

BAB 1

CHAPTER 1

BAB 2

CHAPTER 2

BAB 3

CHAPTER 3

BAB 4

CHAPTER 4

4.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 4.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

4.1.1 Teori

4.1.2 Praktek

4.1.3 Penanganan Error

4.1.4 Bukti Tidak Plagiat



Gambar 4.2 Kecerdasan Buatan.

=====

BAB 5

CHAPTER 5

5.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 5.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

5.1.1 Teori

5.1.2 Praktek

5.1.3 Penanganan Error

5.1.4 Bukti Tidak Plagiat



Gambar 5.2 Kecerdasan Buatan.

BAB 6

CHAPTER 6

6.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 6.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

6.1.1 Teori

6.1.2 Praktek

6.1.3 Penanganan Error

6.1.4 Bukti Tidak Plagiat



Gambar 6.2 Kecerdasan Buatan.

6.2 1174087 - Ilham Muhammad Ariq

6.2.1 Teori

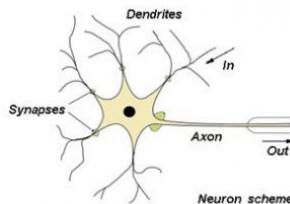
1. Jelaskan kenapa file suara harus di lakukan MFCC. dilengkapi dengan ilustrasi atau gambar.

Nilai-nilai MFCC meniru pendengaran manusia dan mereka biasanya digunakan dalam aplikasi pengenalan suara serta genre musik deteksi. Nilai-nilai MFCC ini akan dimasukkan langsung kejaringan saraf. Agar dapat diubah menjadi bentuk vektor, dan dapat digunakan pada machine learning. Disebabkan machine learning hanya mengerti bilangan vektor saja.

Ilustrasinya, Ketika ingin menggunakan file suara dalam machine learning. Machine learning tidak memahami rekaman suara melainkan vektor. Maka rekaman tersebut akan diubah kedalam bentuk vektor kemudian vektor akan menyesuaikan dengan kata-kata yang sudah disediakan. Jika cocok maka akan mengembalikan waktu yang diinginkan

2. Jelaskan konsep dasar neural network dilengkapi dengan ilustrasi atau gambar

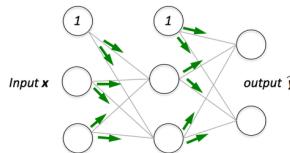
Neural Network ini terinspirasi dari jaringan saraf otak manusia. Dimana setiap neuron terhubung ke setiap neuron di lapisan berikutnya. Lapisan pertama menerima input dan lapisan terakhir memberikan keluaran. Struktur jaringan, yang berarti jumlah neuron dan koneksinya, diputuskan sebelumnya dan tidak dapat berubah, setidaknya tidak selama training. Juga, setiap input harus memiliki jumlah nilai yang sama. Ini berarti bahwa gambar, misalnya, mungkin perlu diubah ukurannya agar sesuai dengan jumlah neuron input.



Gambar 6.3 Contoh Pembobotan Neural Network

3. Jelaskan konsep pembobotan dalam neural network. dilengkapi dengan ilustrasi atau gambar

Bobot mewakili kekuatan koneksi antar unit. Jika bobot dari node 1 ke node 2 memiliki besaran lebih besar, itu berarti bahwa neuron 1 memiliki pengaruh lebih besar terhadap neuron 2. Bobot penting untuk nilai input. Bobot mendekati nol berarti mengubah input ini tidak akan mengubah output. Bobot negatif berarti meningkatkan input ini akan mengurangi output. Bobot menentukan seberapa besar pengaruh input terhadap output. Seperti contoh contoh berikut :



Gambar 6.4 Contoh Pembobotan Neural Network

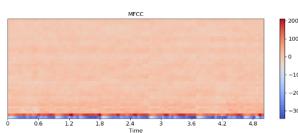
4. Jelaskan konsep fungsi aktifasi dalam neural network. dilengkapi dengan ilustrasi atau gambar

Fungsi aktivasi digunakan untuk memperkenalkan non-linearitas ke jaringan saraf. Ini menekan nilai dalam rentang yang lebih kecil yaitu. fungsi aktivasi Sigmoid memeras nilai antara rentang 0 hingga 1. Ada banyak

fungsi aktivasi yang digunakan dalam industri pembelajaran yang dalam dan ReLU, SELU dan TanH lebih disukai daripada fungsi aktivasi sigmoid. Ilustrasinya, ketika fungsi aktivasi linier, jaringan saraf dua lapis mampu mendekati hampir semua fungsi. Namun, jika fungsi aktivasi identik dengan fungsi aktivasi $F(X) = X$, properti ini tidak puas, dan jika MLP menggunakan fungsi aktivasi yang sama, seluruh jaringan se-tara dengan jaringan saraf lapis tunggal.

- Jelaskan cara membaca hasil plot dari MFCC,dilengkapi dengan ilustrasi atau gambar

Berikut merupakan hasil plot dari rekaman suara :



Gambar 6.5 Cara Membaca Hasil Plot MFCC

Dari gambar tersebut dapat diketahui :

- Terdapat 2 dimensi yaitu x sebagai waktu, dan y sebagai power atau desibel.
- Dapat dilihat bahwa jika berwarna biru maka power dari suara tersebut rendah, dan jika merah power dari suara tersebut tinggi
- Dibagian atas terdapat warna pudar yang menandakan bahwa tidak ada suara sama sekali dalam jangkauan tersebut.

- Jelaskan apa itu one-hot encoding,dilengkapi dengan ilustrasi kode dan atau gambar.

One-hot encoding adalah representasi variabel kategorikal sebagai vektor biner. Mengharuskan nilai kategorikal dipetakan ke nilai integer. Kemudian, setiap nilai integer direpresentasikan sebagai vektor biner yang semuanya bernilai nol kecuali indeks integer, yang ditandai dengan 1.

Label Encoding			→	One Hot Encoding		
Food Name	Categorical #	Calories		Apple	Chicken	Broccoli
Apple	1	95		1	0	0
Chicken	2	231		0	1	0
Broccoli	3	50		0	0	1

Gambar 6.6 One Hot Encoding

- Jelaskan apa fungsi dari np/.unique dan to categorical dalam kode program,dilengkapi dengan ilustrasi atau gambar.

Untuk np.unique fungsinya yaitu menemukan elemen unik array. Mengembalikan elemen unik array yang diurutkan. Ada tiga output opsional selain elemen unik:

- Indeks array input yang memberikan nilai unik
- Indeks array unik yang merekonstruksi array input
- Berapa kali setiap nilai unik muncul dalam array input.

```
>>> np.unique([1, 1, 2, 2, 3, 3])
array([1, 2, 3])
>>> a = np.array([[1, 1], [2, 3]])
>>> np.unique(a)
array([1, 2, 3])
```

Gambar 6.7 Numpy Unique

Untuk To Categorical fungsinya untuk mengubah vektor kelas (integer) ke matriks kelas biner.

```
# Consider an array of 5 labels out of a set of 3 classes (0, 1, 2):
> labels
array([1, 2, 1, 2, 0])
# `to_categorical` converts this into a matrix with as many
# columns as there are classes. The number of rows
# stays the same.
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 1.,  0.,  0.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]])
```

Gambar 6.8 To Categorical

8. Jelaskan apa fungsi dari Sequential dalam kode program,dilengkapi dengan ilustrasi atau gambar.

Sequential berfungsi sebagai tumpukan linear lapisan. COntohnya sebagai berikut :

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(2, input_dim=1))
model.add(Dense(1))
```

Gambar 6.9 Sequential

6.2.2 Praktek Program

1. Soal 1

```
1 # In [1]: Soal Nomor 1
2
3 import librosa
4 import librosa.feature
5 import librosa.display
6 import glob
7 import numpy as np
8 import matplotlib.pyplot as plt
9 from keras.layers import Dense, Activation
10 from keras.models import Sequential
11 from keras.utils.np_utils import to_categorical
12
13 def display_mfcc(song):
14     y, _ = librosa.load(song)
15     mfcc = librosa.feature.mfcc(y)
16
17     plt.figure(figsize=(10, 4))
18     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
19     plt.colorbar()
20     plt.title(song)
21     plt.tight_layout()
22     plt.show()
```

Kode di atas menjelaskan cara mengimport library yang dibutuhkan dan membuat fungsi display mfcc untuk melakukan plot pada file audio nanti. Isi data GTZAN adalah datasets lagu atau suara yang terdiri dari 10 genre yang di simpan kedalam 10 folder yaitu folder blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock ke sepuluh folder tersebut masing-masing berisi 100 data suara sedangkan data freesound merupakan contoh data suara yang akan di gunakan untuk menguji hasil pengolahan data tersebut dengan menggunakan metode mfcc. jika GTZAN memiliki beberapa genre jika freesound hanya untuk 1 lagu dan disini kita membuat fungsi untuk membaca file audio dan outputnya sebagai plot, hasilnya adalah sebagai berikut:

```

In [2]: import librosa.feature
... import librosa.display
... import numpy as np
... import os
... import librosa.util.apg as apg
... from keras.layers import Dense, Activation
... from keras.models import Sequential
... from keras.utils import np_utils, import_to_categorical
... from keras import optimizers

... def display_mfcc(song):
...     y, sr = librosa.load(song)
...     x = librosa.feature.mfcc(y)

...     plt.figure(figsize=(10, 4))
...     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
...     plt.colorbar()
...     plt.title(song)
...     plt.tight_layout()
...     plt.show()

... display_mfcc('song.mp3')

```

Gambar 6.10 Hasil Soal 1.

2. Soal 2

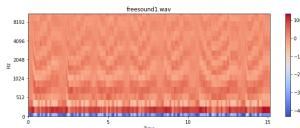
```
1 # In [2]: Soal Nomor 2  
2 display_mfcc('freesound1.wav')  
3 # In [2]: Soal Nomor 2
```

```

4 display_mfcc('freesound2.wav')
5 # In [2]: Soal Nomor 2
6 display_mfcc('genres/blues/blues.00023.au')
7 # In [2]: Soal Nomor 2
8 display_mfcc('genres/classical/classical.00023.au')
9 # In [2]: Soal Nomor 2
10 display_mfcc('genres/country/country.00023.au')
11 # In [2]: Soal Nomor 2
12 display_mfcc('genres/disco/disco.00023.au')
13 # In [2]: Soal Nomor 2
14 display_mfcc('genres/hiphop/hiphop.00023.au')
15 # In [2]: Soal Nomor 2
16 display_mfcc('genres/jazz/jazz.00023.au')
17 # In [2]: Soal Nomor 2
18 display_mfcc('genres/metal/metal.00023.au')
19 # In [2]: Soal Nomor 2
20 display_mfcc('genres/pop/pop.00023.au')
21 # In [2]: Soal Nomor 2
22 display_mfcc('genres/reggae/reggae.00023.au')
23 # In [2]: Soal Nomor 2
24 display_mfcc('genres/rock/rock.00023.au')

```

Kode di atas akan menampilkan hasil dari proses mfcc yang sudah dibuat fungsi pada soal 1, yaitu fungsi display mfcc akan menampilkan plot dari pembacaan file audio. Berikut adalah hasil dari salah satu pembacaan file audio :



Gambar 6.11 Hasil Soal 2.

3. Soal 3

```

1 # In [3]: Soal Nomor 3
2
3 def extract_features_song(f):
4     y, _ = librosa.load(f)
5
6     # get Mel-frequency cepstral coefficients
7     mfcc = librosa.feature.mfcc(y)
8     # normalize values between -1,1 (divide by max)
9     mfcc /= np.amax(np.absolute(mfcc))
10
11    return np.ndarray.flatten(mfcc)[:25000]

```

Kode di atas adalah membuat fungsi yang didefinisikan dengan nama extract_features_song yang nantinya akan di gunakan pada fungsi yang lainnya kemudian dibuat variabel y dengan method librosa load setelah

itu dibuat variabel baru mfcc dengan isi librosa features mfcc dengan isi variabel y tadi kemudian dibuat variabel mfcc dengan isian np.max dan variabel mfcc tadi terakhir di buat array dari data tersebut merupakan data 25000 data pertama. kenapa data 25000 pertama yang digunakan dikarenakan data tersebut digunakan sebagai data testing semakin besar data testing yang di gunakan maka semakin akurat hasil AI. tapi sebenarnya data tersebut relatif bisa lebih besar atau lebih kecil tergantung pada komputer masing masing. Hasilnya adalah sebagai berikut :

```
In [8]: def extract_features_song(f):
    ...
    y, sr = librosa.load(f)
    ...
    # get Mel-frequency cepstral coefficients
    ...
    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=12) # (divide by max)
    ...
    mfcc /= np.max(np.abs(mfcc))
    ...
    return np.ndarray.flatten(mfcc)[:25000]
```

Gambar 6.12 Hasil Soal 3.

4. Soal 4

```
1 # In [4]: Soal Nomor 4
2
3 def generate_features_and_labels():
4     all_features = []
5     all_labels = []
6
7     genres = [ 'blues' , 'classical' , 'country' , 'disco' , '
8         hiphop' , 'jazz' , 'metal' , 'pop' , 'reggae' , 'rock' ]
9     for genre in genres:
10        sound_files = glob.glob('genres/' + genre + '/*.au')
11        print('Processing %d songs in %s genre...' % (len(
12            sound_files), genre))
13        for f in sound_files:
14            features = extract_features_song(f)
15            all_features.append(features)
16            all_labels.append(genre)
17
18    # convert labels to one-hot encoding cth blues :
19    # 1000000000 classic 0100000000
20    label_uniq_ids, label_row_ids = np.unique(all_labels,
21        return_inverse=True) #ke integer
22    label_row_ids = label_row_ids.astype(np.int32, copy=False)
23    onehot_labels = to_categorical(label_row_ids, len(
24        label_uniq_ids)) #ke one hot
25    return np.stack(all_features), onehot_labels
```

Kode di atas adalah mendefinisian nama fungsi yaitu generate features and labels kemudian membuat variabel baru dengan array kosong yaitu all_features dan all_labels kemudian mendefinisikan isian label untuk genre dengan cara membuat variabel genres kemudian di isi dengan 10 genre yang tadi setelah itu dilakukan fungsi if else dengan code for dan in setelah itu akan di buat encoding untuk data tiap tiap label contoh

untuk blues 1000000000 dan untuk clasical 0100000000. Hasilnya adalah sebagai berikut :

```
In [36]: def generate_features_and_labels():
    all_features = []
    all_labels = []
    for genre in genres:
        for file in os.listdir(genre):
            if file.endswith('.mp3'):
                sound_file = glob.glob(genre + '/' + file)[0]
                print('Processing file: ' + file)
                if file in fset:
                    continue
                fset.add(file)
                features, sound_file = extract_features(sound_file, config)
                all_features.append(features)
                all_labels.append(genre)

    # comment out the following line if you don't want to use labels
    # all_labels = np_utils.to_categorical(all_labels, len(genres))

    return np.array(all_features), np.array(all_labels)
```

Gambar 6.13 Hasil Soal 4.

5. Soal 5

```
1 # In [5]: Soal Nomor 5
2 features, labels = generate_features_and_labels()
3 # In [5]: Soal Nomor 5
4 print(np.shape(features))
5 print(np.shape(labels))
```

Hal ini menjadi lama dikarenakan mesin membaca satupersatu file yang ada pada folder dan dalam foldertersebut terdapat 100 file sehingga wajar menjadi lama ditambah lagi mengolah data yang tadinya suara menjadi bentuk vektor. Hasilnya adalah sebagai berikut :

```
In [11]: features, labels = generate_features_and_labels()
Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in rock genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...
In [12]: print(np.shape(features))
...: print(np.shape(labels))
(1000, 25000)
(1000, 10)
```

Gambar 6.14 Hasil Soal 5.

6. Soal 6

```
1 # In [6]: Soal Nomor 6
2 training_split = 0.8
3 # In [6]: Soal Nomor 6
4 alldata = np.column_stack(( features , labels ))
5 # In [6]: Soal Nomor 6
6 np.random.shuffle(alldata)
7 splitidx = int(len(alldata) * training_split)
8 train , test = alldata [:splitidx ,:] , alldata [splitidx :,:]
9 # In [6]: Soal Nomor 6
10 print(np.shape(train))
11 print(np.shape(test))
12 # In [6]: Soal Nomor 6
13 train_input = train [:,:-10]
```

```

14 train_labels = train[:, -10:]
15 # In [6]: Soal Nomor 6
16 test_input = test[:, :-10]
17 test_labels = test[:, -10:]
18 # In [6]: Soal Nomor 6
19 print(np.shape(train_input))
20 print(np.shape(train_labels))

```

Kode diatas berfungsi untuk melakukan training split 80%. Karena supaya mesin dapat terus belajar tentang data baru, jadi ketika prediksi dibuat tentang data yang terlatih itu bisa mendapatkan persentase yang cukup bagus. Hasilnya adalah sebagai berikut :

```
In [13]: training_split = 0.8
```

Gambar 6.15 Hasil Soal 6.

7. Soal 7

```

1 # In [7]: Soal Nomor 7
2 model = Sequential([
3     Dense(100, input_dim=np.shape(train_input)[1]),
4     Activation('relu'),
5     Dense(10),
6     Activation('softmax'),
7 ])

```

Fungsi Sequential() ialah sebuah model untuk menentukan izin pada setiap neuron, di sini adalah 100 dense yang merupakan 100 neuron pertama dari data pelatihan. Fungsi dari relay itu sendiri adalah untuk mengaktifkan neuron atau input yang memiliki nilai maksimum. Sedangkan untuk dense 10 itu adalah output dari hasil neuron yang telah berhasil diaktifkan, untuk dense 10 diaktifkan menggunakan softmax. Hasilnya adalah sebagai berikut :

```

In [20]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])

```

Gambar 6.16 Hasil Soal 7.

8. Soal 8

```

1 # In [8]: Soal Nomor 8
2 model.compile(optimizer='adam',
3                 loss='categorical_crossentropy',
4                 metrics=['accuracy'])
5 print(model.summary())

```

Model Compile di perjelas dengan gambar dibawah, Hasil output pada kode tersebut seperti gambar menjelaskan bahwa dense pertama itu memiliki 100 neurons dengan parameter sekitar 2 juta lebih dengan aktivasi 100, jadi untuk setiap neurons memiliki masing-masing 1 aktivasi. Sama halnya seperti dense 2 memiliki jumlah neurons sebanyak 10 dengan parameter 1010 dan jumlah aktivasinya 10 untuk setiap neurons tersebut dan total parameternya sekitar 2.5 juta data yang akan dilatih pada mesin tersebut.

```
In [21]: model.compile(optimizer='adam',
...                     loss='categorical_crossentropy',
...                     metrics=['accuracy'])
Model: "sequential_1"
Layer (type)                 Output Shape              Param #
dense_1 (Dense)              (None, 100)               2500100
activation_1 (Activation)    (None, 100)               0
dense_2 (Dense)              (None, 10)                1010
activation_2 (Activation)    (None, 10)               0
=====
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0
None
```

Gambar 6.17 Hasil Soal 8.

9. Soal 9

```
1 # In [9]: Soal Nomor 9
2 model.fit(train_input, train_labels, epochs=10, batch_size
3 =32,
4 validation_split=0.2)
```

Kode tersebut berfungsi untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch_size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap 1 kali pengulangan. Sedangkan validation_split itu untuk menentukan presentase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan, hasilnya adalah sebagai berikut :

```
In [21]: model.fit(train_input, train_labels, epochs=10, batch_size=12,
... validation_split=0.2)
Train on 548 samples, validate on 108 samples
Epoch 1/10
646/648 [=====***] - 1s/step - loss: 2.8232 - accuracy: 0.2991 - val_loss:
1.3925 - val_accuracy: 0.4000
1.3925 [=====***] - 1s/step - loss: 1.9893 - accuracy: 0.9051 - val_loss:
1.3925 - val_accuracy: 0.4000
Epoch 2/10
646/648 [=====***] - 1s/step - loss: 1.9893 - accuracy: 0.9051 - val_loss:
1.3925 - val_accuracy: 0.4000
1.3925 [=====***] - 1s/step - loss: 1.1881 - accuracy: 0.6862 - val_loss:
1.4462 - val_accuracy: 0.4437
Epoch 3/10
646/648 [=====***] - 1s/step - loss: 0.9448 - accuracy: 0.7172 - val_loss:
1.3925 - val_accuracy: 0.4000
1.3925 [=====***] - 1s/step - loss: 0.7351 - accuracy: 0.8080 - val_loss:
1.3925 - val_accuracy: 0.4000
Epoch 4/10
646/648 [=====***] - 1s/step - loss: 0.7351 - accuracy: 0.8080 - val_loss:
1.3925 - val_accuracy: 0.4000
1.3925 [=====***] - 1s/step - loss: 0.5571 - accuracy: 0.9120 - val_loss:
1.3925 - val_accuracy: 0.5110
Epoch 5/10
646/648 [=====***] - 1s/step - loss: 0.5581 - accuracy: 0.9022 - val_loss:
1.3925 - val_accuracy: 0.4576
646/648 [=====***] - 1s/step - loss: 0.3614 - accuracy: 0.9422 - val_loss:
1.3925 - val_accuracy: 0.4000
Epoch 6/10
646/648 [=====***] - 1s/step - loss: 0.3122 - accuracy: 0.9510 - val_loss:
1.3925 - val_accuracy: 0.5112
Epoch 7/10
646/648 [=====***] - 1s/step - loss: 0.2368 - accuracy: 0.9659 - val_loss:
1.3925 - val_accuracy: 0.5125
646/648 [=====***] - 1s/step - loss: 0.2368 - accuracy: 0.9659 - val_loss:
0.2368 - val_accuracy: 0.5125
Out[21]: <keras.callbacks.callbacks.History at 0x1d0eccc0>
```

Gambar 6.18 Hasil Soal 9.

10. Soal 10

```

1 # In[10]: Soal Nomor 10
2 loss, acc = model.evaluate(test_input, test_labels,
3                             batch_size=32)
4 # In[10]: Soal Nomor 10
5 print("Done!")
5 print("Loss: %.4f, accuracy: %.4f" % (loss, acc))

```

Fungsi evaluate atau evaluasi ini ialah untuk menguji data pengujian setiap file. Di sini ada prediksi yang hilang, artinya mesin memprediksi data, sedangkan untuk keseluruhan perjanjian sekitar 55%, hasilnya adalah sebagai berikut :

```
In [23]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
200/200 [=====] - 0s 65ms/Step
```

Gambar 6.19 Hasil Soal 10.

11. Soal 11

```

1 # In[11]: Soal Nomor 11
2 model.predict(test_input[:1])

```

Fungsi Predict ialah untuk menghasilkan suatu nilai yang sudah di prediksi dari data training sebelumnya. Gambar dibawah ini menjelaskan file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre rock. Maka lagu tersebut termasuk ke dalam genre rock dengan perbandingan presentase hasil prediksi.

