisno o njihovoj važnosti. Te dvije vrijednosti se množe i zbrajaju s prošlim stanjem ćelije (C_{t-1}) što daje trenutačno stanje ćelije (C_t) [20].

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i) \quad \textbf{(7)}$$

$$N_t = tanh(W_N[h_{t-1}, X_t] + b_N) \quad \textbf{(8)}$$

$$C_t = C_{t-1} + N_t I_t \quad \textbf{(9)}$$

Gdje su C_t i C_{t-1} stanja ćelija u vremenskom trenutku t i t – 1, a W matrice težina i b pragovi.

Posljednji korak mreže računa izlazne vrijednosti (h_t), a baziran je na filtriranom obliku izlaza ćelije (O_t). Sigmoidni sloj odlučuje koji će se dijelovi stanja ćelije proslijediti na izlaz, te izlaz sigmoidnih vrata (O_t) se množi s novim vrijednostima iz tanh sloja stanja ćelije (C_t) koji može poprimiti vrijednosti između –1 i 1 [20].

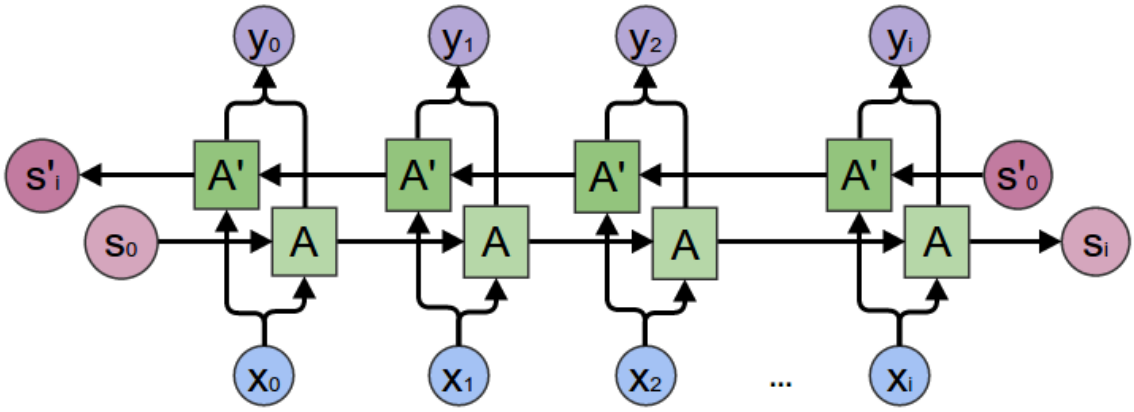
$$O_t = \sigma(W_o[h_{t-1}, X_t] + b_o) \quad \textbf{(10)}$$

$$h_t = O_t \tanh(C_t) \quad \textbf{(11)}$$

Gdje su W_o težinske matrice i b_o prag od sigmoidnih vrata" (O_t).

Bitno je napomenuti da se u gornjim izrazima se radi o vektorima, a ne skalarima. Vektori u mreži su svi pragovi *b*, ulazi sustava X_t i izlazi sustava h_t dok težinski faktori W su matrice.

Na slici [Slika 9] je prikazana dvosmjerna LSTM mreža (engl. *Bidirectional LSTM network*) koja se sastoji od dvije nezavisne LSTM RNN mreže. Takva struktura mreže omogućuje da mreža ima unazadne i unaprijedne informacije u svakom koraku. Sekvencijski ulaz se uči u dva smjera, od početka sekvence do kraja i obrnuto što omogućuje očuvanju informaciju iz početka i kraja sekvence [21].



Slika 9 Struktura dvosmjerne LSTM-e mreže [21]

3.5. Prenaučenost neuronskih mreža

Kod učenja mreže s učiteljem može doći do prenaučenosti mreže. Prenaučenost mreže uzrokuje da model mreže ima odlične rezultate nad skupom ulaznih podataka za treniranje dok kod podataka za validaciju daje loše rezultate. Loši rezultati dolaze zbog razlika u podacima između podataka za treniranje i učenje te što prenaučene mreže pamte šumove u podacima. Uzroci prenaučenosti mreže se uobičajeno mogu podijeliti u tri kategorije [22]:

- Učenje šuma iz podataka – dolazi zbog nedostataka u skupu podataka za učenje kao: premali skupa podataka za učenje, premale količine značajnih podataka ili previše udjela šuma. Zbog tih razloga dolazi do velike šanse učenja šuma, a ne njegovog odbacivanja.
- Kompleksnost hipoteza – Povećanjem hipoteza sustava (ulaza sustava) model postaje kompliciraniji što rezultira povećanjem srednje vrijednosti točnosti, ali s manjom dosljednošću. Takve naučene mreže postaju drastično različite ovisno o skupu podataka na kojima su učene.
- Višestruki usporedni postupci – Kroz procese usporedbe uvijek se uspoređuju više vrsta podataka kojima se daju ocjene iz evaluacijskih funkcija te se biraju oni podaci

maksimalnih vrijednosti. Dolazi do izbora podataka koji se neće moći poboljšavati već čak i smanjiti točnost mreže.

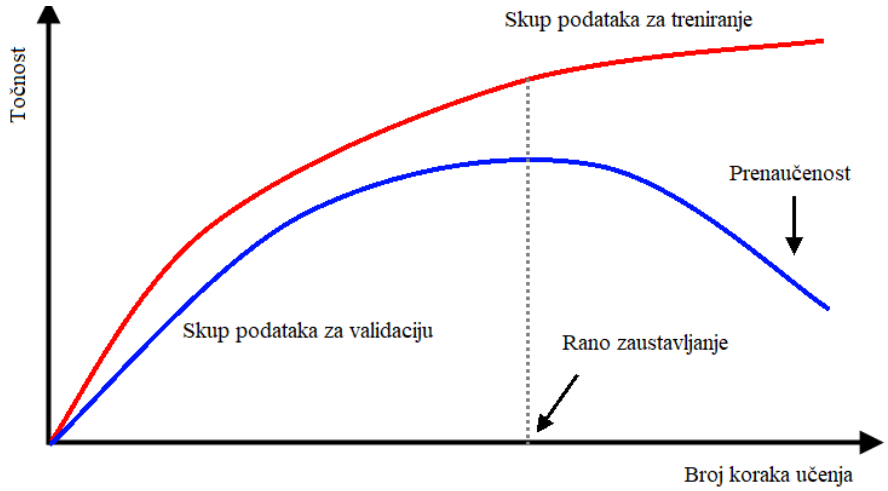
Kako bi se smanjio utjecaj prenaučenosti mreže koriste se različite metode kao: smanjenje mreže, rano zaustavljanje, proširenje skupa podataka za učenje, regularizacija i druge. [22].

Smanjenje mreže:

Smanjenje mreže se koristi kako bi se smanjilo učenje šuma. Koristi se „pruning“ teorija kako bi se odlučilo kako smanjiti kompliciranost klasifikatora uklanjajući manje značajne ili nebitne podatke[22].

Rano zaustavljanje:

Rano zaustavljanje je metoda koja se koristi kako bi se učenje mreže zaustavilo u trenutku kada dodatno učenje šteti točnosti mreže. Problem koji se dešava je da mreža nakon određenog vremena učenja prestaje poboljšavati točnost već se može pogoršati zbog učenja šuma. Na slici [Slika 10] se prikazuje problem prenaučenosti mreže, gdje horizontalna os prikazuje broj koraka učenja, a vertikalna os točnost mreže. Crvena linija pokazuje točnost mreže za ulazne signale iz skupa podataka za učenje dok plava linija prikazuje točnost mreže za ulazne signale iz skupa podatka za validaciju [22].



Slika 10 Graf točnosti kod ranog zaustavljanja [23]

Može se uočiti da ako se nastavi mreža učiti nakon trenutka ranog zaustavljanja točnost mreže za podatke izvan skupa podataka za učenje će padati. Ako se učenje zaustavi prije trenutka ranog zaustavljanja mreža će biti podučena stoga je potrebno naći trenutak u kojemu mreža prestaje povećavati točnost nad validacijskim skupom podataka.

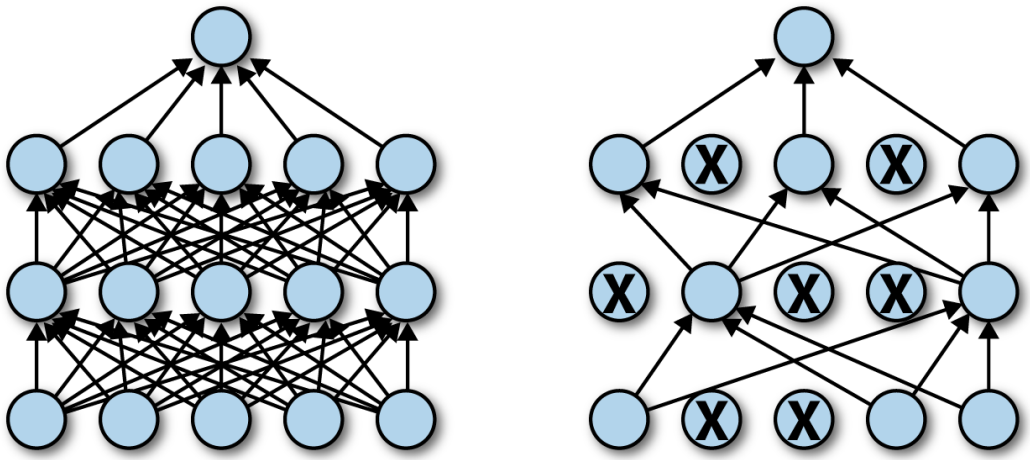
Proširenje skupa podataka za učenje:

Proširenje skupa podataka za učenje je bitno zbog toga što na točnost klasifikatora utječe kvantiteta i kvaliteta podataka za učenje što je izraženije kod učenja s učiteljem. Novi podaci povećavaju vrijeme učenja mreža, ali ih može biti teško nabaviti pošto često zahtijevaju ljudski rad kako bi se prikupili. Stoga se mogu manipulirati postojeći podaci kako bi se proširio skup podataka za treniranje. Postoje četiri glavna načina prikupljanja novih podataka [22]:

- Prikupljanje potpuno novih podataka.
- Dodavanje šuma u postojeće podatke
- Ponovno stjecanje podataka iz postojećih kroz njihovu obradu
- Proizvesti nove podatke na temelju distribucije postojećih podataka

Regularizacija:

Na izlaz modela utječe više različitih značajki. Kako broj značajki raste model postaje sve kompliciraniji. Prenaučene mreže imaju tendenciju uzimanja u obzir svih značajki iako neke nemaju znamenit utjecaj na izlaz mreže, već znaju biti samo šumovi. Stoga je potrebno ukloniti nepotrebne značajke ili ih ograničiti. Kako je vrlo teško odrediti koje značajke su nepotrebne ograničavaju se sve značajke tako što se minimiziraju funkcije cilja uvođenjem kaznenog člana u njih koji se naziva regularizator (engl. *regularizor*). Postoje više različitih regularizacijskih metoda, od kojih od kojih najčešće korištena je metoda isključivanja (engl. *dropout*). U metodi isključivanja [Slika 11] nasumično se izbacuju jedinice i značajne veze u neuronskoj mreži tijekom učenja što onemogućuje mreži prenaučenost [22].



Slika 11 Konceptualni prikaz metode isključivanja [24]

4. Baza podataka za validaciju i učenje neuronskih mreža

U ovom poglavlju napraviti će se baza podataka izvučenih značajki za učenje i validaciju konvolucijske neuronske mreže i LSTM mreže. Prvo će se proći kroz odabrane skupove neobrađenih podataka iz kojih će se izvući odabrane akustičke značajke. Baza podataka će se morati prije izvlačenja značajki obraditi kako bi se izbjeglo krivo učenje mreže. Nakon izvlačenja značajki podaci se spremaju u odgovarajućem obliku za ulaz neuronskih mreža.

4.1. AESDD baza podataka

Acted Emotional Speech Dynamic Database (AESDD) [25] je grčka emocionalna govorna baza podataka besplatno dostupna za istraživačke svrhe. Prikazuje pet emocija: ljutnju, gađenje, strah, sreću i tugu. Emocije su izvedene sa strane pet glumaca od kojih su tri ženske glumice i dva muška glumca. Koristi se kako bi se dobila veća generalizacija podataka pošto je na grčkom jeziku. Konvencija označavanja audio zapisa, primjer „a03 (5)“, je podijeljena u tri dijela, gdje prvi dio (a) predstavlja emociju, drugi dio (03) predstavlja broj rečenice izgovorene i treći dio ((5)) označava broj govornika.

4.2. CREMA-D baza podataka

Crowd Sourced Emotional Multimodal Actors Dataset (CREMA-D) [26] je besplatni skup podataka sačinjen od 7442 originalna isječka. Isječke su snimali 43 ženskih i 48 muških glumaca u dobnoj granici između 20 i 74 godine različitih rasa i etničkih grupa. Svi glumci su govorili istih 12 rečenica. Rečenice su predstavljene koristeći jednu od šest različitih emocija ljutnja, gađenje, strah, sreća, neutralno i tuga te četiri različita intenziteta nisko, srednje, visoko i neodređeno. Prednost korištenja ove baze podataka je što ima velik broj govornika što omogućuje bolju generalizaciju i smanjuje prenaučenost mreže. Konvencija označavanja audio zapisa, primjer „1001_DFA_NEU_XX“, je podijeljena u četiri dijela, gdje prvi dio (1001) je znamenka od četiri broja koja predstavlja broj glumca, drugi dio (DFA) predstavlja koja rečenica je izgovorena, treći dio (NEU) predstavlja korištenu emociju i četvrti dio (XX) predstavlja intenzitet.

4.3. Emo-DB baza podataka

Berlin Database of Emotional Speech (Emo-DB) [27] je slobodno dostupna njemačka emocionalna baza podataka. Sadrži sveukupno 535 govornih izjava na njemačkom jeziku koje je izgovorilo deset profesionalnih govornika sačinjeno od pet ženskih i pet muških govornika u dobnoj granici od 21 do 35 godina. Baza podataka sadrži izjave od sedam emocija: ljutnja, dosada, anksioznost, sreća, tuga, gađenje i neutralno. Konvencija označavanja audio zapisa, primjer „03a01Fa“, gdje pozicija simbola 1-2 (03) predstavlja broj govornika, 3-5 (a01) predstavlja korišteni tekst, 6 (F) predstavlja korištenu emociju i 7 (a) predstavlja verziju izjave ako postoje više od dvije iste. Bitno je napomenuti da slovo u izrazima koje predstavlja emocije uzima početno slovo naziva emocije na njemačkom jeziku.

4.4. RAVDESS baza podataka

Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) [28] je baza podataka koja sadrži video i audio snimke govora i pjesama, no koristit će se samo govorni dio audio baze podataka. Govorni dio audio baze podataka sadrži 1440 govornih audio zapisa. Audio zapisi su izvedeni od 24 profesionalna glumca od kojih je 12 ženskih i 12 muških glumca. Sadržani izrazi emocija u bazi podataka su smirenost, sreća, tuga, ljutnja, strah, iznenađenost i gađenje te neutralna izjava koja nema intenzitet. Svaka emocionalna izjava je predstavljena u dva intenziteta: normalnom i jakom. Konvencija označavanja audio zapisa, primjer „03-01-05-02-02-02-23“, sastoji se od sedam numeričkih identifikacijskih brojeva koji redom predstavljaju: modalitet, govor ili pjesmu, emocija, emocionalni intenzitet, izjavu, broj ponavljanja i broj glumca.

4.5. SAVEE baza podataka

Surrey Audio-Visual Expressed Emotion (SAVEE) [29] je baza podataka koja sadrži audio i video zapise, gdje će se koristiti samo audio dio baze podataka. Audio zapisi su izvedeni od četiri muška britanska govornika u dobnoj granici između 27 do 31 godina. Sadržani izrazi emocija u bazi podataka su: ljutnja, gađenje, strah, sreća, tuga i iznenađenost te 30 neutralnih izjava po govorniku. Konvencija označavanja audio zapisa, primjer „DC_a13“, sastoji se od tri djela, gdje prvi dio (DC) označava govornika, drugi dio (a) predstavlja emociju i treći dio (13) predstavlja broj izjave.

4.6. TESS baza podataka

Toronto emotional speech set (TESS) [30] je baza podataka od 2800 izjava. Audio zapise su izvele dvije glumice od 26 i 64 godine. Skup od 200 ciljanih riječi glumice izgovaraju u rečenici „Say the word _“ za svaku prikazanu emociju. Prikazane emocije su: ljutnja, gađenje, strah, sreća, ugodno iznenađenje i neutralno. Konvencija označavanja audio zapisa, primjer „OAF_bar_disgust“ se sastoji od tri dijela, gdje prvi dio (OAF) označava glumicu, drugi dio (bar) označava ciljanu riječ i treći dio (disgust) označava emociju.

4.7. Priprema baze podataka

Prije izvlačenja akustičnih značajki potrebno je pripremiti podatke za njihovo izvlačenje. Prvo su se normalizirali audio zapisi pošto je bilo velikih razlika u glasnoći među njima, što loše utječe na učenje mreže. Normalizacija audio zapisa izvršena je u programu *Audacity*. *Audacity* je besplatni audio softer otvorenog koda, te se može koristiti na više operativnih sustava [31]. Odabrane emocije za ovaj rad su: ljutnja (engl. *anger*), tuga (engl. *sadness*), sreća (engl. *happiness*), strah (engl. *fear*), gađenje (engl. *disgust*) i neutralno (engl. *neutral*), koje se moraju izvući iz navedenih baza podatka. Svaka emocija će se posebno kategorizirati za ženski i muški spol pošto postoje dostatne zvučne spektralne razlike između spolova. Stoga ukupni broj kategorija modela je 12.

Prvo je potrebno učitati potrebne biblioteke s kojima će se napraviti program [Slika 12].

```
1. import pandas as pd
2. import os
3. import librosa
4. import numpy as np
5. import pickle
```

Slika 12 Učitavanje biblioteka

Nakon učitavanja biblioteka učitava se putanja svih korištenih baza podataka: AESDD, CREMA-D, Emo-DB, RAVDESS, SAVEE i TESS [Slika 13].

```
1. pathAESDD=".\Baze podataka\AESDD/"ž
2. pathCREMAD=".\Baze podataka\CREMA-D/"
3. pathEmoDB=".\Baze podataka\Emo-DB/"
4. pathRAVDESS=".\Baze podataka\RAVDESS/"
5. pathSAVEE=".\Baze podataka\SAVEE/"
6. pathTESS=".\Baze podataka\TESS/"
```

Slika 13 Učitavanje putanje baza podataka

Zatim se iz imena audio zapisa za svaku pojedinu audio bazu podataka izvlače željene emocije ovisne o spolu. Proces se izvršava for petljom, gdje se za svaki pojedini audio zapis provjerava prvo koji je spol te zatim koja je emocija. Ako se emocija podudara s jednom od izabranih emocija, sprema se u varijablu Aemotion, a one emocije koje se ne podudaraju spremaju se kao unknown. Prvo slovo Aemotion označava bazu podataka AESDD. Ujedno se spremaju i putanje audio zapisa u varijablu Apath. Nakon završene for petlje podaci iz AESDD baze podataka se postavljaju u tablični zapis, gdje prvi red označava spol i emociju iz varijable Aemotion, te je označen s "Label". Drugi stupac označen je sa "Source" i predstavlja izvor audio zapisa dok je treći stupac izrađen iz varijable Apath i označen je s "Path". Tablica se sprema naredbom *to_csv*. Na slici [Slika 14] je prikazan gornji postupak u programskom kodu.

```
dir0_list=os.listdir(pathAESDD)
dir0_list.sort()
#CREMA-D Kategorizaja po spolu i emociji
Agender=[]
Aemotion=[]
Apath=[]
female=[1,2,5]
for i in dir0_list:
    if int(i[5:6]) not in female:
        genA='male'
    else:
       genA='female'
  Agender.append(genA)
if i[:1] == 'a' and genA == 'female':
     Aemotion.append('female_angry')
elif i[:1] == 's' and genA == 'female':
    Aemotion.append('female_sad')
elif i[:1] == 'h' and genA == 'female':
    Aemotion.append('female_happy')
elif i[:1] == 'd' and genA == 'female':
     Aemotion.append('female_disgust')
elif i[:1] == 'f' and genA == 'female':
     Aemotion.append('female_fear')
elif i[:1] == 'a' and genA == 'male':
     Aemotion.append('male_angry')
elif i[:1] == 's' and genA == 'male':
    Aemotion.append('male_sad')
elif i[:1] == 'h' and genA == 'male':
     Aemotion.append('male_happy')
elif i[:1] == 'd' and genA == 'male':
      Aemotion.append('male_disgust')
elif i[:1] == 'f' and genA == 'male':
   Aemotion.append('male_fear')
else:
    Aemotion.append('unknown')
Apath.append(pathAESDD + i)
#Postavljanje podataka CREAM-D u tablični zapis
AESDDtable=pd.DataFrame(Aemotion, columns = ['Label'])
AESDDtable['Source']='AESDD'
AESDDtable=pd.concat([AESDDtable,pd.DataFrame(Apath
                      ,columns=['Path'])],axis=1)
AESDDtable.to_csv("Adata_path.csv", index=False)
```

Slika 14 Očitavanje emocija iz imena iz baze podataka AESDD

Na sličan način se izvlače emocije i spremaju u tablični zapisi iz ostalih baza podataka. U prilogu II BazaPodataka.py može se vidjeti programski kod za izvlačenje emocija iz imena ostalih baza podataka. Na [Tablica 2] prikazan je tablični zapis AESDD baze podataka.

