**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU**

**FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA**

Diplomski studij

**KLASIFIKACIJA ZVUKA URBANE OKOLINE**

Marko Grubeša

Osijek, 2020.

Sadržaj

[**1.** **Uvod** 1](#_Toc32521480)

[**2.** **Skup podataka** 2](#_Toc32521481)

[**3.** **Preodobradba podataka** 4](#_Toc32521482)

[3.1 MFCC 5](#_Toc32521483)

[3.2 Obrada podataka 7](#_Toc32521484)

[**4.Izrada neuronske mreže** 8](#_Toc32521485)

[4.1 Test i trening podatci 8](#_Toc32521486)

[4.2 Učenje mreže 9](#_Toc32521487)

[**5.** **Klijentska aplikacija** 12](#_Toc32521488)

[**6.** **Zaključak** 14](#_Toc32521489)

[***7.*** **Literatura** 15](#_Toc32521490)

# **Uvod**

Cilj projektnog zadatka je pomoću neuronske mreže klasificirati zvukove urbane okoline. U ovom projektu korišteno je 10 klasa: klima(„air conditioner“), truba auta(„car horn“), igranje djece(„children playing“), lajanje psa(„dog bark“), bušenje(„drilling“), zvuk motora(„engine idling“), pištolj(„gun shot“), udarni čekić(„jackhammer“), sirena(„siren“) i glazba(„street music“). Mreža će primati kao ulaz spektogram, a izlaz će biti vektor vjerojatnosti da određen zvuk pripada određenoj klasi. U ovom projektu korišteno je deset klasa urbane okoline. Skup podataka je potrebno predobraditi kako bi podatci mreži bili razumljivi. Potom se izgrađuje model, te ga se trenira. Programsko rješenje je napisano u programskom jeziku Python, te su još korištene razne biblioteke od kojih su najbitniji: Librosa, Keras, Sounddevice, Numpy. Programsko rješenje je potom implementirano u klijentsku aplikaciju, gdje se zvuk može snimiti pomoću mikrofona ili ručno učitati željenu zvučnu datoteku za testiranje programskog rješenja.

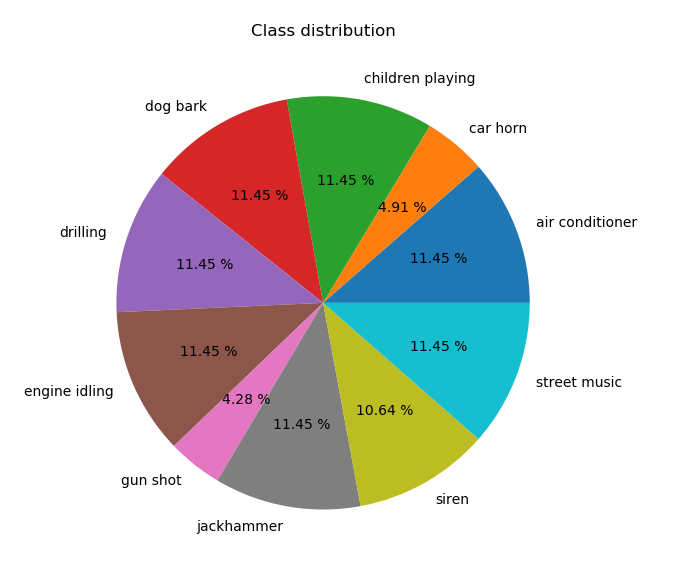
# **Skup podataka**

Korišteni skup podataka sastoji se od 8732 zvučnih datoteka od 10 različitih klasa[1]: klima(„air conditioner“), truba auta(„car horn“), igranje djece(„children playing“), lajanje psa(„dog bark“), bušenje(„drilling“), zvuk motora(„engine idling“), pištolj(„gun shot“), udarni čekić(„jackhammer“), sirena(„siren“) i glazba(„street music“). Datoteke su sortirane u mape, te je također uz skup podataka dostupna CSV datoteka koja sadržava podatke o svakom zvučnom zapisu. Zvučni zapisi su u wav(„waveform audio file“) formatu te ni jedan zapis nije dulji od pet sekundi.