```
In [25]: model.predict(test_input[:1])
Out[25]:
array([[5.2140345e-04, 5.0897381e-06, 5.1087983e-01, 4.7763154e-01,
       1.4319259e-03, 1.4710493e-05, 1.7895336e-06, 9.3976054e-03,
       2.2317986e-05, 9.7474178e-05]], dtype=float32)
```

Gambar 6.20 Hasil Soal 11.

6.2.3 Penanganan Error

1. ScreenShoot Error

```
FileNotFoundError: [Errno 2] No such file or directory: 'E:\VideodanBuat\dataset\dataset\genres\blues\blues.00022.wv'
```

Gambar 6.21 FileNotFoundError

2. Cara Penanganan Error

- **FileNotFoundException**

Error terdapat pada letak file yang tidak terbaca, karena letak file berbeda dengan pemanggilannya, solusi nya ialah dengan meletakkan direktori file yang dibaca dengan benar.

6.2.4 Bukti Tidak Plagiat



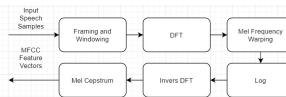
Gambar 6.22 Bukti Tidak Melakukan Plagiat Chapter 6

6.3 1174066 - D.Irga B. Naufal Fakhri

6.3.1 Teori

6.3.1.1 Kenapa file suara harus di lakukan MFCC

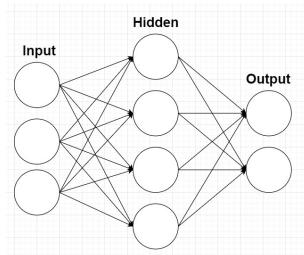
Karena MFCC adalah koefisien yang mewakili audio. Ekstraksi fitur dalam proses ini ditandai dengan konversi data suara menjadi gambar spektrum gelombang. File audio dilakukan oleh MFCC sehingga objek suara dapat dikonversi menjadi matriks. Suara akan menjadi vektor yang akan diproses sebagai output. Selain mempermudah mesin dalam bahasa ini karena mesin tidak dapat membaca teks, maka MFCC perlu mengubah suara menjadi vektor.



Gambar 6.23 Teori 1

6.3.1.2 Konsep dasar neural network

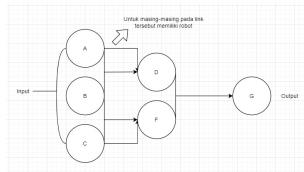
Konsep sederhana dari neural network atau jaringan saraf sederhana dengan proses pembelajaran pada anak-anak dengan memetakan pola-pola baru yang diperoleh dari input untuk membuat pola-pola baru pada output. Contoh sederhana ini menganalogikan kinerja otak manusia. Jaringan saraf itu sendiri terdiri dari unit pemrosesan yang disebut neuron yang berisi fungsi adder dan aktivasi. Fungsi aktivasi itu sendiri untuk publikasi diberikan oleh neuron. Jaringan saraf yang mendukung pemikiran sistem atau aplikasi yang melibatkan otak manusia, baik untuk mendukung berbagai elemen sinyal yang diterima, memeriksa kesalahan, dan juga memproses secara paralel. Karakteristik neural network atau jaringan saraf dilihat dari pola hubungan antar neuron, metode penentuan bobot setiap koneksi, dan fungsi aktivasi mereka.



Gambar 6.24 Teori 2

6.3.1.3 Konsep pembobotan dalam neural network

Pembobotan di dalam neural network juga akan menentukan penanda koneksi. Dalam proses neural network mulai dari input yang diterima oleh neuron bersama dengan nilai bobot masing-masing input. Setelah memasuki neuron, nilai input akan ditambahkan oleh fungsi penerima. Hasil penambahan ini akan diproses oleh masing-masing fungsi neuron, hasil penambahan ini akan dibandingkan dengan nilai ambang tertentu. Jika nilai jumlah ini melebihi nilai ambang batas, aktivasi neuron akan dibatalkan, tetapi sebaliknya jika jumlah hasil di bawah nilai ambang batas, neuron akan diaktifkan. Setelah neuron aktif maka akan mengirimkan nilai output melalui bobot outputnya ke semua neuron yang terkait.



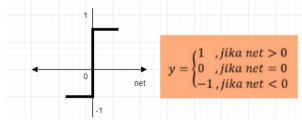
Gambar 6.25 Teori 3

6.3.1.4 Konsep fungsi aktifasi dalam neural network

Fungsi aktivasi dalam neural network ialah merupakan suatu operasi matematik yang dikenakan pada sinyal output. Fungsi ini digunakan untuk mengaktifkan atau menonaktifkan neuron. Fungsi aktivasi ini terbagi setidaknya menjadi 6, adapun sebagai berikut:

1. Fungsi Undak Biner Hard Limit, fungsi ini biasanya digunakan oleh jaringan lapiran tunggal untuk mengkonversi nilai input dari suatu variabel yang bernilai kontinu ke suatu nilai output biner 0 atau 1.
2. Fungsi Undak Biner Threshold, fungsi ini menggunakan nilai ambang sebagai batasnya.
3. Fungsi Bipolar Symetric Hard Limit, fungsi ini memiliki output bernilai 1, 0 atau -1.

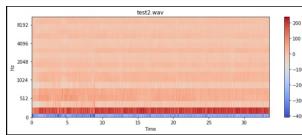
4. Fungsi Bipolar dengan Threshold, fungsi ini mempunyai output yang bernilai 1, 0 atau -1 untuk batas nilai ambang tertentu.
5. Fungsi Linear atau Identitas.



Gambar 6.26 Teori 4

6.3.1.5 Cara membaca hasil plot dari MFCC

Perhatikan gambar dibawah: Gambar menjelaskan bahwa pada waktu ke-5 kekuatan atau layak yang dikeluarkan pada nada ini paling keras pada 20 Hz, selain itu pada 40 -120 Hz daya atau nilai dikeluarkan pada musik yang telah diplot. Demikian juga, warna tersebut adalah kekuatan atau nilai tertinggi dibandingkan dengan warna canggih. Yang merah terdengar di bawah pendengaran manusia, sehingga tidak bisa didengar secara langsung.



Gambar 6.27 Teori 5

6.3.1.6 one-hot encoding

one-hot encoding ini adalah untuk mengubah hasil data vektorisasi menjadi angka biner 0 dan 1 dan membuat informasi tentang atribut yang diberi label.

```
from sklearn import preprocessing
label = np.array([1, 2, 3, 4, 5])
label_row_id = np.arange(len(label))[:, np.newaxis]
label = np.vstack((label_row_id, label))
label = label.astype(np.int32)
onehot_label = np.zeros((len(label), len(np.unique(label))))
onehot_label[np.arange(len(label)), label] = 1
onehot_label
```

Gambar 6.28 Teori 6-1

No	Color
0	Red
1	Green
2	Blue
3	Red
4	Blue

→

No	Red	Green	Blue
0	1	0	0
1	0	1	0
2	0	0	1
3	1	0	0
4	0	0	1

Gambar 6.29 Teori 6-2

6.3.1.7 Apa fungsi dari np.unique dan to_categorical dalam kode program

Fungsi np.unique adalah untuk menemukan berbagai elemen atau array unik, dan dapat mengembalikan elemen unik dari array yang diurutkan. Sebagai

ilustrasi, cukup bisa dilihat pada gambar di bawah ini. Gambar tersebut menjelaskan bahwa unik itu sendiri akan mengambil data yang berbeda dari variabel a dalam fungsi array.

```
>>> import numpy as np
>>> x = np.array([[1,2],[1,1],[2,3],[4,5]])
>>> np.unique(x)
array([1, 2, 3, 4, 5])
```

Gambar 6.30 Teori 7-1

Fungsi dari `to_categorical` ialah untuk mengubah suatu vektor yang berupa integer menjadi matrix dengan kelas biner. Untuk ilustrasinya bisa dilihat pada gambar ??

```
>>> import pandas as pd
>>> var = pd.Categorical(['a', 'a', 'y', 'a', 'd', 'i', 'e', 'g', 'a'])
>>> var
[('a', 0), ('a', 0), ('d', 1), ('r', 0), ('g', 0)]
Categories (7, object): [a, d, g, i, r, s, y].labels
```

Gambar 6.31 Teori 7-2

6.3.1.8 Apa fungsi dari Sequential dalam kode program

Fungsi dari Sequential sebagai salah satu jenis model yang digunakan dalam perhitungan. Sequential ini membangun tumpukan linear yang berurutan.

```
In [35]: model = Sequential([
...     Dense(100, input_dim=np.shape(train_input)[1]),
...     Activation('relu'),
...     Dense(10),
...     Activation('softmax'),
... ])
```

Gambar 6.32 Teori 8

6.3.2 Praktek

6.3.2.1 Nomor 1

```
1 import librosa
2 import librosa.feature
3 import librosa.display
4 import glob
5 import numpy as np
6 import matplotlib.pyplot as plt
7 from keras.layers import Dense, Activation
8 from keras.models import Sequential
9 from keras.utils import np_utils
10
11 def display_mfcc(song):
12     y, _ = librosa.load(song)
13     mfcc = librosa.feature.mfcc(y)
14
15     plt.figure(figsize=(10, 4))
16     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
17     plt.colorbar()
18     plt.title(song)
```

```
19     plt.tight_layout()  
20     plt.show()
```

Kode di atas menjelaskan isi data GTZAN. Ini adalah kumpulan data yang berisi 10 genre lagu dengan masing-masing genre memiliki 100 lagu yang akan kami lakukan proses MFCC dan juga lagu yang saya pilih, jika GTZAN memiliki beberapa genre jika freesound hanya untuk 1 lagu dan disini kita membuat fungsi untuk membaca file audio dan outputnya sebagai plot

```
In [1]: import librosa
import librosa.feature
import numpy as np
import glob
import os
import matplotlib.pyplot as plt
from keras.layers import Dense, Activation
from keras.models import Sequential
from keras.utils import to_categorical
from keras import metrics

def display_fcc(waveform):
    Yfcc = librosa.load(fcc)
    mfcc = librosa.feature.mfcc(Yfcc)

    plt.figure(figsize=(10, 4))
    librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
    plt.title(song)
    plt.colorbar()
    plt.tight_layout()

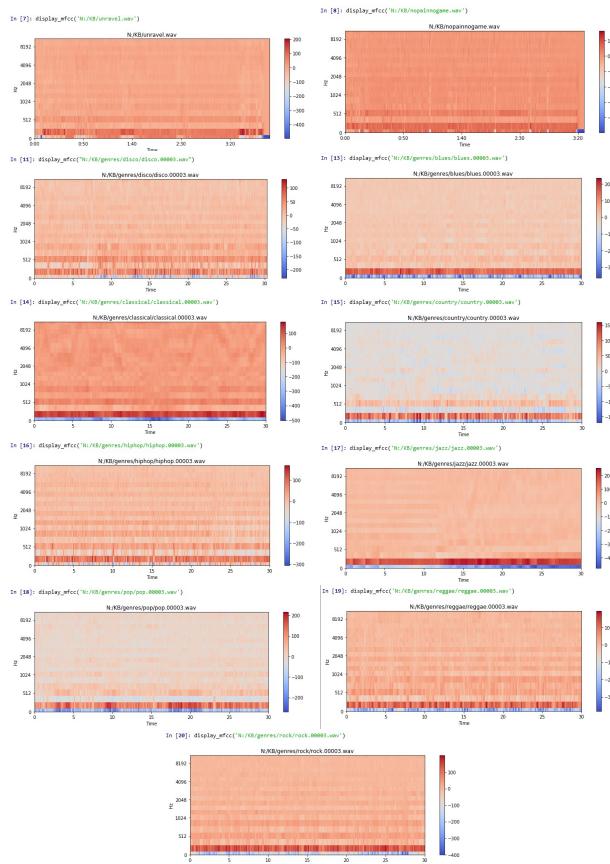
Using TensorFlow backend.
```

Gambar 6.33 Nomor 1

6.3.2.2 Nomor 2

```
1 # In [2]: Soal 2
2 display_mfcc('N:/KB/unravel.wav')
3 # In [2]: Soal 2
4 display_mfcc('N:/KB/nopainnogame.wav')
5 # In [2]: Soal 2
6 display_mfcc("N:/KB/genres/disco/disco.00003.wav")
7 # In [2]: Soal 2
8 display_mfcc('N:/KB/genres/blues/blues.00003.wav')
9 # In [2]: Soal 2
10 display_mfcc('N:/KB/genres/classical/classical.00003.wav')
11 # In [2]: Soal 2
12 display_mfcc('N:/KB/genres/country/country.00003.wav')
13 # In [2]: Soal 2
14 display_mfcc('N:/KB/genres/hiphop/hiphop.00003.wav')
15 # In [2]: Soal 2
16 display_mfcc('N:/KB/genres/jazz/jazz.00003.wav')
17 # In [2]: Soal 2
18 display_mfcc('N:/KB/genres/pop/pop.00003.wav')
19 # In [2]: Soal 2
20 display_mfcc('N:/KB/genres/reggae/reggae.00003.wav')
21 # In [2]: Soal 2
22 display_mfcc('N:/KB/genres/rock/rock.00003.wav')
```

Kode di atas akan menampilkan hasil dari proses mfcc yang sudah dibuat fungsi pada soal 1, yaitu display_mfcc() dan akan menampilkan plot dari pembacaan file audio. Berikut adalah hasil setelah saya lakukan running dan pembacaan file audio:



Gambar 6.34 Nomor 2

6.3.2.3 Nomor 3

```

1 def extract_features_song(f):
2     y, _ = librosa.load(f)
3
4     # get Mel-frequency cepstral coefficients
5     mfcc = librosa.feature.mfcc(y)
6     # normalize values between -1,1 (divide by max)
7     mfcc /= np.absolute(mfcc).max()
8
9     return np.ndarray.flatten(mfcc)[:25000]

```

Baris pertama itu untuk membuat fungsi `extract_features_song(f)`. Pada baris kedua itu akan me-load data inputan dengan menggunakan `librosa`. Lalu selanjutnya untuk membuat sebuah fitur untuk mfcc dari `y` atau parameter inputan. Lalu akan me-return menjadi array dan akan mengambil 25000 data saja dari hasil vektorisasi dalam 1 lagu.

```
In [21]: def extract_features_song(f):
    ...
    y, sr = librosa.load(f)
    ...
    mfcc = librosa.feature.mfcc(y)
    ...
    # normalize features between 0,1 (divide by max)
    ...
    mfcc /= np.max(np.abs(mfcc))
    ...
    return np.ndarray.flatten(mfcc)[:25000]
```

Gambar 6.35 Nomor 3

6.3.2.4 Nomor 4

```
1 def generate_features_and_labels():
2     all_features = []
3     all_labels = []
4
5     genres = ['blues', 'classical', 'country', 'disco', 'hiphop',
6               'jazz', 'metal', 'pop', 'reggae', 'rock']
7     for genre in genres:
8         sound_files = glob.glob('N:/KB/genres/' + genre + '/*.wav')
9         print('Processing %d songs in %s genre...' % (len(
10            sound_files), genre))
11         for f in sound_files:
12             features = extract_features_song(f)
13             all_features.append(features)
14             all_labels.append(genre)
15
16     # convert labels to one-hot encoding cth blues : 1000000000
17     # classic 0100000000
18     label_uniq_ids, label_row_ids = np.unique(all_labels,
19                                              return_inverse=True)#ke integer
20     label_row_ids = label_row_ids.astype(np.int32, copy=False)
21     onehot_labels = to_categorical(label_row_ids, len(
22     label_uniq_ids))#ke one hot
23
24     return np.stack(all_features), onehot_labels
```

Kode di atas dapat digunakan untuk melakukan fungsi yang sebelumnya telah kita lakukan. Kemudian di bagian genre yang disesuaikan dengan dataset nama folder. Untuk baris berikutnya akan mengulang genre folder dengan ekstensi .au. Maka itu akan memanggil fungsi ekstrak lagu. Setiap file dalam folder itu akan diekstraksi menjadi vektor dan akan ditambahkan ke fitur. Dan fungsi yang ditambahkan adalah untuk menumpuk file yang telah di-vektor-kan. Hasil kode tidak menampilkan output.

```
In [26]: def generate_features_and_labels():
    ...
    all_features = []
    all_labels = []
    ...
    genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop', 'reggae', 'rock']
    ...
    for genre in genres:
        sound_files = glob.glob('N:/KB/genres/' + genre + '/*.au')
        for f in sound_files:
            features = extract_features_song(f)
            all_features.append(features)
            all_labels.append(genre)
    ...
    label_uniq_ids, label_row_ids = np.unique(all_labels, return_inverse=True)
    onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids))
    return np.stack(all_features), onehot_labels
```

Gambar 6.36 Nomor 4

6.3.2.5 Nomor 5

```
1 features, labels = generate_features_and_labels()
```

```
2 print(np.shape(features))
3 print(np.shape(labels))
```

Kode diatas berfungsi untuk melakukan load variabel features dan labels. Mengapa memakan waktu yang lama ? Karena mesin akan melakukan vektorisasi terhadap semua file yang berada pada setiap foldernya, di sini terdapat 10 folder dengan masing-masing folder terdiri atas 100 buah lagu, setiap lagu tersebut akan dilakukan vektorisasi atau ekstraksi data menggunakan mfcc.

```
In [27]: features, labels = generate_features_and_labels()
...: print(np.shape(features))
...: print(np.shape(labels))
Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...
(1000, 25000)
(1000, 10)
```

features	float32 [1000, 25000]	[1.0, -0.00000079, -0.00000001, ..., -0.00000004, 0.00000004]
labels	float32 [1000, 10]	[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Gambar 6.37 Nomor 5

6.3.2.6 Nomor 6

```
1 # In [6]: Soal 6
2 training_split = 0.8
3 # In [6]: Soal 6
4 alldata = np.column_stack((features, labels))
5 # In [6]: Soal 6
6 np.random.shuffle(alldata)
7 splitidx = int(len(alldata) * training_split)
8 train, test = alldata[:splitidx, :], alldata[splitidx:, :]
9 # In [6]: Soal 6
10 print(np.shape(train))
11 print(np.shape(test))
12 # In [6]: Soal 6
13 train_input = train[:, :-10]
14 train_labels = train[:, -10:]
15 # In [6]: Soal 6
16 test_input = test[:, :-10]
17 test_labels = test[:, -10:]
18 # In [6]: Soal 6
19 print(np.shape(train_input))
20 print(np.shape(train_labels))
```

Kode diatas berfungsi untuk melakukan training split 80%. Karena supaya mesin dapat terus belajar tentang data baru, jadi ketika prediksi dibuat tentang data yang terlatih itu bisa mendapatkan persentase yang cukup bagus.

The screenshot shows the Jupyter Notebook interface with several code cells and their outputs.

- In [28]:** `training_split = 0.8`
- In [29]:** `alldata = float32 (1000, 2000) [[-0.001366 ... -0.5101305 ... -0.74608847 ... 0. 0. 0.]]`
- In [29]:** `features = float32 (1000, 2000) [[-0.0100975 ... -0.0141395 ... -0.71544417 ... -0.41010354 ... 0.01827242 0. 0. 0.]]`
- In [29]:** `labels = float32 (1000, 10) [[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]`
- In [29]:** `splitidx = int 300`
- In [29]:** `train = float32 (200, 2000) [[-0.001366 ... -0.4121544 ... -0.5101305 ... -0.74608847 ... 0. 0. 0.]]`
- In [29]:** `test = float32 (200, 2000) [[-0.001366 ... -0.5101305 ... -0.74608847 ... 0. 0. 0.]]`
- In [29]:** `training_split = float 1 0.8`
- In [30]:** `alldata = np.column_stack((features, labels))`
- In [31]:** `print(np.shape(train))`
...
`(200, 25010)`
- In [32]:** `train, test = alldata[splitidx:, :], alldata[:splitidx, :]`
- In [33]:** `train_input = float32 (200, 25000) [[-0.001366 ... -0.5101305 ... -0.74608847 ... -0.41010354 ... 0.01827242 0. 0. 0. 1.]]`
- In [33]:** `train_labels = float32 (200, 10) [[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]`
- In [33]:** `test_input = float32 (200, 25000) [[-0.001366 ... -0.5101305 ... -0.74608847 ... -0.41010354 ... 0.01827242 0. 0. 0. 1.]]`
- In [33]:** `test_labels = float32 (200, 10) [[1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]`
- In [34]:** `print(np.shape(train_input))`
...
`(200, 25000)`
- In [34]:** `print(np.shape(train_labels))`
...
`(200, 10)`

Gambar 6.38 Nomor 6

6.3.2.7 Nomor 7

```
1 model = Sequential([
2     Dense(100, input_dim=np.shape(train_input)[1]),
3     Activation('relu'),
4     Dense(10),
5     Activation('softmax'),
6 ])
```

Fungsi Sequential() ialah sebuah model untuk menentukan izin pada setiap neuron, di sini adalah 100 dense yang merupakan 100 neuron pertama dari data pelatihan. Fungsi dari relay itu sendiri adalah untuk mengaktifkan neuron atau input yang memiliki nilai maksimum. Sedangkan untuk dense 10 itu adalah output dari hasil neuron yang telah berhasil diaktifkan, untuk dense 10 diaktifkan menggunakan softmax.

```
In [35]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])
```

Gambar 6.39 Nomor 7

6.3.2.8 Nomor 8

```
1 model.compile(optimizer='adam',
```

```

2         loss='categorical_crossentropy',
3         metrics=['accuracy'])
4 print(model.summary())

```

Model Compile di perjelas dengan gambar dibawah, Hasil output pada kode tersebut seperti gambar menjelaskan bahwa dense pertama itu memiliki 100 neurons dengan parameter sekitar 2 juta lebih dengan aktivasi 100, jadi untuk setiap neurons memiliki masing-masing 1 aktivasi. Sama halnya seperti dense 2 memiliki jumlah neurons sebanyak 10 dengan parameter 1010 dan jumlah aktivasinya 10 untuk setiap neurons tersebut dan total parameternya sekitar 2.5 juta data yang akan dilatih pada mesin tersebut.

```

In [36]: model.compile(optimizer='adam',
...                     loss='categorical_crossentropy',
...                     metrics=['accuracy'])
... print(model.summary())
Model: "sequential_1"
Layer (Type)                 Output Shape            Param #
dense_1 (Dense)              (None, 100)           25000.00
activation_1 (Activation)    (None, 100)           0
dense_2 (Dense)              (None, 10)            1010
activation_2 (Activation)    (None, 10)           0
=====
Total params: 2,501,110
Trainable params: 2,401,110
Non-trainable params: 0
None

```

Gambar 6.40 Nomor 8

6.3.2.9 Nomor 9

```

1 model.fit(train_input, train_labels, epochs=10, batch_size=32,
2 validation_split=0.2)

```

Kode tersebut berfungsi untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch_size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap 1 kali pengulangan. Sedangkan validation_split itu untuk menentukan presentase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan.

```

In [37]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
... validation_split=0.2)
Epoch 1/10
Train on 888 samples, validate on 222 samples
1/1 [====] - ETA: 0.000s - loss: 1.769 - accuracy: 0.400
2/10 [=====] - ETA: 0.000s - loss: 1.009 - accuracy: 0.524 - val_loss: 1.769 - val_accuracy: 0.400
3/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
4/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
5/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
6/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
7/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
8/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
9/10 [=====] - ETA: 0.000s - loss: 1.000 - accuracy: 0.524 - val_loss: 1.430 - val_accuracy: 0.500
10/10 [=====] - ETA: 0.000s - loss: 0.629 - accuracy: 0.875 - val_loss: 1.347 - val_accuracy: 0.682
None

```

Gambar 6.41 Nomor 9

6.3.2.10 Nomor 10

```

1 loss, acc = model.evaluate(test_input, test_labels, batch_size
2 =32)

```

```
2 print("Done!")
3 print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

Fungsi evaluate atau evaluasi ini ialah untuk menguji data pengujian setiap file. Di sini ada prediksi yang hilang, artinya mesin memprediksi data, sedangkan untuk keseluruhan perjanjian sekitar 55%.

```
In [38]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
...:     ...:     ...: print("Done!")
...:     ...:     ...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
...:     ...:     ...: 200/200 [=====] - 0s 473ms/step
...: Done!
...: Loss: 1.3699, accuracy: 0.5159
```

Gambar 6.42 Nomor 10

6.3.2.11 Nomor 11

```
1 model.predict(test_input[:1])
```

Fungsi Predict ialah untuk menghasilkan suatu nilai yang sudah di prediksi dari data training sebelumnya. Gambar dibawah ini menjelaskan file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre rock. Maka lagu tersebut termasuk ke dalam genre rock dengan perbandingan presentase hasil prediksi.

```
In [39]: model.predict(test_input[:1])
Out[39]: array([[3.0364503e-01, 1.6553223e-04, 6.0861975e-02, 3.5557214e-02,
   1.2559776e-01, 1.8407228e-01, 1.3252003e-04, 1.2221429e-03,
   1.3126246e-01, 1.5698114e-01]], dtype=float32)
```

Gambar 6.43 Nomor 11

6.3.3 Penanganan Error

6.3.3.1 Error

- NoBackendError

```
D:\Z\K\Aku\kerjaku\kuis_kuisan\kerjaku\11\main\kerjaku.py:101: Warning: Pydubfile failed. Trying audioread
  audioread.wav('pydubfile.wav')
audioread.wav('pydubfile.wav') failed. Trying audioread instead.
  audioread.wav('pydubfile.wav')
File "C:\Python36\lib\site-packages\audiotools\__init__.py", line 1, in module
  __import__(os.path.join(os.path.dirname(__file__), "audioread"))
  File "C:\Python36\lib\site-packages\audioread\__init__.py", line 1, in __init__
    __import__(os.path.join(os.path.dirname(__file__), "audiofile"))
  File "C:\Python36\lib\site-packages\audioread\audiofile.py", line 1, in __init__
    __import__(os.path.join(os.path.dirname(__file__), "formats"))
  File "C:\Python36\lib\site-packages\audioread\formats\__init__.py", line 1, in __init__
    __import__(os.path.join(os.path.dirname(__file__), "mp3"))
  File "C:\Python36\lib\site-packages\audioread\formats\mp3.py", line 1, in __init__
    __import__(os.path.join(os.path.dirname(__file__), "mp3file"))
  File "C:\Python36\lib\site-packages\audioread\formats\mp3file.py", line 1, in __init__
    __import__(os.path.join(os.path.dirname(__file__), "mp3info"))

NoBackendError
```

Gambar 6.44 NoBackendError

6.3.3.2 Solusi Error

- NoBackendError

Cek kembali file format audionya dipastikan menggunakan wav

6.3.4 Bukti Tidak Plagiat



Gambar 6.45 Bukti tidak plagiat

6.3.5 Link Youtube

https://youtu.be/zA7U-zfvI_w

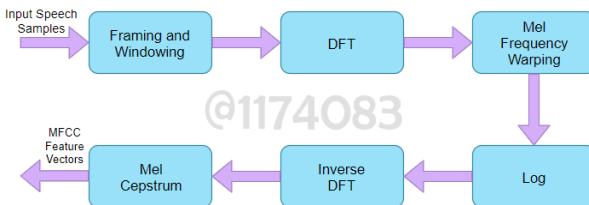
6.4 1174083 - Bakti Qilan Mufid

Chapter 6 - MFCC dan Neural Network

6.4.1 Teori

6.4.1.1 Jelaskan kenapa file suara harus di lakukan MFCC. delengkapi dengan ilustrasi atau gambar

Mel-Frequency Cepstral Coefficient (MFCC) itu ya cuma koefisien(bisa lebih dari satu) yang disebut Koefisien Mel Frequency. Nah, itu tu buat apa sih??. jadi gini, MFCC itu biasa digunakan dalam speech processing. lebih khususnya lagi dalam pengenalan suara atau yang lebih kerennya kita kenal dengan speech recognition. MFCC adalah metode untuk memproses sinyal suara, agar bisa kelihatan deh tu ciri-cirinya. Dilakukan proses MFCC sedemikian rupa, sehingga suatu sinyal suara bisa diekstraksi ciri2 yang membedakannya. Uniknya lagi, MFCC itu seperti pendengaran kita, karena dia memproses suara itu hampir seperti telinga kita. Tau ga sih, telinga kita itu filternya kan beda2. Kalau tinggi filternya gimana, kalau rendah gimana, nah MFCC itu seperti itu, memprosesnya secara logaritmik. Jadi Mel-Frequency Cepstral Coefficients (MFCC) dapat digunakan sebagai vektor ciri yang baik untuk merepresentasikan suara manusia dan sinyal musik. Lebih khusus lagi, MFCC telah terbukti bermanfaat untuk pengenalan suara.