Tablica 2 Izvadak iz tabličnog zapisa AESDD baze podataka

Label	Source	Path
female_angry	AESDD	.\Baze podataka\AESDD/a01 (1).wav
female_angry	AESDD	.\Baze podataka\AESDD/a01 (2).wav
male_angry	AESDD	.\Baze podataka\AESDD/a01 (3).wav
male_angry	AESDD	.\Baze podataka\AESDD/a01 (4).wav
female_angry	AESDD	.\Baze podataka\AESDD/a01 (5).wav
male_angry	AESDD	.\Baze podataka\AESDD/a01 (6).wav

Nakon što se za svaku pojedinačnu bazu podataka emocije spreme u tablični zapis one se spajaju u jednu zajedničku tablicu te se svi članovi unutar stupca "Label" s imenom unknown izbacuju iz tablice [Slika 15].

```
1. Baza_Podataka=pd.concat([AESDDtable, CREMA_Dtable, Emo_DBtable,
2.                          RAVDESStable, SAVEEtable, TESStable], axis=0)
3. Baza_Podataka=Baza_Podataka[Baza_Podataka.Label !='unknown']
4. Baza_Podataka.to_csv("Baza_podataka.csv", index=False)
5. print(Baza_Podataka.Label.value_counts())
```

Slika 15 Spajanje baza podataka

Zadnji redak koda [Slika 15] nam daje raspodjelu audio zapisa po emocijama i spolu zbrajajući imena unutar stupca "Label". Na slici [Slika 16] se može vidjeti raspodjela emocija po spolovima, gdje se vidi da raspodjela emocija po spolovima nije jednaka, ali je prihvatljiva.

```
female_angry      1210
female_happy      1186
female_sad        1181
female_disgust    1178
female_fear       1172
female_neutral     988
male_angry         961
male_fear          940
male_happy         928
male_sad           925
male_disgust       913
male_neutral       794
Name: Label, dtype: int64
```

Slika 16 Raspodjela emocija unutar baze podataka

4.8. Izdvajanje akustičkih značajki

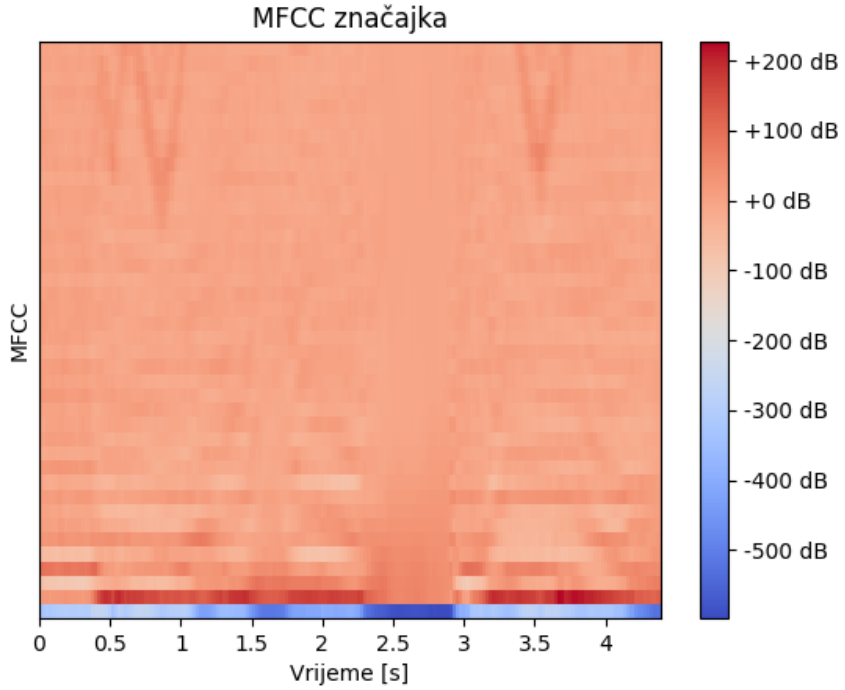
U prethodnom poglavlju pripremila se ukupna baza podataka u jednoj zajedničkoj tablici, gdje imamo označenu emociju i spol, izvor audio zapisa i putanju do audio zapisa. Iz takvog tabličnog zapisa će se izvući akustičke značajke glasa za svaki pojedini audio zapis. Izabrane akustičke značajke su MFCC i spektrogram po Mel skali. MFCC značajke su izabrane jer su najčešće korištena spektralna reprezentacija govora. Prikazuju kratkoročnu snagu spektra govornog signala. Bazira se na ljudskom sluhu koji ne može percipirati frekvencije iznad 1000 Hz [9].

Za izvlačenje MFCC značajki i Mel spektrograma koristi se biblioteka *Librosa*. *Librosa* je snažna biblioteka za analizu, obradu i izvlačenje audio podataka [32]. Pomoću for petlje se prolazi kroz bazu podataka, te se uklanja tišina s početka i kraja zvučnog zapisa pomoću *trim* funkcije. Zatim se izvlače srednje vrijednosti MFCC značajki po čitavom vremenu zapisa, kako parametri ne bi bili ovisni o duljini trajanja zvučnog zapisa. Nakon izvlačenja MFCC značajki izvlači se spektrogram po Mel skali na jednak način. Zatim se vrijednosti značajki spremaju u varijablu "Značajke". Na slici [Slika 17] je prikazan programski kod za izvlačenje akustičnih značajki.

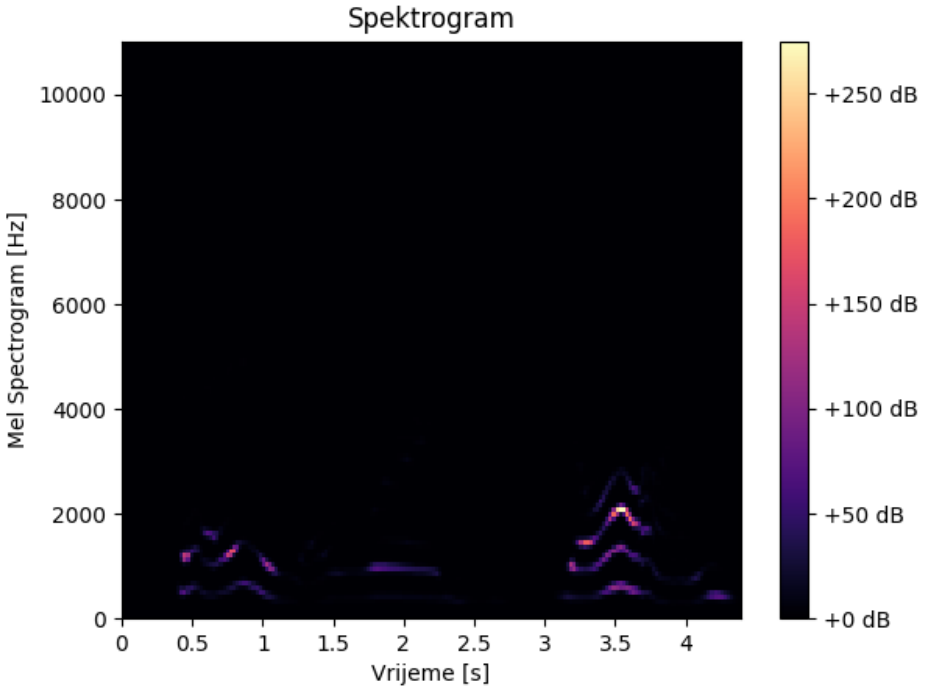
```
1.  BazaPodataka_Značajke=pd.read_csv("Baza_podataka.csv")
2.  Značajke=pd.DataFrame(columns=['Features'])
3.
4.  #Izvlačenje značajki iz podataka
5.  counter=0
6.  for index, Path in enumerate(BazaPodataka_Značajke.Path):
7.      #resample baze podataka s brzinom uzrokovanja 44kHz
8.      resdata, brzina_uzoraka=librosa.load(Path,
9.                                      res_type='kaiser_best', sr=44000)
10.   brzina_uzoraka=np.array(brzina_uzoraka)
11.   #Uklanjanje tišine iz zvučnih zapisa ispod 20dB
12.   resdata, _ =librosa.effects.trim(y=resdata, top_db= 20)
13.   rezultat = np.array([])
14.   #Izvlačenje srednje vrijednosti MFCC značajki
15.   MFCC_značajke=np.mean(librosa.feature.mfcc(y=resdata,
16. sr=brzina_uzoraka, n_mfcc=40), axis=1)
17.   rezultat=np.hstack((rezultat,MFCC_značajke))
18.   #Izvlačenje srednje vrijednosti MEL spektograma
19.   MEL_značajke=np.mean(librosa.feature.melspectrogram(resdata,
20. sr=brzina_uzoraka), axis=1)
21.   rezultat=np.hstack((rezultat,MEL_značajke))
22.   Značajke.loc[counter]=[rezultat]
23.   counter=counter+1
```

Slika 17 Izvlačenje akustičnih značajki

Prikaz izvučenih značajki za jedan audio zapis se može vidjeti na slici [Slika 18] za MFCC značajke i na slici [Slika 19] za Mel spektrogram. Bitno je napomenuti da grafovi nisu srednje vrijednosti značajki po vremenu.



Slika 18 Prikaz MFCC značajki



Slika 19 Prikaz Mel spektrograma

Izvučene značajke se potom spremaju kako se proces ne bi morao ponavljati pošto izvlačenje akustičkih značajki na velikom broju audio zapisa traje duže vrijeme. Spremljene značajke se pridodaju postojećoj tablici bazi podataka, te se nova cjelokupna baza podataka s vrijednostima izvučenih značajki sprema za daljnju uporabu. Na slici [Slika 20] je prikazan programski kod za pridodavanje i spremanje akustičnih značajki.

```
#Spremanje izvučenih značajki
Značajke.to_pickle('Baze_Značajki.pk1')
#Združivanje značajki u tablicu Baze Podataka
Značajke=pd.read_pickle('Baze_Značajki.pk1')
Značajke=pd.concat([BazaPodataka_Značajke,
                    pd.DataFrame(Značajke['Features'].values.tolist())]
                    ,axis=1)
Značajke.to_csv("Tablica_Značajki.csv", index=False)
```

Slika 20 Spremanje akustičkih značajki

Naredbom *shape* nad tablicom podataka možemo dobiti oblik tablice. Dobiveni oblik tablice je (12376, 171), što znači da tablica ima 12376 redaka i 171 stupaca. Prvi redak je informacijski te nam govori što se nalazi u kojemu stupcu, što znači da je obrađeno 12375 audio podataka. Od 171 stupca 168 stupaca reprezentiraju izvučene akustične značajke dok prva tri označuju: emociju i spol, izvor audio zapisa i treći stupac putanju do audio zapisa. Na tablici [Tablica 3] je prikazan ispis prvih nekoliko članove baze podataka s prvih 6 izvučenim akustičkim značajkama.

Tablica 3 Izvadak iz tabličnog zapisa baze podataka

Label	Source	Path	0	1	2	3	4	5
female_ar	AESDD	.\Baze poc	-321.612	105.4343	-16.8409	5.756296	-17.5826	6.672817
female_ar	AESDD	.\Baze poc	-326.324	121.0987	-24.356	-9.35334	-22.9938	-12.4014
male_ang	AESDD	.\Baze poc	-304.772	111.1229	-15.8466	17.40725	-19.8535	12.32804
male_ang	AESDD	.\Baze poc	-278.254	120.1116	-28.0784	21.53141	-29.9616	16.2933
female_ar	AESDD	.\Baze poc	-331.282	100.1277	-35.6313	11.16901	-1.06299	-7.26114
male_ang	AESDD	.\Baze poc	-289.046	98.84982	-0.12137	12.54923	-3.52136	12.55623
female_ar	AESDD	.\Baze poc	-335.458	91.7455	-49.75	11.5284	-12.7111	-8.8102

5. Akustički model duboke neuronske mreže

Prije samoga početka izrade neuronske mreže potrebno je učitati u program sve potrebne biblioteke s kojima će se izraditi sljedeći zadaci:

- Podjela podataka na skup za učenje i validaciju, te njihovu prilagodbu
- Izrada CNN modela mreže
- Spremanje naučenog modela
- Izradu grafova točnosti i gubitka
- Izradu matrice konfuzije

Na slici [Slika 21] prikazane su učitane biblioteke.

```
1.  import pandas as pd
2.  import numpy as np
3.  import os
4.  from sklearn.model_selection import train_test_split
5.  from sklearn.preprocessing import LabelEncoder
6.  from sklearn.metrics import confusion_matrix
7.  import keras
8.  from keras.utils import to_categorical, np_utils
9.  from keras import regularizers
10. from keras.optimizers import RMSprop
11. from keras.preprocessing import sequence
12. from keras.preprocessing.sequence import pad_sequences
13. from keras.models import Sequential, Model, model_from_json
14. from keras.layers import Dense, Input, Flatten, Dropout, Activation
15. from keras.layers import BatchNormalization, Conv1D, MaxPooling1D,
16.                          AveragePooling1D
17. from keras.callbacks import ModelCheckpoint
18. import matplotlib.pyplot as plt
19. import seaborn as sns
```

Slika 21 Učitavanje biblioteka za CNN mrežu

Nakon učitavanja potrebnih biblioteka učitava se i tablica baze podataka. Zatim je potrebno podijeliti podatke iz tablice na skup za validaciju i skup za treniranje neuronske mreže. Odabrana je raspodjela podataka tako da se 75% podataka koristi za učenje, a 25% podataka za validaciju. Prvo se spremaju podaci značajki u varijable "Značajke_train" i "Značajke_test", te podaci spola i emocija u varijable "Label_train" i "Label_test". Kod podataka značajki odbacuju se prva tri stupca tablice kako bi ostali samo podaci značajki dok kod varijabli spola i emocija promatra se samo stupac "Label". Nakon toga izvučene varijable prebacuju se u niz naredbom *array*. Na slici [Slika 22] prikazan je programski kod dijeljenja varijabli na skup za učenje i validaciju.

```
#Čitanje Značajki i njihova podjela za treniranje i validaciju
Značajke=pd.read_csv('Tablica_Značajki.csv')
Značajke_train, Značajke_test, Label_train,
Label_test=train_test_split(Značajke.drop(['Path','Source','Label'],
                                          axis=1)
                            ,Značajke.Label,test_size=0.25
                            ,shuffle=True)
Značajke_train=np.array(Značajke_train)
Značajke_test=np.array(Značajke_test)
Label_train=np.array(Label_train)
Label_test=np.array(Label_test9
```

Slika 22 Dijeljenje podataka na skup za učenje i validaciju

Dobivene četiri varijable potrebno je još pretvoriti u oblik s kojim mreža može raditi. Pošto mreža ne može raditi s tekstualnim oznakama emocija i spola varijable "Label_train" i "Label_test" će se kodirati metodom *One Hot Encoding*. Metoda uzima sve kategorije iz stupca "Label" te ih sprema pod klase (engl. *classes*), te se naredbom *to_categorical* pretvaraju u zapis čitljiv mreži. Oblik zapisa je binaran te za svaku kategoriju ako je postojeća daje 1, odnosno 0 ako ne spada pod tu kategoriju. Na slici [Slika 23] prikazan je kod za *One hot encoding* i ispod prikaz klasa, te binarni zapis labela skupa za treniranje imenom "Label_trainKategorizirano".

```
#One Hot Encoding
KodiranjeLabel=LabelEncoder()
Label_trainKodiran=np_utils.to_categorical(KodiranjeLabel.fit_transfo
                                                        rm(Label_train))
Label_testKodiran=np_utils.to_categorical(KodiranjeLabel.fit_transfor
                                                          m(Label_test))
```

```
klase:  ['female_angry'   'female_disgust'  'female_fear'   'female_happy'
         'female_neutral' 'female_sad'      'male_angry'    'male_disgust'
         'male_fear'      'male_happy'      'male_neutral'  'male_sad']

Label_trainKategoriziran:
 [[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 1.]
  [0. 0. 0. ... 0. 0. 1.]
  ...
  [0. 0. 0. ... 0. 0. 1.]
  [0. 0. 0. ... 0. 1. 0.]
  [0. 1. 0. ... 0. 0. 0.]]
```

Slika 23 Kod za *One Hot Encoding*, klase i zapis labela nakon kodiranja

Potom se proširuje dimenzija varijabli "Značajke_train" i "Značajke_test" kako bi bili kompatibilni za konvolucijsku neuronsku mrežu. Dodavanjem jedne dodatne osi već napravljenim varijablama postiže se proširivanje dimenzije. Na slici [Slika 24] prikazan je programski kod i oblik varijabli.

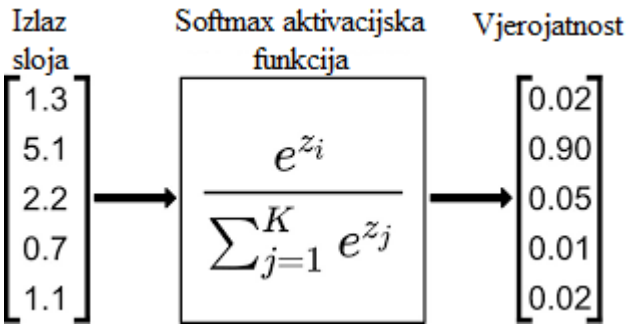
```
#Podešenje ulaza za CNN
Značajke_train_prošireno=np.expand_dims(Značajke_train, axis=2)
Značajke_test_prošireno=np.expand_dims(Značajke_test, axis=2)

Značajke_train_prošireno: (9282, 168, 1)
Značajke_test_prošireno:  (3094, 168, 1)
```

Slika 24 Kod za proširivanje dimenzije ulaznih podataka i njihov oblik

Nakon prilagodbe ulaznih podataka mreže, izrađuje se model mreže. Za izradu konvolucijske neuronske mreže izabrana je biblioteka *keras*. *Keras* je sučelje za programiranje aplikacija (engl. A*pplication Programming Interface*, API*)* namijenjenih za duboko učenje koji je izrađen u programskom jeziku *Python* [33]. Model mreže napravljen je po uzoru na [34]. Na slici [Slika 25] prikazan je model CNN mreže. Model započinje naredbom *Sequential()*, te se zatim naredbom *add()* dodaje željeni sljedeći sloj. Prvi sloj mreže je ulazni jednodimenzijski konvolucijski sloj, te se u njemu definiraju ulazni podaci za učenje mreže. Aktivacijska funkcija ulaznog sloja je *ReLU*. Kroz mrežu je postavljeno još 8 jednodimenzijskih konvolucijskih slojeva koji također imaju *ReLU* aktivacijsku funkciju. Osim prethodno navedenih slojeva, kroz mrežu se pojavljuju 2 *Dropout* sloja koji smanjuju prenaučenost mreže i 2 *BatchNormalization* slojeva koji radi standardnu normalnu distribuciju, čime se poboljšava stabilnost i performanse mreže. Ujedno se nalaze i dva *Maxpooling1D* sloja kako bi se smanjila prenaučenost mreže i količina parametara. Na kraju mreže je potpuno povezan sloj s 12 izlaza kako ima 12 klasa koje kategorizira. Aktivacijska funkcija potpuno povezanog sloja je *Softmax*. *Softmax* funkcija se koristi kod kategorizacijskih problema i problema predviđanja. Postavlja izlaze neuronske mreže na vrijednosti između 0 i 1, te zbroj svih izlaza je jednak 1 [Slika 26].

```
#Model CNN mreže
CNN_Model=Sequential()
CNN_Model.add(Conv1D(256, 8, padding='same'
                ,input_shape=(Značajke_train_prošireno.shape[1],1)))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(256, 8, padding='same'))
CNN_Model.add(BatchNormalization())
CNN_Model.add(Activation('relu'))
CNN_Model.add(Dropout(0.25))
CNN_Model.add(MaxPooling1D(pool_size=(8)))
CNN_Model.add(Conv1D(128, 8,padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(128, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(128, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(128, 8, padding='same'))
CNN_Model.add(BatchNormalization())
CNN_Model.add(Activation('relu'))
CNN_Model.add(Dropout(0.25))
CNN_Model.add(MaxPooling1D(pool_size=(8)))
CNN_Model.add(Conv1D(64, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(64, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Flatten())
CNN_Model.add(Dense(12))
CNN_Model.add(Activation('softmax'))
optimize=keras.optimizers.RMSprop(lr=0.00001, decay=1e-6)
CNN_Model.summary()
CNN_Model.compile(loss='categorical_crossentropy'
                  ,optimizer=optimize, metrics=['accuracy'])
```