Slika 2.2. Instanca

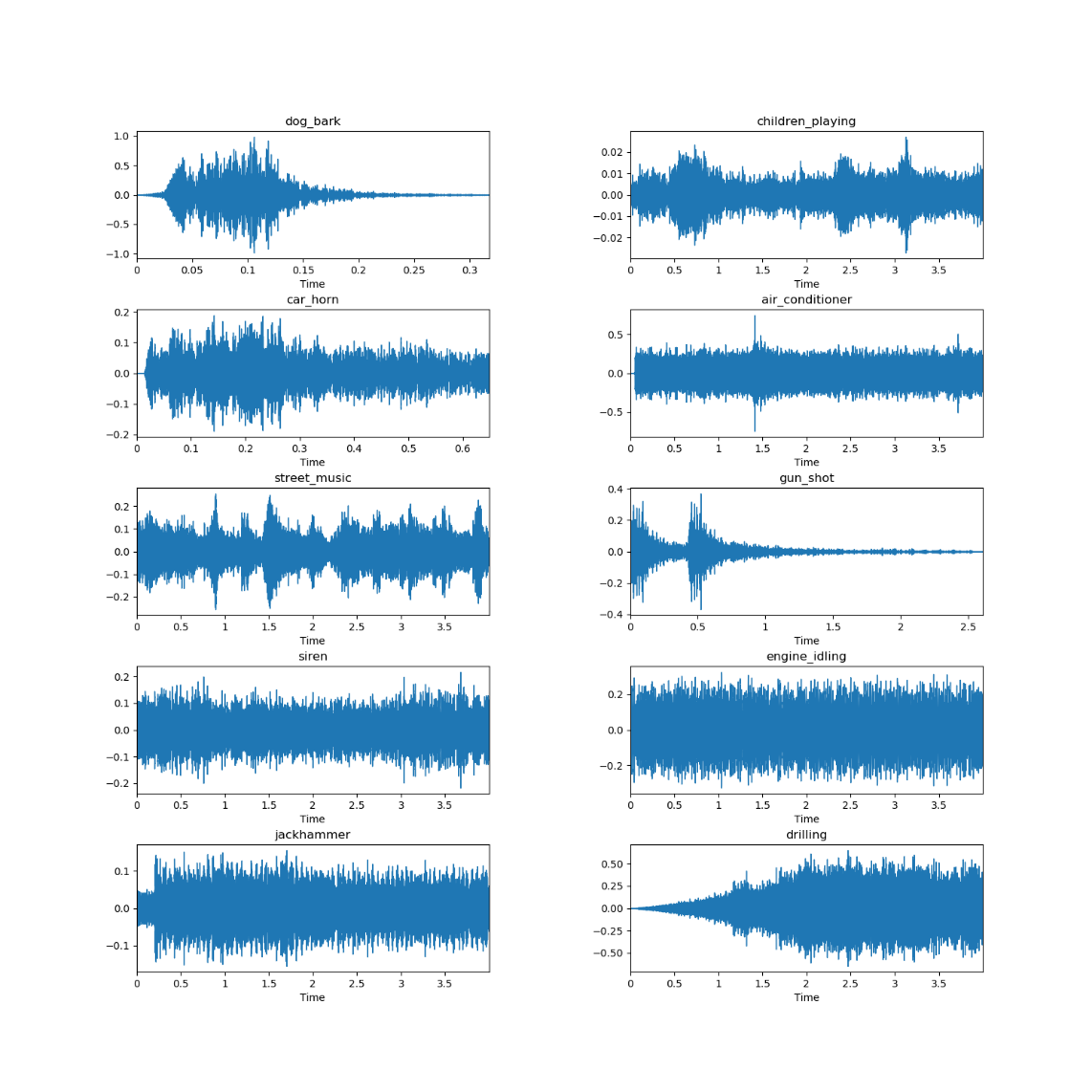
Slika 2.1 prikazuje primjer jedne instance iz skupa podataka. Instanca je definirana nazivom(„slice\_file\_name“), „fsID“(skup podataka podatke povlači s [www.freesound.org](http://www.freesound.org), te „fsID“ predstavlja ID zvučnog zapisa s te stranice), početak zapisa(„start“), kraj zapisa(„end“), vidljivost zvuka(„salience“, 1-prvi plan, 2-pozadinski zvuk), mapa(„fold“), id klase(„classID“, numerički identifikator za naziv klase), klasa(„class“).