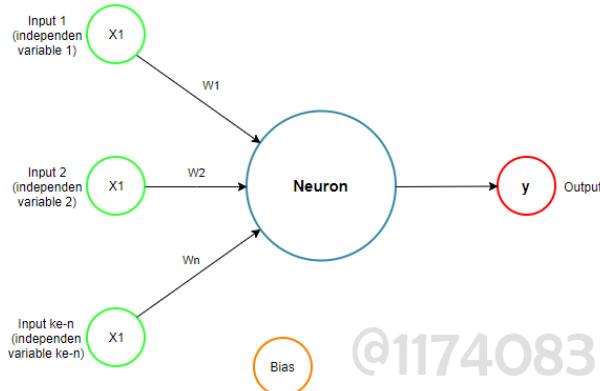
Gambar 6.46 gambaran penjelasan no. 1

6.4.1.2 Jelaskan konsep dasar neural network dilengkapi dengan ilustrasi atau gambar

Melalui ilustrasi di bawah bisa dilihat bahwa secara umum sebuah neural network (NN) terbagi menjadi tiga bagian, yaitu input, neuron (hidden layer) dan output. Dalam konteks deep learning, neuron memiliki istilah lain yaitu perceptron.

Input dari sebuah NN adalah variabel independen yang kita miliki. Output adalah variabel dependen yang kita cari. Misal kita ingin memprediksi harga rumah, maka variabel independennya misalnya luas tanah, luas bangunan, usia bangunan, dan lain-lain. Variabel dependennya adalah harga rumah itu sendiri. Pada contoh ini jenis output (variabel dependen) nya adalah variabel kontinu (continuous variable).

Untuk kasus yang lain, misal kita ingin mendeteksi apakah seorang calon pelanggan masuk ke kriteria target pemasaran atau tidak, maka variabel dependennya adalah berjenis binary (hanya ada 2 pilihan, ya/tidak). Jika terdiri dari lebih dari 2 kategori, maka ia masuk ke jenis kategori (categorical dependent variable). Perlu diperhatikan, jika berjenis kategori, maka outputnya juga lebih dari satu sesuai jumlah kategorinya.

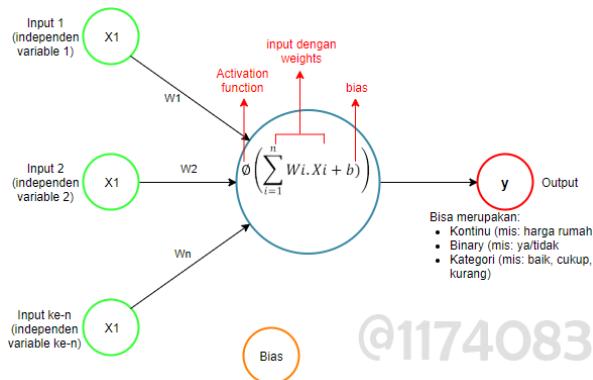


Gambar 6.47 gambaran penjelasan no. 2

6.4.1.3 Jelaskan konsep pembobotan dalam neural network. dilengkapi dengan ilustrasi atau gambar

Setiap input memiliki nilai W yang berbeda. Bobot (weights) sangat krusial bagi neuron, karena ini merupakan bagian agar neuron bisa belajar. Nilai w (simbol untuk weights) bisa sama dan bisa juga tidak untuk setiap input yang diterima dari neuron.

Semakin besar nilai w sebuah input, maka sinyal yang berasal dari input tertentu memiliki prioritas yang semakin besar untuk bisa berkontribusi kepada neuron di depannya. Selain itu, nilai w juga menentukan apakah sinyal dari input tertentu bisa melewati neuron di depannya atau tidak.

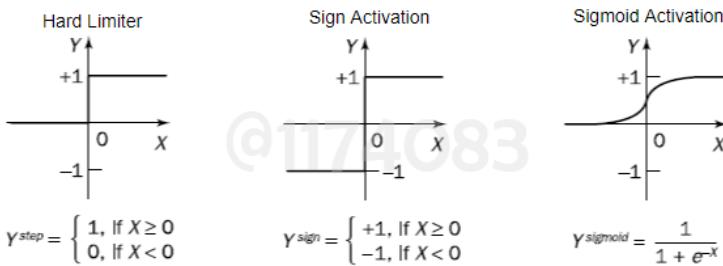


Gambar 6.48 gambaran penjelasan no. 3

6.4.1.4 Jelaskan konsep fungsi aktifasi dalam neural network. dilengkapi dengan ilustrasi atau gambar

Fungsi aktivasi (activation function), merupakan fungsi matematis yang digunakan untuk mendapatkan output neuron dari nilai inputnya. Disebut aktifasi karena output akan bernilai jika melampaui nilai threshold-nya. Beberapa fungsi aktivasi yang sering digunakan, yaitu :

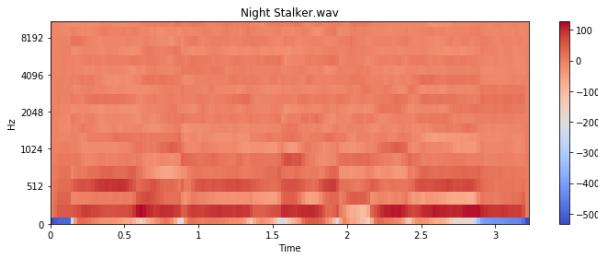
- hard limiter
- signum activation
- sigmoid activation.



Gambar 6.49 gambaran penjelasan no. 4

6.4.1.5 Jelaskan cara membaca hasil plot dari MFCC, dilengkapi dengan ilustrasi atau gambar

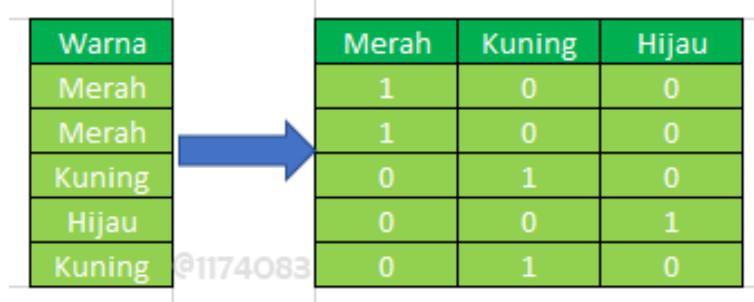
Sumbu y untuk tinggi rendah frekuensi dan sumbu x untuk lama waktu audio. Semakin gelap warnanya, atau semakin dekat ke merah, semakin banyak daya dalam rentang frekuensi pada waktu itu.



Gambar 6.50 gambaran penjelasan no. 5

6.4.1.6 Jelaskan apa itu one-hot encoding, dilengkapi dengan ilustrasi kode dan atau gambar

one-hot encoding merupakan proses untuk mengkoversi variable categorical menjadi bentuk yang dapat diterapkan untuk Machine Learning.



The diagram illustrates the conversion of categorical data into a binary matrix. On the left, a vertical list of colors is shown: Warna, Merah, Merah, Kuning, Hijau, and Kuning. An arrow points from this list to a 6x3 binary matrix on the right. The matrix has columns labeled Merah, Kuning, and Hijau. The rows correspond to the colors in the list. The values in the matrix are binary (0 or 1), indicating the presence or absence of each color category.

Warna	Merah	Kuning	Hijau
Merah	1	0	0
Merah	1	0	0
Kuning	0	1	0
Hijau	0	0	1
Kuning	0	1	0

Gambar 6.51 gambaran penjelasan no. 6

6.4.1.7 Jelaskan apa fungsi dari np.unique dan to_categorical dalam kode program, dilengkapi dengan ilustrasi atau gambar

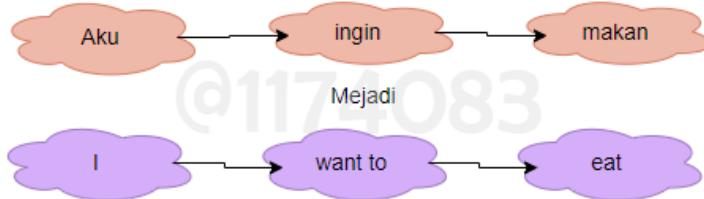
Fungsi np.unique adalah untuk mengembalikan array elemen unik dari inputan array. Contoh sebuah array [5, 2, 6, 2, 7, 5, 6, 8, 2, 9] diterapkan fungsi np.unique hasilnya menjadi [2, 5, 6, 7, 8, 9]. Fungsi to_categorical adalah untuk mengubah nama class menjadi beberapa genre atau mengkategorikan class sesuai jenisnya.

```
>>> import numpy as np
>>> x = np.array([5, 2, 6, 2, 7, 5, 6, 8, 2, 9])
>>> np.unique(x)
array([2, 5, 6, 7, 8, 9])
<<<
>>> import pandas as pd
>>> bakti = pd.Categorical(['n','o','c','o','p','a','s','-','c','o','p','a','s','c','l','u','b'])
>>> print(bakti)
[n, o, c, o, p, ..., s, c, l, u, b]
Length: 17
Categories (10, object): [-, a, b, c, ..., o, p, s, u]
```

Gambar 6.52 gambaran penjelasan no. 7

6.4.1.8 Jelaskan apa fungsi dari Sequential dalam kode program, dilengkapi dengan ilustrasi atau gambar

Fungsi dari Sequential adalah untuk memodelkan atau membuat prediksi jenis data yang sequential atau berurutan, seperti audio, teks, dan lain-lain. Contoh program untuk mengambil sepotong teks dalam bahasa Indonesia dan menerjemahkannya ke bahasa Inggris.



Gambar 6.53 gambaran penjelasan no. 9

6.4.2 Praktek

6.4.2.1 *Jelaskan isi dari data GTZAN Genre Collection dan data dari freesound. Buat kode program untuk meload data tersebut untuk digunakan pada MFCC. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas)*

Data GTZAN Genre Collection berisikan 1000 lagu dari 10 genre berbeda. Di-tiap genrenya masing-masing terdapat 100 lagu yang kurang lebih durasinya 30 detik. sedangkan freesound adalah repository yang emmiliki lisensi berisikan sampel audio, contohnya suara burung, suaran drum, suara alam, dll.

Kode program untuk meload data tersebut untuk digunakan pada MFCC.

```

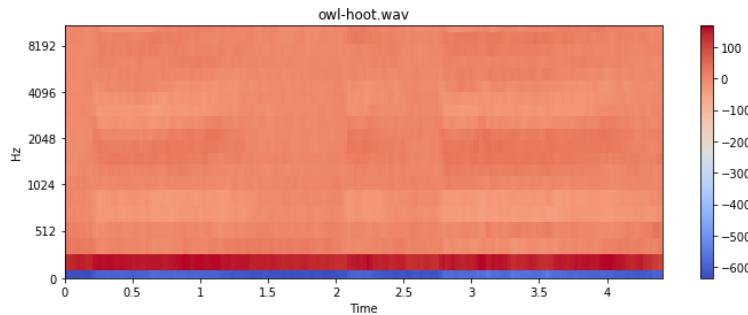
1 # In [ Soal Nomor 1]:
2 import librosa
3 import librosa.feature
4 import librosa.display
5 import glob
6 import numpy as np
7 import matplotlib.pyplot as plt
8 from keras.layers import Dense, Activation
9 from keras.models import Sequential
10 from keras.utils import np_utils
11 from scipy import signal
  
```

Import library librosa untuk mengekstrak fitur dari lagu, import library glob untuk me-list file lagu ke berbagai direktori genre, import library numpy untuk melakukan operasi numerical, import library matplotlib untuk menggambarkan grafik MFCC,import layer dense dari library keras yang memiliki banyak neuron di dalamnya, import layer activation dari library keras untuk menggunakan activation function di tiap layer neuron, import model sequential dari library keras, import to categorical dari library keras untuk mengubah nama class menjadi beberapa genre, dan import signal dari library scipy untuk mengatasi error.

Ini spectogram dari owl-hoot(source dari freesound) yang menampilkan frekuensi rendah.

```

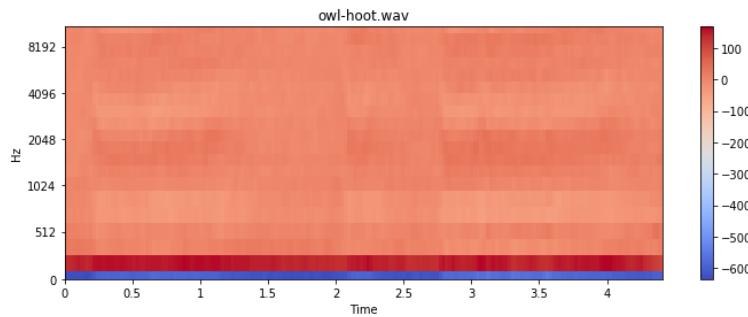
1 display_mfcc('owl-hoot.wav')
  
```



Gambar 6.54 spectogram owl-hoot

Ini spectrogram dari the-canadian-loon(source dari freesound) yang menampilkan frekuensi rendah.

```
1 display_mfcc('the-canadian-loon.wav')
```



Gambar 6.55 spectogram the-canadian-loon

Untuk contoh spectrogram dari GTZAN Genre Collection, bisa dilihat di nomor selanjutnya.

6.4.2.2 Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari `display_mfcc()`

```
1 def display_mfcc(song):
2     y, _ = librosa.load(song)
3     mfcc = librosa.feature.mfcc(y)
4
5     plt.figure(figsize=(10, 4))
6     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
7     plt.colorbar()
8     plt.title(song)
9     plt.tight_layout()
10    plt.show()
```

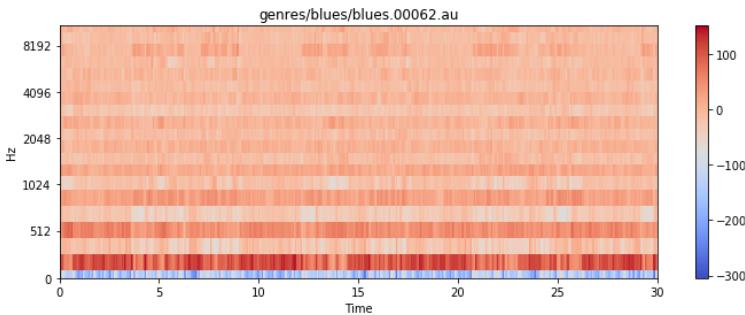
Membuat fungsi untuk menampilkan nilai dari MFCC dari tiap lagu. Pertama load lagunya, lalu ekstrak lagunya dengan fungsi mfcc dan tampung nilainya. Kemudian tampilkan spectogramnya dengan fungsi specshow.

- Nama fungsinya display_mfcc() dengan parameter song untuk lagu yang nantinya diproses.
- Fungsi load() untuk meload lagunya kemudian ditampung nilainya.
- Fungsi mfcc() untuk mengekstrak fitur dari lagunya kemudian ditampung nilainya.
- Fungsi figure() untuk membuat grafiknya.
- Fungsi specshow() untuk membuat spectogramnya.
- Fungsi colorbar() untuk memberi warna pada grafiknya.
- Fungsi title() untuk memberi judul pada grafiknya.
- Fungsi tight layout() untuk menyesuaikan grafiknya sesuai layout.
- Fungsi show() untuk menampilkan grafiknya.

berikut adalah beberapa contoh penggunaan display_mfcc(), beserta contoh data dari GTZAN Genre Collection

Ini spectogram dari lagu genre blues (source dari GTZAN Genre Collection).

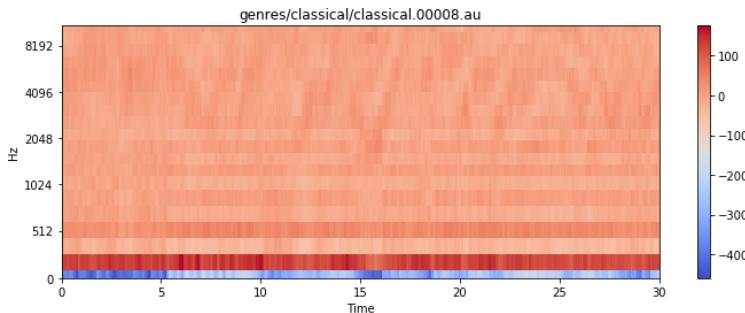
```
1 display_mfcc('genres/blues/blues.00062.au')
```



Gambar 6.56 spectogram dari lagu genre blues

Ini spectogram dari lagu genre classical (source dari GTZAN Genre Collection).

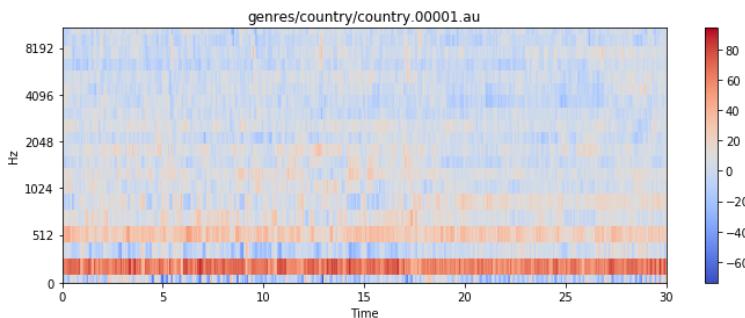
```
1 display_mfcc('genres/classical/classical.00008.au')
```



Gambar 6.57 spectogram dari lagu genre classical

Ini spectogram dari lagu genre country (source dari GTZAN Genre Collection).

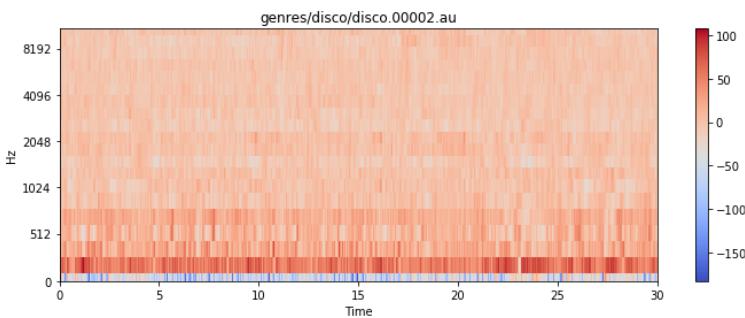
```
1 display_mfcc('genres/country/country.00001.au')
```



Gambar 6.58 spectogram dari lagu genre country

Ini spectogram dari lagu genre disco (source dari GTZAN Genre Collection).

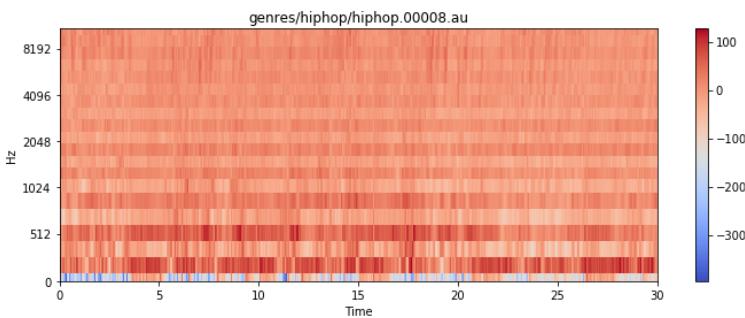
```
1 display_mfcc('genres/disco/disco.00002.au')
```



Gambar 6.59 spectogram dari lagu genre disco

Ini spectogram dari lagu genre hiphop (source dari GTZAN Genre Collection).

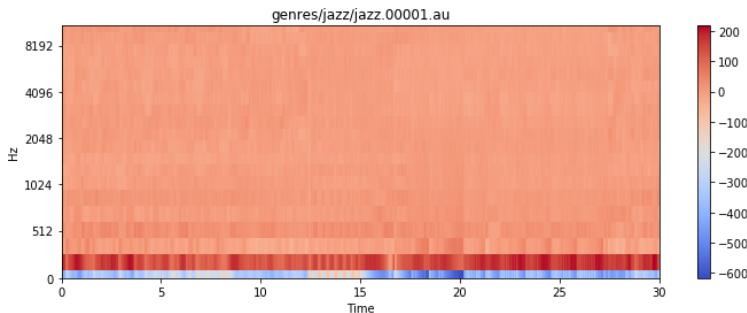
```
1 display_mfcc('genres/hiphop/hiphop.00008.au')
```



Gambar 6.60 spectogram dari lagu genre hiphop

Ini spectogram dari lagu genre jazz (source dari GTZAN Genre Collection).

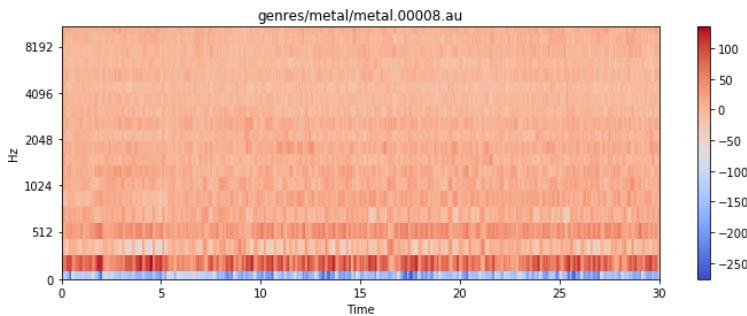
```
1 display_mfcc('genres/jazz/jazz.00001.au')
```



Gambar 6.61 spectogram dari lagu genre jazz

Ini spectrogram dari lagu genre metal (source dari GTZAN Genre Collection).