Slika 25 Model CNN neuronske mreže



Slika 26 Softmax aktivacijska funkcija [35]

Korišteni optimizator za učenje CNN mreže je *RMSprop*. *RMSprop* je robustan optimizator koji adaptivno prilagođava stopu učenja smanjujući gradijent učenja kod malih serija

podataka (engl. *batch size*) te povećavajući kod velikih serija podataka [36]. Učenje se izvršava u 65 epoha te se težinske vrijednosti mijenjaju nakon svake serije podataka, gdje je postavljena serija podataka na 32, što se može vidjeti na slici [Slika 27].

```
#Učenje mreže
CNN_Model_History=CNN_Model.fit(Značajke_train_prošireno
                        ,Label_trainKodiran
                        ,batch_size=32
                        ,epochs=65
                        ,validation_data=(Značajke_test_prošireno
                                          ,Label_testKodiran))
```

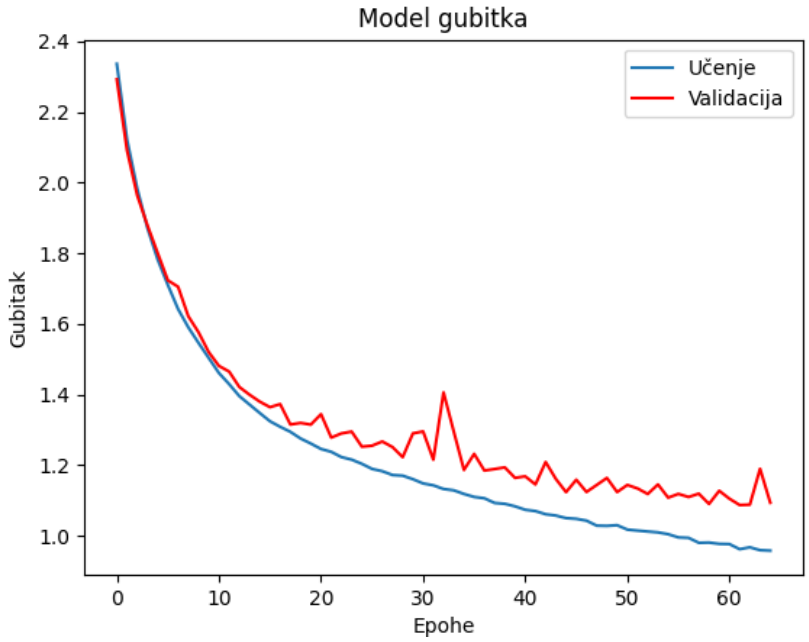
Slika 27 Podešavanje parametara za učenje neuronske mreže

Nakon završavanja učenja CNN mreže, što je dugotrajan proces, model se sprema za kasniju uporabu. Sprema se naučeni model mreže, što znači da se sprema arhitektura mreže i postignute težinske vrijednosti cijele mreže. Na slici [Slika 28] prikazan je programski kod za spremanje naučene CNN mreže.

```
#Spremanje CNN modela
CNN_Model_json=CNN_Model.to_json()
with open("CNN_Model.json", "w") as json_file:
    json_file.write(CNN_Model_json)
CNN_Model.save_weights("CNN_Model.h5")
```

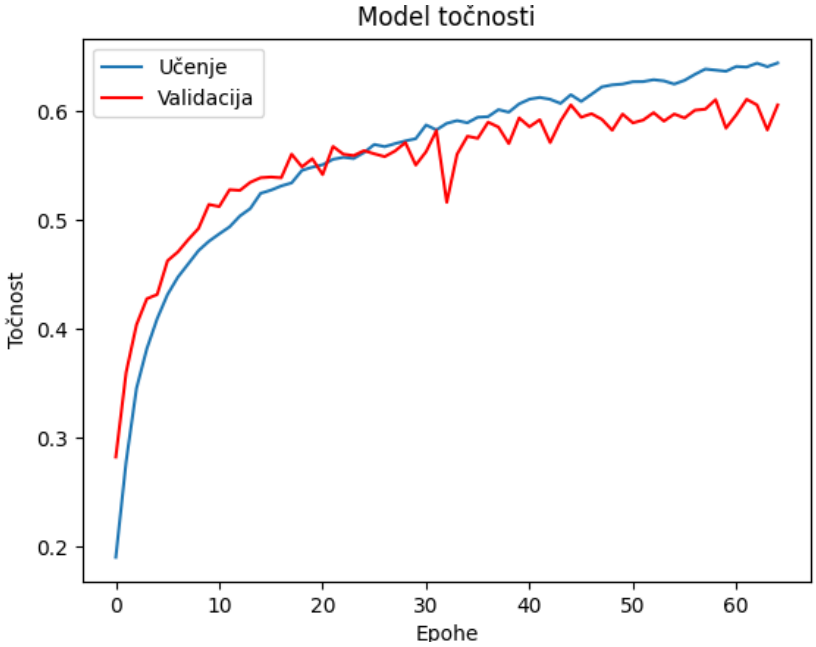
Slika 28 Spremanje naučenog CNN modela

Na slici [Slika 29] prikazan je graf gubitka CNN modela kroz proces učenja mreže te se može uočiti zadovoljavajući oblik krivulje za prikupljenu bazu podataka i količinu klasa koje se kategoriziraju. Treba napomenuti da se graf dobiva iz funkcije gubitka (engl. *Loss function*) kojoj je cilj minimizirati dobivene vrijednosti svakom iteracijom tj. u ovom slučaju svakom epohom. Crvena linija pokazuje rezultate nad validacijskim podacima dok plava linija označava rezultate nad podacima za treniranje.



Slika 29 Graf gubitka kroz učenje CNN mreže

Drugi graf za provjeru rezultata prikazan je na slici [Slika 30], gdje se prikazuje točnost prepoznavanja klasa kroz epohe učenja. Kao i na prijašnjem grafu krivulja je zadovoljavajuća, te se može vidjeti da se nije postigla prenaučenost mreže. Postignuta točnost mreže pri završetku učenja je 60.57%.

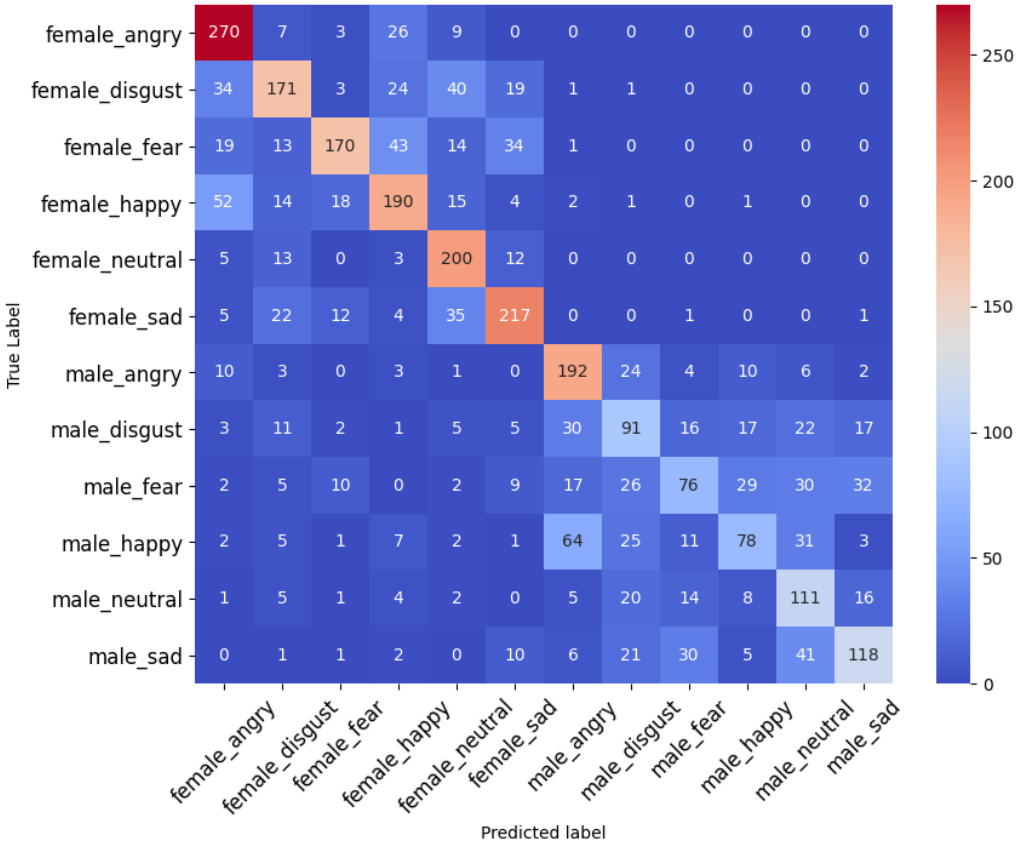


Slika 30 Graf točnosti kroz učenje CNN mreže

Matrica konfuzije CNN mreže [Slika 31] pokazuje ovisnost stvarnih klasa i predviđenih klasa. Redci matrice označuju stvarne klase dok stupci matrice označuju predviđene klase. Glavna

dijagonala matrice predstavlja ispravno predviđene klase, stoga je cilj imati što manju raspodjelu vrijednosti izvan glavne dijagonale. Matrica pruža pogled gdje u klasifikaciji neuronska mreža ima najviše problema kod razlučivanja klasa.

Iz pregleda matrice konfuzije može se uočiti relativno dobra razlučivost između spolova, gdje mreža vrlo dobro prepoznaje ženske glasove, dok muške glasove zna zamijeniti za ženske glasove u malom postotku. Što se tiče kategorizacije ispravnih emocija, ženski glasovi su se znatno točnije prepoznavali, nego emocije kod muških glasova. Razlog tomu je moguć što baza podataka sadrži više ženskih iskaza emocija te lošije pokazivanje emocija sa strane muškaraca čime se razlučivost između emocija gubi. Najviše odstupanja kod prepoznavanja ženskih emocija je kod stvarnih emocija sreće i predviđenih emocija ljutnje što je očekivano pošto su oba osjećaja pokazivana s visokim intenzitetom. Parovi koji su češće pogrešno kategorizirani kod ženskih glasova su (stvarno-predikcija): sreća-ljutnja, strah-sreća, gađenje-neutralno, tuga-neutralno, gađenje-ljutnja i strah-tuga. Najviše odstupanja kod prepoznavanja muških emocija je između stvarne emocije sreće i predviđene ljutnje, što je ista situacija kao i kod ženskih emocija. Kod muških glasova se uočava puno veće rasipavanje muških emocija s glavne dijagonale.



Slika 31 Matrica konfuzije CNN mreže

6. Akustički model povratne neuronske mreže

Prije samoga početka izrade neuronske mreže potrebno je učitati u program sve potrebne biblioteke s kojima će se izraditi sljedeći zadaci:

- Podjela podataka na skup za učenje i validaciju, te njihovu prilagodbu
- Izrada LSTM modela mreže
- Spremanje naučenog modela
- Izradu grafova točnosti i gubitka
- Izradu matrice konfuzije

Na slici [Slika 32] prikazane su učitane biblioteke.

```
1.  import pandas as pd
2.  import numpy as np
3.  import os
4.  from sklearn.model_selection import train_test_split
5.  from sklearn.preprocessing import LabelEncoder
6.  from sklearn.metrics import confusion_matrix
7.  from tensorflow import keras
8.  from keras.utils import to_categorical, np_utils
9.  import keras
10. from keras import optimizers
11. from keras.optimizers import Adam
12. from keras.models import Sequential, model_from_json
13. from keras.layers import LSTM, Dense, Bidirectional
14. from keras.layers import Dropout
15. import matplotlib.pyplot as plt
16. import seaborn as sns
```

Slika 32 Učitavanje biblioteka za LSTM mrežu

Programski kod za podjelu i pripremu baza podataka za učenje i validaciju LSTM mreže je potpuno jednak kao i kod CNN mreže. Uzela se jednaka raspodjela podataka za treniranje i validaciju, 75% za treniranje i 25% za učenje. Provodi se metoda *One Hot Encoding*, te proširuju dimenzije varijabli značajki. Potpuni programski kod navedenog se može naći u prilogu II LSTM_Model.py.

Izrada LSTM modela mreže je izrađena pomoću biblioteke *keras*. Napravljen model mreže je dvosmjerna LSTM mreža, što je prikazano na slici [Slika 33], koja započinje naredbom *Sequential()*. Naredbom *add()* se dodaju sljedeći slojevi, te se prvo dodaje ulazni sloj. Ulazni sloj mreže je dvosmjerni LSTM sloj, nakon kojeg se nalazi potpuno povezani sloj s *ReLU* aktivacijskom funkcijom. U modelu se nalazi jedan *Dropout* sloj kako bi se smanjila prenaučenost mreže. Zadnji sloj mreže je potpuno povezani sloj s 12 izlaza kako ima 12 klasa koje se kategoriziraju. Aktivacijska funkcija potpuno povezanog sloja je *Softmax*. Korišteni

optimizator za učenje LSTM mreže je *Adam* koji adaptivno primjenjuje stopu učenja čime se ubrzava proces učenja i njegova efikasnost.

```
#Model LSTM mreže
LSTM_Model= Sequential()
LSTM_Model.add(Bidirectional(LSTM(128),
input_shape=(Značajke_train_prošireno.shape[1], 1)))
LSTM_Model.add(Dense(64,activation='relu'))
LSTM_Model.add(Dropout(0.25))
LSTM_Model.add(Dense(12,activation='softmax'))
opti= optimizers.Adam(lr=0.001)
LSTM_Model.summary()
LSTM_Model.compile(loss='categorical_crossentropy'
                   ,optimizer=opti, metrics=['accuracy'])
```

Slika 33 Model dvosmjerne LSTM mreže

Proces učenja se izvršava u 25 epoha, gdje je korištena serija podataka od 16 [Slika 34].

```
#Učenje mreže
LSTM_Model_History=LSTM_Model.fit(Značajke_train_prošireno
                                  ,Label_trainKodiran
                                  ,batch_size=16
                                  ,epochs=25
                                  ,validation_data=(Značajke_test_prošireno
                                                    ,Label_testKodiran))
```

Slika 34 Učenje dvosmjerne LSTM mreže

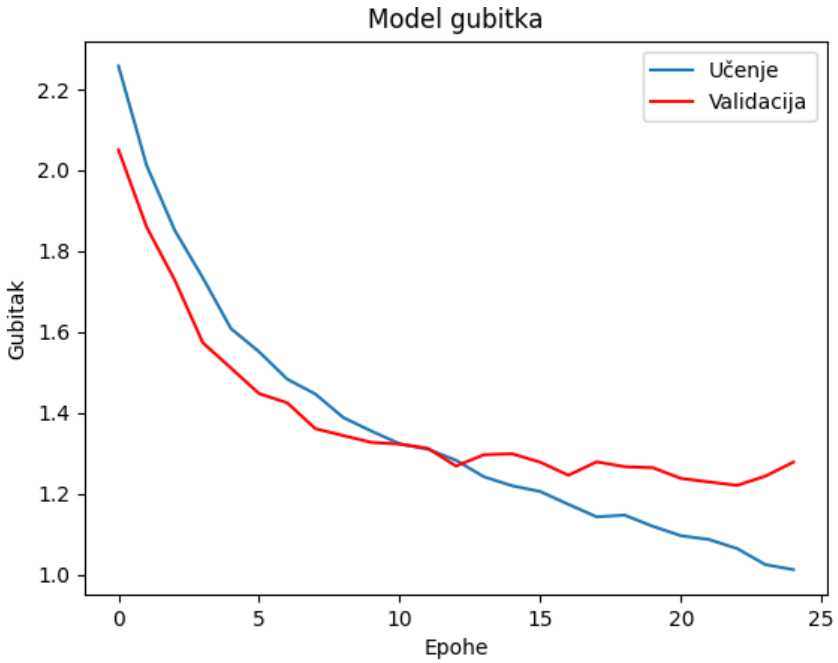
Nakon završavanja učenja dvosmjerne LSTM mreže model se sprema za kasniju uporabu. Sprema se arhitektura modela mreže zajedno s dobivenim težinskim vrijednostima mreže. Na slici [Slika 35] prikazan je programski kod za spremanje naučene dvosmjerne LSTM mreže.

```
#Spremanje LSTM modela
LSTM_Model_json=LSTM_Model.to_json()
with open("LSTM_Model.json", "w") as json_file:
    json_file.write(LSTM_Model_json)
LSTM_Model.save_weights("LSTM_Model.h5")
```

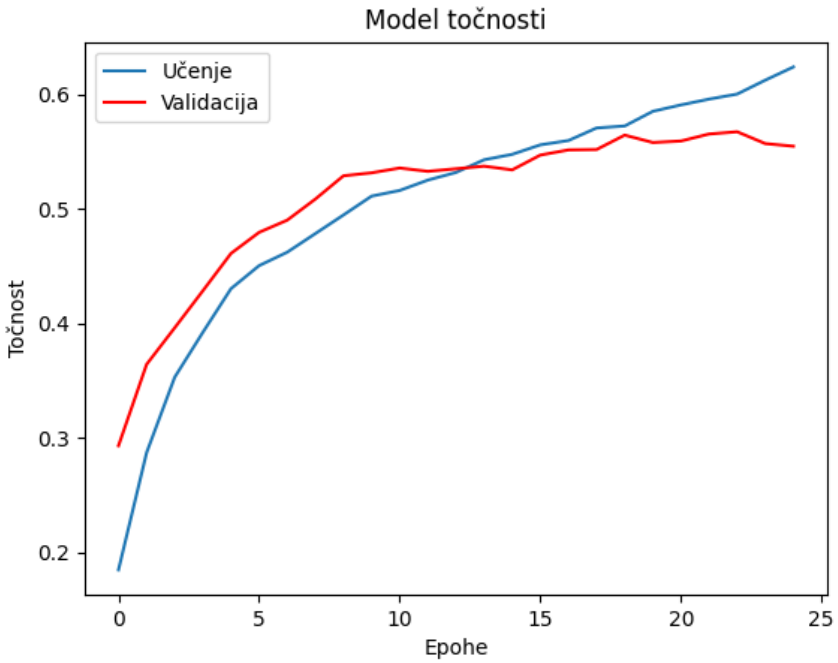
Slika 35 Spremanje dvosmjernog LSTM modela mreže

Na slici [Slika 36] prikazan je graf gubitka za dvosmjerni LSTM model mreže kroz proces učenja mreže, te se može uočiti zadovoljavajući oblik krivulje za prikupljenu bazu podataka i količinu klasa koje se kategoriziraju. Crvena linija pokazuje rezultate nad validacijskim podacima dok plava linija označava rezultate nad podacima za treniranje.

Na slici [Slika 37] prikazan je graf točnosti dvosmjerne LSTM mreže, gdje se prikazuje točnost prepoznavanja klasa kroz epohe učenja. Kao i na prijašnjem grafu krivulja je zadovoljavajuća, te se može vidjeti da se nije postigla prenaučenost mreže. Crvena linija pokazuje dobivenu točnost nad validacijskim podacima dok plava linija označava točnost nad podacima za treniranje. Postignuta točnost mreže pri završetku učenja je 55.47%.



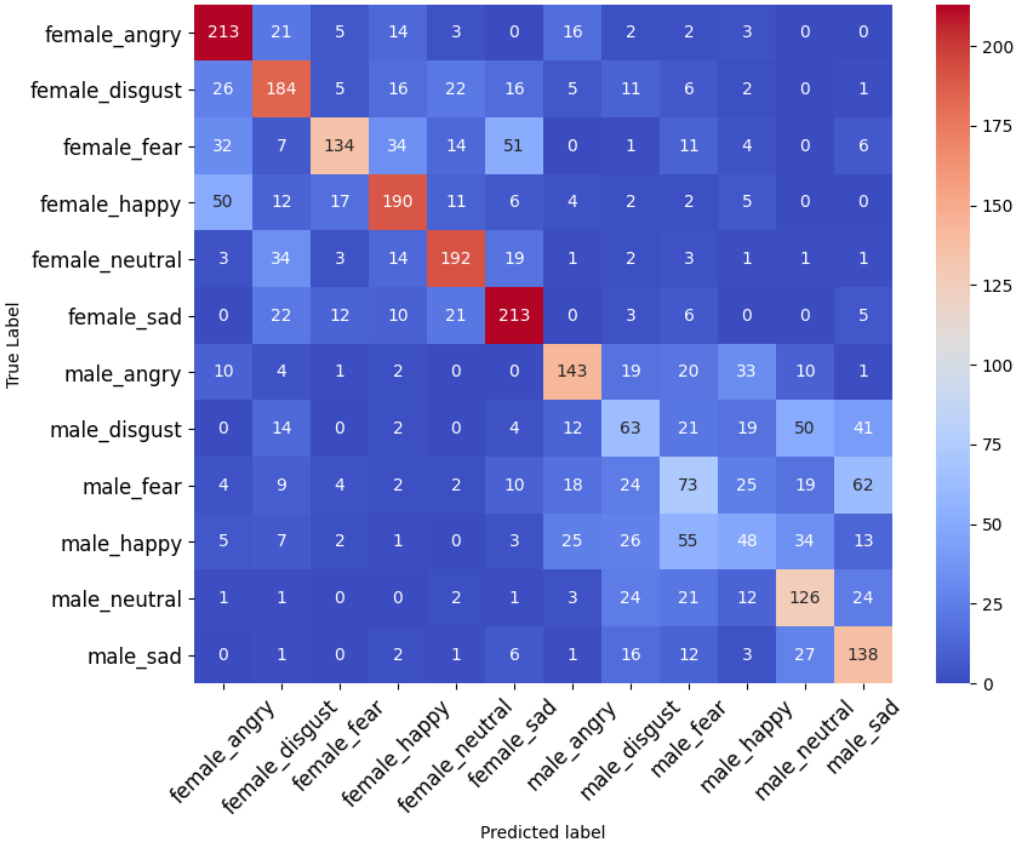
Slika 36 Graf gubitka dvosmjerne LSTM mreže



Slika 37 Graf točnosti dvosmjerne LSTM mreže

Matrica konfuzije dvosmjerne LSTM mreže [Slika 38] pokazuje ovisnost stvarnih klasa i predviđenih klasa. Redci matrice označuju stvarne klase dok stupci matrice označuju predviđene klase. Glavna dijagonala matrice predstavlja ispravno predviđene klase.