Slika 2.2 Kružni graf raspodjele klasa

Slika 2.2 prikazuje prisutnost pojedine klase u skupu podataka. Kao što je vidljivo iz kružnog grafa, klima, bušenje, glazba, ,lajanje psa, igranje djece su najzastupljenije klase, dok su zvukovi trube auta i pištolja manje zastupljeni. Također pri treniranju korištena je ista takva raspodjela podataka kao na slici 2.2.

# **Preodobradba podataka**

Svaki audio signal predstavljen je u vremenskoj domeni, gdje se amplituda nalazi na y-osi.

Slika 3.3 Signali audio zapisa

Na slici 3.1 predstavljeni su audio zapisi u vremenskoj domeni. Zapisi u vremenskoj domeni ne daju previše informacija, te je teško iz te domene izvršiti klasifikaciju(pr. klasa „engine idling“ i jackhammer“ su jako slične).

Jedan od boljih načina za izvlačenje informacija iz zvučnog signala je korištenje kepstralne domene(„MFCC“)[2]

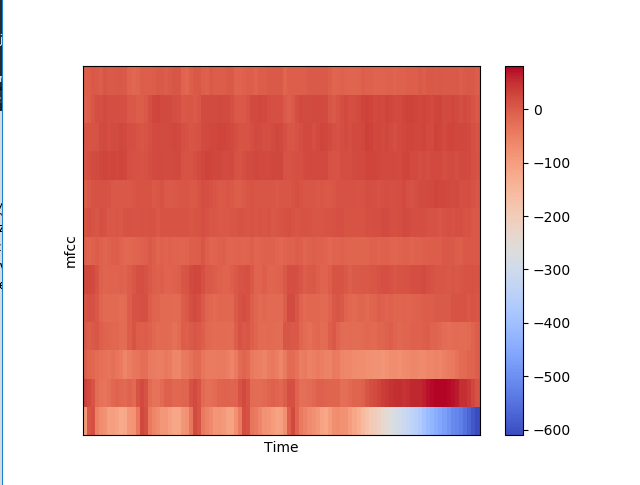
## 3.1 MFCC

Melfrekventni kepstralni koeficijenti (MFCC)[2] široko se koriste u automatskom prepoznavanju govora i zvučnika. Mel-ova ljestvica predstavlja odnos između percipirane frekvencije i njene stvarne izmjerene frekvencije. Ljudi bolje prepoznavaju promjene kod niskih frekvencija, nego kod visokih. S ovom ljestvicom bolje će se približiti značajke ljudskom sluhu. Formula za pretvaranje u Mel-ovu ljestvicu glasi:

|  |  |  |
| --- | --- | --- |
|  |  | (3-1) |

Algoritam za izračun MFCC se sastoji od šest koraka[3]:

1. Segmentacija zvučnog signala u kratke okvire, trajanja oko 25 ms
2. Izračun spektra snage ulaznog signala diskretnom Fourierovom transformacijom
3. Mapiranje frekvencija iz linearne skale u Hz u logaritamsku mel skalu
4. Logaritmiranje spektra snage u točkama odabranim u prethodnom koraku
5. Izračun diskretne kosinusne transformacije logaritmiranog spektra snage
6. MFCC su definirani kao amplitude dobivenog kepstra, pri čemu su najznačajniji prvih 2-13 koeficijenata



Slika 3.2. Izgled MFCC spektra

Na slici 3.2 je prikazan spektogram nakon primjene MFCC algoritma. Spektrogram je zapravo graf kojemu na osi apscisa leži proteklo vrijeme, na osi ordinata frekvencija, a amplituda signala u tom vremenu je dana kao intenzitet svjetline piksela gdje tamni pikseli predstavljaju manje amplitude, a svijetli pikseli predstavljaju veće amplitude.[4]

## 3.2 Obrada podataka

Kao što je navedeno u prošlom poglavlju, prije nego što se krene graditi i trenirati model za strojno učenje, potrebno je proći kroz svaki audio podatak iz skupa podataka i primijeniti MFCC algoritam.