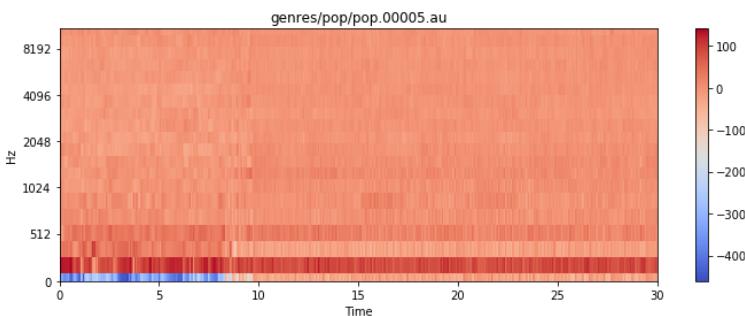
```
1 display_mfcc('genres/metal/metal.00008.au')
```



Gambar 6.62 spectogram dari lagu genre metal

Ini spectrogram dari lagu genre pop (source dari GTZAN Genre Collection).

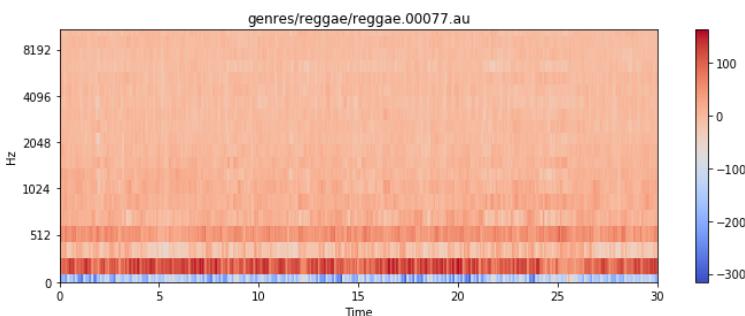
```
1 display_mfcc('genres/pop/pop.00005.au')
```



Gambar 6.63 spectogram dari lagu genre pop

Ini spectrogram dari lagu genre reggae (source dari GTZAN Genre Collection).

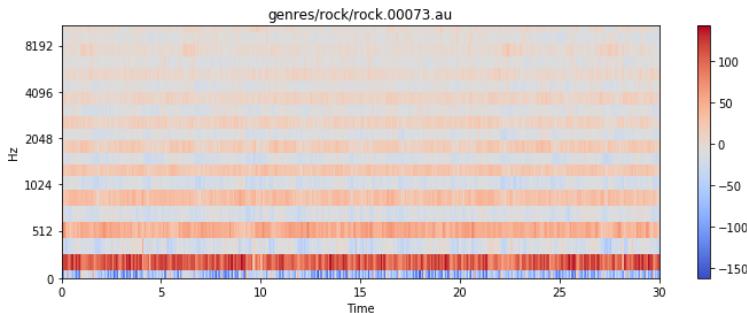
```
1 display_mfcc('genres/reggae/reggae.00077.au')
```



Gambar 6.64 spectogram dari lagu genre reggae

Ini spectrogram dari lagu genre rock (source dari GTZAN Genre Collection).

```
1 display_mfcc('genres/rock/rock.00073.au')
```



Gambar 6.65 spectrogram dari lagu genre rock

6.4.2.3 *Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari extract_features_song(). Jelaskan juga mengapa yang diambil 25.000 baris pertama?*

Kode :

```

1 # In [Soal Nomor 3]:
2 def extract_features_song(f):
3     y, _ = librosa.load(f)
4     # get Mel-frequency cepstral coefficients
5     mfcc = librosa.feature.mfcc(y)
6     # normalize values between -1,1 (divide by max)
7     mfcc /= np.amax(np.absolute(mfcc))
8
9     return np.ndarray.flatten(mfcc)[:25000]

```

- nama fungsinya adalah extract_features_song() dengan paramete f untuk lagu yang nantinya diproses.
- fungsi load() untuk meload lagunya kemudian ditampung nilainya.
- fungsi mfcc() untuk mengekstrak fitur dari lagunya kemudian ditampung nilainya.
- fungsi absolute() untuk menjadikannya nilai absolut. Panggil fungsi amax untuk mencari nilai max. Kemudian hasil tadi dibagi dengan nilai mfcc sebelumnya.
- lalu mengembalikan hasil convert array ke dalam bentuk satu dimensi dari nilai mfcc dan diambil 25000 baris pertama.

Alasan diambil 25000 baris pertama, karena setiap lagu memiliki panjang nilai MFCC yang berbeda-beda dan yang dimasukkan ke dalam neuron harus memiliki ukuran yang sama.

```
In [27]: def extract_features_song(f):
...:     y, _ = librosa.load(f)
...:
...:     # get Mel-frequency cepstral coefficients
...:     mfcc = librosa.feature.mfcc(y)
...:     # normalize values between -1,1 (divide by max)
...:     mfcc /= np.amax(np.absolute(mfcc))
...:
...:     return np.ndarray.flatten(mfcc)[:25000]
```

Gambar 6.66 Hasil dari kode program praktik nomor 3

6.4.2.4 Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari `generate_features_and_labels()`

Kode :

```
1 # In [Soal Nomor 4]:  
2 def generate_features_and_labels():  
3     all_features = []  
4     all_labels = []  
5  
6     genres = ['blues', 'classical', 'country', 'disco', 'hiphop',  
7                'jazz', 'metal', 'pop', 'reggae', 'rock']  
8     for genre in genres:  
9         sound_files = glob.glob('genres/' + genre + '/*.au')  
10        print('Processing %d songs in %s genre ...' % (len(  
11            sound_files), genre))  
12        for f in sound_files:  
13            features = extract_features_song(f)  
14            all_features.append(features)  
15            all_labels.append(genre)  
16  
17    # convert labels to one-hot encoding cth blues : 100000u800000  
18    # classic 0100000000  
19    label_uniq_ids, label_row_ids = np.unique(all_labels,  
20    return_inverse=True)  
21    #ke integer  
22    label_row_ids = label_row_ids.astype(np.int32, copy=False)  
23    onehot_labels = to_categorical(label_row_ids, len(  
24    label_uniq_ids))  
25    #ke one hot  
26    return np.stack(all_features), onehot_labels
```

Kode di atas dapat digunakan untuk melakukan fungsi yang sebelumnya telah kita lakukan. Kemudian di bagian genre yang disesuaikan dengan dataset nama folder. Untuk baris berikutnya akan mengulang genre folder dengan ekstensi .au. Maka itu akan memanggil fungsi ekstrak lagu. Setiap file dalam folder itu akan diekstraksi menjadi vektor dan akan ditambahkan ke fitur. Dan fungsi yang ditambahkan adalah untuk menumpuk file yang telah di-vektor-kan.

```
In [30]: def generate_features_and_labels():
    ...
    all_features = []
    all_labels = []
    ...
    genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop',
    'reggae', 'rock']
    ...
    for genre in genres:
        sound_files = glob.glob('genres/' + genre + '*.au')
        print('Processing %d songs in %s genre...' % (len(sound_files), genre))
        for f in sound_files:
            features = extract_features_song(f)
            all_features.append(features)
            all_labels.append(genre)
    ...
    # convert labels to one-hot encoding cth blues : 1000000000 classic 0100000000
    label_uniq_ids, label_row_ids = np.unique(all_labels, return_inverse=True) #ke integer
    label_row_ids = label_row_ids.astype(np.int32, copy=False)
    ...
    onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids))#ke one hot
    ...
    return np.stack(all_features), onehot_labels
```

Gambar 6.67 Hasil dari kode program praktek nomor 4

6.4.2.5 Jelaskan dengan kata dan praktek kenapa penggunaan fungsi `generate_features_and_labels()` sangat lama ketika meload dataset genre. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

Kode :

```
1 # In [Soal Nomor 5]:
2 features, labels = generate_features_and_labels()
3 # In [Soal Nomor 5]:
4 print(np.shape(features))
5 print(np.shape(labels))
```

Kode diatas berfungsi untuk melakukan load variabel features dan labels. Mengapa memakan waktu yang lama ? Karena mesin akan melakukan vektorisasi terhadap semua file yang berada pada setiap foldernya, di sini terdapat 10 folder dengan masing-masing folder terdiri atas 100 buah lagu, setiap lagu tersebut akan dilakukan vektorisasi atau ekstraksi data menggunakan mfcc. Oleh karena itu, proses cukup memakan waktu.

<pre>In [31]: features, labels = generate_features_and_labels() Processing 100 songs in blues genre... Processing 100 songs in classical genre... Processing 100 songs in country genre... Processing 100 songs in disco genre... Processing 100 songs in hiphop genre... Processing 100 songs in jazz genre... Processing 100 songs in metal genre... Processing 100 songs in pop genre... Processing 100 songs in reggae genre... Processing 100 songs in rock genre...</pre>	<pre>In [32]: print(np.shape(features)) ... print(np.shape(labels)) (1000, 25000) (1000, 10)</pre>												
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Name</th> <th>Type</th> <th>Size</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>features</td> <td>float32</td> <td>(1000, 25000)</td> <td>[[-0.81999075 -0.81014305 -0.75184417 ... -0.01818394 0.01827242 0 ...]]</td> </tr> <tr> <td>labels</td> <td>float32</td> <td>(1000, 10)</td> <td>[[1. 0. 0. ... 0. 0. 0.] [1. 0. 0. ... 0. 0. 0.]</td> </tr> </tbody> </table>	Name	Type	Size	Value	features	float32	(1000, 25000)	[[-0.81999075 -0.81014305 -0.75184417 ... -0.01818394 0.01827242 0 ...]]	labels	float32	(1000, 10)	[[1. 0. 0. ... 0. 0. 0.] [1. 0. 0. ... 0. 0. 0.]	
Name	Type	Size	Value										
features	float32	(1000, 25000)	[[-0.81999075 -0.81014305 -0.75184417 ... -0.01818394 0.01827242 0 ...]]										
labels	float32	(1000, 10)	[[1. 0. 0. ... 0. 0. 0.] [1. 0. 0. ... 0. 0. 0.]										

Gambar 6.68 Hasil dari kode program praktek nomor 5

6.4.2.6 Jelaskan kenapa harus dilakukan pemisahan data training dan data set sebesar 80%? Praktekan dengan kode dan Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

Kode :

```

1 # In [Soal Nomor 6]:
2 training_split = 0.8
3 # In [Soal Nomor 6]:
4 alldata = np.column_stack((features , labels))
5 # In [Soal Nomor 6]:
6 np.random.shuffle(alldata)
7 splitidx = int(len(alldata) * training_split)
8 train , test = alldata [:splitidx ,:] , alldata [splitidx :,:]
9 # In [Soal Nomor 6]:
10 print(np.shape(train))
11 print(np.shape(test))
12 # In [Soal Nomor 6]:
13 train_input = train[:, :-10]
14 train_labels = train[:, -10:]
15 # In [Soal Nomor 6]:
16 test_input = test[:, :-10]
17 test_labels = test[:, -10:]
18 # In [Soal Nomor 6]:
19 print(np.shape(train_input))
20 print(np.shape(train_labels))

```

Dilakukannya pemisahan data training dan data set sebesar 80% agar model yang dihasilkan lebih akurat dan meminimalisir kesalahan dalam pembelajaran mesin.

In [33]:	training_split = 0.8	In [36]:	print(np.shape(train)) ...: print(np.shape(test)) (800, 25010) (200, 25010)
In [39]:		print(np.shape(train_input)) ...: print(np.shape(train_labels)) (800, 25000) (800, 10)	

training_split	float	1	0.8
alldata	float32	(1000, 25010)	[[-0.81999075 -0.81014305 -0.75184417 ... 0. 0. ...]
<hr/>			
Name	Type	Size	Value
training_split	float	1	0.8
train	float32	(800, 25010)	[[-0.16444726 -0.08589051 0.2347324 ... 1. 0. ...]
test	float32	(200, 25010)	[[-0.3358299 -0.33254293 -0.4001908 ... 0. 1. ...]
splitidx	int	1	800
train_input	float32	(800, 25000)	[[-0.16444726 -0.08589051 0.2347324 ... 0.06372784 0.05331... -0. ...]
train_labels	float32	(800, 10)	[[0. 0. 0. ... 1. 0. 0.] [0. 0. 1. ... 0. 0. 0.]
test_input	float32	(200, 25000)	[[-0.3358299 -0.33254293 -0.4001908 ... 0.05416475 0.05123... 0. ...]
test_labels	float32	(200, 10)	[[0. 0. 0. ... 0. 0. 1.] [0. 0. 0. ... 0. 0. 1.]

Gambar 6.69 Hasil dari kode program praktek nomor 6

6.4.2.7 Praktekkan dan jelaskan masing-masing parameter dari fungsi Sequential(). Tunjukkan keluaranya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

Kode :

```
1 # In [ Soal Nomor 7]:
2 model = Sequential([
3     Dense(100, input_dim=np.shape(train_input)[1]),
4     Activation('relu'),
5     Dense(10),
6     Activation('softmax'),
7 ])
```

fungsi Sequential() adalah sebuah model untuk menentukan izin pada setiap neuron, di sini adalah 100 dense yang merupakan 100 neuron pertama dari data pelatihan. Fungsi dari relay itu sendiri adalah untuk mengaktifkan neuron atau input yang memiliki nilai maksimum. Sedangkan untuk dense 10 itu adalah output dari hasil neuron yang telah berhasil diaktifkan, untuk dense 10 diaktifkan menggunakan softmax.

```
In [40]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])
```

Gambar 6.70 Hasil dari kode program praktek nomor 7

6.4.2.8 Praktekkan dan jelaskan masing-masing parameter dari fungsi compile(). Tunjukkan keluaranya dengan fungsi summary dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

Kode :

```
1 # In [ Soal Nomor 8]:
2 model.compile(optimizer='adam',
3                 loss='categorical_crossentropy',
4                 metrics=['accuracy'])
5 print(model.summary())
```

Model Compile di perjelas dengan gambar dibawah, Hasil output pada kode tersebut seperti gambar menjelaskan bahwa dense pertama itu memiliki 100 neurons dengan parameter sekitar 2 juta lebih dengan aktivasi 100, jadi untuk setiap neurons memiliki masing-masing 1 aktivasi. Sama halnya seperti dense 2 memiliki jumlah neurons sebanyak 10 dengan parameter 1010 dan jumlah aktivasinya 10 untuk setiap neurons tersebut dan total parameternya sekitar 2.5 juta data yang akan dilatih pada mesin tersebut.

```
In [42]: model.compile(optimizer='adam',
...:                     loss='categorical_crossentropy',
...:                     metrics=['accuracy'])
...: print(model.summary())
Model: "sequential_2"

Layer (type)                 Output Shape              Param #
=====
dense_3 (Dense)            (None, 100)             2500100
activation_3 (Activation)   (None, 100)              0
dense_4 (Dense)            (None, 10)               1010
activation_4 (Activation)   (None, 10)               0
=====
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0
=====
None
```

Gambar 6.71 Hasil dari kode program praktek nomor 8

6.4.2.9 Praktekkan dan jelaskan masing-masing parameter dari fungsi *fit()*. dan tunjukkan maksud setiap luaran yang didapatkan

Kode :

```
# In [Soal Nomor 9]:
model.fit(train_input, train_labels, epochs=10, batch_size=32,
           validation_split=0.2)
```

Kode tersebut berfungsi untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap 1 kali pengulangan. Sedangkan validation split itu untuk menentukan presentase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan.

```
[44]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
Train on 640 samples, validate on 160 samples
Epoch 1/10
640/640 [=====] - 3s 4ms/step - loss: 1.0998 - accuracy: 0.2703 - val_loss:
1.252 - val_accuracy: 0.3500
Epoch 2/10
640/640 [=====] - 2s 3ms/step - loss: 1.4819 - accuracy: 0.5078 - val_loss:
1.4663 - val_accuracy: 0.4812
Epoch 3/10
640/640 [=====] - 2s 3ms/step - loss: 1.0866 - accuracy: 0.6669 - val_loss:
1.4845 - val_accuracy: 0.5063
Epoch 4/10
640/640 [=====] - 2s 3ms/step - loss: 0.8257 - accuracy: 0.7563 - val_loss:
1.4801 - val_accuracy: 0.8075
Epoch 5/10
640/640 [=====] - 2s 3ms/step - loss: 0.6824 - accuracy: 0.8266 - val_loss:
1.4952 - val_accuracy: 0.4798
Epoch 6/10
640/640 [=====] - 2s 3ms/step - loss: 0.5573 - accuracy: 0.8969 - val_loss:
1.3768 - val_accuracy: 0.5132
Epoch 7/10
640/640 [=====] - 2s 3ms/step - loss: 0.4153 - accuracy: 0.9250 - val_loss:
1.3843 - val_accuracy: 0.5375
Epoch 8/10
640/640 [=====] - 2s 3ms/step - loss: 0.3512 - accuracy: 0.9469 - val_loss:
1.4800 - val_accuracy: 0.4625
Epoch 9/10
640/640 [=====] - 2s 3ms/step - loss: 0.2584 - accuracy: 0.9828 - val_loss:
1.3868 - val_accuracy: 0.5437
Epoch 10/10
640/640 [=====] - 2s 3ms/step - loss: 0.1953 - accuracy: 0.9986 - val_loss:
1.4956 - val_accuracy: 0.5926

```

Gambar 6.72 Hasil dari kode program praktek nomor 9

**6.4.2.10 Praktekkan dan jelaskan masing-masing parameter dari fungsi *evalu-*
ate(). Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap
luaran yang didapatkan**

Kode :

```

1 # In [Soal Nomor 10]:
2 loss, acc = model.evaluate(test_input, test_labels, batch_size
3 =32)
4 # In [Soal Nomor 10]:
5 print("Done!")
5 print("Loss: %.4f, accuracy: %.4f" % (loss, acc))

```

Fungsi evaluate atau evaluasi ini ialah untuk menguji data pengujian se- tiap file. Di sini ada prediksi yang hilang, artinya mesin memprediksi data, sedangkan untuk keseluruhan perjanjian sekitar 55%.

```

In [45]: print("Done!")
...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
Done!
Loss: 1.2897, accuracy: 0.5800

```

Gambar 6.73 Hasil dari kode program praktek nomor 10

6.4.2.11 Praktekkan dan jelaskan masing-masing parameter dari fungsi predict(). Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan

Kode :

```

1 # In [Soal Nomor 11]:
2 model.predict(test_input[:1])

```

Fungsi Predict ialah untuk menghasilkan suatu nilai yang sudah di prediksi dari data training sebelumnya. Gambar dibawah ini menjelaskan file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre rock. Maka lagu tersebut termasuk ke dalam genre rock dengan perbandingan presentase hasil prediksi.

```

In [46]: model.predict(test_input[:1])
Out[46]:
array([[2.30070949e-03, 2.77521031e-06, 2.15052031e-02, 3.57471257e-01,
       9.54874605e-03, 1.09723915e-05, 3.56088638e-01, 1.69718251e-09,
      1.03918152e-04, 2.52967805e-01]], dtype=float32)

```

Gambar 6.74 Hasil dari kode program praktek nomor 11

6.4.3 Penanganan Error

6.4.3.1 Terjadi error

1. terjadi error no module named 'librosa', seperti pada gambar berikut:

```

ModuleNotFoundError: No module named 'librosa'

```

Gambar 6.75 terjadi Error 1

2. terjadi error no module named 'keras', seperti pada gambar berikut:

```
ModuleNotFoundError: No module named 'keras'
```

Gambar 6.76 terjadi Error 2

3. terjadi error no module named 'tensorflow', seperti pada gambar berikut:

```
ModuleNotFoundError: No module named 'tensorflow'
```

Gambar 6.77 terjadi Error 3

4. terjadi error module 'scipy' has no attribute 'signal', seperti pada gambar berikut:

```
AttributeError: module 'scipy' has no attribute 'signal'
```

Gambar 6.78 terjadi Error 4

6.4.3.2 Solusi

1. solusi dari error 1 ialah:

```
(base) C:\Windows\system32>pip install librosa
Collecting librosa
  Downloading https://files.pythonhosted.org/packages/77/b5/181786/librosa-0.7.2.tar.gz (1.6MB)
    |██████████| 1.2MB 187kB/s eta 0:00:03
```

Gambar 6.79 solusi error 1

2. solusi dari error 2 ialah:

```
(base) C:\Windows\system32>pip install keras
Collecting keras
  Downloading https://files.pythonhosted.org/p/Keras-2.3.1-py2.py3-none-any.whl (377kB)
```

Gambar 6.80 solusi error 2

3. solusi dari error 3 ialah:

```
(base) C:\Windows\system32>pip install tensorflow
Collecting tensorflow
  Downloading https://files.pythonhosted.org/packages/3
/tensorflow-2.1.0-cp37-cp37m-win_amd64.whl (355.8MB)
    |                                                 1.5MB 91kB/s et
```

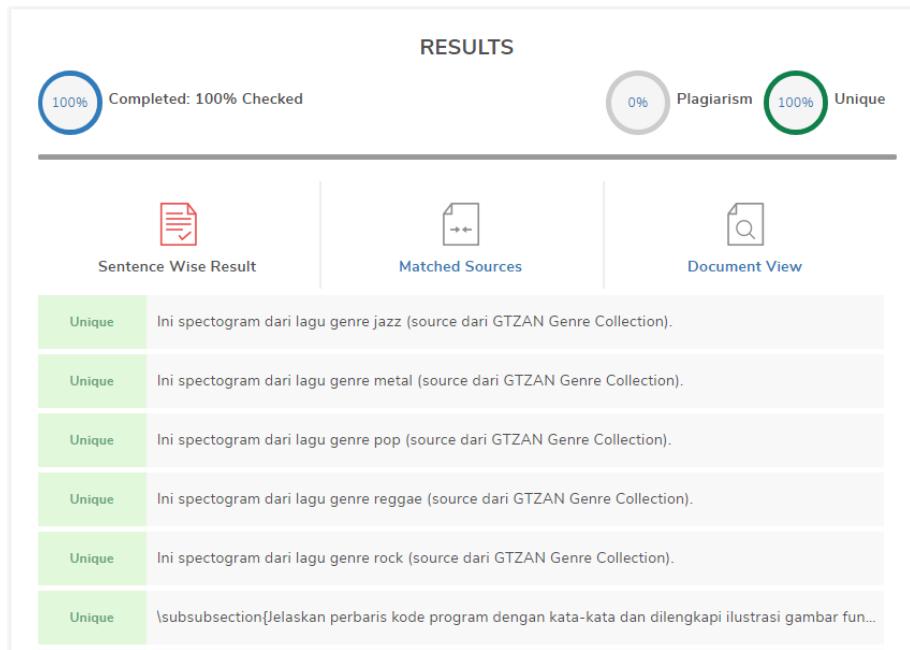
Gambar 6.81 solusi error 3

4. solusi dari error 4 ialah menambahkan kode berikut:

```
from scipy import signal
```

Gambar 6.82 solusi error 4

6.4.4 Bukti Tidak Plagiat



Gambar 6.83 Bukti tidak plagiat

6.4.5 Link Youtube

<https://youtu.be/RgBkUfYZHgY>

6.5 Arrizal Furqona Gifary / 1174070

6.5.1 Teori

1. Jelaskan kenapa file suara harus dilakukan MFCC dilengkapi dengan ilustrasi atau gambar.

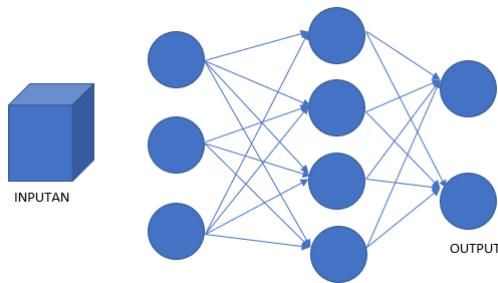
digunakan untuk mengidentifikasi jenis suara misalkan jenis suara gendre lagu jes pop metal dan klasikal atau suara ultra sonic.



Gambar 6.84 Ilustrasi gambar metode MFCC

2. Jelaskan konsep dasar neural network. dilengkapi dengan ilustrasi gambar.

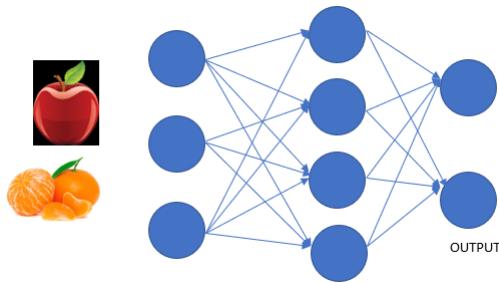
konsep neural network dilakukan ada inputan pasti ada outputan sesuai dengan kategori inputan dan fungsi di dalamnya.