Iz pregleda matrice konfuzije može se uočiti relativno dobra razlučivost između spolova, gdje postoji mali postotak pogreške kod prepoznavanja spola kod emocija ljutnje i gađenja. Što se tiče kategorizacije ispravnih emocija ženski glasovi su se izuzetno točnije prepoznavali, nego emocije kod muških glasova. Razlog tomu je moguć što baza podataka sadrži više ženskih iskaza emocija, te lošije pokazivanje emocija sa strane muškaraca čime se razlučivost između emocija gubi. Najviše odstupanja kod prepoznavanja ženskih emocija je kod stvarnih emocija straha i predviđenih emocija tuge. Parovi koji su češće pogrešno kategorizirani kod ženskih glasova su (stvarno-predikcija): strah-tuga, sreća-ljutnja, strah-sreća, neutralno-gađenje i strah-ljutnja. Najviše odstupanja kod prepoznavanja muških emocija je između stvarne emocije straha i predviđene tuge. Kod muških glasova se uočava puno veće rasipavanje muških emocija s glavne dijagonale, te je očito da mreža kod muških glasova loše kategorizira osjećaje gađenja,straha i sreće.



Slika 38 Matrica konfuzije LSTM mreže

7. Prepoznavanje emocija iz glasa

Nakon izrađenih CNN i LSTM modela neuronskih mreža, te njihovog učenja, izrađena su dva programa za očitavanje emocija iz govora. Jedan program koristi CNN model mreže dok drugi koristi LSTM model mreže. Oba programa se u cijelosti nalaze u prilogu II CNN_Prepoznavanje.py i LSTM_Prepoznavanje.py. Kako su oba programa slična proći će se kroz program CNN_Prepoznavanja uz napomenu razlika između oba programa.

Kako se radi o programskom kodu on započinje učitavanjem potrebnih biblioteka za rad, te se mogu vidjeti na slici [Slika 39]. *PyAudio* je besplatna *Python* biblioteka koja omogućuje jednostavno snimanje i reprodukciju audio zapisana na različitim platformama kao što su: *Windows, Linux* i *Apple mac OS X* [37].

```
1.  #bibliote
2.  import pyaudio
3.  import pandas as pd
4.  import numpy as np
5.  import os
6.  import pyaudio
7.  import wave
8.  import sys
9.  import json
10. from keras.models import model_from_json
11. import librosa
12. import matplotlib
13. import matplotlib.pyplot as plt
14.  import pyloudnorm as pyln
```

Slika 39 Učitavanje potrebnih biblioteka

Nakon učitavanja biblioteka učitava se spremljeni model neuronske mreže zajedno s težinskim vrijednostima mreže. Razlika između programa je ta da program CNN_Prepoznavanje učitava prethodno spremljen CNN model mreže zajedno s pripadajućim težinskim faktorima, a program LSTM_Prepoznavanje učitava LSTM model mreže s pripadajućim težinskim faktorima. Na slici [Slika 40] može se vidjeti kod za učitavanje CNN modela mreže dok je na slici [Slika 41] prikazan kod za učitavanje LSTM modela mreže.

```
#Učitavanje Modela
#CNN
CNN_json=open('CNN_Model.json', 'r')
loaded_CNN_json=CNN_json.read()
CNN_json.close()
loaded_CNN_Model=model_from_json(loaded_CNN_json)
loaded_CNN_Model.load_weights("CNN_Model.h5")
```

Slika 40 Učitavanje CNN modela

```
#Učitavanje Modela
#LSTM
LSTM_json=open('LSTM_Model.json', 'r')
loaded_LSTM_json=LSTM_json.read()
LSTM_json.close()
loaded_LSTM_Model=model_from_json(loaded_LSTM_json)
loaded_LSTM_Model.load_weights("LSTM_Model.h5")
```

Slika 41 Učitavanje LSTM modela

Potom dolazi kod koji će stvoriti mapu imenom "temp" gdje će se pohranjivati snimljeni audio zapisi. Prvo će se provjeriti je li mapa postojeća te ako nije postojana, izrađuje se. Mjesto mape "temp" nalazit će se u radnoj mapi gdje se nalazi program. Zadnji redak koda stvara globalnu varijablu putanje s imenom audio, na kojega će se kasnije nadovezati drugi dio imena trenutačnog zapisa. Kod je prikazan na slici [Slika 42]

```
#Mapa za snimanje audio datoteka
cwd = os.getcwd()
Output_Folder = os.path.join(cwd, "temp")
if not os.path.exists(Output_Folder):
    os.mkdir(Output_Folder)
cat_str =Output_Folder + '/audio'
```

Slika 42 Stvaranje mape za spremanje audio zapisa

Nadalje, napravljene su dvije tablice koje će se popunjavati kako se snimaju audio zapisi i varijabla "classes" sa šest emocija [Slika 43]. Varijabla "classes" koristi se u daljnjem dijelu programa za spajanje dobivenih vrijednosti s emocijama. Tablica Značajke_potpuno služi za pohranu numeričkih vrijednosti zbrojenih izlaza neuronske mreže dok tablica Izlaz služi za pohranu emocija s njihovom numeričkom vrijednošću koja predstavlja zastupljenost određene emocije. Ujedno se vrši ispis upute za prekid snimanja. U zadnje dvije varijable na slici [Slika 43] unosi se broj uzorka i emocije kako bi se mogli naknadno izraditi grafovi.

```
#Podjela Emocija
classes=['angry','disgust','fear','happy','neutral','sad']
#Podatci izcučenih značajki
Značajke_potpuno=pd.DataFrame(columns=['Emocije'])
Izlaz=pd.DataFrame(columns=['Kategorizacija'])
#Uputa za prekid snimanja
print('Za prekid snimanja pritisnite Ctrl-C')
#graf varijable
t=[]
Emocija=[]
```

Slika 43 Priprema prostora i varijabli za program

Nadalje se postavljaju globalni parametri za snimanje audio zapisa i brojilo unutar programa koje će se povisivati prolaskom kroz *while* petlju u nastavku programa [Slika 44]. *RATE* je varijabla uzimanja uzoraka te je postavljena na 44100 Hz. *CHUNK* je broj uzoraka koji se procesiraju odjednom kako ne bi došlo do zagušenja procesora. Namješteni format snima uzorke veličine od 2 bajta tj. 16 bita.

```
#Parametri za snimanje audio datoteka
RATE = 44100
CHUNK = 1024
CHANNELS = 1
FORMAT = pyaudio.paInt16
RECORD_SECONDS = 2
sample_number = 0
```

Slika 44 Postavljanje parametara snimanja

Zatim se određuje funkcija koja će snimati audio zapis od trajanja dvije sekunde i spremati snimku u *wave* formatu [Slika 45]. Parametri snimanja se učitavaju iz već određenih globalnih varijabli pomoću naredbe *audio.open*. Postavlja se duljina trajanja snimanja preko for petlje u ovisnosti o broju uzimanja uzoraka i veličine *Chunk-a*, te se također spremaju snimane informacije u varijablu *frame*. Nakon završetka for petlje gasi se snimanje, te se postavlja kompatibilni *wave* format u koji se tada spremaju podaci iz varijable *frame*.

```
def capture_audio(test):
    audio = pyaudio.PyAudio()
    #Početak snimanja isječka
    stream = audio.open(format=FORMAT, channels=CHANNELS, rate=RATE
                        ,input=True, frames_per_buffer=CHUNK)
    frames = []
    for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
        data = stream.read(CHUNK)
        frames.append(data)
    #Prekid snimanja isječka
    stream.stop_stream()
    stream.close()
    audio.terminate()
    waveFile = wave.open(test, 'wb')
    waveFile.setnchannels(CHANNELS)
    waveFile.setsampwidth(audio.get_sample_size(FORMAT))
    waveFile.setframerate(RATE)
    waveFile.writeframes(b''.join(frames))
    waveFile.close()
```

Slika 45 Funkcija snimanja audio zapisa

Na slici [Slika 46] može se uočiti da je *while* petlja unutar postupka *try except*. Postupak *try except* se koristi kako bi se omogućio prekid snimanja pritiskom tipki ctrl-c. Postupak snimanja se vrti dok se nije ponovio 900 puta, što je nešto više od 30 minuta snimanja zbog kratke stanke između uzimanja svakog koraka zbog procesiranja podataka. Zatim se stvara varijabla u koju će se snimati zvučni zapis koja je jednaka već napravljenoj varijabli *cat_str*, te se imenu varijable nadodaje broj koraka i nastavak ".*wav*". Kako bi zvučni zapis mogli pustiti kroz napravljeni model mreže, moramo prvo pretprocesirati audio podatak i izvući akustične značajke iz njega. Prvo se učitava zapis putem L*ibrosa* biblioteke, te se izvršava uklanjanje tišine s početka i kraja. Završetkom uklanjanja tišine s početka i kraja audio zapisa završeno je pretprocesiranje. Zatim se izvlače jednake akustičke značajke kao i kod baze podatka za učenje mreže. Nakon izvlačenja značajki postavlja se zapis kompatibilan ulazu neuronske mreže te se podatci postavljaju u kompatibilnu formu za mrežu. Razlika između CNN programa i LSTM programa je samo izmjena u učitanom modelu mreže za klasifikaciju. Pri završetku klasifikacije, neuronska mreža nam daje 12 izlaza. Izlazi koji predstavljaju istu emociju kod ženskih i muških osoba zbrajaju se kako bi imali klasifikaciju samo emocija. Trenutačno zbrojeno stanje emocija iz izlaza mreže je označeno varijablom "Predikcija". Varijabla se zatim sprema u ovisnosti na broj koraka u tablicu Značajke_potpuno, zbog mogućnosti korištenja vrijednosti iz prethodnih stanja varijable "Predikcija".

Zatim se koristi fuzijska tehnika, gdje trenutačna vrijednost emocija je jednaka vrijednostima iz varijable “Predikcija“ u prvih 4 koraka. S petim korakom i nadalje trenutačna vrijednost emocija ovisi o zadnjih 5 spremljenih varijabli “Predikcija“ iz tablice Značajke_potpuno. Numeričke vrijednosti se spremaju u varijablu “fusion“. Zatim se varijabla “fusion“ spaja s varijablom “classes“ [Slika 43] kako bi spojili numeričke vrijednosti i imena emocija. Na kraju se uzima maksimalna vrijednost emocije za ispis i povisuje broj koraka za 1.

Kod retka 53 [Slika 46] izvršava se brisanje trenutačnog audio zapisa kako se ne bi prenapunila memorija računala, ali se može ignorirati redak postavljanjem simbola ljestve ispred njih. Postavljanjem simbola ljestve zadržat će se pohranjeni isječci na računalu.

```
try:
  while sample_number < 900:#30min
      WAVE_OUTPUT_FILENAME = cat_str + str(sample_number) + ".wav"
      capture_audio(WAVE_OUTPUT_FILENAME)
      # Izvlačenje značajki
      # resample baze podataka s brzinom uzrokovanja 44kHz
      Značajke = pd.DataFrame(columns=['Features'])
      resdata, brzina_uzoraka = librosa.load(WAVE_OUTPUT_FILENAME
                                  ,res_type='kaiser_best',sr=44000)
      brzina_uzoraka = np.array(brzina_uzoraka)
      # Uklanjanje tišine iz zvučnih zapisa ispod 20dB
      resdata,   = librosa.effects.trim(y=resdata, top db=30)
      rezultat = np.array([])
      # Izvlačenje srednje vrijednosti MFCC značajki
      MFCC_značajke = np.mean(librosa.feature.mfcc(y=resdata,
                          sr=brzina_uzoraka, n_mfcc=40,axis=1)
      rezultat = np.hstack((rezultat, MFCC_značajke))
      # Izvlačenje srednje vrijednosti MEL spektograma
       MEL_značajke =
 np.mean(librosa.feature.melspectrogram(resdata, sr=brzina uzoraka)
                                        ,axis=1)
      rezultat = np.hstack((rezultat, MEL_značajke))
      Značajke.loc[sample_number] = [rezultat]
      Značajke = pd.DataFrame(Značajke['Features'].values.tolist())
      ZnačajkeMreža = np.array(Značajke)
      ZnačajkeMreža = np.expand_dims(ZnačajkeMreža, axis=2)
      # Klasifikacija CNN
      CNN = loaded_CNN_Model.predict(ZnačajkeMreža, batch_size=16
                                     ,verbose=0)
      Predikcija = [CNN[0][0] + CNN[0][6], CNN[0][1] + CNN[0][7]
                   ,CNN[0][2] + CNN[0][8],CNN[0][3] + CNN[0][9]
                   ,CNN[0][4] + CNN[0][10], CNN[0][5] + CNN[0][11]]
      Predikcija = np.array(Predikcija)
      Značajke_potpuno.loc[sample_number] = [Predikcija]
      if sample_number <= 3:
          fusion = Predikcija
      else:
          Značajke_potpunoArray =
 np.array(Značajke_potpuno['Emocije'])
          fusion = 0.45 * Značajke_potpunoArray[sample_number]+\
                   0.175 * Značajke_potpunoArray[sample_number-1]+\
                   0.15 * Značajke_potpunoArray[sample_number-2]+\
                   0.125 * Značajke_potpunoArray[sample_number-3]+\
                   0.1 * Značajke_potpunoArray[sample_number-4]
      CNNp = dict(zip(classes, fusion))
      Izlaz.loc[sample_number] = [CNNp]
      maximum = max(CNNp, key=CNNp.get)
      print(maximum)
      Emocija.append(maximum)
      os.remove(WAVE_OUTPUT_FILENAME)
      sample_number += 1
      t.append(sample_number)
except KeyboardInterrupt
  print("Snimanje završeno")
  pass
```

Slika 46 Kod While petlje snimanja i obrade audio zapisa

Pri završetku programa iscrtava se graf koji prikazuje koja emocija je bila kategorizirana u određenom koraku snimanja. Ujedno se i spremaju u tablični zapis kategorizirane numeričke vrijednosti pripadajućim emocijama [Slika 47].

```
1. fig=plt.figure(1)
2. plt.plot(t,Emocija, linestyle='--', marker='o', color='b')
3. plt.yticks(classes)
4. plt.show()
5. fig.savefig('CNN_kategorizacija.png')
6. Izlaz.to_csv('Kategorizacija2_po_2s.csv',index=False)
```

Slika 47 Kod ispisa grafa emocija i spremanja tablice kategoriziranih emocija

8. Testiranje programa prepoznavanja emocija

U ovom dijelu će se prikazati rezultati izrađenih programa snimanjem ljudskih subjekata. Za svaku emociju osim gađenje čitatelj će proći kroz skup rečenica koje sadrže ciljani emocionalni sadržaj kako bi se lakše uspjela odglumiti emocija. Skup rečenica sadrži 130 različitih rečenica od kojih 30 zauzima neutralnu emociju, a po 25 rečenica za emocije: ljutnje, tuge, sreće i straha [38].

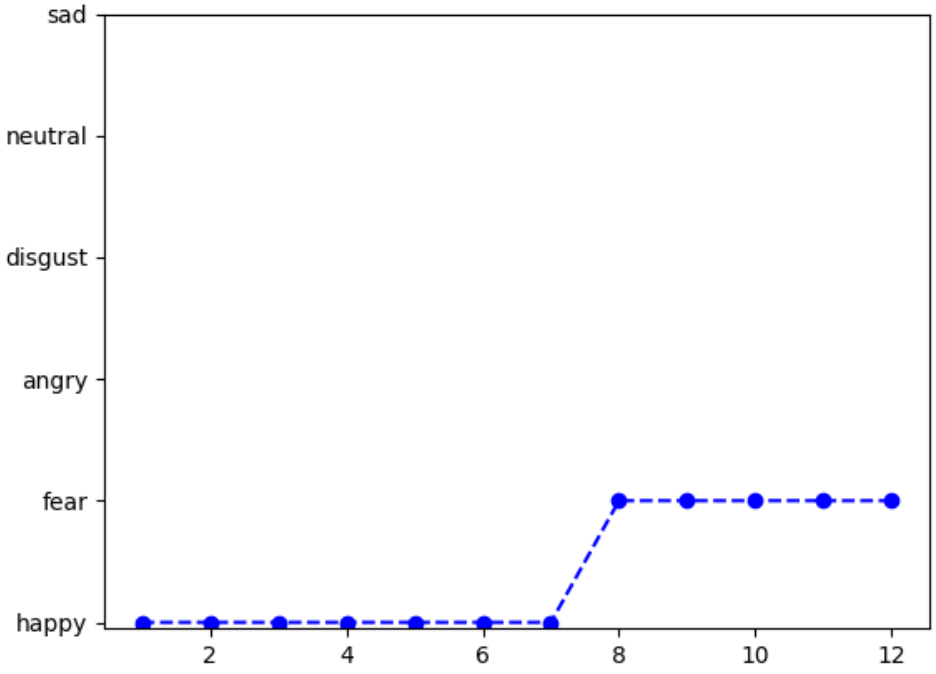
Testiranje će se izvesti na dva govornika, jednog ženskog govornika i jednog muškog. Zbog mogućnosti usporedbe podataka između mreža prethodno će se snimiti govor i pustiti drugim reprodukcijskim uređajem kako bi se jednak govor mogao snimiti i obraditi s oba programa. Svaki skup rečenica je snimljen zasebno. Bitno je napomenuti kako govornici nisu profesionalni glumci ni mogućnost snimanja nije u profesionalnim i kontroliranim uvjetima.

U nastavku slijede rezultati snimanja i kategorizacije emocija.