**def** extract\_features**(**file\_name**):**

audio**,** sample\_rate **=** librosa**.**load**(**file\_name**,** res\_type**=**'kaiser\_fast'**)**

mfccs **=** librosa**.**feature**.**mfcc**(**y**=**audio**,** sr**=**sample\_rate**,** n\_mfcc**=**40**)**

mfccs\_processed **=** np**.**mean**(**mfccs**.**T**,**axis**=**0**)**

**return** mfccs\_processed

Slika 3.3 MFCC algoritam

Na slici 3.3 prikazana je implementacija MFCC algoritma. Pomoću biblioteke Librosa učitava se zvučni zapis, te potom iz iste biblioteke koristi se gotova funkcija koja izvršava MFFC algoritam. Funkcija prima audio signal, frekvenciju uzorkovanja i broj mffc-ova. Frekvencija uzorkovanja predstavlja slijed jednako razmaknutih uzoraka, te n\_mffc predstavlja kolika će biti veličina polja nakon obrade algoritma.



Slika 3.4. Prolazak kroz sve datoteke

Na slici 3.4 prikazan je postupak prolaženje kroz sve audio zapise, te njihova obradba. Prolaskom kroz skup podataka primjenjuje se MFFC algoritam te se u polje „features“ sprema obrađeni podatak zajedno s imenom njene klase, na kraju se spremaju ti podaci u novi „Dataframe“ nazivom „featuresdf“.

# **4.Izrada neuronske mreže**

## 4.1 Test i trening podatci

Prije nego što se krene izrađivati neuronska mreža, potrebne je podatke podijeli na test i trening skupove. Najčešće se uzima 70% podataka za učenje modela, te se ostalih 30% koristi za ocjenjivanje modela.



Slika 4.1 Priprema test i trening podataka

Na slici 4.1 prikazana je podjela podataka na test i trening skup. Podaci se najprije pretvore u numpy polja, te se još vrši pretvorba imena klasa u numeričke vrijednosti. Podaci se potom pomoću sklearn biblioteke i korištenjem train\_test\_split() metode dijele u testne i trening podatke. U ovom slučaju 80% podataka je uzeto za treniranje modela, dok se ostalih 20% uzima za testiranje modela.

## 4.2 Učenje mreže

Neuronska mreža nastaje ulančavanjem različitih slojeva mreže. Što je sloj dublji, to on izvlači kompleksnije značajke. Neuronska mreža može se opisati kao umjetna preslika ljudskog mozga kojom se nastoji simulirati postupak učenja. Neuronska mreža je skup međusobno povezanih jednostavnih elemenata, jedinica ili čvorova, čija se funkcionalnost temelji na biološkom neuronu. Moć analize pohranjena je u snazi veza između pojedinih neurona,tj. težinama do kojih se dolazi postupkom prilagodbe (učenjem iz skupa podataka).[5]



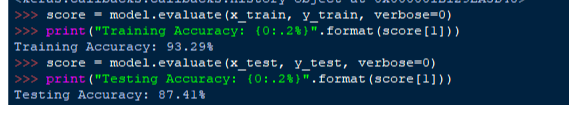
Slika 4. 2 Neuronska mreža

Na slici 4.2 prikazana je neuronska mreža za ovaj zadatak. Stvara se sekvencijalni model, na koji se onda nadodavaju slojevi modela. Limitiran je tako što nema mogućnost stvaranja modela koji dijele slojeve ili imaju više ulaza ili izlaza. Potom se dodaje Dense sloj. Dense sloj je sloj neurona u neuronskoj mreži. Svaki neuron prima ulaz svih neurona iz prethodnog sloja čime je gusto povezan. Potom slijedi aktivacijski sloj. Aktivacijski sloj predstavlja „vrata“ između sloja neurona te idućeg sloja u koji prelazi. Potom se dodaje Dropout sloj[6]. Dropout sloj predstavlja regulacijski sloj. To je metoda koja nasumično odabrane neuron ignorira tijekom treniranja. Efekt je da mreža postaje manje osjetljiva na specifične težine neurona. To zauzvrat rezultira mrežom koja je sposobna bolja generalizirati i manja je vjerojatnost da će doći do „overfitinga“. Te sve slojeve još se jednom implementiraju te na kraju se dodaje sloj neurona koji će imati broj labela u modelu i još jedan aktivacijski sloj.[7]