Gambar 6.85 Ilustrasi Konsep dasar neural network

3. Jelaskan konsep pembobotan dalam neural network. dilengkapi dengan ilustrasi gambar.

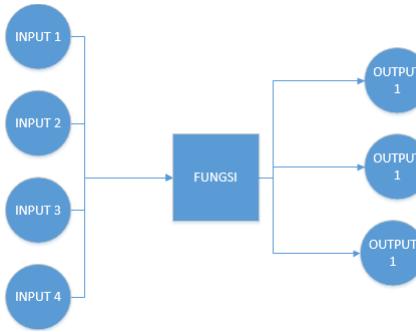
pembobotan dalam neural network yaitu digunakan untuk membedakan objek inputan atau variabel inputan untuk AI misalkan apel dan jeruk digunakan untuk variabel inputan maka dibuat pembobotan antara kedua benda tersebut untuk menentukan output yang pasti dari inputan yang dilakukan.



Gambar 6.86 Ilustrasi Konsep pembobotan pada neural network

4. Jelaskan konsep aktifitas dalam neural network. dilengkapi dengan ilustrasi gambar.

cara aktifitas dalam neural network dilakukan terhadap input pada neural network inputan tersebut dimasukan kepada fungsi pada mesin misalkan fungsi $\tanh(x)$ sehingga di hasilkanlah output yang sesuai dengan fungsi tersebut.

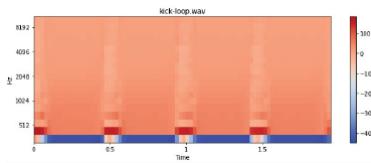


Gambar 6.87 Gambar yang dibaca hasil plotnya

5. Jelaskan cara membaca hasil plot dari MFCC dilengkapi dengan ilustrasi gambar.

cara membaca hasil plotting dari MFCC yaitu tentukan terlebih dahulu batas minimal Hz dari gelombang suara dan batas maksimal dari suara tersebut. kemudian warna yang paling pekat merupakan hasil dari pengolahan data tersebut misalkan muncul warna orange pekat di bagian bawah dan orange muda di bagian atas yang berarti suara tersebut kuat bagian basnya dan biasanya juga antara warna yang pekat tersebut ada jarak.

6. Jelaskan apa itu one-hot encoding, dilengkapi dengan ilustrasi kode atau gambar.



Gambar 6.88 Ilustrasi Cara Membaca Hasil Plot

one-hot encoding merupakan pemberian nilai pada suatu variabel jika nilai itu iya maka nilainya satu dan jika tidak maka nilainya nol.

	pop	rock	blues	rege	clasical
Lagu 1	1	0	0	0	0
Lagu 2	1	0	0	0	0
Lagu 3	0	1	0	0	0
Lagu 4	0	0	1	0	0
Lagu 5	0	0	0	1	0
Lagu 6	0	0	0	0	1
Lagu 7	0	0	0	0	1

Gambar 6.89 Ilustrasi Konsep one-hot encoding

7. Jelaskan apa dari np.unique dan to_categorical dalam kode program, dilengkapi dengan ilustrasi atau gambar.
digunakan untuk membuat array sedangkan to_categorical digunakan untuk membuat matrix bauititu 64 bit atau 32 bit.

```
>>> np.unique ([ 1 , 1 , 2 , 2 , 3 , 3 ])
array([1, 2, 3])
>>> a = np.array ([[ 1 , 1 ], [ 2 , 3 ]])
>>> np.unique ( a )
array([1, 2, 3])
```

Gambar 6.90 Ilustrasi np.unique

to_categorical

```
keras.utils.to_categorical(y, num_classes=None, dtype='float32')
```

Gambar 6.91 Ilustrasi to_categorical

8. Jelaskan apa fungsi dari Sequential dalam kode program, dilengkapi dengan ilustrasi atau gambar.
sequential adalah proses perbandingan setiap elemen satu persatu mulai dari dari objek pertama hingga yang di tuju atau jika mencari angka 100

maka sequential akan membagi bagian misalnya dari satu sampai 20 dan seterusnya sampai mendapat nilai seratus.

Bobot 1	Bobot 2	Bobot 3	Bobot 4	Bobot 5
1-20	21-40	41-60	61-80	81-100

Gambar 6.92 Ilustrasi Konsep pembobotan pada neural network

6.5.2 Praktikum

1. Jelaskan isi dari data GTZAN Genre Collection dan data dari freesound. Buat kode program untuk meload data tersebut untuk digunakan pada MFCC. Jelaskan arti dari perbaris kode yang dibuat (harus beda dengan teman satukelas).

Isi data data merupakan datasets lagu atau suara yang tersirri dari 10 gendre yang di simpan kedalam 10 folder yaitu folder blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock ke sepu-luh folder tersebut masing-masing berisi 100 data suara sedangkan data freesound merupakan contoh data suara yang akan di gunakan untuk menguji hasil pengolahan data tersebut dengan menggunakan metode mfcc. apakah suara dari freesound termasuk kategori jazz pop atau sebagainya ?.

```

1 import librosa
2 import librosa.feature
3 import librosa.display
4 import glob
5 import numpy as np
6 import matplotlib.pyplot as plt
7 from keras.layers import Dense, Activation
8 from keras.models import Sequential
9 from keras.utils.np_utils import to_categorical
10
11 # In[1]: buat fungsi mfcc untuk ngetest ajah
12 def display_mfcc(song):
13     y, _ = librosa.load(song)
14     mfcc = librosa.feature.mfcc(y)
15
16     plt.figure(figsize=(10, 4))
17     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
18     plt.colorbar()
19     plt.title(song)
20     plt.tight_layout()
21     plt.show()
```

dapat dilihat pada kode diatas pada baris kesatu dilakukan import librosa tang digunakan untuk fungsi mfcc pada suara. pada baris kedua dilakukan import librosa feature dan pada baris ke tiga dilakukan librosa

display selanjutnya pada baris ke empat dilakukan import glob kemudian insert numpy untuk pengolahan data menjadi vektor setelah itu dilakukan import matplotlib untuk melakukan plotting setelah itu dilakukan import librari keras.

Selanjutnya yaitu membuat fungsi mfcc dengan nama display_mfcc yang didalamnya terdapat variabel y yang berisi method librosa load kemudian variabel mfcc yang berisi method librosa featurea mfcc. Setelah itu membuat float figure dengan ukuran 10 banding 4 kemudian di isi oleh data librosa display dengan variabel x nya yaitu waktu dan y yaitu mel atau Hz kemudian melakukan plot warna setelah itu melakukan plot judul dan terakhir float di tampilkan.

2. Jelaskan perbaris kode program dengan kata-kata dan di lengkapi ilustrasi gambar fungsi dari display_mfcc().

```

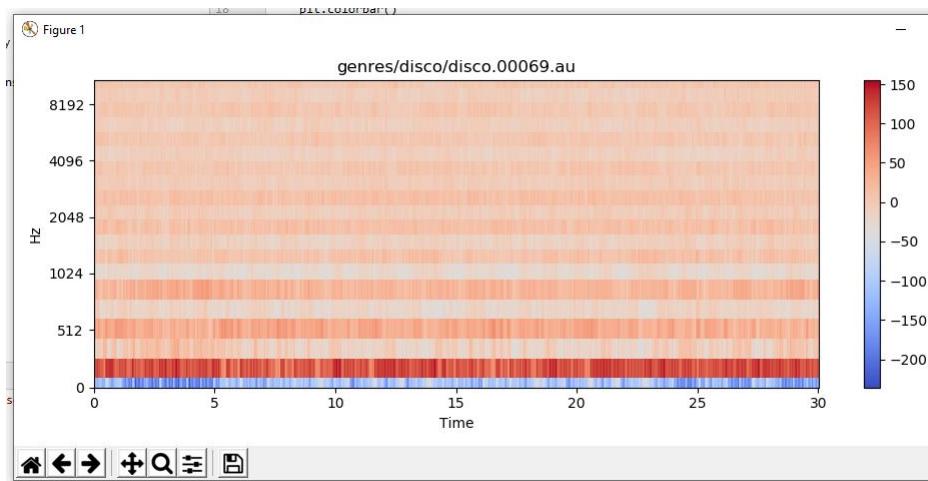
1 # In [2]: cek fungsi
2 display_mfcc('genres/disco/disco.00069.au')
3 # In [2]: cek fungsi
4 display_mfcc('genres/blues/blues.00069.au')
5 # In [2]: cek fungsi
6 display_mfcc('genres/classical/classical.00069.au')
7 # In [2]: cek fungsi
8 display_mfcc('genres/country/country.00069.au')
9 # In [2]: cek fungsi
10 display_mfcc('genres/hiphop/hiphop.00069.au')
11 # In [2]: cek fungsi
12 display_mfcc('genres/jazz/jazz.00069.au')
13 # In [2]: cek fungsi
14 display_mfcc('genres/pop/pop.00069.au')
15 # In [2]: cek fungsi
16 display_mfcc('genres/reggae/reggae.00069.au')
17 # In [2]: cek fungsi
18 display_mfcc('genres/rock/rock.00069.au')
```

pada baris ke dua program diatas digunakan untuk mendisplay tampilan glombang suara dari file 266093_stereo-surgeon_kick-loop-5.wav menggunakan metode mfcc dengan menggunakan fungsi display_mfcc yang telah tadi di buat pada nomer dua begitu juga pada baris ke 4 6 8 sampai ke 22 secara teksis sama menggunakan fungsi display_mfcc hanyasaja beda penyimpanan data yang akan di tampilkan atau di eksekusi. untuk contoh hasilnya dapat dilihat pada gambar ?? berikut:

Gambar tersebut merupakan hasil dari mfcc dari salah satu gender lagu pop yang ada pada datasets yang 1000 atau terdapat dalam sepuluh folder tadi.

3. Jelaskan perbaris dengan kata-kata dan dilengkapi dengan ilustrasi gambar fungsi dari extract_features_song jelaskan kenapa data yangdiambil merupakan data 25.000 baris pertama ?

```
1 def extract_features_song(f):
```



Gambar 6.93 Ilustrasi gambar fungsi dari `display_mfcc()`

```

2     y, _ = librosa.load(f)
3
4     # get Mel-frequency cepstral coefficients
5     mfcc = librosa.feature.mfcc(y)
6     # normalize values between -1,1 (divide by max)
7     mfcc /= np.amax(np.absolute(mfcc))
8
9     return np.ndarray.flatten(mfcc)[:25000]

```

pada baris ke tiga di definisikan nama `extract_features_song` yang nantinya akan di gunakan pada fungsi yang lainnya kemudian dibuat variabel `y` dengan method `librosa load` setelah itu dibuat variabel baru `mfcc` dengan isi `librosa features mfcc` dengan isi variabel `y` tadi kemudian dibuat variabel `mfcc` dengan isian `np.max` dan variabel `mfcc` tadi terakhir di buat array dari data tersebut merupakan data 25000 data pertama. kenapa data 25000 pertama yang digunakan dikarenakan data tersebut digunakan sebagai data testing semakin besar data testing yang di gunakan maka semakin akurat hasil AI. tapi sebenarnya data tersebut relatif bisa lebih besar atau lebih kecil tergantung pada komputer masing masing.

4. Jelaskan Perbaris kode program dengan kata-kata dan lengkapi ilustrasi gambar fungsi dari `generate_features_and_labels`.

```

1 def generate_features_and_labels():
2     all_features = []
3     all_labels = []
4
5     genres = [ 'blues', 'classical', 'country', 'disco', 'hiphop',
6               'jazz', 'metal', 'pop', 'reggae', 'rock' ]

```

```

6   for genre in genres:
7       sound_files = glob.glob('genres/' +genre+'/*.au')
8       print('Processing %d songs in %s genre...', % (len(
9           sound_files), genre))
10      for f in sound_files:
11          features = extract_features_song(f)
12          all_features.append(features)
13          all_labels.append(genre)
14
15      # convert labels to one-hot encoding cth blues :
16      1000000000 classic 0100000000
17      label_uniq_ids, label_row_ids = np.unique(all_labels,
18          return_inverse=True)#ke integer
18      label_row_ids = label_row_ids.astype(np.int32, copy=False)
19      onehot_labels = to_categorical(label_row_ids, len(
20          label_uniq_ids))#ke one hot
21
22      return np.stack(all_features), onehot_labels

```

pada baris ke tiga merupakan pendefinisian nama fungsi yaitu generate features and labels kemudian membuat variabel baru dengan array kosong yaitu all_features dan all_labels kemudian mendefinisikan isian label untuk gendre dengan cara membuat variabel genres kemudian di isi dengan 10 gendre yang tadi setelah itu dilakukan fungsi if else dengan code for dan in setelah itu akan di buat encoding untuk data tiap tiap label contoh untuk blues 1000000000 dan untuk clasical 0100000000.

- Jelaskan dengan kata dan praktek kenapa penggunaan fungsi generate features and labels sangat lama saat meload dataset gendre tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

halnini menjadi lama dikarenakan mesin membaca satupersatu file yang ada pada folder dan dalam foldertersebut terdapat 100 file sehingga wajar menjadi lama ditambah lagi mengolah data yang tadinya suara menjadi bentuk vektor. berikut merupakan codenya.

```

1 # In [3]: passing parameter dari fitur ekstraksi menggunakan
2                         mfcc
3 features, labels = generate_features_and_labels()

```

- jelaskan kenapa harus dilakukan pemisahan data training dan data testing sebesar 80 persen praktekan dengan kode dan tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut. untuk code nya adalah sebagai berikut yang merupakan code untuk membagi data sebanyak 80 persen untuk data training maka data musik tadi yang total jumlahnya 1000 akan di bagi dua untuk data training sebanyak 800 dan 200 untuk data testing.

```

1 # In [3]: fitur ekstraksi
2 training_split = 0.8

```

```

57     features, labels = generate_features_and_labels()
--> 58     generate_features_and_labels()
Run: 1 ×
  Using TensorFlow backend.
  Processing 100 songs in blues genre...
  Processing 100 songs in classical genre...
  Processing 100 songs in country genre...
  Processing 100 songs in disco genre...
  Processing 100 songs in hiphop genre...
  Processing 100 songs in jazz genre...
  Processing 100 songs in metal genre...
  Processing 100 songs in pop genre...
  Processing 100 songs in reggae genre...
  Processing 100 songs in rock genre...

Process finished with exit code 0

```

Gambar 6.94 Hasil dari fungsi generate features and labels

- praktekan dan jelaskan masing masing parameter dari fungsi Sequential(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

fungsi sequential digunakan untuk mengolah data inputan sesuai dengan fungsi yang ada pada fungsi sequential pada fungsi sequential kali ini menggunakan dua fungsi sequential mengkompile data dari 100 neuron atau dari 1 folder file dengan menggunakan fungsi relu dan softmax untuk menghasilkan outputan yang sesuai dengan keriteria.

```

1 # In [3]: membuat seq NN, layer pertama dense dari 100 neurons
2 model = Sequential([
3     Dense(100, input_dim=np.shape(train_input)[1]),
4     Activation('relu'),
5     Dense(10),
6     Activation('softmax'),
7 ])

```

- praktekan dan jelaskan masing masing parameter dari fungsi compile(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

yaitu fungsi kompile yang digunakan untuk mengetahui parameter yang digunakan dari data yang telah diolah untuk caranya dapat menggunakan codingan sebagai berikut, pada gambar tersebut memunculkan parameternya berapasaja dan total parameter yang digunakan.

```

1 # In [3]: fitur ekstraksi
2 model.compile(optimizer='adam',
3                 loss='categorical_crossentropy',
4                 metrics=['accuracy'])
5 print(model.summary())

```

- praktekan dan jelaskan masing masing parameter dari fungsi fit(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

```

Run: 1 ×
Layer (type)          Output Shape         Param #
dense_1 (Dense)      (None, 100)        2500100
activation_1 (Activation) (None, 100)        0
dense_2 (Dense)      (None, 10)          1010
activation_2 (Activation) (None, 10)        0
=====
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0
None
Process finished with exit code 0

```

Gambar 6.95 Hasil fungsi compile

pada fungsi ini dilakukan pengolahan data dari 10 label tadi atau 10 file data sets tadi kemudian di hitung tingkat akurasi masing masing dan tingkat kegagalan atau loss data darisetiap file tersebut caranya dengan melakukan codingan berikut. pada gambar tersebut menunjukan 10 pengolahan data untuk menentukan nilai akurasi dan loss dari data tersebut dan selanjutnya dilakukan fingsi evaluasi.

```

1 # In [3]: fitur_ekstraksi
2 model.fit(train_input, train_labels, epochs=10, batch_size
            =32,
            validation_split=0.2)

```

```

Run: 1 ×
32/640 [>.....] - ETA: 1s - loss: 0.2331 - accuracy: 1.0000
64/640 [==>.....] - ETA: 1s - loss: 0.2113 - accuracy: 1.0000
96/640 [==>.....] - ETA: 1s - loss: 0.2268 - accuracy: 0.9896
128/640 [====>.....] - ETA: 1s - loss: 0.2172 - accuracy: 0.9922
160/640 [=====>.....] - ETA: 1s - loss: 0.2371 - accuracy: 0.9812
192/640 [=====>.....] - ETA: 0s - loss: 0.2391 - accuracy: 0.9844
224/640 [=====>.....] - ETA: 0s - loss: 0.2484 - accuracy: 0.9821
256/640 [=====>.....] - ETA: 0s - loss: 0.2529 - accuracy: 0.9727
288/640 [=====>.....] - ETA: 0s - loss: 0.2549 - accuracy: 0.9722
320/640 [=====>.....] - ETA: 0s - loss: 0.2469 - accuracy: 0.9719
352/640 [=====>.....] - ETA: 0s - loss: 0.2512 - accuracy: 0.9688
384/640 [=====>.....] - ETA: 0s - loss: 0.2525 - accuracy: 0.9661
416/640 [=====>.....] - ETA: 0s - loss: 0.2508 - accuracy: 0.9688
448/640 [=====>.....] - ETA: 0s - loss: 0.2511 - accuracy: 0.9710
480/640 [=====>.....] - ETA: 0s - loss: 0.2503 - accuracy: 0.9708
512/640 [=====>.....] - ETA: 0s - loss: 0.2519 - accuracy: 0.9727
544/640 [=====>.....] - ETA: 0s - loss: 0.2543 - accuracy: 0.9724
576/640 [=====>.....] - ETA: 0s - loss: 0.2503 - accuracy: 0.9740
608/640 [=====>.....] - ETA: 0s - loss: 0.2602 - accuracy: 0.9720
640/640 [=====>.....] - 1s 2ms/step - loss: 0.2544 - accuracy: 0.9734 - val_loss: 1.2366 - val_accuracy: 0.5625

Process finished with exit code 0

```

Gambar 6.96 Hasil fungsi fit

10. praktekan dan jelaskan masing masing parameter dari fungsi evaluate(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

pada fungsi ini dilakukan evaluasi terhadap datayang telah di runing sebelumnya untuk lebih jelasnya dapat di lihat codingan tersebut pada codingan tersebut dilakukan evaluasi pada tingkat kegagalan dan akurasi kebenaran maka hasilnya munculkan hasil evaluasi dari 10 proses dari setiap gendre yaitu akurasi sebesar 51 persen dan loss data sebesar 1.4105 data.

```
1 # In [3]: fitur_ekstraksi
2 loss, acc = model.evaluate(test_input, test_labels,
3                             batch_size=32)
4 # In [3]: fitur_ekstraksi
5 print("Done!")
6 print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

Gambar 6.97 Hasil fungsi evaluasi

11. praktekan dan jelaskan masing masing parameter dari fungsi predict tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

fungsi predict merupakan fungsi untuk membandingkan tingkat akurasi pada setiap label yang sepuluh tadi maka data akan di sandingkan ke masing masing tingkat akurasinya, yang akurasinya paling tinggi maka itulah jawaban untuk setiap inputan yang dilakukan.

```
1 # In [3]: fitur_ekstraksi
2 model.predict(test_input[:1])
```

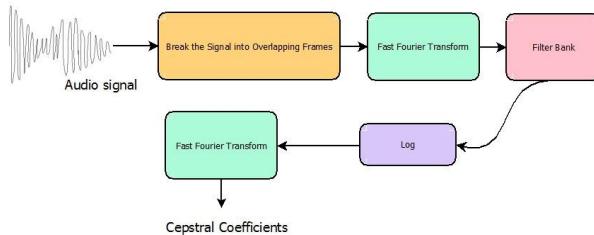
Gambar 6.98 Hasil fungsi prediksi

6.6 1174079 - Chandra Kirana Poetra

6.6.1 Teori

1. Jelaskan kenapa file suara harus di lakukan MFCC. dilengkapi dengan ilustrasi atau gambar.

Mel Frequency Cepstral Coefficient (MFCC) berguna untuk melakukan proses ekstraksi data khususnya audio dan juga merupakan suatu fitur yang ada di aplikasi pengenalan suara yang bisa digunakan untuk men- genali kata atau angka yang dikenal.

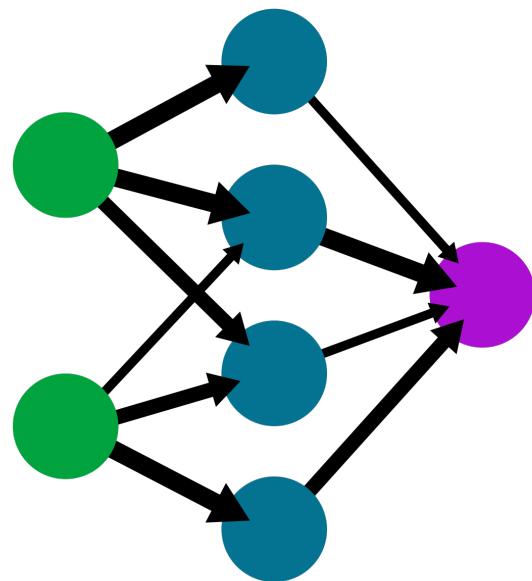


2. Jelaskan konsep dasar neural network.dilengkapi dengan ilustrasi atau gambar.

Neural network merupakan suatu set algoritma yang dibuat berdasarkan otak manusia. didesain untuk mengenali pola, neural network meng- intepresentasikan data sensor melalui berbagai macam persepsi mesin, melabeli atau mengelompokkan input. pola yang dapat dikenali dapat berupa angka atau vektor atau bisa juga data real seperti gambar atau suara dan juga teks.

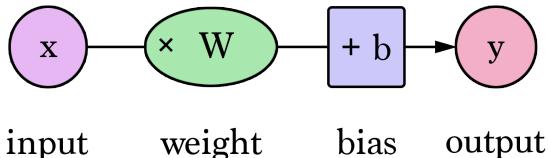
A simple neural network

input layer hidden layer output layer

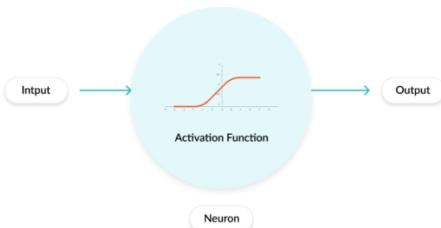


3. Jelaskan konsep pembobotan dalam neural network. dilengkapi dengan ilustrasi atau gambar.

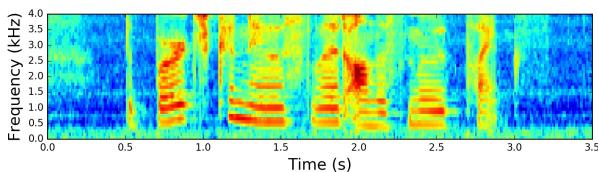
Weight merupakan parameter dalam neural network yang berguna untuk melakukan transformasi data input yang akan masuk kebagian hidden layer dari neural network. suatu neural network merupakan sekelompok dari nodes atau neuron. di dalam tiap node ada beberapa set instruksi, bobot, dan juga bias value. suatu input masuk ke bagian node kemudian di jumlahkan dengan value dari weight dan hasilnya akan di lihat atau di berikan ke layer berikutnya yang ada pada neural network. seringkali bobot yang ada pada neural network sering berada pada bagian hidden layer di neural network



4. Jelaskan konsep fungsi aktifasi dalam neural network. dilengkapi dengan ilustrasi atau gambar.
- function aktifasi merupakan sebuah rumus matematika yang ada diantara input neuron dan output yang akan menuju ke layer berikutnya. Hal ini bisa sesimple seperti langkah langkah function yang berfungsi untuk menyalakan atau mematikan suatu output dari neuron yang tergantung oleh beberapa aturan.



5. Jelaskan cara membaca hasil plot dari MFCC, dilengkapi dengan ilustrasi atau gambar.



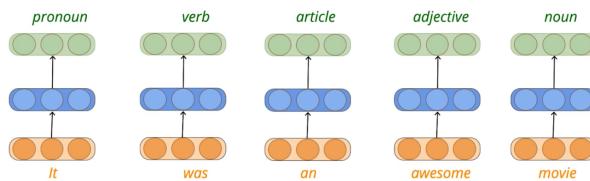
Sumbu y digunakan untuk mengukur tingginya khz, sementara sumbu x digunakan untuk mengukur durasi dari audio tersebut.

6. Jelaskan apa itu one-hot encoding, dilengkapi dengan ilustrasi kode dan atau gambar. One hot encoding digunakan untuk memisahkan columnm

yang berisi kategori berdasarkan angka menjadi ke beberapa column berdasarkan jumlah kategori yang ada. setiap kolumn berisi 0 atau 1. one hot encoding dipakai untuk kasus kasus seperti misalkan pada suatu dataset, kita menemukan kolumn yang berisi angka tanpa susunan. data ini biasanya menggambarkan kategori atau value dari suatu kategori. ini akan membuat machine learning kebingungan sehingga kita butuh yang namanya one hot encoding agar mesin mengerti.

Label Encoding			One Hot Encoding			
Food Name	Categorical #	Calories	Apple	Chicken	Broccoli	Calories
Apple	1	95	1	0	0	95
Chicken	2	231	0	1	0	231
Broccoli	3	50	0	0	1	50

7. Jelaskan apa fungsi dari Sequential dalam kode program,dilengkapi dengan ilustrasi atau gambar.
 Sequential digunakan untuk membuat suatu prediksi dari data yang dimasukkan. contohnya seperti gambar dibawah ini yang digunakan untuk memprediksi adjective, noun,verb dan pronoun dari teks bahasa inggris



6.6.2 Praktek

1. Jelaskan isi dari data GTZAN Genre Collection dan data dari freesound. Buat kode program untuk meload data tersebut untuk digunakan pada MFCC. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas).

GTZAN Genre Collection merupakan sebuah koleksi lagu dari berbagai jenis genre yang durasinya rata rata 30 detik.

Data dari freesound yang chandra gunakan ada 2 yaitu kick.wav dan juga spaceysynth.wa.

Kode program untuk meload data tersebut untuk digunakan pada MFCC.