Skup neutralno

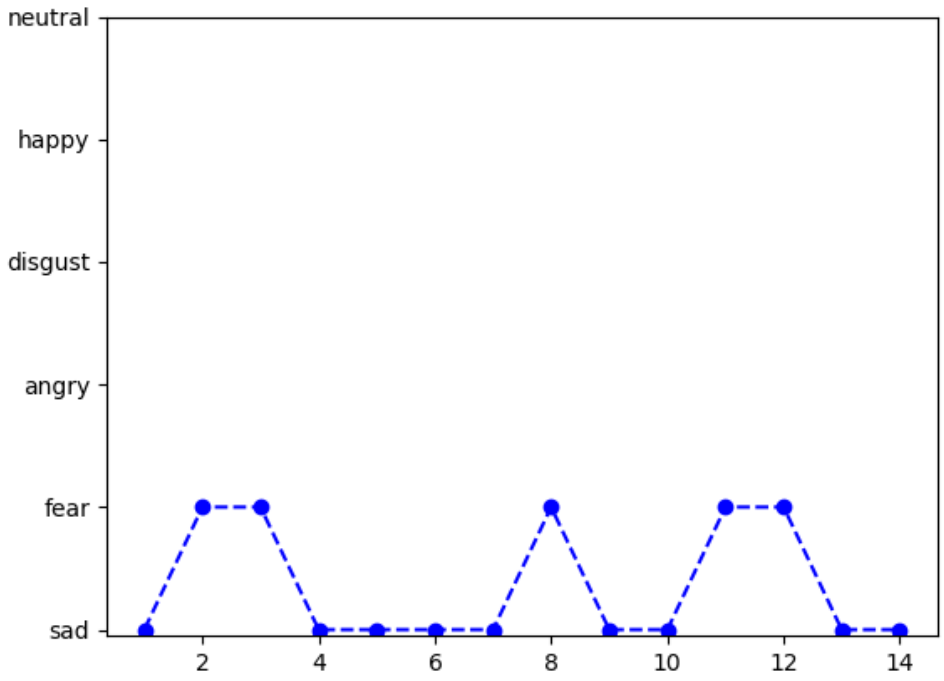
Ženski govornik

CNN program:



Slika 48 Ženski govornik neutralno CNN

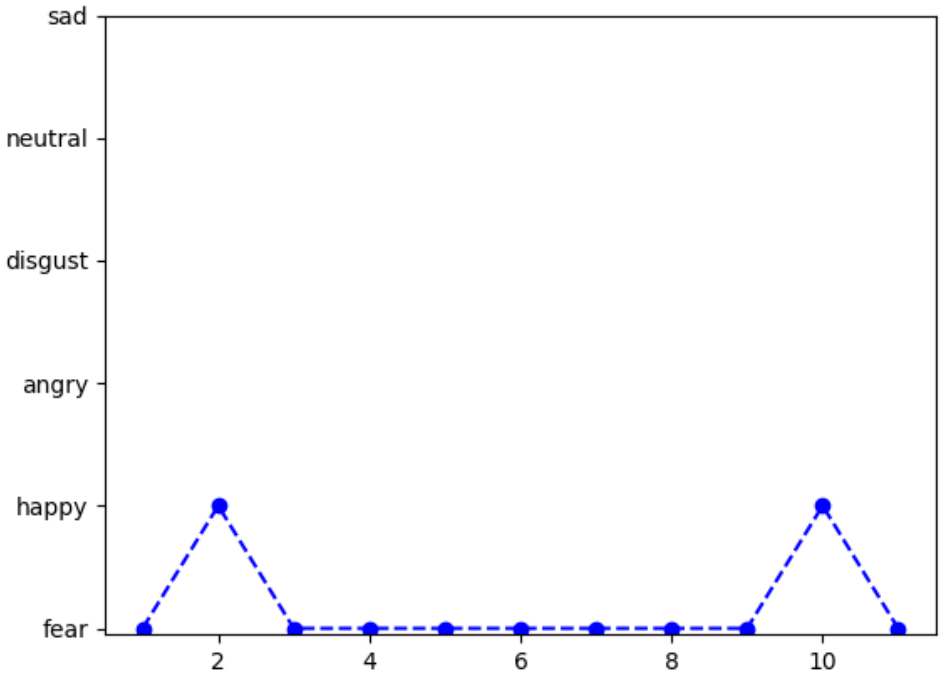
LSTM program:



Slika 49 Ženski govornik neutralno LSTM

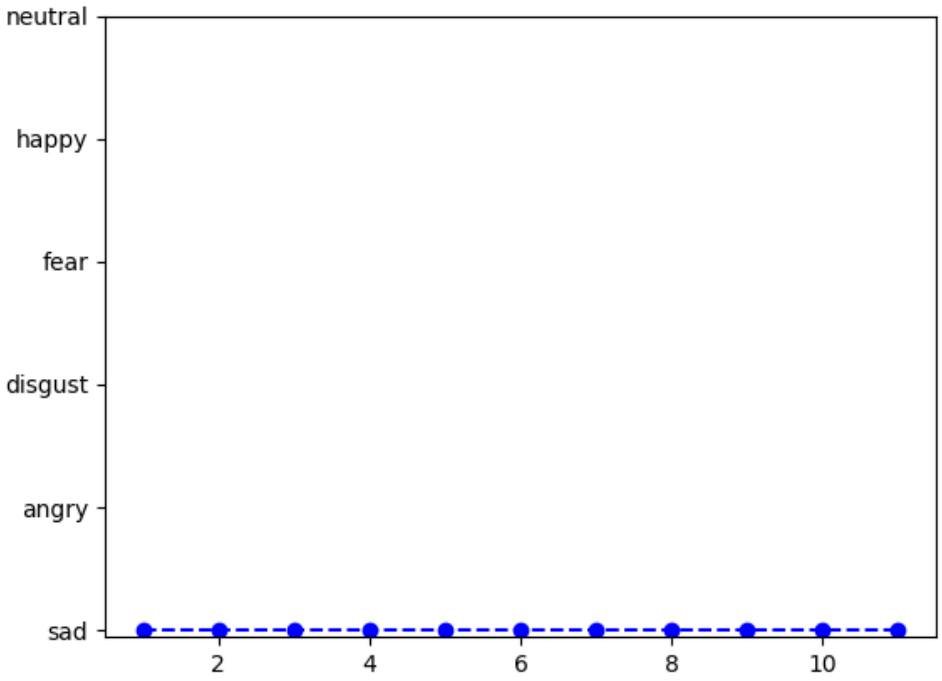
Muški govornik

CNN program:



Slika 50 Muški govornik neutralno CNN

LSTM program:

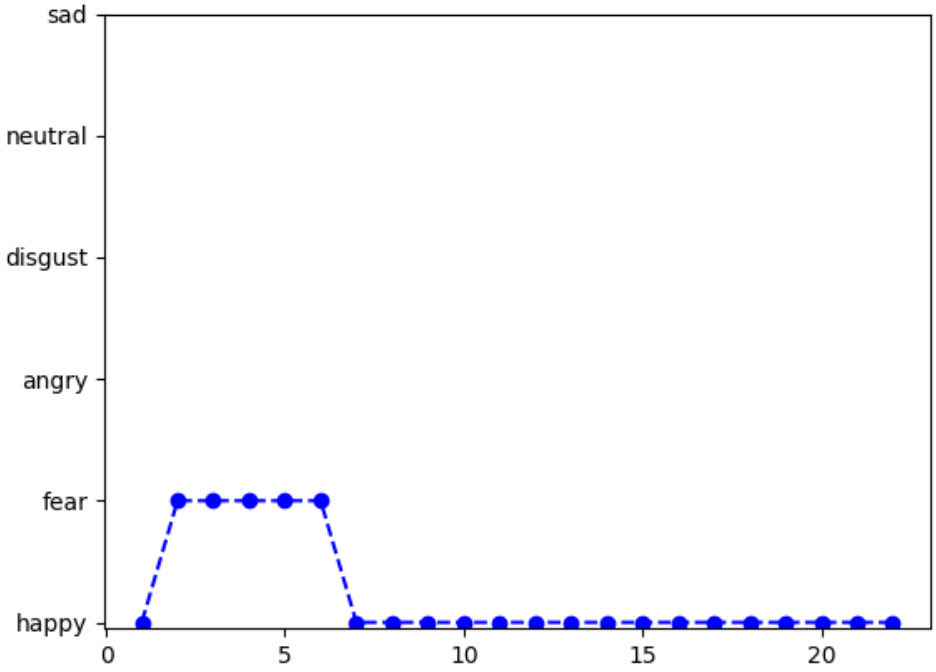


Slika 51 Muški govornik neutralno LSTM

Skup sreća

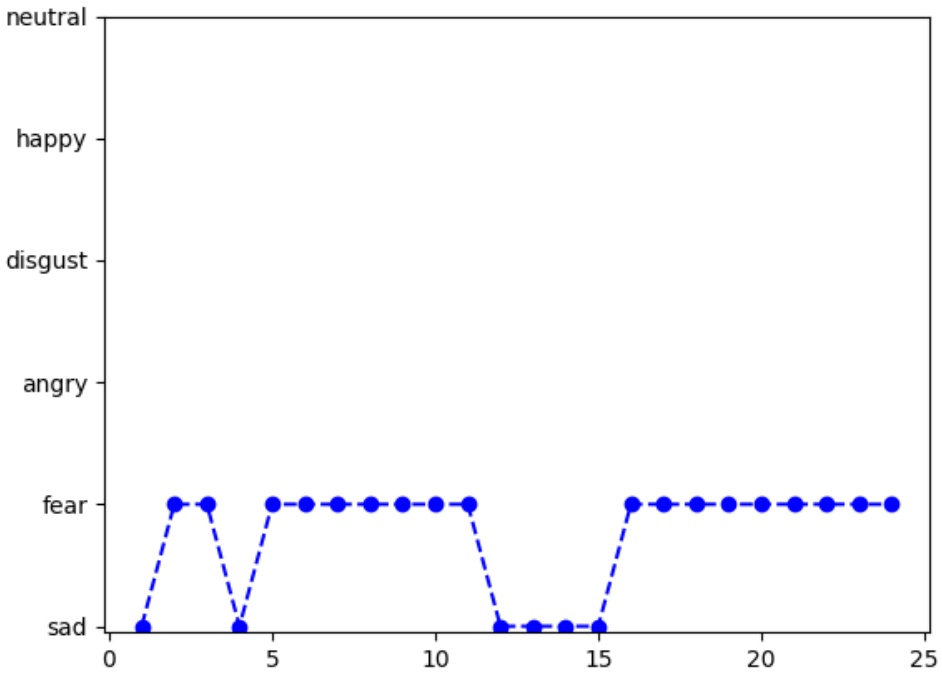
Ženski govornik

CNN program:



Slika 52 Ženski govornik sreća CNN

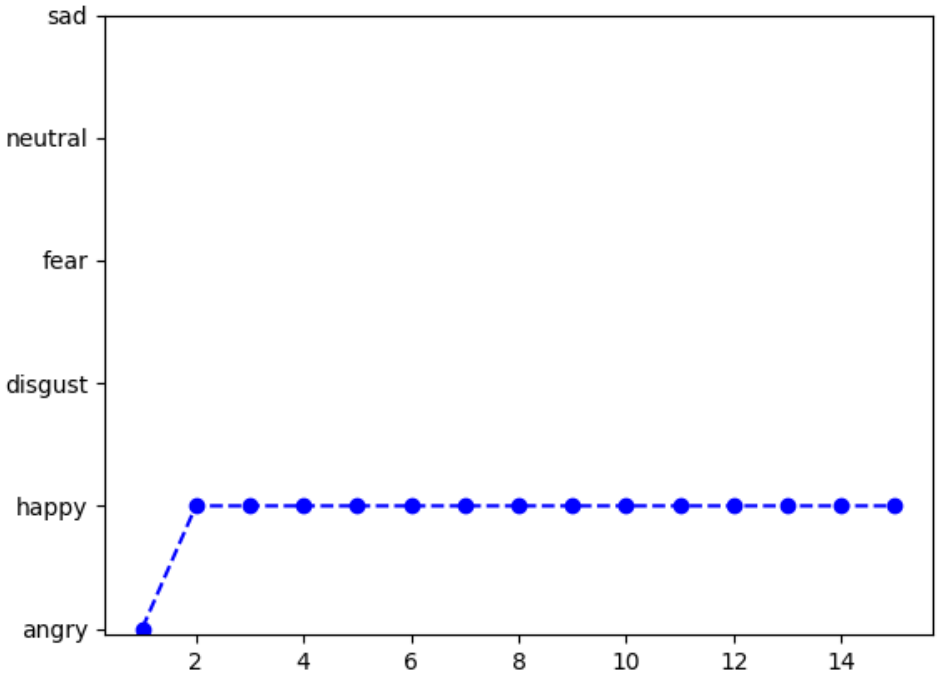
LSTM program:



Slika 53 Ženski govornik sreća LSTM

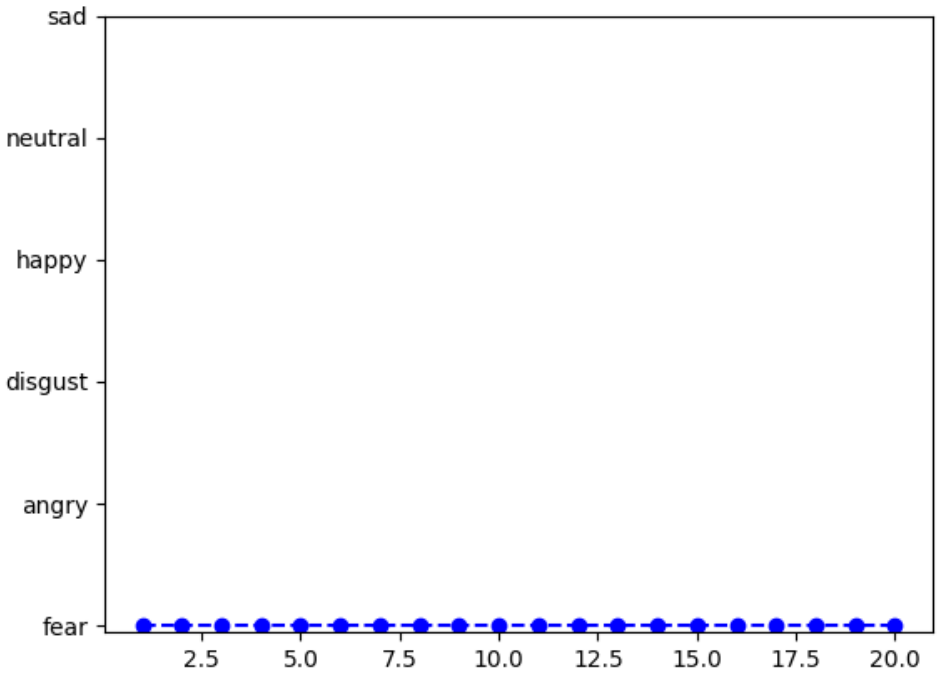
Muški govornik

CNN program:



Slika 54 Muški govornik sreća CNN

LSTM program:

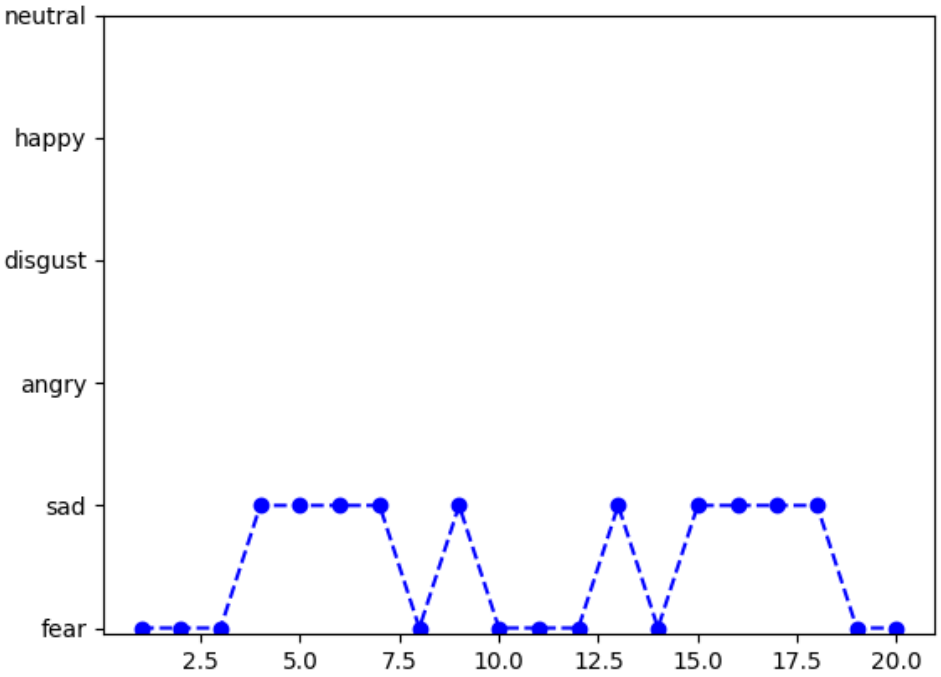


Slika 55 Muški govornik sreća LSTM

Skup tuga

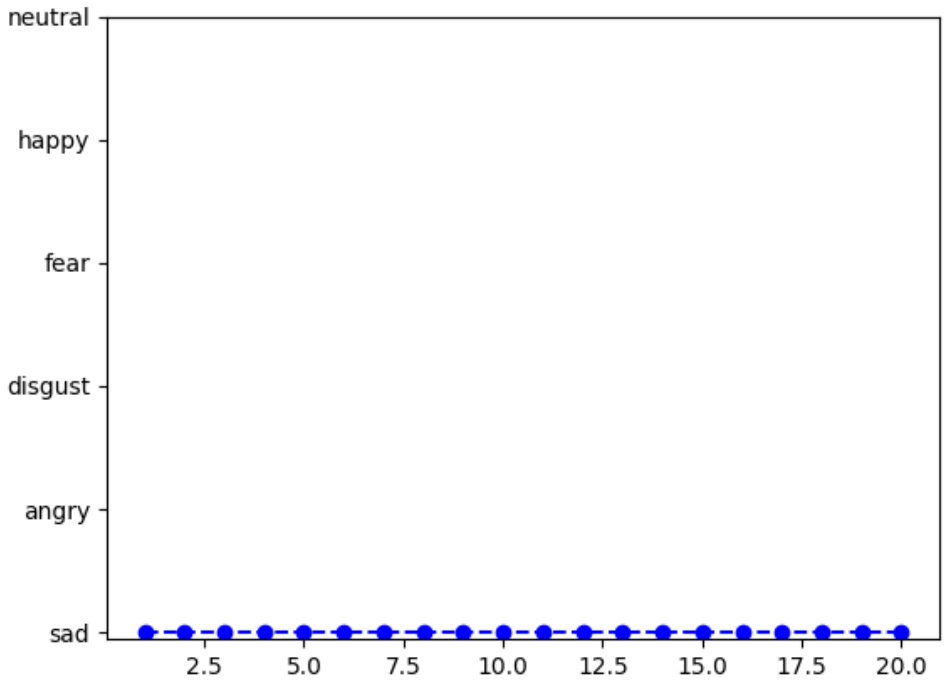
Ženski govornik

CNN program:



Slika 56 Ženski govornik tuga CNN

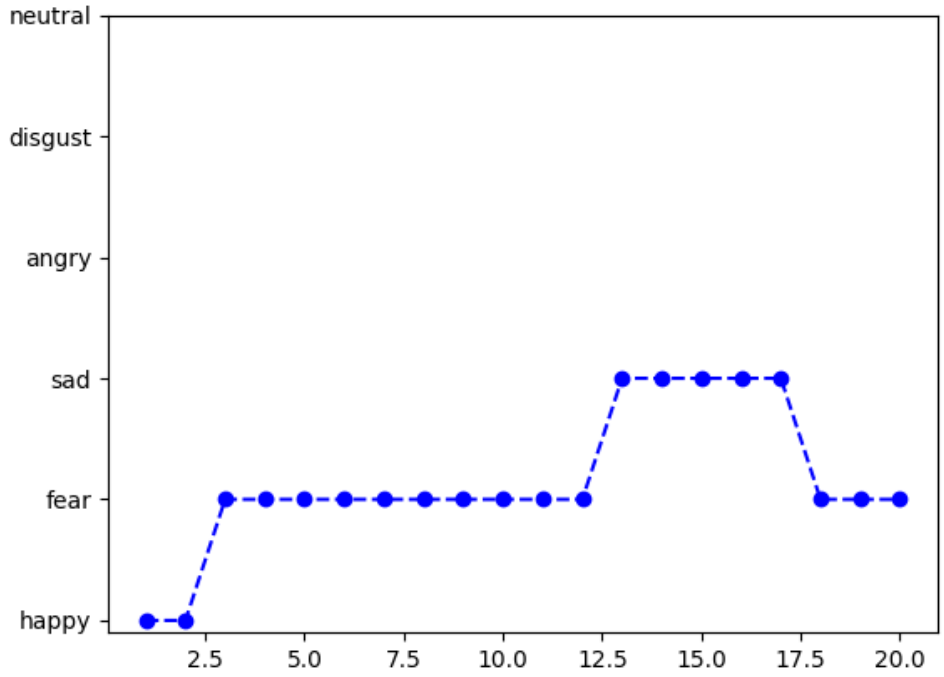
LSTM program:



Slika 57 Ženski govornik tuga LSTM

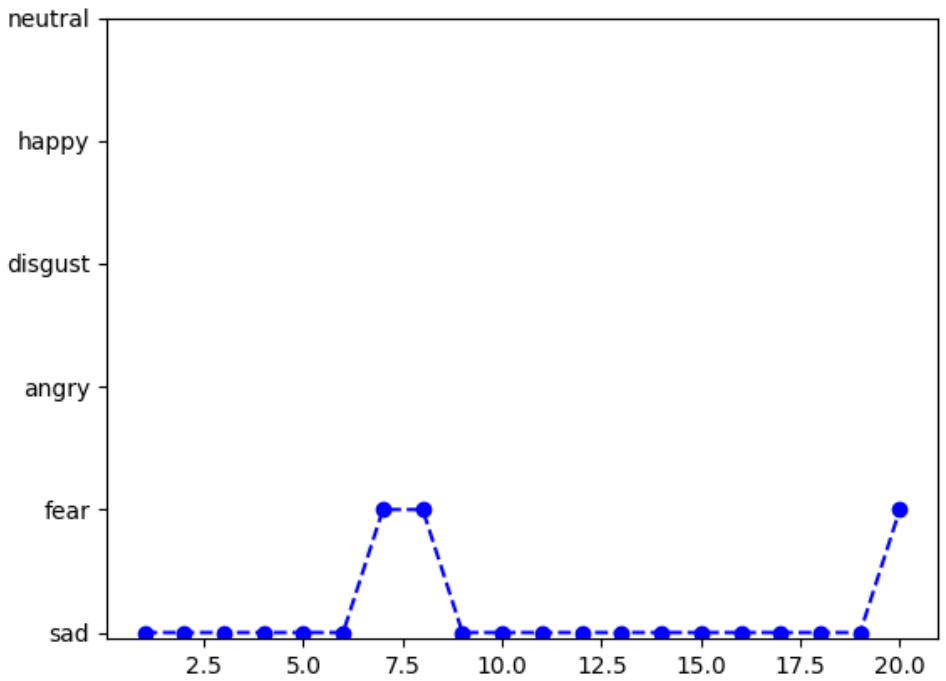
Muški govornik

CNN program:



Slika 58 Muški govornik tuga CNN

LSTM program:

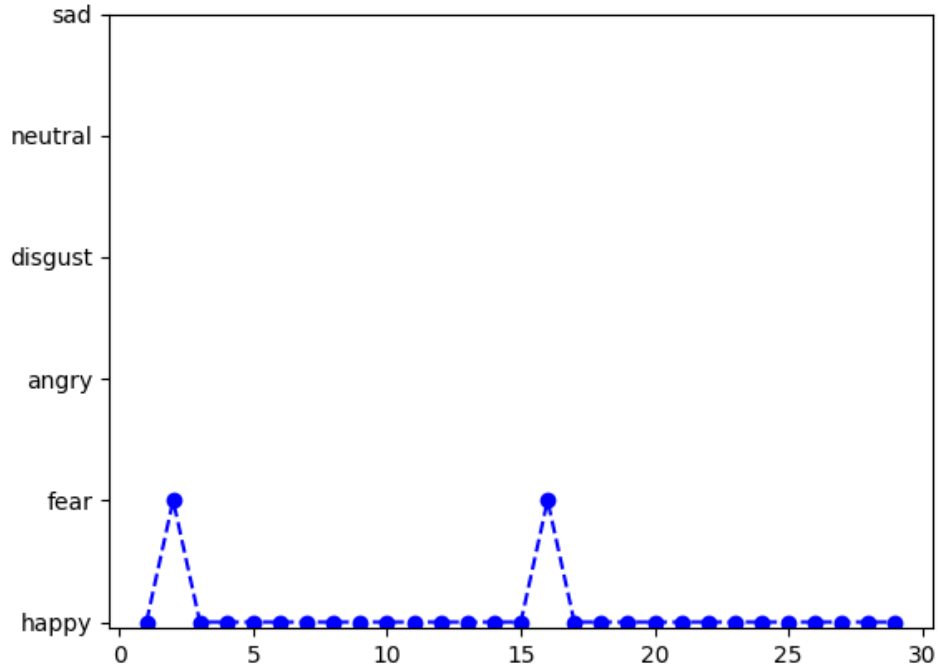


Slika 59 Muški govornik tuga LSTM

Skup ljutnja

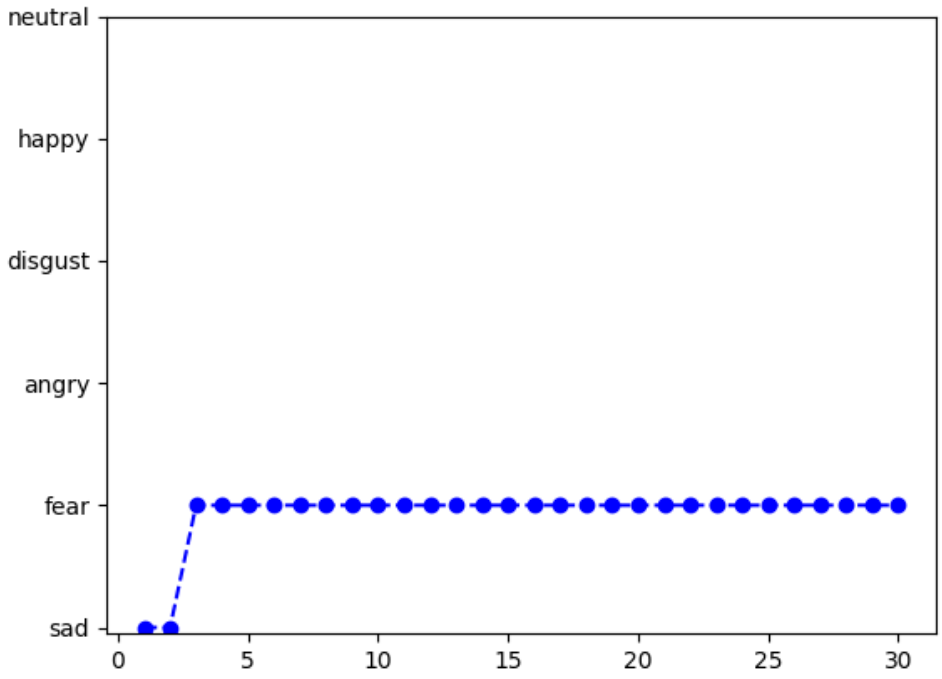
Ženski govornik

CNN program:



Slika 60 Ženski govornik ljutnja CNN

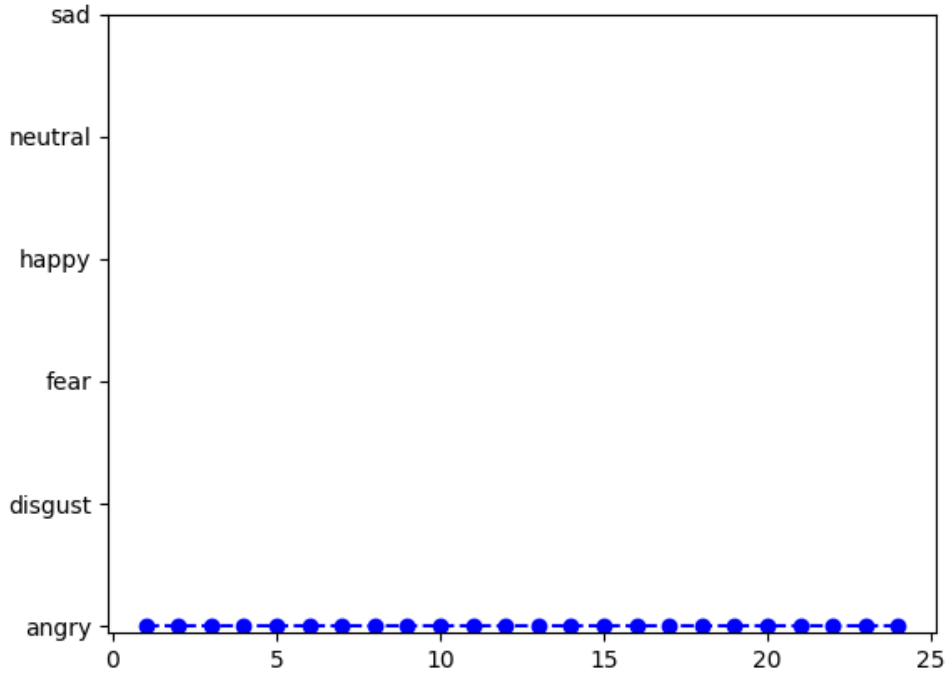
LSTM program:



Slika 61 Ženski govornik ljutnja LSTM

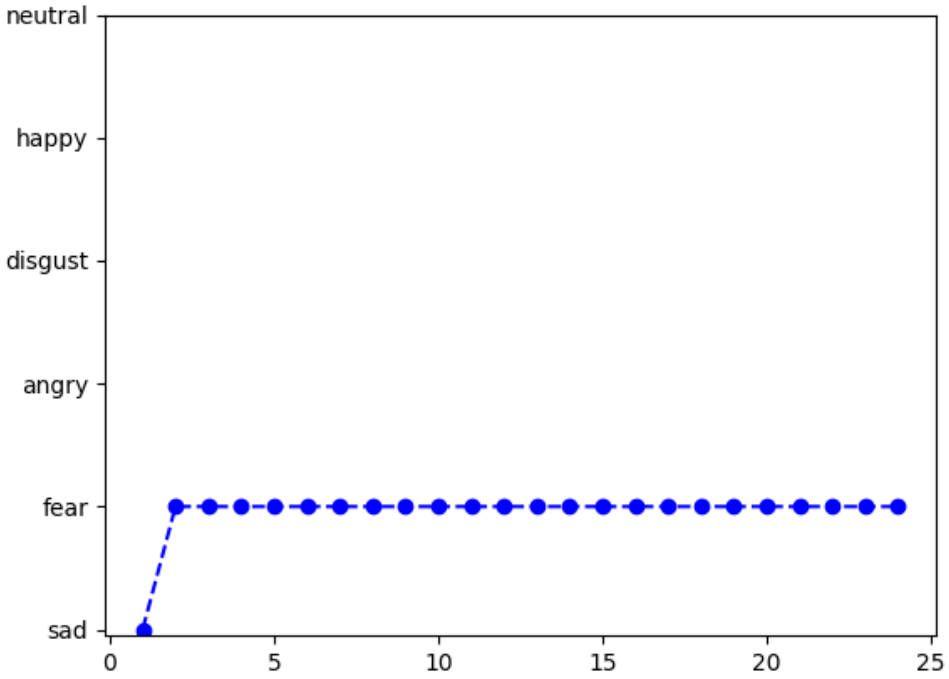
Muški govornik

CNN program:



Slika 62 Muški govornik ljutnja CNN

LSTM program:

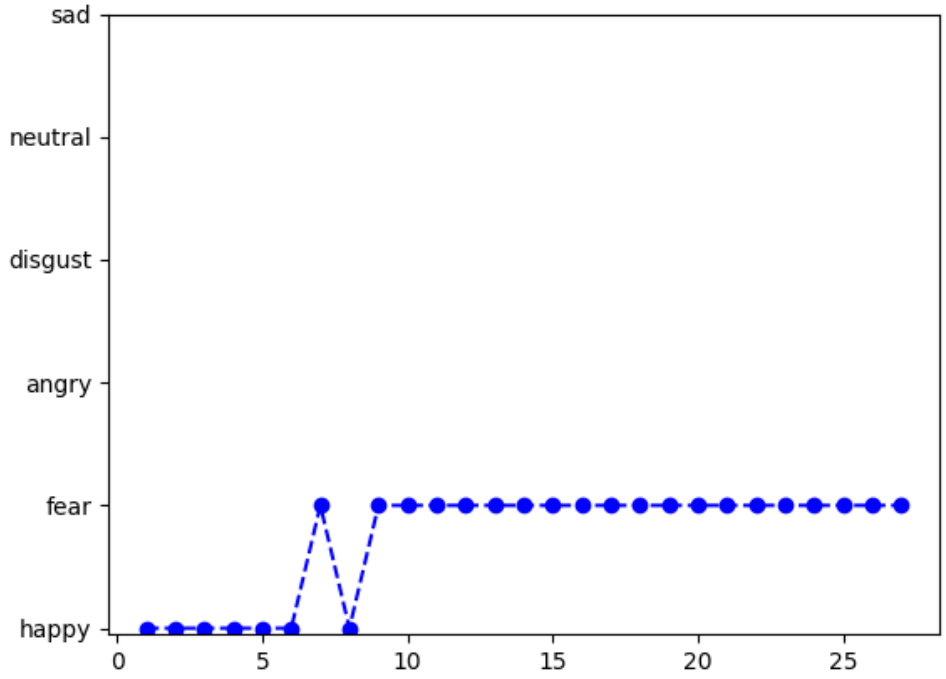


Slika 63 Muški govornik ljutnja LSTM

Skup strah

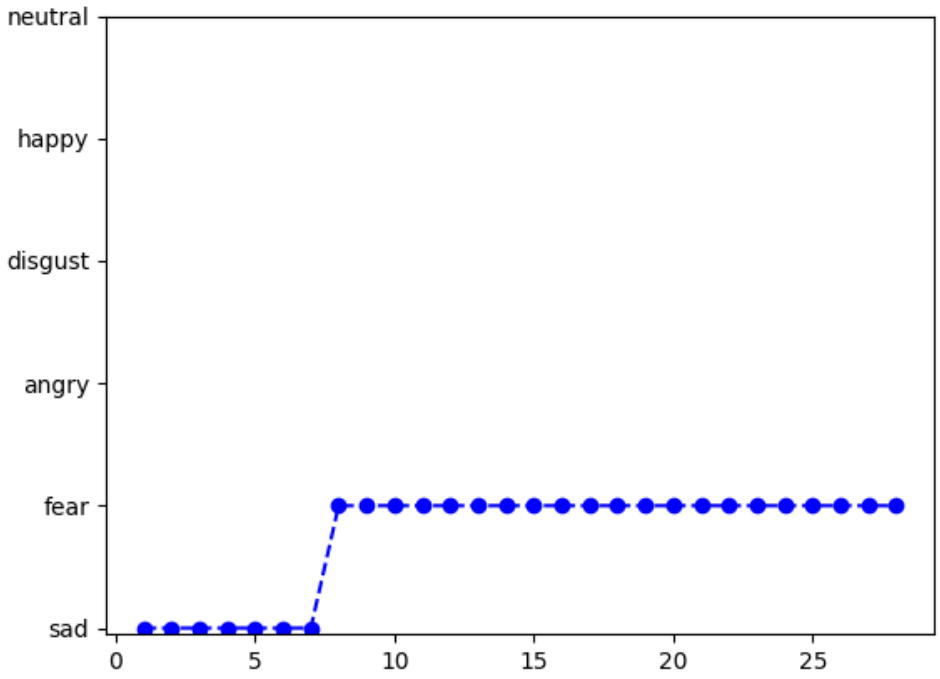
Ženski govornik:

CNN program:



Slika 64 Ženski govornik strah CNN

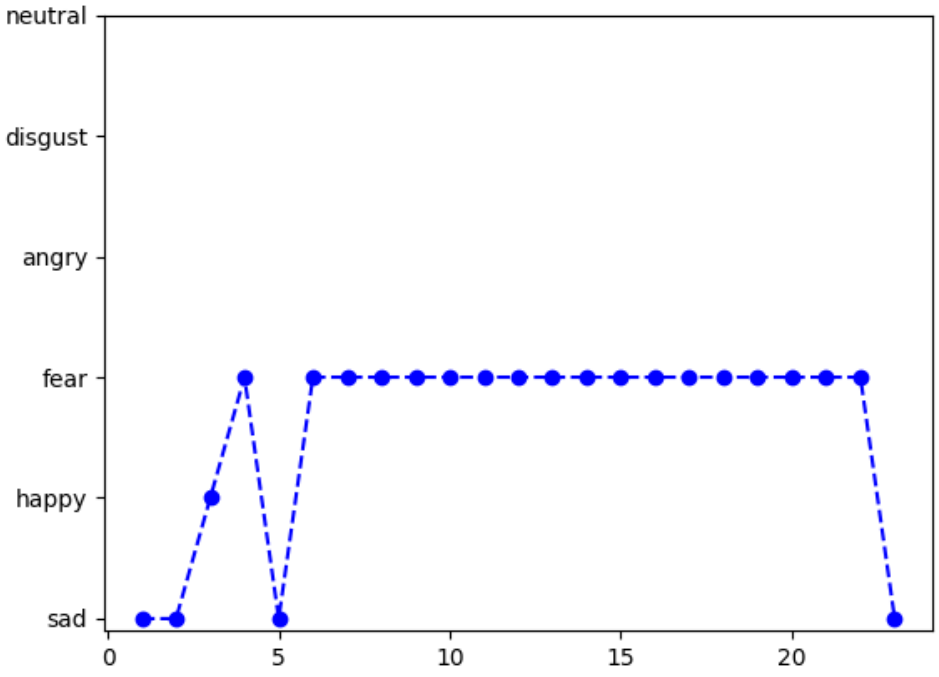
LSTM program:



Slika 65 Ženski govornik strah LSTM

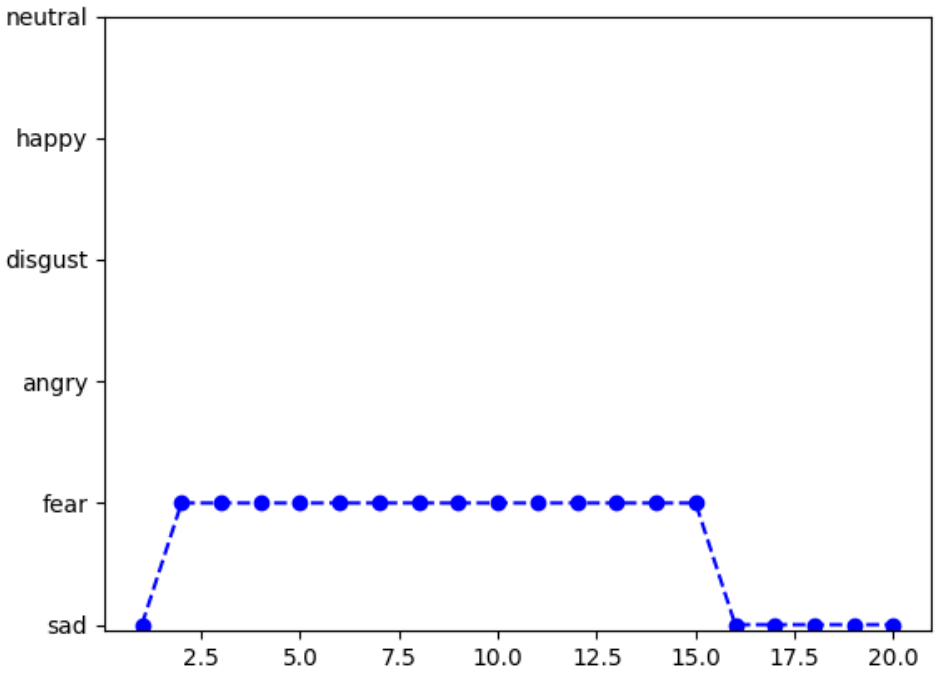
Muški govornik

CNN program:



Slika 66 Muški govornik strah CNN

LSTM program:



Slika 67 Muški govornik strah LSTM

9. Zaključak

U ovome radu izrađena su dva modela neuronskih mreža. Prvi model mreže je konvolucijska neuronska mreža dok je drugi model dvostruka LSTM povratna neuronska mreža. Obje mreže su bile učene i validirane nad istim skupom podataka koji se sastoji od izvučenih akustičkih značajki iz audio zapisa. Audio zapisi prikazuju odglumljene emocije sa strane profesionalnih glumaca. Kod validacije podataka nakon učenja mreže se moglo uočiti malo bolja točnost konvolucijske mreže. Obje mreže su imale poteškoća kod prepoznavanja muških emocija što je vrlo velika vjerojatnost lošijeg izražavanja emocija glumaca iz izabranih baza podataka, te kvantitativni nedostatak podataka za učenje neuronskih mreža.

Kod implementacije programa za snimanje uvrstilo se kašnjenje sustava od dvije sekunde, čije se trajanje može izmijeniti, pošto dvosmjerna LSTM mreža zahtjeva gotovi podatak kako bi ga mogla obraditi od početka prema kraju i obrnuto. Ujedno se gubi i kratki dio razgovora između dva koraka snimanja zbog obrade podataka što može utjecati na loš ishod rezultata.

Kod evaluacije programa može se vidjeti da su neki osjećaji bili vrlo dobro pogođeni dok su neki osjećaji bili potpuno krivo kategorizirani. Greške prilikom evaluacije nad ljudskim subjektima dolaze iz mnogo različitih kutova. Prvi problem je što snimanje nije izvršeno u kontroliranim i profesionalnim uvjetima, što uvelike unosi smetnje u snimku, te je zbog loše opreme kvaliteta zvuka ograničena. Dvoje izabranih govornika u evaluaciji nisu profesionalni glumci pa je upitna kvalitetu prikazanih emocija. Pošto se htjela pokazati usporedba između LSTM neuronske mreže i CNN neuronske mreže snimane emocije su prvo bile snimljene, te potom puštene preko reprodukcijskog uređaja kako bi zapis bio identičan za obje mreže. Prilikom ponovne reprodukcije i njenog snimanja za programsku obradu, gubi se dio akustičkih značajki, te dodatno smanjuje kvalitetu podataka. Također se mora uzeti u obzir da se izvršila normalizacija glasnoće podataka za učenje i validaciju mreža te da ista nije napravljena unutar programa. Problem implementacije normalizacije podataka je u tome što bi se ujedno i utjecaj smetnji povisio u snimci, te je potrebno implementirati neku vrstu dodatnog pretprocesiranja, gdje bi se audio zapis očistio smetnji i normalizirao prikladno podatcima iz skupa za učenje i validaciju. Trenutno bez normalizacije glasnoće prevelik utjecaj u kategorizaciji emocija igra glasnoća. Ako je snimka tiša sigurno će biti kategorizirana tuga ili strah dok kod glasnijih zapisa je bila kategorizirana sreća. Ljutnja je bila kategorizirana jedino kod muškog govornika konvolucijske mreže, gdje je snimka bila

izuzetno glasnija od ostalih. Zadnji problem evaluacije se veže na podatke za učenje i validaciju mreže. Kako se koristio ograničeni broj baza podataka broj različitih govornika je još manji stoga je upitno jesu li govornici u evaluaciji iz spektra govornika uzetih u procesu učenja mreže.

Kako bi se pospješila točnost i pouzdanost kategorizacije emocija dva parametra dolaze do isticanja. Prvi parametar je povećanje količine i raznolikosti baze podataka. Drugi parametar je implementacija normalizacije glasnoće i uklanjanje šuma iz snimanih podataka. Ujedno jednostavan način povećanja točnosti mreže bio bi smanjenje broja kategoriziranih emocija.