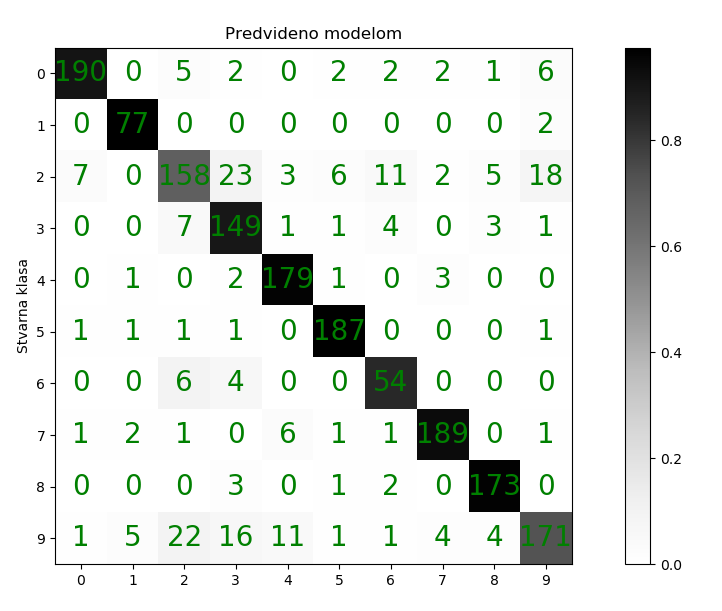
Slika 4.3 Implementiranje trening i test podataka

Na slici 4.3 prikazana je implementacija trening i test podataka. Treniranje će se izvršiti kroz 100 epoha te će se uzimati 32 primjerka po prolasku. Nakon treniranja mreža se testira.



Slika 4.4 Preciznost mreže

Iako nije napravljena pretežito komplicirana mreža, rezultati na testnim podacima pokazuju preciznost od 87.41%. Nakon Treniranja i testiranja, model se može spremiti pomoću biblioteke Pickle, te se model može koristiti za testiranje novih nepoznatih podataka.