```

1 #%%Soal1
2 import librosa
3 import librosa.feature
4 import librosa.display
5 import glob
6 import numpy as np

```

```

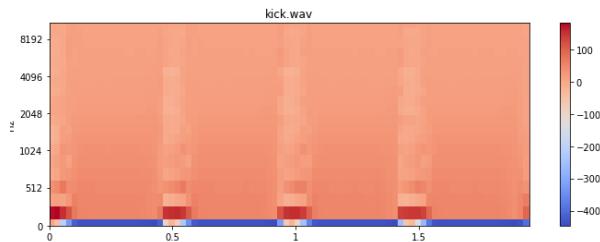
7 import matplotlib.pyplot as plt
8 from keras.models import Sequential
9 from keras.layers import Dense, Activation
10 from keras.utils.np_utils import to_categorical
11
12 def display_mfcc(song):
13     y, _ = librosa.load(song)
14     mfcc = librosa.feature.mfcc(y)
15
16     plt.figure(figsize=(10, 4))
17     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
18     plt.colorbar()
19     plt.title(song)
20     plt.tight_layout()
21     plt.show()
22
23 display_mfcc('kick.wav')
24
25 display_mfcc('spaceysynth.wav')
26
27 display_mfcc('country.00000.au')
28
29 display_mfcc('blues.00000.au')

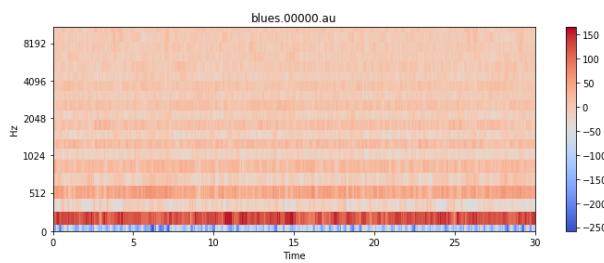
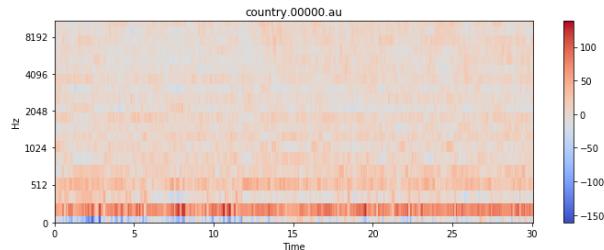
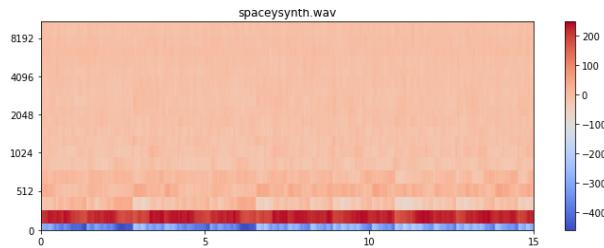
```

Import librarynya terlebih dahulu seperti librosa, glob dan numpy serta matplotlib dan juga keras. Librosa digunakan untuk ekstrak fitur yang ada pada lagu, glob digunakan untuk membuat list file, sedangkan numpy untuk melakukan operasi yang berhubungan dengan angka dan matplotlib untuk membuat grafik. import juga sequential dan lain lain dari library keras untuk penggunaan function activation serta dense yang memiliki banyak sekali neuron didalamnya serta tocategorycal untuk merubah vector menjadi binary

Setelah itu buat class dengan parameter bernama song yang isinya berisi function untuk load lagu dan fitur serta spectrogramnya

Berikut hasilnya





2. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari displaymfcc()

```

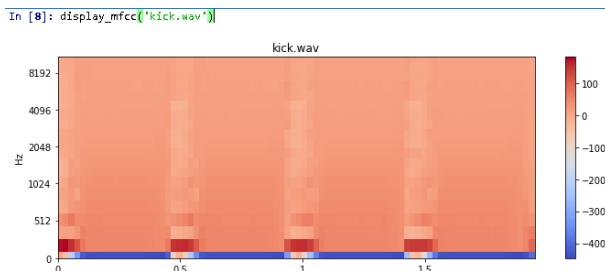
1 %%Soal2
2 def display_mfcc(song):
3     y, _ = librosa.load(song)
4     mfcc = librosa.feature.mfcc(y)
5
6     plt.figure(figsize=(10, 4))
7     librosa.display.specshow(mfcc, x_axis='time',
8     y_axis='mel')
9     plt.colorbar()
10    plt.title(song)

```

```
10     plt.tight_layout()
11     plt.show()
```

- Nama fungsinya adalah displaymfcc dengan satu parameter bernama song.
- Fungsi load() yang dipanggil digunakan untuk meload lagunya yang kemudian ditampung di variable.
- Fungsi mfcc() dipakai untuk melakukan ekstrak fitur file lagu yang dimuat kemudian ditampung pada variable nilainya.
- Fungsi figure() digunakan untuk membuat grafik.
- Fungsi specshow() untuk membuat spectrogram
- Fungsi colorbar() digunakan untuk memberi warna pada grafik
- Fungsi title() untuk memberikan judul pada grafik.
- Fungsi tightlayout() untuk menyesuaikan layout grafik.
- Fungsi show() menampilkan output grafik.

Jika kita memanggil fungsi displaymfcc() nantinya akan seperti ini.



3. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari extractfeaturesong(). Jelaskan juga mengapa yang diambil 25.000 baris pertama?

```
1 def extract_features_song(f):
2     y, _ = librosa.load(f)
3     # get Mel-frequency cepstral coefficients
4     mfcc = librosa.feature.mfcc(y)
5     # normalize values between -1,1 (divide by max)
6     mfcc /= np.amax(np.absolute(mfcc))
7     return np.ndarray.flatten(mfcc)[:25000]
8
9 extract_features_song('blues.00000.au')
```

- Pertama-tama nama fungsinya adalah extractfeaturesong dengan parameter bernama f
- Fungsi load() dipakai untuk memuat lagunya yang kemudian disimpan di variable.
- Fungsi mfcc() untuk melakukan ekstrak fitur file lagu yang dimuat lalu disimpan nilainya di variable.
- Fungsi absolute() digunakan supaya nilainya menjadi nilai absolut. Sedangkan fungsi amax dipakai untuk mencari nilai max. Lalu hasil yang didapatkan sebelumnya dibagi dengan nilai mfcc.
- Terakhir, hasil yang telah ditampung di variable mfcc di convert ke array 1 dimensi dan diambil 25000 baris pertama saja mengembalikan hasil convert array ke dalam bentuk satu dimensi dari nilai mfcc dan diambil 25000 baris pertama.
- Alasannya mengapa hanya diambil 25000 baris pertama karena hampir setiap lagu memiliki durasi atau panjang yang berbeda beda se mentara data yang harus dimasukan ke machine learning harus ukuran yang sama.

```

....: extract_features_song('blues.
00000.au')
Out[12]:
array([-0.81999071, -0.81014303, -0.75184401,
..., -0.01818394,
         0.01827243,  0.01200242])

```

4. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari generatefeaturesandlabels().

```

1 def generate_features_and_labels():
2     all_features = []
3     all_labels = []
4
5     genres = [ 'blues', 'classical', 'country', 'disco', 'hiphop',
6               'jazz', 'metal', 'pop', 'reggae', 'rock' ]
7     for genre in genres:
8         sound_files = glob.glob('*.*au')
9         print('Processing %d songs in %s genre...' % (len(
10            sound_files), genre))
11         for f in sound_files:
12             features = extract_features_song(f)
13             all_features.append(features)
14             all_labels.append(genre)
15
16     # convert labels to one-hot encoding

```

```

15     label_uniq_ids , label_row_ids = np.unique( all_labels ,
16         return_inverse=True)
17     label_row_ids = label_row_ids.astype(np.int32 , copy=False)
18     onehot_labels = to_categorical(label_row_ids , len(
19         label_uniq_ids))
    return np.stack(all_features) , onehot_labels
features , labels = generate_features_and_labels()

```

- Buat fungsi dengan nama generatefeaturesandlabels().
- Kemudian buat array list allfeatures
- Kemudian buat array list alllabels
- Lalu buat array list genres untuk menyimpan daftar genre musik
- Gunakan loop untuk melakukan proses secara berulang agar semua file dapat dimuat
- Pakai fungsi glob() untuk mencari file lagu yang ingin dimuat dengan mendefinisikan direktoriya
- Memperlihatkan panjang daftar lagu berdasarkan genre
- looping pada soundfiles
- Gunakan fungsi extractfeaturessong() untuk mengambil fitur yang ada pada file musik yang di load
- tambahkan hasil dari fungsi extractfeaturessong() ke allfeatures
- Isi allfeatures dengan features dan alllabels dengan genrelabels
- Gunakan fungsi unique untuk membuat label unik
- Gunakan fungsi astype untuk mengubah type menjadi type int32
- Gunakan function tocategorical untuk melakukan konversi ke one-hot-encoding
- Return nilainya kemudian gabungkan dengan fungsi stack() dari allfeatures ke matrix
- panggil fungsinya

```

.....
.... features , labels = generate_features_and_labels()
Processing 2 songs in blues genre...
Processing 2 songs in classical genre...
Processing 2 songs in country genre...
Processing 2 songs in disco genre...
Processing 2 songs in hiphop genre...
Processing 2 songs in jazz genre...
Processing 2 songs in metal genre...
Processing 2 songs in pop genre...
Processing 2 songs in reggae genre...
Processing 2 songs in rock genre...

```

5. Jelaskan dengan kata dan praktik kenapa penggunaan fungsi generate-featuresandlabels() sangat lama ketika meload dataset genre. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

Sangat lama karena file yang dimuat bisa banyak yaitu ada 1000 file serta kita juga mencari 25000 nilai pertama dari setiap file yang ada sehingga akibatnya lama, apabila hanya dua file seperti gambar dibawah ini tentunya lebih cepat.

```
.....
....: features, labels = generate_features_and_labels()
Processing 2 songs in blues genre...
Processing 2 songs in classical genre...
Processing 2 songs in country genre...
Processing 2 songs in disco genre...
Processing 2 songs in hiphop genre...
Processing 2 songs in jazz genre...
Processing 2 songs in metal genre...
Processing 2 songs in pop genre...
Processing 2 songs in reggae genre...
Processing 2 songs in rock genre...
```

6. Jelaskan kenapa harus dilakukan pemisahan data training dan data set sebesar 80 persen? Praktekkan dengan kode dan Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

Karena agar nantinya data yang akan kita hitung menghasilkan hasil yang lebih akurat dan juga meminimalisir kemungkinan kesalahan oleh mesinnya itu sendiri.

```
1 print(np.shape(features))
2 print(np.shape(labels))
3 training_split = 0.8
4 # last column has genre, turn it into unique ids
5 alldata = np.column_stack((features, labels))
6 np.random.shuffle(alldata)
7 splitidx = int(len(alldata) * training_split)
8 train, test = alldata[:splitidx, :], alldata[splitidx:, :]
9 print(np.shape(train))
10 print(np.shape(test))
11 train_input = train[:, :-10]
12 train_labels = train[:, -10:]
13 test_input = test[:, :-10]
14 test_labels = test[:, -10:]
15 print(np.shape(train_input))
16 print(np.shape(train_labels))
```

- (20, 25000)
- (20, 10)
- (16, 25010)
- (4, 25010)
- (16, 25000)
- (16, 10)

- Ukuran array atau jumlah data pada data features
- Ukuran array atau jumlah data pada data labels
- Ukuran array atau jumlah data pada data train
- Ukuran array atau jumlah data data test
- Ukuran array atau jumlah data data traininput
- Ukuran array atau jumlah data data trainlabels

7. Praktekkan dan jelaskan masing-masing parameter dari fungsi Sequential(). Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```
1 model = Sequential([
2     Dense(100, input_dim=np.shape(train_input)[1]),
3     Activation('relu'),
4     Dense(10),
5     Activation('softmax'),
6 ])
```

Layer pertama terdapat 100 neuron dengan parameter berjumlah 25000 yang didapat dari variable train input dengan neuron sebanyak 100, layer kedua memiliki 10 neuron dengan parameter sekitar 1010 input ditambah biasnya 10. Dengan total params 250110. Setelah itu data dilakukan training.

- Layer pertama adalah layer dense dengan jumlah 100 neuron, kemudian inputkan datanya
- Eksekusi activation dengan tipe relu
- Lalu ada layer kedua dengan jumlah sebanyak 10 neuron
- Melakukan fungsi activation dengan tipe softmax

```
Model: "sequential_1"
=====
Layer (type)          Output Shape
Param #
=====
dense_1 (Dense)      (None, 100)
2500100
=====
activation_1 (Activation)  (None, 100)      0
=====
dense_2 (Dense)      (None, 10)           1010
=====
activation_2 (Activation)  (None, 10)           0
=====
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0
```

8. Praktekkan dan jelaskan masing-masing parameter dari fungsi compile(). Tunjukkan keluarannya dengan fungsi summary dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```
1 model.compile(optimizer='adam',
2                     loss='categorical_crossentropy',
3                     metrics=['accuracy'])
```

- Optimizer yang dipakai yaitu adam
- Loss dipakai adalah categoricalexentropy
- Metrics yang dipakai accuracy

```

Model: "sequential_1"
=====
Layer (type)          Output Shape
Param #
=====
dense_1 (Dense)      (None, 100)
2500100
=====
activation_1 (Activation)  (None, 100)      0
=====
dense_2 (Dense)      (None, 10)           1010
=====
activation_2 (Activation)  (None, 10)           0
=====
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0

```

Layer pertama terdapat 100 neuron dengan parameter berjumlah 25000 yang didapat dari variable train input dengan neuron sebanyak 100, layer kedua memiliki 10 neuron dengan parameter sekitar 1010 input ditambah biasnya 10. Dengan total params 2501110.

9. Praktekkan dan jelaskan masing-masing parameter dari fungsi fit(). Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```

1 model.fit(train_input, train_labels, epochs=10, batch_size
           =32, validation_split=0.2)

```

- Parameter yang pertama adalah variable train input
- Parameter yang pertama adalah variable train labels
- Jumlah epoch yang dipakai adalah 10
- Batch Size yang dipakai sebesar 32
- Validation Splitnya adalah 0.2 atau 20%

```
In [20]: model.fit(train_input, train_labels, epochs=10,
batch_size=32, validation_split=0.2)
Train on 12 samples, validate on 4 samples
Epoch 1/10
12/12 [=====] - 0s 4ms/step -
loss: 1.9453 - accuracy: 0.1667 - val_loss: 10.9742 -
val_accuracy: 0.0000e+00
Epoch 2/10
12/12 [=====] - 0s 4ms/step -
loss: 1.9359 - accuracy: 0.1667 - val_loss: 11.0833 -
val_accuracy: 0.0000e+00
Epoch 3/10
12/12 [=====] - 0s 3ms/step -
loss: 1.9135 - accuracy: 0.1667 - val_loss: 11.2846 -
val_accuracy: 0.0000e+00
Epoch 4/10
12/12 [=====] - 0s 4ms/step -
loss: 1.8996 - accuracy: 0.1667 - val_loss: 11.5726 -
```

10. Praktekkan dan jelaskan masing-masing parameter dari fungsi evaluate(). Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```
1 print("Done!")
2 print("Loss: %.4f, accuracy: %.4f" % (loss , acc))
```

- Parameter yang pertama adalah variable test input
- Parameter yang kedua adalah variable test labels
- Batch Size yang dibutuhkan 32

```
In [22]: loss, acc = model.evaluate(test_input,
test_labels, batch_size=32)
...: print("Done!")
...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
4/4 [=====] - 0s 750us/step
Done!
Loss: 11.3426, accuracy: 0.0000
```

Lossnya 11.3426,dan akurasinya 0.000

11. Praktekkan dan jelaskan masing-masing parameter dari fungsi predict(). Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```
1 print(prediction)
```

- Parameter yang pertama adalah variable test input

- Batch Size yang dibutuhkan 32

```
In [23]: prediction = model.predict(test_input,
batch_size=32)
....: print(prediction)
[[1.87984303e-01 2.17278704e-01 8.68563291e-07
4.76343166e-06
1.12103038e-01 1.13045134e-01 1.48350701e-01
2.88178694e-06
2.21214622e-01 1.49135822e-05]
[2.13587453e-04 1.58349320e-01 3.92913398e-06
1.15934108e-05
2.38298401e-01 1.33187622e-01 2.75805682e-01
2.39947894e-05
7.94716701e-02 1.14634179e-01]
[2.13587453e-04 1.58349320e-01 3.92913398e-06
1.15934108e-05
2.38298401e-01 1.33187622e-01 2.75805682e-01
2.39947894e-05
```

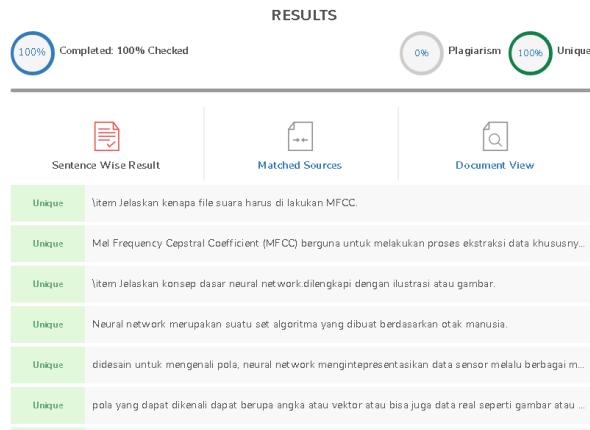
6.6.3 Penanganan Error

No module named librosa

```
C:\Users\acer>pip install librosa
Collecting librosa
  Downloading https://files.pythonhosted.org/packages/77d05/1817862d64a7c211afdf15419d8418aeff000742cac275e95c74b219cbccb
/librosa-0.7.2.tar.gz (1.6MB) |████████| 1.6MB 1.7MB/s
Collecting audiostream >=2.0.0
  Downloading https://files.pythonhosted.org/packages/2e/0b/940e7861e0e9840f09dcfd72a0e9ae55f607c17c209a323f0140f913d2
/audiostream-2.0.0.tar.gz (1.4MB) |████████| 1.4MB 1.4MB/s
Requirement already satisfied: numpy >=1.15.0 in c:\users\acer\appdata\local\programs\python\python38-32\lib\site-packages
  (from librosa) (1.18.2)
Requirement already satisfied: scipy >=1.6.0 in c:\users\acer\appdata\local\programs\python\python38-32\lib\site-packages
  (from librosa) (1.4.1)
Requirement already satisfied: scikit-learn >=0.19.0, <0.19.1 in c:\users\acer\appdata\local\programs\python\python38-32\lib\site-packages
  (from librosa) (0.22.2.post1)
Requirement already satisfied: joblib >=0.12 in c:\users\acer\appdata\local\programs\python\python38-32\lib\site-packages
  (from librosa) (0.14.1)
```

6.6.3.1 Solusi Error Install library terlebih dahulu

6.6.4 Bukti Tidak Plagiat



6.6.5 Link Youtube

<https://youtu.be/LPeaSGrQSgA>

6.7 1174077 - Alvan Alvanzah

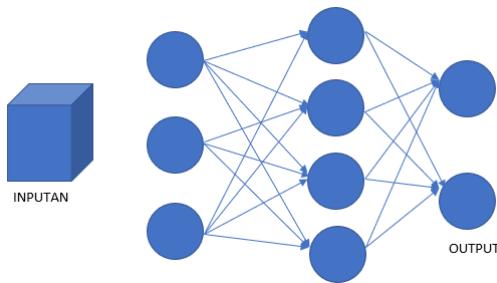
6.7.1 Teori

1. Jelaskan kenapa file suara harus dilakukan MFCC dilengkapi dengan ilustrasi atau gambar.
digunakan untuk mengidentifikasi jenis suara misalkan jenis suara gendre lagu jes pop metal dan klasikal atau suara ultra sonic.



Gambar 6.99 Ilustrasi gambar metode MFCC

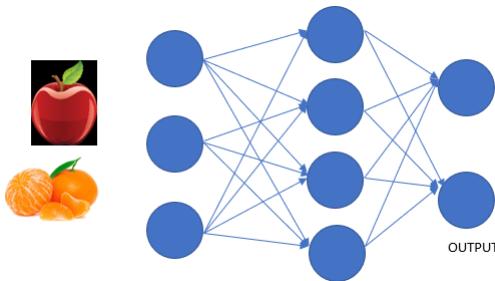
2. Jelaskan konsep dasar neural network. dilengkapi dengan ilustrasi gambar.
konsep neural network dilakukan ada inputan pasti ada outputan sesuai dengan kategori inputan dan fungsi di dalamnya.



Gambar 6.100 Ilustrasi Konsep dasar neural network

3. Jelaskan konsep pembobotan dalam neural network. dilengkapi dengan ilustrasi gambar.

pembobotan dalam neural network yaitu digunakan untuk membedakan objek inputan atau variabel inputan untuk AI misalkan apel dan jeruk digunakan untuk variabel inputan maka dibuat pembobotan antara kedua benda tersebut untuk menentukan output yang pasti dari inputan yang dilakukan.



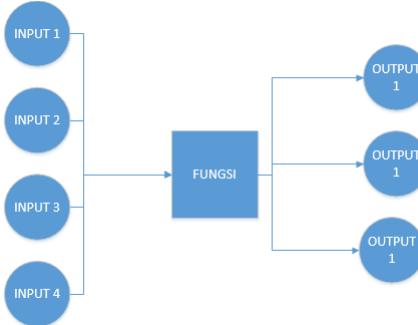
Gambar 6.101 Ilustrasi Konsep pembobotan pada neural network

4. Jelaskan konsep aktifitas dalam neural network. dilengkapi dengan ilustrasi gambar.

cara aktifitas dalam neural network dilakukan terhadap input pada neural network inputan tersebut dimasukan kepada fungsi pada mesin misalkan fungsi $\tanh(x)$ sehingga di hasilkanlah output yang sesuai dengan fungsi tersebut.