LITERATURA

[1] Definicija emocija: https://www.merriam-webster.com/dictionary/emotion, Pristupljeno: 19.11. 2020.

[2] Sreeja P.S., G.S.Mahalakshmi, "Emotional Models: A Review", International Journal of Control Theory and Applications, 2017, ISSN 0974-5572.

[3] A.M.Alfaisal, "Universal Emotions", The International Journal of Humanities & Social Studies, 2019, ISSN 2321 – 9203.

[4] A.Dzedzickis, A.Kaklauskas, V.Bucinskas, "Human Emotion Recognition: Review of Sensors and Methods", 2020, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7037130/ Pristupljeno: 20.11.2020.

[5] Paul Eckman, "Emotions Revealed", 1st. ed. New York: Times Books; 2003.

[6] A.Batliner, B.Schuller, L.Devillers, V.Aharonson, "The Automatic Recognition of Emotions in Speech", Cognitive Technologies, 2011, DOI: 10.1007/978-3-642-15184-2_6.

[7] Definicija umijetne inteligencije: https://builtin.com/artificial-intelligence Pristupljeno: 22.11.2020.

[8] J.G.Razuri, D.Sundgren, R.Rahmani, A.M.Cardenas, "Automatic emotion recognition through facial expression analysis in merged images based on an Artificial Neural Network", Mexican International Conference on Artificial Intelligence, 2013, MICAI.2013.16.

[9] K.R.Anne, S.Kuchibhotla, H.D.Vankayalapati, "Acoustic Modeling for Emotion Recognition", SpringerBriefs in Electrical and Computer Engineering, 2015, ISSN 2191-8112.

[10] "Python" https://www.python.org/doc/essays/blurb/ Pristupljeno: 23.11.2020

[11] "PyCharm" https://www.jetbrains.com/help/pycharm/quick-start-guide.html Pristupljeno: 23.11.2020

[12] Definicija Strojno učenje: https://www.ibm.com/cloud/learn/machine-learning Pristupljeno: 25.11.2020

[13] B.D.Bašić, M.Čupić, J.Šnajder, "Umjetne neuronske mreže", Zagreb: Fakultet elektrotehnike i računarstva, 2008

[14] S.Haykin, "Neural Networks and Learning Machines", 3rd. ed. , McMaster University, Hamilton, Ontario, Canada, 2009.

[15] A.Fred, M.Agarap, "Deep Learning using Rectified Linear Units", Manila, Philippines, Adam University, 2018.

[16] S.Hijazi, R.Kumar, C.Rowen, "Using Convolutional Neural Networks for Image Recognition", 2015.

[17] M. Rizwan, "Convolutional Neural Network – In a Nut Shell", 2018, https://engmrk.com/convolutional-neural-network-3/ Pristupljeno: 01.12.2020.

[18] A.Azzouni, G.Pujolle, "A Long Short-Term Memory Recurrent Neural Network Framework for Network Traffic Matrix Prediction", 2017. ⟨hal-01538471⟩.

[19] P.T.Perez, "Deep Learning: Recurrent Neural Networks", 2018, https://medium.com/deeplearningbrasilia/deep-learning-recurrent-neural-networks-f9482a24d010 Pristupljeno: 05.12.2020

[20] Le, Xuan Hien & Ho, Hung & Lee, Giha & Jung, Sungho. (2019), "Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting", Water. 11. 1387. 10.3390/w11071387.

[21] R.Aggarwal, "Bi-LSTM", 2019, https://medium.com/@raghavaggarwal0089/bi-lstm-bc3d68da8bd0 Pristupljeno: 15.12.2020.

[22] Xue Ying 2019, "An Overview of Overfitting and its Solutions", J. Phys.: Conf. Ser. 1168 022022.

[23] Graf Prenaučenosti mreže: https://medium.com/autonomous-agents/part-2-error-analysis-the-wild-west-algorithms-to-improve-neuralnetwork-accuracy-6121569e66a5 Pristupljeno: 04.01.2021

[24] "Chapter 4. Fully Connected Deep Networks": https://www.oreilly.com/library/view/tensorflow-for-deep/9781491980446/ch04.html Pristupljeno: 07.12.2020.

[25] "Acted Emotional Speech Dynamic Database – AESDD", http://m3c.web.auth.gr/research/aesdd-speech-emotion-recognition/ Pristupljeno: 15.12.2020

[26] "Crowd Sourced Emotional Multimodal Actors Dataset (CREMA-D)", https://www.kaggle.com/ejlok1/cremad Pristupljeno: 07.12.2020.

[27] "Berlin Database of Emotional Speech", https://www.kaggle.com/piyushagni5/berlin-database-of-emotional-speech-emodb Pristupljeno: 07.12.2020.

[28] Livingstone SR, Russo FA (2018) The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. https://doi.org/10.1371/journal.pone.0196391.

[29] "Surrey Audio-Visual Expressed Emotion (SAVEE) Database", http://kahlan.eps.surrey.ac.uk/savee/ Pristupljeno: 07.12.2020

[30] Pichora-Fuller, M. Kathleen; Dupuis, Kate, 2020., "Toronto emotional speech set (TESS)", https://doi.org/10.5683/SP2/E8H2MF, Scholars Portal Dataverse, V1.

[31] "Audacity ": https://www.audacityteam.org/ Pristupljeno: 09.12.2020.

[32] "Librosa": https://librosa.org/doc/latest/index.html Pristupljeno: 10.12.2020.

[33] "Keras": https://keras.io/ Pristupljeno: 12.12.2020.

[34] E. J. Lok, "Audio Emotion, Part 3 - Baseline model", 2019., https://www.kaggle.com/ejlok1/audio-emotion-part-3-baseline-model Pristupljeno: 12.12.2020.

[35] D.Radečić, "Softmax Activation Function Explained", 2020., https://towardsdatascience.com/softmax-activation-function-explained-a7e1bc3ad60 Pristupljeno: 12.12.2020.

[36] "RMSProp", https://deepai.org/machine-learning-glossary-and-terms/rmsprop Pristupljeno: 13.12.2020

[37] "PyAudio", https://people.csail.mit.edu/hubert/pyaudio/ Pristupljeno: 28.12.2020.

[38] J.B.Russ, R.C.Gur, W.B.Bilker, "Validation of affective and neutral sentence content for prosodic testing", 2008., doi: 10.3758/BRM.40.4.935.

PRILOZI

I. CD-R disk

II. Python programski kod

BazaPodataka.py

```
#Učitavanje Pandas biblioteke
import pandas as pd
import os
import librosa
import numpy as np
import pickle
#Lokacije baza podataka
pathAESDD=".\Baze podataka\AESDD/"
pathCREMAD=".\Baze podataka\CREMA-D/"
pathEmoDB=".\Baze podataka\Emo-DB/"
pathRAVDESS=".\Baze podataka\RAVDESS/"
pathSAVEE=".\Baze podataka\SAVEE/"
pathTESS=".\Baze podataka\TESS/"

#CREMA-D Očitanje baze podataka
dir0_list=os.listdir(pathAESDD)
dir0_list.sort()
#CREMA-D Kategorizaja po spolu i emociji
Agender=[]
Aemotion=[]
Apath=[]
female=[1,2,5]
for i in dir0_list:
    if int(i[5:6]) not in female:
        genA='male'
    else:
        genA='female'
    Agender.append(genA)
    if i[:1] == 'a' and genA == 'female':
        Aemotion.append('female_angry')
    elif i[:1] == 's' and genA == 'female':
        Aemotion.append('female_sad')
    elif i[:1] == 'h' and genA == 'female':
        Aemotion.append('female_happy')
    elif i[:1] == 'd' and genA == 'female':
        Aemotion.append('female_disgust')
    elif i[:1] == 'f' and genA == 'female':
        Aemotion.append('female_fear')
    elif i[:1] == 'a' and genA == 'male':
        Aemotion.append('male_angry')
    elif i[:1] == 's' and genA == 'male':
        Aemotion.append('male_sad')
    elif i[:1] == 'h' and genA == 'male':
        Aemotion.append('male_happy')
    elif i[:1] == 'd' and genA == 'male':
        Aemotion.append('male_disgust')
    elif i[:1] == 'f' and genA == 'male':
        Aemotion.append('male_fear')
    else:
        Aemotion.append('unknown')
    Apath.append(pathAESDD + i)
#Postavljanje podataka CREAM-D u tablični zapis
AESDDtable=pd.DataFrame(Aemotion, columns = ['Label'])
AESDDtable['Source']='AESDD'
AESDDtable=pd.concat([AESDDtable,pd.DataFrame(Apath,
columns=['Path'])],axis=1)
AESDDtable.to_csv("Adata_path.csv", index=False)
```

```
#CREMA-D Očitanje baze podataka
dir1_list=os.listdir(pathCREMAD)
dir1_list.sort()
#CREMA-D Kategorizaja po spolu i emociji
Cgender=[]
Cemotion=[]
Cpath=[]
female=[1002,1003,1004,1006,1007,1008,1009,1010,1012,1013,1018,1020,1021,10
24,1025,1028,1029,1030,1037,1043,1046,1047,1049,1052,1053,1054,1055,1056,10
58,1060,1061,1063,1072,1073,1074,1075,1076,1078,1079,1082,1084,1089,109]
for i in dir1_list:
    if int(i[0:4]) not in female:
        gen='male'
    else:
        gen='female'
    Cgender.append(gen)
    if i[9:12] == 'ANG' and gen == 'female':
        Cemotion.append('female_angry')
    elif i[9:12] == 'SAD' and gen == 'female':
        Cemotion.append('female_sad')
    elif i[9:12] == 'HAP' and gen == 'female':
        Cemotion.append('female_happy')
    elif i[9:12] == 'NEU' and gen == 'female':
        Cemotion.append('female_neutral')
    elif i[9:12] == 'DIS' and gen == 'female':
        Cemotion.append('female_disgust')
    elif i[9:12] == 'FEA' and gen == 'female':
        Cemotion.append('female_fear')
    elif i[9:12] == 'ANG' and gen == 'male':
        Cemotion.append('male_angry')
    elif i[9:12] == 'SAD' and gen == 'male':
        Cemotion.append('male_sad')
    elif i[9:12] == 'HAP' and gen == 'male':
        Cemotion.append('male_happy')
    elif i[9:12] == 'NEU' and gen == 'male':
        Cemotion.append('male_neutral')
    elif i[9:12] == 'DIS' and gen == 'male':
        Cemotion.append('male_disgust')
    elif i[9:12] == 'FEA' and gen == 'male':
        Cemotion.append('male_fear')
    else:
        Cemotion.append('unknown')
    Cpath.append(pathCREMAD + i)
#Postavljanje podataka CREAM-D u tablični zapis
CREMA_Dtable=pd.DataFrame(Cemotion, columns = ['Label'])
CREMA_Dtable['Source']='CREMA-D'
CREMA_Dtable=pd.concat([CREMA_Dtable,pd.DataFrame(Cpath,
columns=['Path'])],axis=1)
CREMA_Dtable.to_csv("Cdata_path.csv", index=False)
#Emo-DB Očitavanje baze podataka
dir2_list=os.listdir(pathEmoDB)
dir2_list.sort()
#Emo-DB Kategorizaja po spolu i emociji
Egender=[]
Eemotion=[]
Epath=[]
femaleE=[8, 9, 13, 14, 16]
for i in dir2_list:
if int(i[0:2]) not in femaleE:
```

```
        genE='male'
    else:
        genE='female'
    Egender.append(genE)
    if i[5:6] == 'W' and genE == 'female':
        Eemotion.append('female_angry')
    elif i[5:6] == 'T' and genE == 'female':
        Eemotion.append('female_sad')
    elif i[5:6] == 'F' and genE == 'female':
        Eemotion.append('female_happy')
    elif i[5:6] == 'N' and genE == 'female':
        Eemotion.append('female_neutral')
    elif i[5:6] == 'E' and genE == 'female':
        Eemotion.append('female_disgust')
    elif i[5:6] == 'A' and genE == 'female':
        Eemotion.append('female_fear')
    elif i[5:6] == 'W' and genE == 'male':
        Eemotion.append('male_angry')
    elif i[5:6] == 'T' and genE == 'male':
        Eemotion.append('male_sad')
    elif i[5:6] == 'F' and genE == 'male':
        Eemotion.append('male_happy')
    elif i[5:6] == 'N' and genE == 'male':
        Eemotion.append('male_neutral')
    elif i[5:6] == 'E' and genE == 'male':
        Eemotion.append('male_disgust')
    elif i[5:6] == 'A' and genE == 'male':
        Eemotion.append('male_fear')
    else:
        Eemotion.append('unknown')
    Epath.append(pathEmoDB + i)
#Postavljanje podataka Emo-DB u tablični zapis
Emo_DBtable=pd.DataFrame(Eemotion, columns = ['Label'])
Emo_DBtable['Source']='Emo_DB'
Emo_DBtable=pd.concat([Emo_DBtable,pd.DataFrame(Epath,
columns=['Path'])],axis=1)
Emo_DBtable.to_csv("Edata_path.csv", index=False)

#RAVDESS Očitavanje baze podataka
dir3_list=os.listdir(pathRAVDESS)
dir3_list.sort()
#RAVDESS Kategorizaja po spolu i emociji
Rgender=[]
Remotion=[]
Rpath=[]
femaleR=[2, 4, 6, 8, 10, 12, 14, 16 ,18, 20, 22, 24]
for i in dir3_list:
    if int(i[-6:-4]) not in femaleR:
        genR='male'
    else:
        genR='female'
    Rgender.append(genR)
    if i[6:8] == '05' and genR == 'female':
        Remotion.append('female_angry')
    elif i[6:8] == '04' and genR == 'female':
        Remotion.append('female_sad')
    elif i[6:8] == '03' and genR == 'female':
        Remotion.append('female_happy')
    elif i[6:8] == '01' and genR == 'female':
```

```
        Remotion.append('female_neutral')
    elif i[6:8] == '07' and genR == 'female':
        Remotion.append('female_disgust')
    elif i[6:8] == '06' and genR == 'female':
        Remotion.append('female_fear')
    elif i[6:8] == '05' and genR == 'male':
        Remotion.append('male_angry')
    elif i[6:8] == '04' and genR == 'male':
        Remotion.append('male_sad')
    elif i[6:8] == '03' and genR == 'male':
        Remotion.append('male_happy')
    elif i[6:8] == '01' and genR == 'male':
        Remotion.append('male_neutral')
    elif i[6:8] == '07' and genR == 'male':
        Remotion.append('male_disgust')
    elif i[6:8] == '06' and genR == 'male':
        Remotion.append('male_fear')
    else:
        Remotion.append('unknown')
    Rpath.append(pathRAVDESS + i)
#Postavljanje podataka RAVDESS u tablični zapis
RAVDESStable=pd.DataFrame(Remotion, columns = ['Label'])
RAVDESStable['Source']='RAVDESS'
RAVDESStable=pd.concat([RAVDESStable,pd.DataFrame(Rpath,
columns=['Path'])],axis=1)
RAVDESStable.to_csv("Rdata_path.csv", index=False)

#SAVEE Očitavanje baze podataka
dir4_list=os.listdir(pathSAVEE)
dir4_list.sort()
#SAVEE Kategorizaja po spolu i emociji
Sgender=[]
Semotion=[]
Spath=[]
for i in dir4_list:
    genS='male'
    if i[-8:-6] == '_a' and genS == 'male':
        Semotion.append('male_angry')
    elif i[-8:-6] == 'sa' and genS == 'male':
        Semotion.append('male_sad')
    elif i[-8:-6] == '_h' and genS == 'male':
        Semotion.append('male_happy')
    elif i[-8:-6] == '_n' and genS == 'male':
        Semotion.append('male_neutral')
    elif i[-8:-6] == '_d' and genS == 'male':
        Semotion.append('male_disgust')
    elif i[-8:-6] == '_f' and genS == 'male':
        Semotion.append('male_fear')
    else:
        Semotion.append('unknown')
    Spath.append(pathSAVEE + i)
#Postavljanje podataka SAVEE u tablični zapis
SAVEEtable=pd.DataFrame(Semotion, columns = ['Label'])
SAVEEtable['Source']='SAVEE'
SAVEEtable=pd.concat([SAVEEtable,pd.DataFrame(Spath
                                              ,columns=['Path'])],axis=1)
SAVEEtable.to_csv("Sdata_path.csv", index=False)
```

```
#TESS Očitavanje baze podataka
dir5_list=os.listdir(pathTESS)
dir5_list.sort()
#TESS Kategorizaja po spolu i emociji
Tgender=[]
Temotion=[]
Tpath=[]
for i in dir5_list:
    genT='female'
    if i[-6:-4] == 'ry' and genT == 'female':
        Temotion.append('female_angry')
    elif i[-6:-4] == 'ad' and genT == 'female':
        Temotion.append('female_sad')
    elif i[-6:-4] == 'py' and genT == 'female':
        Temotion.append('female_happy')
    elif i[-6:-4] == 'al' and genT == 'female':
        Temotion.append('female_neutral')
    elif i[-6:-4] == 'st' and genT == 'female':
        Temotion.append('female_disgust')
    elif i[-6:-4] == 'ar' and genT == 'female':
        Temotion.append('female_fear')
    else:
        Temotion.append('unknown')
    Tpath.append(pathTESS + i)
#Postavljanje podataka TESS u tablični zapis
TESStable=pd.DataFrame(Temotion, columns = ['Label'])
TESStable['Source']='TESS'
TESStable=pd.concat([TESStable,pd.DataFrame(Tpath,
columns=['Path'])],axis=1)
TESStable.to_csv("Tdata_path.csv", index=False)

#Spajanje svih baza podataka u tabilčnom zapisu
Baza_Podataka=pd.concat([AESDDtable, CREMA_Dtable, Emo_DBtable,
RAVDESStable, SAVEEtable, TESStable], axis=0)
Baza_Podataka=Baza_Podataka[Baza_Podataka.Label !='unknown']
Baza_Podataka.to_csv("Baza_podataka.csv", index=False)
print(Baza_Podataka.Label.value_counts())

#Izvlačenje značajki
#Postavljanje tabličnog stupca u koji će se spremati značajke
BazaPodataka_Značajke=pd.read_csv("Baza_podataka.csv")
Značajke=pd.DataFrame(columns=['Features'])
#Izvlačenje značajki iz podataka
counter=0
for index, Path in enumerate(BazaPodataka_Značajke.Path):
    #resample baze podataka s brzinom uzrokovanja 44kHz
    resdata, brzina_uzoraka=librosa.load(Path, res_type='kaiser_best',
sr=44000)
    brzina_uzoraka=np.array(brzina_uzoraka)
    #Uklanjanje tišine iz zvučnih zapisa ispod 20dB
    resdata, _ =librosa.effects.trim(y=resdata, top_db= 20)
    rezultat = np.array([])
#Izvlačenje srednje vrijednosti MFCC značajki
MFCC_značajke=np.mean(librosa.feature.mfcc(y=resdata, sr=brzina_uzoraka,
n_mfcc=40), axis=1)
rezultat=np.hstack((rezultat,MFCC_značajke))
#Izvlačenje srednje vrijednosti MEL spektograma
MEL_značajke=np.mean(librosa.feature.melspectrogram(resdata
                                                    ,sr=brzina_uzoraka), axis=1)
```

```
    rezultat=np.hstack((rezultat,MEL_značajke))
    Značajke.loc[counter]=[rezultat]
    counter=counter+1
print(Značajke)
#Spremanje izvučenih značajki
Značajke.to_pickle('Baze_Značajki.pk1')
#Združivanje značajki u tablicu Baze Podataka
Značajke=pd.read_pickle('Baze_Značajki.pk1')
Značajke=pd.concat([BazaPodataka_Značajke,
pd.DataFrame(Značajke['Features'].values.tolist())], axis=1)
Značajke.to_csv("Tablica_Značajki.csv", index=False)
print(Značajke.shape)
```