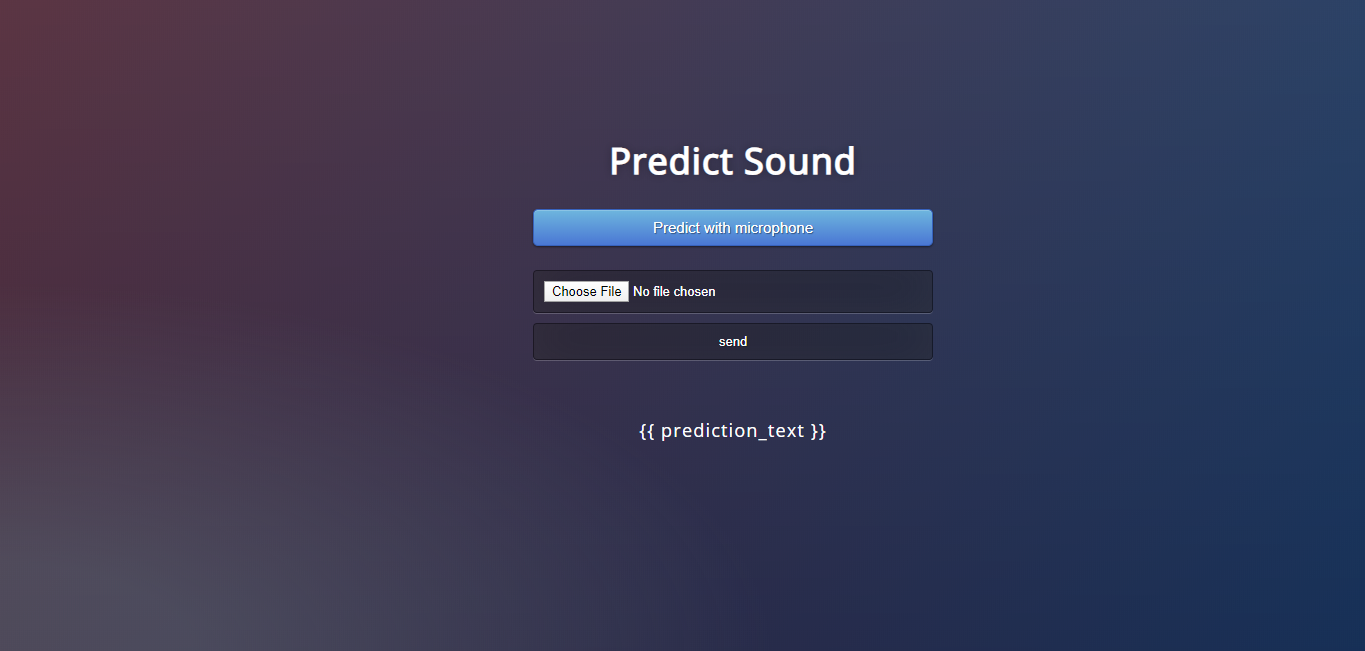


Slika 4.5 Matrica konfuzije

Slika 4.4 prikazuje matricu konfuzije. Dijagonala prikazuje koliko puta su podatci točno klasificirani(True positives), dok podaci koje se ne nalaze na dijagonali prikazuju broj koliko je puta došlo do krive klasifikacije(False positives).

# **Klijentska aplikacija**

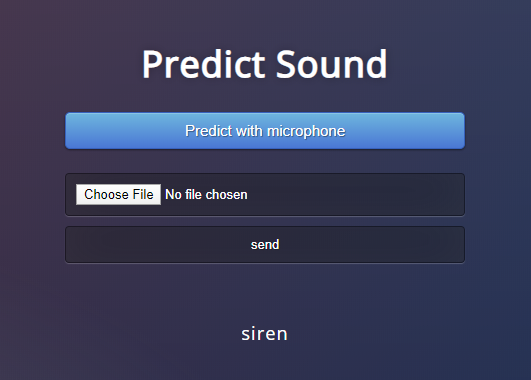
Za klijentsku aplikaciju korišten je Flask. Flask je „framework“ napisan u Pythonu za implementiranje web rješenja.[8] Napravljeno je jednostavno web sučelje gdje se može testirati rad modela. Podaci se mogu ručno implementirati ili se može snimiti pomoću mikrofona. Za snimanje zvuka preko mikrofona korištena je biblioteka sounddevice.



Slika 5.1 Izgled klijentske aplikacije

Na slici 5.1 prikazan je izgled web aplikacije. Na mjestu prediction\_text bude prikazana klasa koju je

model predvidio.



Slika 5.2 Pogađanje klase

Slika 5.2 prikazuje primjer ispisa kategorije koji je model smatrao da je zvuk sirene.

# **Zaključak**

#### U projektu je napravljana neuronska mreža, koja prima spektogram te klasificira ga. Kroz nekoliko slojeva se izvlače bitne značajke koji potom klasificiraju podatak. Testiranjem podataka dobiva se visok postotak preciznosti mreže. Iako je preciznost testnih podataka visoka, pri klasifikaciji može doći do problema. Dosta utječe pozadinski zvuk koji onda utječe na klasifikaciju. Model se može još zakomplicirati, te bi se s time vjerojatno riješio i problem pozadinskog zvuka. Preciznost modela je potom prikazana na klijentskoj aplikaciji, gdje korisnik ima mogućnost snimanja zvuka preko mikrofona ili može učitati već snimljeni zvuk.

# **Literatura**

[1] <https://urbansounddataset.weebly.com/urbansound8k.html>

[2] <https://en.wikipedia.org/wiki/Mel-frequency_cepstrum>

[3] DIPLOMSKI RAD br. 1154METODE RAZLIKOVANJA GOVORA I GLAZBE U DIGITALNIM ZVUČNIM ZAPISIMA Ivan Križanić, Zagreb, Lipanj 2015.

[4] Određivanje glazbenog žanra pjesme pomoću duboke neuronske mreže, Antun Kakuk, Osijek, 2019.

[5] Suzana Dumančić, Neuronske mreže, Diplomski Rad, Odjel za fiziku Osijek,2014

[6 ]<https://machinelearningmastery.com/dropout-regularization-deep-learning-models-keras/>

[7] <https://keras.io/getting-started/sequential-model-guide/>

[8 ]<https://www.palletsprojects.com/p/flask/>