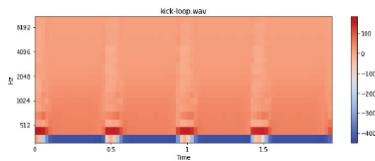
5. Jelaskan cara membaca hasil plot dari MFCC dilengkapi dengan ilustrasi gambar.

cara membaca hasil plotting dari MFCC yaitu tentukan terlebih dahulu batas minimal Hz dari gelombang suara dan batas maksimal dari suara



Gambar 6.102 Gambar yang dibaca hasil plotnya

tersebut. kemudian warna yang paling pekat merupakan hasil dari pengolahan data tersebut misalkan muncul warna orange pekat di bagian bawah dan orange muda di bagian atas yang berarti suara tersebut kuat bagian basnya dan biasanya juga antara warna yang pekat tersebut ada jarak.



Gambar 6.103 Ilustrasi Cara Membaca Hasil Plot

6. Jelaskan apa itu one-hot encoding, dilengkapi dengan ilustrasi kode atau gambar.

one-hot encoding merupakan pemberian nilai pada suatu variabel jika nilai itu iya maka nilainya satu dan jika tidak maka nilainya nol.

	pop	rock	blues	rege	clasical
Lagu 1	1	0	0	0	0
Lagu 2	1	0	0	0	0
Lagu 3	0	1	0	0	0
Lagu 4	0	0	1	0	0
Lagu 5	0	0	0	1	0
Lagu 6	0	0	0	0	1
Lagu 7	0	0	0	0	1

Gambar 6.104 Ilustrasi Konsep one-hot encoding

7. Jelaskan apa dari np.unique dan to_categorical dalam kode program, dilengkapi dengan ilustrasi atau gambar.

digunakan untuk membuat array sedangkan to_categorical digunakan untuk membuat matrix bauit 64 bit atau 32 bit.

```
>>> np.unique ([[ 1 , 1 , 2 , 2 , 3 , 3 ]])
array([1, 2, 3])
>>> a = np.array ([[ [ 1 , 1 ], [ 2 , 3 ] ]])
>>> np.unique ( a )
array([1, 2, 3])
```

>>>

Gambar 6.105 Ilustrasi np.unique

to_categorical

```
keras.utils.to_categorical(y, num_classes=None, dtype='float32')
```

Gambar 6.106 Ilustrasi to_categorical

8. Jelaskan apa fungsi dari Sequential dalam kode program, dilengkapi dengan ilustrasi atau gambar.

sequential adalah proses perbandingan setiap elemen satu persatu mulai dari dari objek pertama hingga yang di tuju atau jika mencari angka 100 maka sequential akan membagi bagian misalnya dari satu sampai 20 dan seterusnya sampai mendapat nilai seratus.

Bobot 1	Bobot 2	Bobot 3	Bobot 4	Bobot 5
1-20	21-40	41-60	61-80	81-100

Gambar 6.107 Ilustrasi Konsep pembobotan pada neural network

6.7.2 Praktek Program

1. Soal 1

```
1 # In [1]: Soal Nomor 1
2
3 import librosa
4 import librosa.feature
5 import librosa.display
6 import glob
7 import numpy as np
8 import matplotlib.pyplot as plt
9 from keras.layers import Dense, Activation
10 from keras.models import Sequential
11 from keras.utils.np_utils import to_categorical
12
```

```

13 def display_mfcc(song):
14     y, _ = librosa.load(song)
15     mfcc = librosa.feature.mfcc(y)
16
17     plt.figure(figsize=(10, 4))
18     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
19     plt.colorbar()
20     plt.title(song)
21     plt.tight_layout()
22     plt.show()

```

Kode di atas menjelaskan cara mengimport library yang dibutuhkan dan membuat fungsi display_mfcc untuk melakukan plot pada file audio nanti. Isi data GTZAN adalah datasets lagu atau suara yang terdiri dari 10 genre yang di simpan kedalam 10 folder yaitu folder blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock ke sepuluh folder tersebut masing-masing berisi 100 data suara sedangkan data freesound merupakan contoh data suara yang akan di gunakan untuk menguji hasil pengolahan data tersebut dengan menggunakan metode mfcc. jika GTZAN memiliki beberapa genre jika freesound hanya untuk 1 lagu dan disini kita membuat fungsi untuk membaca file audio dan outputnya sebagai plot, hasilnya adalah sebagai berikut:

```

In [2]: import librosa
...: import librosa.feature
...: import librosa.display
...: import numpy as np
...: import matplotlib.pyplot as plt
...: from keras.layers import Dense, Activation
...: from keras.models import Sequential
...: from keras.utils import to_categorical
...:
...: def display_mfcc(song):
...:     y, sr = librosa.load(song)
...:     mfcc = librosa.feature.mfcc(y)
...:
...:     plt.figure(figsize=(10, 4))
...:     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
...:     plt.title(song)
...:     plt.tight_layout()
...:     plt.show()

```

Gambar 6.108 Hasil Soal 1.

2. Soal 2

```

1 # In [2]: Soal Nomor 2
2 display_mfcc('freesound1.wav')
3 # In [2]: Soal Nomor 2
4 display_mfcc('freesound2.wav')
5 # In [2]: Soal Nomor 2
6 display_mfcc('genres/blues/blues.00023.au')
7 # In [2]: Soal Nomor 2
8 display_mfcc('genres/classical/classical.00023.au')
9 # In [2]: Soal Nomor 2
10 display_mfcc('genres/country/country.00023.au')
11 # In [2]: Soal Nomor 2
12 display_mfcc('genres/disco/disco.00023.au')
13 # In [2]: Soal Nomor 2
14 display_mfcc('genres/hiphop/hiphop.00023.au')
15 # In [2]: Soal Nomor 2

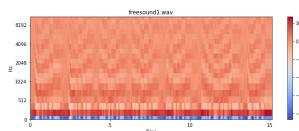
```

```

16 display_mfcc('genres/jazz/jazz.00023.au')
17 # In [2]: Soal Nomor 2
18 display_mfcc('genres/metal/metal.00023.au')
19 # In [2]: Soal Nomor 2
20 display_mfcc('genres/pop/pop.00023.au')
21 # In [2]: Soal Nomor 2
22 display_mfcc('genres/reggae/reggae.00023.au')
23 # In [2]: Soal Nomor 2
24 display_mfcc('genres/rock/rock.00023.au')

```

Kode di atas akan menampilkan hasil dari proses mfcc yang sudah dibuat fungsi pada soal 1, yaitu fungsi display mfcc akan menampilkan plot dari pembacaan file audio. Berikut adalah hasil dari salah satu pembacaan file audio :



Gambar 6.109 Hasil Soal 2.

3. Soal 3

```

1 # In [3]: Soal Nomor 3
2
3 def extract_features_song(f):
4     y, _ = librosa.load(f)
5
6     # get Mel-frequency cepstral coefficients
7     mfcc = librosa.feature.mfcc(y)
8     # normalize values between -1,1 (divide by max)
9     mfcc /= np.amax(np.absolute(mfcc))
10
11    return np.ndarray.flatten(mfcc)[:25000]

```

Kode di atas adalah membuat fungsi yang didefinisikan dengan nama extract_features_song yang nantinya akan digunakan pada fungsi yang lainnya kemudian dibuat variabel y dengan method librosa load setelah itu dibuat variabel baru mfcc dengan isi librosa features mfcc dengan isi variabel y tadi kemudian dibuat variabel mfcc dengan isian np.max dan variabel mfcc tadi terakhir dibuat array dari data tersebut merupakan data 25000 data pertama. kenapa data 25000 pertama yang digunakan dikarenakan data tersebut digunakan sebagai data testing semakin besar data testing yang digunakan maka semakin akurat hasil AI. tapi sebenarnya data tersebut relatif bisa lebih besar atau lebih kecil tergantung pada komputer masing masing. Hasilnya adalah sebagai berikut :

```
In [8]: def extract_features_song(f):
...:     y, sr = librosa.load(f)
...:     # get Mel-frequency cepstral coefficients
...:     mfcc = librosa.feature.mfcc(y)
...:     # normalize values between -1,1 (divide by max)
...:     mfcc /= np.amax(np.absolute(mfcc))
...: 
...:     return np.ndarray.flatten(mfcc)[:25000]
```

Gambar 6.110 Hasil Soal 3.

4. Soal 4

```
1 # In [4]: Soal Nomor 4
2
3 def generate_features_and_labels():
4     all_features = []
5     all_labels = []
6
7     genres = [ 'blues' , 'classical' , 'country' , 'disco' , '
8         hiphop' , 'jazz' , 'metal' , 'pop' , 'reggae' , 'rock' ]
9     for genre in genres:
10        sound_files = glob.glob('genres/' + genre + '/*.au')
11        print('Processing %d songs in %s genre...' % (len(
12            sound_files), genre))
13        for f in sound_files:
14            features = extract_features_song(f)
15            all_features.append(features)
16            all_labels.append(genre)
17
18    # convert labels to one-hot encoding cth blues :
19    # 10000000000 classic 01000000000
20    label_uniq_ids, label_row_ids = np.unique(all_labels,
21        return_inverse=True) #ke integer
22    label_row_ids = label_row_ids.astype(np.int32, copy=False)
23    onehot_labels = to_categorical(label_row_ids, len(
24        label_uniq_ids)) #ke one hot
25
26    return np.stack(all_features), onehot_labels
```

Kode di atas adalah mendefinisian nama fungsi yaitu generate features and labels kemudian membuat variabel baru dengan array kosong yaitu all_features dan all_labels kemudian mendefinisikan isian label untuk genre dengan cara membuat variabel genres kemudian di isi dengan 10 genre yang tadi setelah itu dilakukan fungsi if else dengan code for dan in setelah itu akan di buat encoding untuk data tiap tiap label contoh untuk blues 10000000000 dan untuk clasical 01000000000. Hasilnya adalah sebagai berikut :

```
In [8]: def generate_features_and_labels():
...:     all_features = []
...:     all_labels = []
...:     genres = [ 'blues' , 'classical' , 'country' , 'disco' , '
...:         hiphop' , 'jazz' , 'metal' , 'pop' , 'reggae' , 'rock' ]
...:     for genre in genres:
...:         sound_files = glob.glob('genres/' + genre + '/*.au')
...:         for f in sound_files:
...:             features = extract_features_song(f)
...:             all_features.append(features)
...:             all_labels.append(genre)
...: 
...:     # convert labels to one-hot encoding cth blues :
...:     # 10000000000 classic 01000000000
...:     label_uniq_ids, label_row_ids = np.unique(all_labels,
...:         return_inverse=True) #ke integer
...:     label_row_ids = label_row_ids.astype(np.int32, copy=False)
...:     onehot_labels = to_categorical(label_row_ids, len(
...:         label_uniq_ids)) #ke one hot
...: 
...:     return np.stack(all_features), onehot_labels
```

Gambar 6.111 Hasil Soal 4.

5. Soal 5

```

1 # In [5]: Soal Nomor 5
2 features , labels = generate_features_and_labels()
3 # In [5]: Soal Nomor 5
4 print(np.shape(features))
5 print(np.shape(labels))

```

Hal ini menjadi lama dikarenakan mesin membaca satupersatu file yang ada pada folder dan dalam folder tersebut terdapat 100 file sehingga wajar menjadi lama ditambah lagi mengolah data yang tadinya suara menjadi bentuk vektor. Hasilnya adalah sebagai berikut :

```

In [11]: features, labels = generate_features_and_labels()
Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal...
Processing 100 songs in pop...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...
In [12]: print(np.shape(features))
...: print(np.shape(labels))
(1000, 25000)
(1000, 10)

```

Name	Type	Size	Value
features	Float64	(1000, 25000)	[1.0, 0.1599871, -0.03814985, -0.7516449, ... -0.0181594, 0.0127245]
labels	Float32	(1000, 10)	[1.0, 0.0, ..., 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

Gambar 6.112 Hasil Soal 5.

6. Soal 6

```

1 # In [6]: Soal Nomor 6
2 training_split = 0.8
3 # In [6]: Soal Nomor 6
4 alldata = np.column_stack((features , labels))
5 # In [6]: Soal Nomor 6
6 np.random.shuffle(alldata)
7 splitidx = int(len(alldata) * training_split)
8 train , test = alldata [:splitidx ,:] , alldata [splitidx :,:]
9 # In [6]: Soal Nomor 6
10 print(np.shape(train))
11 print(np.shape(test))
12 # In [6]: Soal Nomor 6
13 train_input = train [:,:-10]
14 train_labels = train [:,-10:]
15 # In [6]: Soal Nomor 6
16 test_input = test [:,:-10]
17 test_labels = test [:,-10:]
18 # In [6]: Soal Nomor 6
19 print(np.shape(train_input))
20 print(np.shape(train_labels))

```

Kode diatas berfungsi untuk melakukan training split 80%. Karena supaya mesin dapat terus belajar tentang data baru, jadi ketika prediksi dibuat tentang data yang terlatih itu bisa mendapatkan persentase yang cukup bagus. Hasilnya adalah sebagai berikut :

```
In [13]: training_split = 0.8
```

Gambar 6.113 Hasil Soal 6.

7. Soal 7

```
1 # In [7]: Soal Nomor 7
2 model = Sequential([
3     Dense(100, input_dim=np.shape(train_input)[1]),
4     Activation('relu'),
5     Dense(10),
6     Activation('softmax'),
7 ])
```

Fungsi Sequential() adalah sebuah model untuk menentukan izin pada setiap neuron, di sini adalah 100 dense yang merupakan 100 neuron pertama dari data pelatihan. Fungsi dari relay itu sendiri adalah untuk mengaktifkan neuron atau input yang memiliki nilai maksimum. Sedangkan untuk dense 10 itu adalah output dari hasil neuron yang telah berhasil diaktifkan, untuk dense 10 diaktifkan menggunakan softmax. Hasilnya adalah sebagai berikut :

```
In [20]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])
```

Gambar 6.114 Hasil Soal 7.

8. Soal 8

```
1 # In [8]: Soal Nomor 8
2 model.compile(optimizer='adam',
3                 loss='categorical_crossentropy',
4                 metrics=['accuracy'])
5 print(model.summary())
```

Model Compile di perjelas dengan gambar dibawah, Hasil output pada kode tersebut seperti gambar menjelaskan bahwa dense pertama itu memiliki 100 neurons dengan parameter sekitar 2 juta lebih dengan aktivasinya 100, jadi untuk setiap neurons memiliki masing-masing 1 aktivasi. Sama halnya seperti dense 2 memiliki jumlah neurons sebanyak 10 dengan parameter 1010 dan jumlah aktivasinya 10 untuk setiap neurons tersebut dan total parameternya sekitar 2.5 juta data yang akan dilatih pada mesin tersebut.

```
In [21]: model.compile(optimizer='adam',
    ...,
    loss='categorical_crossentropy',
    ...,
    metrics=['accuracy'])
...; print(model.summary())
Model: "sequential_1"
-----
```

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 100)	2900100
activation_3 (Activation)	(None, 100)	0
dense_2 (Dense)	(None, 10)	1010
activation_2 (Activation)	(None, 10)	0

```
-----  
Total params: 2,901,110  
Trainable params: 2,901,110  
Non-trainable params: 0  
-----
```

Gambar 6.115 Hasil Soal 8.

9. Soal 9

```
1 # In [9]: Soal Nomor 9
2 model.fit(train_input, train_labels, epochs=10, batch_size=
3           =32,
4           validation_split=0.2)
```

Kode tersebut berfungsi untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch_size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap 1 kali pengulangan. Sedangkan validation_split itu untuk menentukan presentase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan, hasilnya adalah sebagai berikut :

```
In [21]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
    ...,
    validation_split=0.2)
Epoch 1/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.7901 - val_accuracy: 0.4000
Epoch 2/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.6794 - val_accuracy: 0.4180
Epoch 3/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.5893 - val_accuracy: 0.4361
Epoch 4/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.4977 - val_accuracy: 0.4457
Epoch 5/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.4061 - val_accuracy: 0.4553
Epoch 6/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.3145 - val_accuracy: 0.4640
Epoch 7/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.2229 - val_accuracy: 0.4731
Epoch 8/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.1313 - val_accuracy: 0.4818
Epoch 9/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
1.0397 - val_accuracy: 0.4906
Epoch 10/10
1/100 [=====] - 1s/step - loss: 2.3432 - accuracy: 0.2391 - val_loss:
0.9481 - val_accuracy: 0.5023
100/100 [=====] - 1s/step - loss: 0.8364 - accuracy: 0.9424 - val_loss:
0.8475 - val_accuracy: 0.9532
200/200 [=====] - 1s/step - loss: 0.7368 - accuracy: 0.9530 - val_loss:
0.7461 - val_accuracy: 0.9528
300/300 [=====] - 1s/step - loss: 0.6361 - accuracy: 0.9532 - val_loss:
0.6453 - val_accuracy: 0.9528
400/400 [=====] - 1s/step - loss: 0.5354 - accuracy: 0.9534 - val_loss:
0.5445 - val_accuracy: 0.9528
500/500 [=====] - 1s/step - loss: 0.4347 - accuracy: 0.9536 - val_loss:
0.4438 - val_accuracy: 0.9528
600/600 [=====] - 1s/step - loss: 0.3340 - accuracy: 0.9538 - val_loss:
0.3431 - val_accuracy: 0.9528
700/700 [=====] - 1s/step - loss: 0.2333 - accuracy: 0.9540 - val_loss:
0.2424 - val_accuracy: 0.9528
800/800 [=====] - 1s/step - loss: 0.1326 - accuracy: 0.9542 - val_loss:
0.1417 - val_accuracy: 0.9528
900/900 [=====] - 1s/step - loss: 0.0319 - accuracy: 0.9544 - val_loss:
0.0410 - val_accuracy: 0.9528
1000/1000 [=====] - 1s/step - loss: 0.0312 - accuracy: 0.9545 - val_loss:
0.0403 - val_accuracy: 0.9528
100/100 [=====] - 1s/step - loss: 0.0312 - accuracy: 0.9545 - val_loss:
0.0403 - val_accuracy: 0.9528
val_loss: 0.0403 - val_accuracy: 0.9528
```

Gambar 6.116 Hasil Soal 9.

10. Soal 10

```
1 # In [10]: Soal Nomor 10
2 loss, acc = model.evaluate(test_input, test_labels,
3                             batch_size=32)
4 # In [10]: Soal Nomor 10
5 print("Done!")
5 print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

Fungsi evaluate atau evaluasi ini ialah untuk menguji data pengujian setiap file. Di sini ada prediksi yang hilang, artinya mesin memprediksi

data, sedangkan untuk keseluruhan perjanjian sekitar 55%, hasilnya adalah sebagai berikut :

```
In [23]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
200/200 [=====] - 0s 656us/step
```

Gambar 6.117 Hasil Soal 10.

11. Soal 11

```
1 # In [11]: Soal Nomor 11
2 model.predict(test_input [:1])
```

Fungsi Predict ialah untuk menghasilkan suatu nilai yang sudah di prediksi dari data training sebelumnya. Gambar dibawah ini menjelaskan file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre rock. Maka lagu tersebut termasuk ke dalam genre rock dengan perbandingan presentase hasil prediksi.

```
In [25]: model.predict(test_input[:1])
Out[25]:
array([1.2146345e-04, 5.0897301e-06, 5.1087093e-01, 4.7763154e-01,
       1.4319259e-03, 1.4710484e-05, 1.7895336e-05, 9.3976054e-03,
       2.2317965e-05, 9.7474178e-05], dtype=float32)
```

Gambar 6.118 Hasil Soal 11.

6.7.3 Penanganan Error

1. ScreenShoot Error

```
FileNotFoundError: [Errno 2] No such file or directory: 'E:\Koleksi\dataset\musik\dataset\genres\blues\blues.00001.wv'
```

Gambar 6.119 FileNotFoundError

2. Cara Penanganan Error

- **FileNotFoundException**

Error terdapat pada letak file yang tidak terbaca, karena letak file berbeda dengan pemanggilannya, solusi nya ialah dengan meletakkan direktori file yang dibaca dengan benar.

6.7.4 Bukti Tidak Plagiat

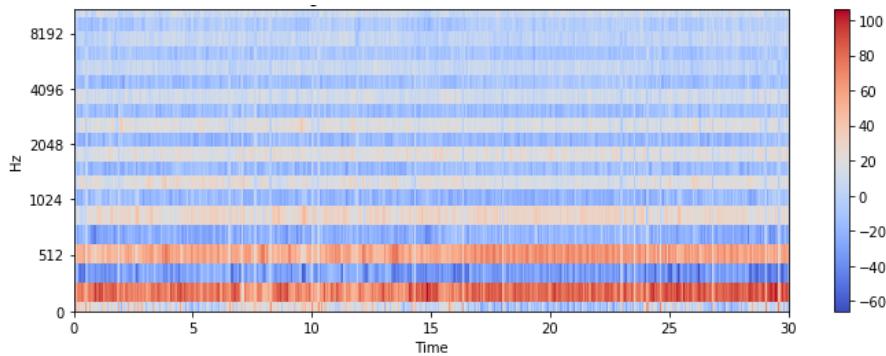


Gambar 6.120 Bukti Tidak Melakukan Plagiat Chapter 6

6.8 Muhammad Reza Syachrani - 1174084

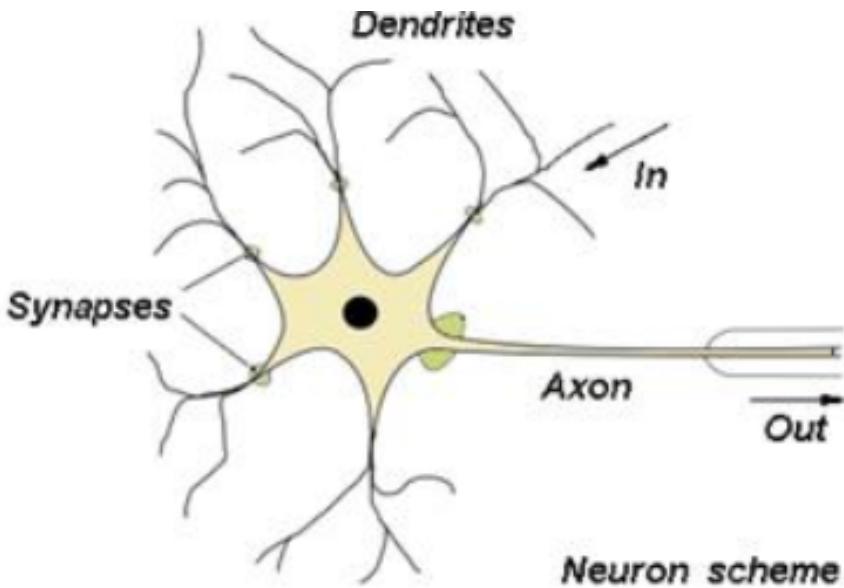
6.8.1 Teori

1. Jelaskan kenapa file suara harus di lakukan MFCC. dilengkapi dengan ilustrasi atau gambar
digunakan untuk mendapatkan nilai dari vektor suara tersebut yang berguna untuk mendapatkan suatu parameter dan informasi mengenai ciri dari suara.



Gambar 6.121 Teori 1

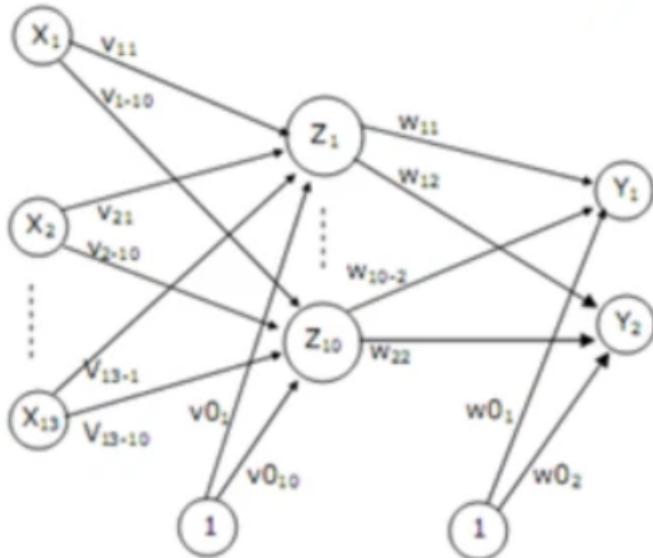
2. Jelaskan konsep dasar neural network.dilengkapi dengan ilustrasi atau gambar.
Konsep dasar neural network sendiri di mulai dari ide dasar neural network otak manusia, dimana otak terdiri dari neuron, neuron ini berfungsi memproses setiap informasi yang masuk. Satu neuron memiliki satu akson dan minimal satu dendrit. sehingga konsep dasar ini membangun neural network buatan yang disebut (Artificial Neural Network).



Gambar 6.122 Teori 2

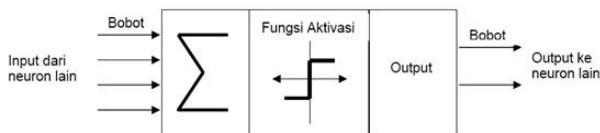
3. Jelaskan konsep pembobotan dalam neural network.dilengkapi dengan ilustrasi atau gambar
Konsep pembobotan dalam neural network menggunakan algoritma back-propagation untuk memperkecil tingkat eror dengan menyesuaikan nilai bobot berdasarkan perbedaan output serta target yang diinginkan.

Input Layer Hidden Layer Output Layer



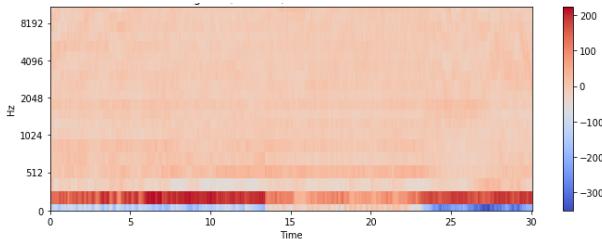
Gambar 6.123 Teori 3

4. Jelaskan konsep fungsi aktifasi dalam neural network. dilengkapi dengan ilustrasi atau gambar
merupakan fungsi matematis yang digunakan untuk mendapatkan output neuron dari nilai inputnya. Disebut suatu aktivasi karena output akan bernilai jika dapat melampaui nilai threshold-nya.



Gambar 6.124 Teori 4

5. Jelaskan cara membaca hasil plot dari MFCC,dilengkapi dengan ilustrasi atau gambar.
Dapat dilakukan dengan sampel suara pengguna diubah dalam bentuk file *.wav yang digunakan sebagai inputan kemudian direpresentasikan menjadi sinyal suara dalam bentuk matriks dengan menggunakan perintah audioread di Matlab R2017a. hasil sampel suara pada gambar dibawah.



Gambar 6.125 Teori 5

6. Jelaskan apa itu one-hot encoding,dilengkapi dengan ilustrasi kode dan atau gambar.

proses dimana variabel kategorikal dikonversi menjadi bentuk yang dapat disediakan untuk algoritma ML untuk melakukan pekerjaan yang lebih baik dalam prediksi.

	CompanyName	Categoricalvalue	Price
1			
2	VW	1	20
3	Acura	2	10011
4	Honda	3	50000
5	Honda	3	10000
6			
7			
8			

	VW	Acura	Honda	Price
1	1	0	0	20000
2	0	1	0	10011
3	0	0	1	50000
4	0	0	1	10000
5				
6				
7				
8				

Gambar 6.126 Teori 6

7. Jelaskan apa fungsi dari np.unique dan to_categorical dalam kode program,dilengkapi dengan ilustrasi atau gambar.

np.unique digunakan untuk membuat array, sedangkan to_categorical digunakan untuk mengubah vektor kelas yang berupa integer (number) menjadi matrix kelas biner.