CNN_Model.py

```
import pandas as pd
import numpy as np
import os
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
import keras
from keras.utils import to_categorical, np_utils
from keras import regularizers
from keras.optimizers import RMSprop
from keras.preprocessing import sequence
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential, Model, model_from_json
from keras.layers import Dense, Input, Flatten, Dropout, Activation
from keras.layers import BatchNormalization, Conv1D, MaxPooling1D,
AveragePooling1D
from keras.callbacks import ModelCheckpoint
import matplotlib.pyplot as plt
import seaborn as sns
#Čitanje Značajki i njihova podjela za treniranje i validaciju
Značajke=pd.read_csv('Tablica_Značajki.csv')
Značajke_train, Značajke_test, Label_train,
Label_test=train_test_split(Značajke.drop(['Path','Source','Label'],
axis=1)
                                                    , Značajke.Label
                                                    , test_size=0.25
                                                    , shuffle=True)
Značajke_train=np.array(Značajke_train)
Značajke_test=np.array(Značajke_test)
Label_train=np.array(Label_train)
Label_test=np.array(Label_test)
#One Hot Encoding
KodiranjeLabel=LabelEncoder()
Label_trainKodiran=np_utils.to_categorical(KodiranjeLabel.fit_transform(Lab
el_train))
Label_testKodiran=np_utils.to_categorical(KodiranjeLabel.fit_transform(Labe
l_test))
print('Značajke_train dimezije', Značajke_train)
print('Značajke_test dimenzije', Značajke_test)
print('classes:', KodiranjeLabel.classes_)
print('Label_trainKategoriziran:', Label_testKodiran)
#Podešenje ulaza za CNN
Značajke_train_prošireno=np.expand_dims(Značajke_train, axis=2)
Značajke_test_prošireno=np.expand_dims(Značajke_test, axis=2)
print('Značajke_train_prošireno:', Značajke_train_prošireno.shape)
print('Značajke_test_prošireno:', Značajke_test_prošireno.shape)
#Model CNN mreže
CNN_Model=Sequential()
CNN_Model.add(Conv1D(256, 8, padding='same'
                     ,input_shape=(Značajke_train_prošireno.shape[1],1)))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(256, 8, padding='same'))
CNN_Model.add(BatchNormalization())
CNN_Model.add(Activation('relu'))
CNN_Model.add(Dropout(0.25))
CNN_Model.add(MaxPooling1D(pool_size=(8)))
```

```
CNN_Model.add(Conv1D(128, 8,padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(128, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(128, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(128, 8, padding='same'))
CNN_Model.add(BatchNormalization())
CNN_Model.add(Activation('relu'))
CNN_Model.add(Dropout(0.25))
CNN_Model.add(MaxPooling1D(pool_size=(8)))
CNN_Model.add(Conv1D(64, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Conv1D(64, 8, padding='same'))
CNN_Model.add(Activation('relu'))
CNN_Model.add(Flatten())
CNN_Model.add(Dense(12))
CNN_Model.add(Activation('softmax'))
optimize=keras.optimizers.RMSprop(lr=0.00001, decay=1e-6)
CNN_Model.summary()
CNN_Model.compile(loss='categorical_crossentropy'
                  ,optimizer=optimize, metrics=['accuracy'])
#Učenje mreže
CNN_Model_History=CNN_Model.fit(Značajke_train_prošireno
                                ,Label_trainKodiran, batch_size=32
,epochs=65, validation_data=(Značajke_test_prošireno, Label_testKodiran))
# Evaluacija modela
scores=CNN_Model.evaluate(Značajke_test_prošireno, Label_testKodiran,
                          verbose=0)
print("%s: %.2f%%" % (CNN_Model.metrics_names[1], scores[1]*100))
#Spremanje CNN modela
CNN_Model_json=CNN_Model.to_json()
with open("CNN_Model.json", "w") as json_file:
    json_file.write(CNN_Model_json)
CNN_Model.save_weights("CNN_Model.h5")
print("Model spremljen na disk")
#Grafovi gubitka i točnosti
#Graf gubitka
fig1=plt.figure(1)
plt.plot(CNN_Model_History.history['loss'])
plt.plot(CNN_Model_History.history['val_loss'], color='red')
plt.title('Model gubitka')
plt.xlabel("Epohe")
plt.ylabel("Gubitak")
plt.legend(['Učenje', 'Validacija'], loc='upper right')
plt.show()
fig1.savefig('Graf_Gubitka.png')
plt.close()
#Graf točnosti
fig2=plt.figure(2)
plt.plot(CNN_Model_History.history['accuracy'])
plt.plot(CNN_Model_History.history['val_accuracy'], color='red')
plt.title('Model točnosti')
plt.xlabel("Epohe")
plt.ylabel("Točnost")
plt.legend(['Učenje', 'Validacija'], loc='upper left')
plt.show()
fig2.savefig('Graf_Točnosti.png')
plt.close()
```

```
#Matrica konfuzije
#Predviđene emocije iz Značajke test prošireno
Predviđanja=CNN_Model.predict(Značajke_test_prošireno)
Predviđanja=Predviđanja.argmax(axis=1)
Predviđanja=Predviđanja.astype(int).flatten()
Predviđanja=(KodiranjeLabel.inverse_transform(Predviđanja))
#Točne emocije
Točna_Labela= Label_testKodiran.argmax(axis=1)
Točna_Labela=Točna_Labela.astype(int).flatten()
Točna_Labela=(KodiranjeLabel.inverse_transform(Točna_Labela))

Matrica_konfuzije= confusion_matrix(Točna_Labela, Predviđanja)
fig3=plt.figure(figsize=(10,8))
heatmap=sns.heatmap(Matrica_konfuzije, annot=True, fmt="d",
cmap='coolwarm')
heatmap.yaxis.set_ticklabels(KodiranjeLabel.classes_, rotation=0,
fontsize=12)
heatmap.xaxis.set_ticklabels(KodiranjeLabel.classes_, rotation=45,
fontsize=12)
plt.ylabel('True Label')
plt.xlabel('Predicted label')
plt.subplots_adjust(left=0.18, bottom=0.18)
fig3.savefig('Matrica_Konfuzije.png')
```

LSTM_Model.py

```
import pandas as pd
import numpy as np
import os
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from tensorflow import keras
from keras.utils import to_categorical, np_utils
import keras
from keras import optimizers
from keras.optimizers import Adam
from keras.models import Sequential, model_from_json
from keras.layers import LSTM, Dense, Bidirectional
from keras.layers import Dropout
import matplotlib.pyplot as plt
import seaborn as sns

#Čitanje Značajki i njihova podjela za treniranje i validaciju
Značajke=pd.read_csv('Tablica_Značajki.csv')
Značajke_train, Značajke_test, Label_train,
Label_test=train_test_split(Značajke.drop(['Path','Source','Label'],
axis=1)
                                                ,Značajke.Label
                                                ,test_size=0.25
                                                ,shuffle=True)
Značajke_train=np.array(Značajke_train)
Značajke_test=np.array(Značajke_test)
Label_train=np.array(Label_train)
Label_test=np.array(Label_test)
#One Hot Encoding
KodiranjeLabel=LabelEncoder()
Label_trainKodiran=np_utils.to_categorical(KodiranjeLabel.fit_transform(Lab
el_train))
Label_testKodiran=np_utils.to_categorical(KodiranjeLabel.fit_transform(Labe
l_test))
print('Značajke_train dimezije', Značajke_train)
print('Značajke_test dimenzije', Značajke_test)
print('classes:', KodiranjeLabel.classes_)
print('Label_trainKategoriziran:', Label_testKodiran)
#Podešenje ulaza za LSTM
Značajke_train_prošireno=np.expand_dims(Značajke_train, axis=2)
Značajke_test_prošireno=np.expand_dims(Značajke_test, axis=2)
print('Značajke_train_prošireno:', Značajke_train_prošireno.shape)
print('Značajke_test_prošireno:', Značajke_test_prošireno.shape)
#Model LSTM mreže
LSTM_Model= Sequential()
LSTM_Model.add(Bidirectional(LSTM(128),
input_shape=(Značajke_train_prošireno.shape[1], 1)))
LSTM_Model.add(Dense(64,activation='relu'))
LSTM_Model.add(Dropout(0.25))
LSTM_Model.add(Dense(12,activation='softmax'))
opti= optimizers.Adam(lr=0.001)
LSTM_Model.summary()
LSTM_Model.compile(loss='categorical_crossentropy',
                   optimizer=opti, metrics=['accuracy'])
```

```
#Učenje mreže
LSTM_Model_History=LSTM_Model.fit(Značajke_train_prošireno
                        ,Label_trainKodiran, batch_size=16, epochs=25
             ,validation_data=(Značajke_test_prošireno, Label_testKodiran))
#Evaluacija modela
scores=LSTM_Model.evaluate(Značajke_test_prošireno, Label_testKodiran,
                           verbose=0)
print("%s: %.2f%%" % (LSTM_Model.metrics_names[1], scores[1]*100))
#Spremanje dvosmjernog LSTM modela
LSTM_Model_json=LSTM_Model.to_json()
with open("LSTM_Model.json", "w") as json_file:
    json_file.write(LSTM_Model_json)
LSTM_Model.save_weights("LSTM_Model.h5")
print("Model spremljen na disk")
#Grafovi gubitka i točnosti
#Graf gubitka
fig1=plt.figure(1)
plt.plot(LSTM_Model_History.history['loss'])
plt.plot(LSTM_Model_History.history['val_loss'], color='red')
plt.title('Model gubitka')
plt.xlabel("Epohe")
plt.ylabel("Gubitak")
plt.legend(['Učenje', 'Validacija'], loc='upper right')
plt.show()
fig1.savefig('Graf_GubitkaLSTM.png')
plt.close()
#Graf točnosti
fig2=plt.figure(2)
plt.plot(LSTM_Model_History.history['accuracy'])
plt.plot(LSTM_Model_History.history['val_accuracy'], color='red')
plt.title('Model točnosti')
plt.xlabel("Epohe")
plt.ylabel("Točnost")
plt.legend(['Učenje', 'Validacija'], loc='upper left')
plt.show()
fig2.savefig('Graf_TočnostiLSTM.png')
plt.close()
#Matrica konfuzije
#Predviđene emocije iz Značajke_test_prošireno
Predviđanja=LSTM_Model.predict(Značajke_test_prošireno)
Predviđanja=Predviđanja.argmax(axis=1)
Predviđanja=Predviđanja.astype(int).flatten()
Predviđanja=(KodiranjeLabel.inverse_transform(Predviđanja))
#Točne emocije
Točna_Labela= Label_testKodiran.argmax(axis=1)
Točna_Labela=Točna_Labela.astype(int).flatten()
Točna_Labela=(KodiranjeLabel.inverse_transform(Točna_Labela))
Matrica_konfuzije= confusion_matrix(Točna_Labela, Predviđanja)
fig3=plt.figure(figsize=(10,8))
heatmap=sns.heatmap(Matrica_konfuzije, annot=True, fmt="d",
cmap='coolwarm')
heatmap.yaxis.set_ticklabels(KodiranjeLabel.classes_, rotation=0,
fontsize=12)
heatmap.xaxis.set_ticklabels(KodiranjeLabel.classes_, rotation=45,
fontsize=12)
plt.ylabel('True Label')
plt.xlabel('Predicted label')
plt.subplots_adjust(left=0.18, bottom=0.18)
fig3.savefig('Matrica_KonfuzijeLSTM.png')
```

CNN_Prepoznavanje.py

```
#bibliote
import pyaudio
import pandas as pd
import numpy as np
import os
import pyaudio
import wave
import sys
import json
from keras.models import model_from_json
import librosa
import matplotlib
import matplotlib.pyplot as plt
import pyloudnorm as pyln
import time

#Učitavanje Modela
#CNN
CNN_json=open('CNN_Model.json', 'r')
loaded_CNN_json=CNN_json.read()
CNN_json.close()
loaded_CNN_Model=model_from_json(loaded_CNN_json)
loaded_CNN_Model.load_weights("CNN_Model.h5")
#Mapa za snimanje audio datoteka
cwd = os.getcwd()
Output_Folder = os.path.join(cwd, "temp")
if not os.path.exists(Output_Folder):
    os.mkdir(Output_Folder)
cat_str =Output_Folder + '/audio'
#Podijela Emocija
classes=['angry','disgust','fear','happy','neutral','sad']
#Podatci izcučenih značajki
Značajke_potpuno=pd.DataFrame(columns=['Emocije'])
#Podaci prisutnosti osjećaja
Izlaz=pd.DataFrame(columns=['Kategorizacija'])
#Uputa za prekid snimanja
print('Za prekid snimanja pritisnite Ctrl-C')
#graf varijable
t=[]
Emocija=[]
#Parametri za snimanje audio datoteka
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
CHUNK = 1024
RECORD_SECONDS = 2
sample_number = 0
def capture_audio(test):
    audio = pyaudio.PyAudio()
    #Početak snimanja isječka
    stream = audio.open(format=FORMAT, channels=CHANNELS,rate=RATE
                        ,input=True, frames_per_buffer=CHUNK)
    frames = []
for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
    data = stream.read(CHUNK)
    frames.append(data)
```

```
    #Kraj snimanja isječka
    stream.stop_stream()
    stream.close()
    audio.terminate()
    waveFile = wave.open(test, 'wb')
    waveFile.setnchannels(CHANNELS)
    waveFile.setsampwidth(audio.get_sample_size(FORMAT))
    waveFile.setframerate(RATE)
    waveFile.writeframes(b''.join(frames))
    waveFile.close()

print('Početak Snimanja')
time.sleep(3)
try:
    while sample_number < 900:#30min
        WAVE_OUTPUT_FILENAME = cat_str + str(sample_number) + ".wav"
        capture_audio(WAVE_OUTPUT_FILENAME)
        # Izvlačenje značajki
        # resample baze podataka s brzinom uzrokovanja 44kHz
        Značajke = pd.DataFrame(columns=['Features'])
        resdata, brzina_uzoraka = librosa.load(WAVE_OUTPUT_FILENAME,
res_type='kaiser_best', sr=44000)
        brzina_uzoraka = np.array(brzina_uzoraka)
        # Uklanjanje tišine iz zvučnih zapisa ispod 20dB
        resdata, _ = librosa.effects.trim(y=resdata, top_db=20)
        rezultat = np.array([])
        # Izvlačenje srednje vrijednosti MFCC značajki
        MFCC_značajke = np.mean(librosa.feature.mfcc(y=resdata,
sr=brzina_uzoraka, n_mfcc=40), axis=1)
        rezultat = np.hstack((rezultat, MFCC_značajke))
        # Izvlačenje srednje vrijednosti MEL spektograma
        MEL_značajke = np.mean(librosa.feature.melspectrogram(resdata,
sr=brzina_uzoraka), axis=1)
        rezultat = np.hstack((rezultat, MEL_značajke))
        Značajke.loc[sample_number] = [rezultat]
        Značajke = pd.DataFrame(Značajke['Features'].values.tolist())
        ZnačajkeMreža = np.array(Značajke)
        ZnačajkeMreža = np.expand_dims(ZnačajkeMreža, axis=2)
        # Klasifikacija CNN
        CNN = loaded_CNN_Model.predict(ZnačajkeMreža, batch_size=16,
verbose=0)
        Predikcija = [CNN[0][0] + CNN[0][6], CNN[0][1] + CNN[0][7]
                      ,CNN[0][2] + CNN[0][8], CNN[0][3] + CNN[0][9]
                      ,CNN[0][4] + CNN[0][10], CNN[0][5] + CNN[0][11]]
Predikcija = np.array(Predikcija)
Značajke_potpuno.loc[sample_number] = [Predikcija]
if sample_number <= 3:
    fusion = Predikcija
else:
    Značajke_potpunoArray = np.array(Značajke_potpuno['Emocije'])
    fusion = 0.5 * Značajke_potpunoArray[sample_number] + \
             0.15 * Značajke_potpunoArray[sample_number - 1] + \
             0.125 * Značajke_potpunoArray[sample_number - 2] + \
             0.125 * Značajke_potpunoArray[sample_number - 3] + \
             0.1 * Značajke_potpunoArray[sample_number - 4]
CNNp = dict(zip(classes, fusion))
Izlaz.loc[sample_number] = [CNNp]
maximum = max(CNNp, key=CNNp.get)
```

```
        print(maximum)
        Emocija.append(maximum)
        os.remove(WAVE_OUTPUT_FILENAME)
        sample_number += 1
        t.append(sample_number)
except KeyboardInterrupt:
    print("Snimanje završeno")
    pass
fig=plt.figure(1)
plt.plot(t,Emocija, linestyle='--', marker='o', color='b')
plt.yticks(classes)
plt.show()
fig.savefig('CNN_kategorizacija.png')
Izlaz.to_csv('Kategorizacija_po_2s.csv',index=False)
```

LSTM_Prepoznavanje.py

```
#bibliote
import pyaudio
import pandas as pd
import numpy as np
import os
import pyaudio
import wave
import sys
import json
from keras.models import model_from_json
import librosa
import matplotlib
import matplotlib.pyplot as plt
import pyloudnorm as pyln
import time

#Učitavanje Modela
#LSTM
LSTM_json=open('LSTM_Model.json', 'r')
loaded_LSTM_json=LSTM_json.read()
LSTM_json.close()
loaded_LSTM_Model=model_from_json(loaded_LSTM_json)
loaded_LSTM_Model.load_weights("LSTM_Model.h5")
#Mapa za snimanje audio datoteka
cwd = os.getcwd()
Output_Folder = os.path.join(cwd, "temp")
if not os.path.exists(Output_Folder):
    os.mkdir(Output_Folder)
cat_str =Output_Folder + '/audio'
#Podjela Emocija
classes=['angry','disgust','fear','happy','neutral','sad']
#Podatci izcučenih značajki
Značajke_potpuno=pd.DataFrame(columns=['Emocije'])
Izlaz=pd.DataFrame(columns=['Kategorizacija'])
#Uputa za prekid snimanja
print('Za prekid snimanja pritisnite Ctrl-C')
#graf varijable
t=[]
Emocija=[]
#Parametri za snimanje audio datoteka
RATE = 44100
CHUNK = 1024
CHANNELS = 1
FORMAT = pyaudio.paInt16
RECORD_SECONDS = 2
sample_number = 0

def capture_audio(test):
    audio = pyaudio.PyAudio()
    #Početak snimanja isječka
    stream = audio.open(format=FORMAT, channels=CHANNELS, rate=RATE
                        ,input=True, frames_per_buffer=CHUNK)
    frames = []
    for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
        data = stream.read(CHUNK)
        frames.append(data)
```

```
    # Kraj snimanja isječka
    stream.stop_stream()
    stream.close()
    audio.terminate()
    waveFile = wave.open(test, 'wb')
    waveFile.setnchannels(CHANNELS)
    waveFile.setsampwidth(audio.get_sample_size(FORMAT))
    waveFile.setframerate(RATE)
    waveFile.writeframes(b''.join(frames))
    waveFile.close()

print('Početak Snimanja')
time.sleep(3)
try:
    while sample_number < 900 :#30min
        WAVE_OUTPUT_FILENAME = cat_str + str(sample_number) + ".wav"
        capture_audio(WAVE_OUTPUT_FILENAME)
        # Izvlačenje značajki
        # resample baze podataka s brzinom uzrokovanja 44kHz
        Značajke = pd.DataFrame(columns=['Features'])
        resdata, brzina_uzoraka = librosa.load(WAVE_OUTPUT_FILENAME,
res_type='kaiser_best', sr=44000)
        brzina_uzoraka = np.array(brzina_uzoraka)
        # Uklanjanje tišine iz zvučnih zapisa ispod 20dB
        resdatat, _ = librosa.effects.trim(y=resdata, top_db=20)
        rezultat = np.array([])
        # Izvlačenje srednje vrijednosti MFCC značajki
        MFCC_značajke = np.mean(librosa.feature.mfcc(y=resdata,
sr=brzina_uzoraka, n_mfcc=40), axis=1)
        rezultat = np.hstack((rezultat, MFCC_značajke))
        # Izvlačenje srednje vrijednosti MEL spektograma
        MEL_značajke = np.mean(librosa.feature.melspectrogram(resdata,
sr=brzina_uzoraka), axis=1)
        rezultat = np.hstack((rezultat, MEL_značajke))
        Značajke.loc[sample_number] = [rezultat]
        Značajke = pd.DataFrame(Značajke['Features'].values.tolist())
        ZnačajkeMreža = np.array(Značajke)
        ZnačajkeMreža = np.expand_dims(ZnačajkeMreža, axis=2)
        # Klasifikacija CNN
        LSTM = loaded_LSTM_Model.predict(ZnačajkeMreža, batch_size=16,
verbose=0)
        Predikcija = [LSTM[0][0] + LSTM[0][6], LSTM[0][1] + LSTM[0][7]
                      ,LSTM[0][2] + LSTM[0][8], LSTM[0][3] + LSTM[0][9]
                      ,LSTM[0][4] + LSTM[0][10], LSTM[0][5] + LSTM[0][11]]
        Predikcija = np.array(Predikcija)
        Značajke_potpuno.loc[sample_number] = [Predikcija]
        if sample_number <= 3:
            fusion = Predikcija
else:
    Značajke_potpunoArray = np.array(Značajke_potpuno['Emocije'])
    fusion = 0.4 * Značajke_potpunoArray[sample_number] + \
             0.15 * Značajke_potpunoArray[sample_number - 1] + \
             0.125 * Značajke_potpunoArray[sample_number - 2] + \
             0.125 * Značajke_potpunoArray[sample_number - 3] + \
             0.1 * Značajke_potpunoArray[sample_number - 4]
CNNp = dict(zip(classes, fusion))
Izlaz.loc[sample_number] = [CNNp]
maximum = max(CNNp, key=CNNp.get)
print(maximum)
```

```
        Emocija.append(maximum)
        os.remove(WAVE_OUTPUT_FILENAME)
        sample_number += 1
        t.append(sample_number)
except KeyboardInterrupt:
    print("Snimanje završeno")
    pass
fig=plt.figure(1)
plt.plot(t,Emocija, linestyle='--', marker='o', color='b')
plt.yticks(classes)
plt.show()
fig.savefig('CNN_kategorizacija.png')
Izlaz.to_csv('Kategorizacija2_po_2s.csv',index=False)
```