```
In [85]: a = np.array([4,3,4,4,3,2,2,1,1,6,6,6,3,3,2,2,0,0])
...: np.unique(a)
Out[85]: array([0, 1, 2, 3, 4, 6])
```

Gambar 6.127 Teori 7

```
In [90]: a = np.array([4,3,4,4,3,2,2,1,1,6,6,6,3,3,2,2,0,0])
...: to_categorical(np.unique(a))
Out[90]:
array([[1., 0., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 0., 1., 0.]]], dtype=float32)
```

Gambar 6.128 Teori 7

8. Jelaskan apa fungsi dari Sequential dalam kode program,dilengkapi dengan ilustrasi atau gambar.

Sequential berfungsi sebagai tumpukan linear lapisan.

```
from keras.models import Sequential
from keras.layers import Dense, Activation

model = Sequential([
    Dense(32, input_shape=(784,)),
    Activation('relu'),
    Dense(10),
    Activation('softmax'),
])
```

Gambar 6.129 Teori 8

6.8.2 Praktek

1. Jelaskan isi dari data GTZAN Genre Collection dan data dari freesound. GTZAN Genre Collection berisikan klasifikasi genre musik. terdapat 10 genre musik didalamnya yaitu blues, clasisscal, country, disco, hiphop, jazz, metal, pop, reggae, dan rock. Dalam setiap genre tersebut berisikan 100 tracks musik.

```
1 import librosa #import librosa digunakan untuk fungsi mfcc
2 import librosa.feature #import librosa feattuse
3 import librosa.display #import librosa display
4 import glob #import glob
5 import numpy as np #import numpy untuk pengolahan data
       menjadi vektor sebagai np
6 import matplotlib.pyplot as plt #import matplotlib untuk
       melakukan plotting sebagai plt
7 from keras.models import Sequential #import sequential dari
       library keras
8 from keras.layers import Dense, Activation #import dense dan
       activation dari library keras
9 from keras.utils import np_utils #import
       to_categorical dari library keras
10
```

```

11 # In [1]: membuat fungsi mfcc untuk melakukan pengujian
12 def display_mfcc(song): #membuat fungsi display_mfcc
13     y, _ = librosa.load(song) # variabel Y yang berisi method
14         librosa load
15     mfcc = librosa.feature.mfcc(y) # method librosa featurea
16         mfcc
17
18     plt.figure(figsize=(10, 4)) #membuat ot gure dengan
19         ukuran 10 banding 4
20     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
21         ) #data librosa display dengan variabel x nya yaitu waktu
22         dan y yaitu mel atau Hz
23     plt.colorbar() #plot bar warna
24     plt.title(song) #judul plot
25     plt.tight_layout()
26     plt.show() #plot di tampilkan

```

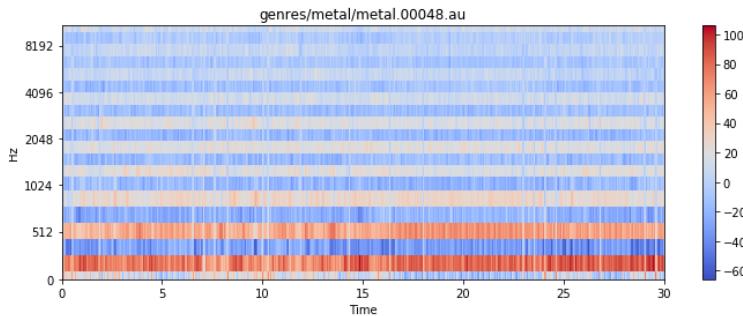
2. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari display_mfcc()

Ini spectogram dari lagu genre metal (source dari GTZAN Genre Collection).

```

1 # In [2]: cek fungsi
2 display_mfcc('genres/metal/metal.00048.au') #mendisplay
3         tampilan glombang suara

```



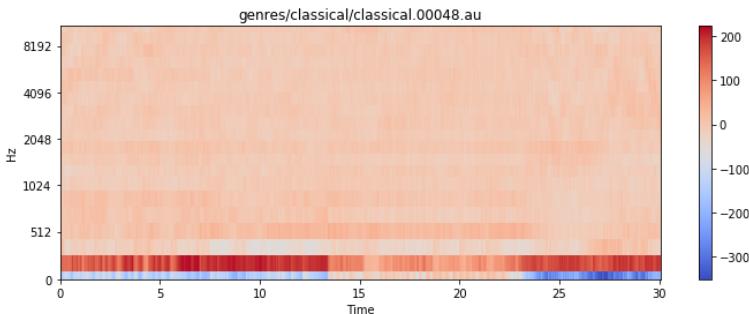
Gambar 6.130 spectrogram dari lagu genre metal

Ini spectogram dari lagu genre classical (source dari GTZAN Genre Collection).

```

1 # In [2]: cek fungsi
2 display_mfcc('genres/classical/classical.00048.au') #
3         mendisplay tampilan glombang suara

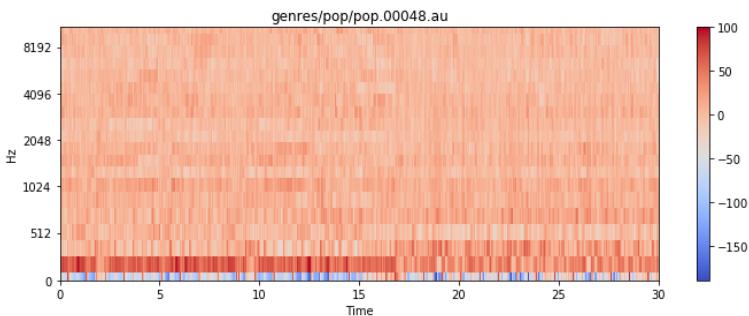
```



Gambar 6.131 spectogram dari lagu genre classical

Ini spectogram dari lagu genre pop (source dari GTZAN Genre Collection).

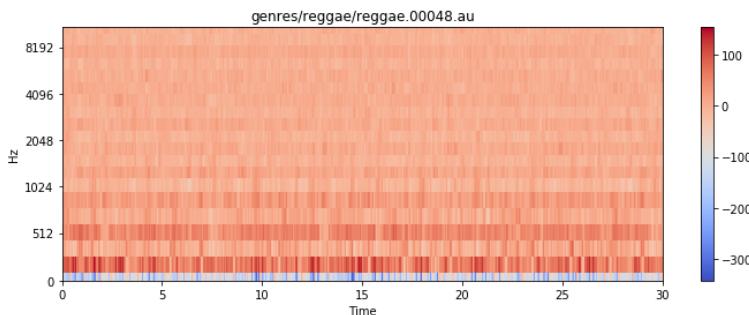
```
1 # In [2]: cek fungsi
2 display_mfcc('genres/pop/pop.00048.au') #mendisplay tampilan
glombang suara
```



Gambar 6.132 spectogram dari lagu genre pop

Ini spectogram dari lagu genre reggae (source dari GTZAN Genre Collection).

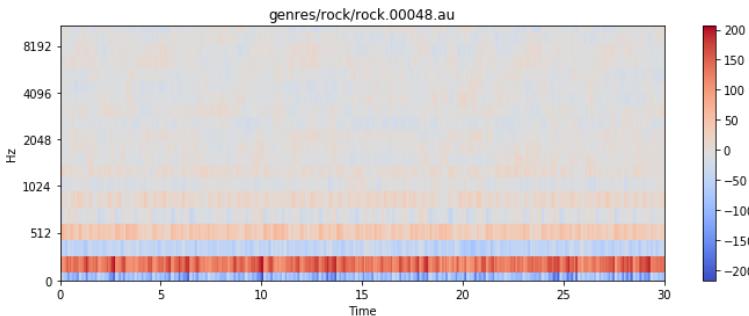
```
1 # In [2]: cek fungsi
2 display_mfcc('genres/reggae/reggae.00048.au') #mendisplay
tampilan glombang suara
```



Gambar 6.133 spectogram dari lagu genre reggae

Ini spectrogram dari lagu genre rock (source dari GTZAN Genre Collection).

```
1 # In [2]: cek fungsi
2 display_mfcc ('genres/rock/rock.00048.au') #mendisplay
    tampilan glombang suara
```



Gambar 6.134 spectogram dari lagu genre rock

3. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari extract features song()
Mengapa mengambil 25000 row pertama? dikarenakan Audio yang terdapat pada dataset ini tidak menentu durasinya. Dan kita harus mengambil data yang memiliki durasi yang sama untuk mempermudah dalam melakukan training.

```
1 # In [3]:
2 def extract_features_song(f): #nama extract features song
    yang nantinya akan di gunakan pada fungsi yang lainnya
3     y, _ = librosa.load(f) #membuat variabel y dengan method
        librosa load
4
```

```

5     mfcc = librosa.feature.mfcc(y) #membuat variabel baru
6     mfcc dengan isi librosa features mfcc dengan isi variabel
7     y
8
9     mfcc /= np.amax(np.absolute(mfcc)) #variabel mfcc dengan
10    isian np.max
11
12    return np.ndarray.flatten(mfcc)[:25000] #membuat array
13    dari data tersebut merupakan data 25000 data pertama

```

```

In [98]: def extract_features_song(f): #nama extract features song yang nantinya akan
di gunakan pada fungsi yang lainnya
...     y, _ = librosa.load(f) #membuat variabel y dengan method Librosa Load
...     ...
...     mfcc = librosa.feature.mfcc(y) #membuat variabel baru mfcc dengan isi
Librosa features mfcc dengan isi variabel y
...     ...
...     mfcc /= np.amax(np.absolute(mfcc)) #variabel mfcc dengan isian np.max
...     ...
...     return np.ndarray.flatten(mfcc)[:25000] #membuat array dari data
tersebut merupakan data 25000 data pertama

```

Gambar 6.135 Praktek 3

4. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari generate features and labels().

```

1 # In [4]:
2 def generate_features_and_labels(): #definisi nama fungsi
3     yaitu generate features and labels
4     all_features = [] #membuat variabel baru dengan array
5     kosong yaitu all\features
6     all_labels = [] ##membuat variabel baru dengan array
7     kosong yaitu all\label
8
9     genres = [ 'blues' , 'classical' , 'country' , 'disco' , '
10    hip hop' , 'jazz' , 'metal' , 'pop' , 'reggae' , 'rock' ] #
11    mendefinisikan isian label untuk genres dengan cara
12    membuat variabel genres kemudian di isi dengan 10 genre
13    for genre in genres:
14        sound_files = glob.glob('genres/' + genre + '/*.au')
15        print('Processing %d songs in %s genre...' % (len(
16            sound_files), genre))
17        for f in sound_files:
18            features = extract_features_song(f)
19            all_features.append(features)
20            all_labels.append(genre)
21
22    # convert labels to one-hot encoding cth blues :
23    1000000000 classic 0100000000
24    label_uniq_ids , label_row_ids = np.unique(all_labels ,
25    return_inverse=True)#ke integer
26    label_row_ids = label_row_ids.astype(np.int32 , copy=False
27    )
28    onehot_labels = to_categorical(label_row_ids , len(
29    label_uniq_ids))#ke one hot
30
31    return np.stack(all_features) , onehot_labels

```

```
In [99]: def generate_features_and_labels(): #definisiakan nama fungsi yaitu generate features and
labels
...:
    all_features = [] ##membuat variabel baru dengan array kosong yaitu all_features
    ...:
    all_labels = [] ##membuat variabel baru dengan array kosong yaitu all_label
    ...:
    genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop',
'reggae', 'rock'] ##mendefinisikan isian label untuk genres dengan cara membuat variabel genres
kemudian di isi dengan 10 genre
    ...:
    for genre in genres:
        sound_files = glob.glob('genres/'+genre+'/*.au')
        print('Processing %d songs in %s genre...' % (len(sound_files), genre))
        ...:
        for f in sound_files:
            features = extract_features_song(f)
            all_features.append(features)
            all_labels.append(genre)
        ...:
        # convert Labels to one-hot encoding cth blues : 1000000000 classic 0100000000
    ...:
    label_uniq_ids, label_row_ids = np.unique(all_labels, return_inverse=True)## integer
    label_row_ids = label_row_ids.astype(np.int32, copy=False)
    onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids))##be one hot
    ...:
    return np.stack(all_features), onehot_labels
```

Gambar 6.136 Praktek 4

- Jelaskan dengan kata dan praktek kenapa penggunaan fungsi generate features and labels() sangat lama ketika meload dataset genre.
Karena mesin membaca file satu persatu yang ada pada folder dan dalam folder tersebut dan setiap folder terdapat 100 file sehingga menjadi lama dan mengolah data yang tadinya suara menjadi bentuk vektor.

```
1 # In [5]: passing parameter dari fitur ekstraksi menggunakan
    mfcc
2 features , labels = generate_features_and_labels()
```

```
In [100]: features, labels = generate_features_and_labels()
Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...

In [101]: print(np.shape(features))
...: print(np.shape(labels))
(1000, 25000)
(1000, 10)
```

Gambar 6.137 Praktek 5

- Jelaskan kenapa harus dilakukan pemisahan data training dan data set sebesar 80 persen? Praktekkan dengan kode dan Tunjukkan keluarannya
Melakukan training split 80% supaya mesin dapat terus belajar tentang data baru, jadi ketika prediksi dibuat tentang data yang terlatih itu bisa mendapatkan persentase yang cukup bagus.

```
1 training_split = 0.8
2
3 # In []
4 # last column has genre, turn it into unique ids
```

```

5 alldata = np.column_stack((features, labels))
6
7 # In []
8 np.random.shuffle(alldata)
9 splitidx = int(len(alldata) * training_split)
10 train, test = alldata[:splitidx,:], alldata[splitidx,:,:]
11
12 # In []
13 print(np.shape(train))
14 print(np.shape(test))
15
16 # In []
17 train_input = train[:, :-10]
18 train_labels = train[:, -10:]
19
20 test_input = test[:, :-10]
21 test_labels = test[:, -10:]
22
23 # In []
24 print(np.shape(train_input))
25 print(np.shape(train_labels))

```

In [102]: `training_split = 0.8`

In [103]: `alldata = np.column_stack((features, labels))`

In [104]: `np.random.shuffle(alldata)`
`....: splitidx = int(len(alldata) * training_split)`
`....: train, test = alldata[:splitidx,:], alldata[splitidx,:,:]`

In [105]: `print(np.shape(train))`
`....: print(np.shape(test))`
`(800, 25010)`
`(200, 25010)`

In [106]: `train_input = train[:, :-10]`
`....: train_labels = train[:, -10:]`
`....:`
`....: test_input = test[:, :-10]`
`....: test_labels = test[:, -10:]`

In [107]: `print(np.shape(train_input))`
`....: print(np.shape(train_labels))`
`(800, 25000)`
`(800, 10)`

Gambar 6.138 Praktek 6

- Praktekkan dan jelaskan masing-masing parameter dari fungsi Sequential().

Fungsi Sequential() ialah sebuah model untuk menentukan izin pada setiap neuron, di sini adalah 100 dense yang merupakan 100 neuron pertama dari data pelatihan. Fungsi dari relay itu sendiri adalah untuk mengaktifkan neuron atau input yang memiliki nilai maksimum. Sedangkan untuk dense 10 itu adalah output dari hasil neuron yang telah berhasil diaktifkan, untuk dense 10 diaktifkan menggunakan softmax.

```

1 model = Sequential([
2     Dense(100, input_dim=np.shape(train_input)[1]),
3     Activation('relu'),
4     Dense(10),
5     Activation('softmax'),
6 ])

```

```

In [108]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])

```

Gambar 6.139 Praktek 7

8. Praktekkan dan jelaskan masing-masing parameter dari fungsi compile().

Model Compile dijelaskan dengan gambar dibawah, Hasil output pada kode tersebut seperti gambar menjelaskan bahwa dense pertama itu memiliki 100 neurons dengan parameter sekitar 2 juta lebih dengan aktivasi 100, jadi untuk setiap neurons memiliki masing-masing 1 aktivasi. Sama halnya seperti dense 2 memiliki jumlah neurons sebanyak 10 dengan parameter 1010 dan jumlah aktivasinya 10 untuk setiap neurons tersebut dan total parameternya sekitar 2.5 juta data yang akan dilatih pada mesin tersebut.

```

1 model.compile(optimizer='adam',
2                 loss='categorical_crossentropy',
3                 metrics=['accuracy'])
4 print(model.summary())

```

```
In [109]: model.compile(optimizer='adam',
    ...
        loss='categorical_crossentropy',
    ...
        metrics=['accuracy'])
    ...: print(model.summary())
Model: "sequential_3"

```

Layer (type)	Output Shape	Param #
dense_5 (Dense)	(None, 100)	2500100
activation_5 (Activation)	(None, 100)	0
dense_6 (Dense)	(None, 10)	1010
activation_6 (Activation)	(None, 10)	0

```
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0
```

```
None
```

Gambar 6.140 Praktek 8

9. Praktekkan dan jelaskan masing-masing parameter dari fungsi fit(). Kode tersebut berfungsi untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch_size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap 1 kali pengulangan. Sedangkan validation_split itu untuk menentukan presentase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan.

```
1 model.fit(train_input, train_labels, epochs=10, batch_size
2 =32,
3 validation_split=0.2)
```

```
In [110]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
...
Train on 640 samples, validate on 160 samples
Epoch 1/10
640/640 [=====] - 1s 2.163s/step - loss: 2.1635 - accuracy: 0.3109 - val_loss: 1.8495 - val_accuracy: 0.3375
Epoch 2/10
640/640 [=====] - 1s 891us/step - loss: 1.3906 - accuracy: 0.5125 - val_loss: 1.6384 - val_accuracy: 0.4313
Epoch 3/10
640/640 [=====] - 1s 890us/step - loss: 1.0521 - accuracy: 0.6486 - val_loss: 1.5986 - val_accuracy: 0.4375
Epoch 4/10
640/640 [=====] - 1s 893us/step - loss: 0.8351 - accuracy: 0.7328 - val_loss: 1.5962 - val_accuracy: 0.4563
Epoch 5/10
640/640 [=====] - 1s 890us/step - loss: 0.6233 - accuracy: 0.8313 - val_loss: 1.4628 - val_accuracy: 0.5125
Epoch 6/10
640/640 [=====] - 1s 888us/step - loss: 0.5007 - accuracy: 0.8844 - val_loss: 1.5120 - val_accuracy: 0.4812
Epoch 7/10
640/640 [=====] - 1s 898us/step - loss: 0.3688 - accuracy: 0.9375 - val_loss: 1.5988 - val_accuracy: 0.5000
Epoch 8/10
640/640 [=====] - 1s 899us/step - loss: 0.2823 - accuracy: 0.9594 - val_loss: 1.5113 - val_accuracy: 0.4437
Epoch 9/10
640/640 [=====] - 1s 919us/step - loss: 0.2226 - accuracy: 0.9828 - val_loss: 1.5454 - val_accuracy: 0.5312
Epoch 10/10
640/640 [=====] - 1s 923us/step - loss: 0.1822 - accuracy: 0.9891 - val_loss: 1.5895 - val_accuracy: 0.4625
0x110]: <keras.callbacks.Callback>.History at 0x1f7584d2f00
```

Gambar 6.141 Praktek 9

10. Praktekkan dan jelaskan masing-masing parameter dari fungsi evaluate(). Fungsi evaluate atau evaluasi ini ialah untuk menguji data pengujian setiap file. Di sini ada data yang hilang sebesar 1.6111 data, sedangkan untuk akurasinya sekitar 51%.

```

1 loss , acc = model.evaluate(test_input , test_labels ,
2   batch_size=32)
3
4 print("Done!")
5 print("Loss: %.4f , accuracy: %.4f" % (loss , acc))

```

```

In [122]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
...:
...:
...: print("Done!")
...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
200/200 [=====] - 0s 174us/step
Done!
Loss: 1.6111, accuracy: 0.5150

```

Gambar 6.142 Praktek 10

11. Praktekkan dan jelaskan masing-masing parameter dari fungsi predict(). Fungsi Predict ialah untuk menghasilkan suatu nilai yang sudah di prediksi dari data training sebelumnya. Gambar dibawah ini menjelaskan file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre classical. Maka lagu tersebut termasuk ke dalam genre classical dengan perbandingan presentase hasil prediksi.

```

1 model.predict(test_input[:1])

```

```

In [115]: model.predict(test_input[:1])
Out[115]:
array([[1.7469719e-03, 9.4805568e-01, 8.8974694e-04, 4.0234820e-04,
       6.9074599e-07, 4.7569752e-02, 6.0497349e-09, 1.8554716e-04,
       1.0829815e-03, 6.6229710e-05]], dtype=float32)

```

Gambar 6.143 Praktek 11

6.8.3 Bukti Tidak Plagiat



Gambar 6.144 plagiarism

6.8.4 Link Video Youtube

<https://youtu.be/MnrV7s-Tdy0>

BAB 7

CHAPTER 7

7.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 7.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

7.1.1 Teori

7.1.2 Praktek

7.1.3 Penanganan Error

7.1.4 Bukti Tidak Plagiat



Gambar 7.2 Kecerdasan Buatan.