

# **CERDAS MENGUASAI GIT**



---

# CERDAS MENGUASAI GIT

## Dalam 24 Jam

---

**Rolly M. Awangga**  
Informatics Research Center



**Kreatif Industri Nusantara**

***Penulis:***

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

***Editor:***

M. Yusril Helmi Setyawan

***Penyunting:***

Syafrial Fachrie Pane

Khaera Tunnisa

Diana Asri Wijayanti

***Desain sampul dan Tata letak:***

Deza Martha Akbar

***Penerbit:***

Kreatif Industri Nusantara

***Redaksi:***

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

***Distributor:***

Informatics Research Center

Jl. Sariasih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara  
apapun tanpa ijin tertulis dari penerbit

*‘Jika Kamu tidak dapat  
menahan lelahnya  
belajar, Maka kamu harus  
sanggup menahan  
perihnya Kebodohan.’  
Imam Syafi’i*

# CONTRIBUTORS

---

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indonesia, Bandung, Indonesia



# CONTENTS IN BRIEF

---

<b>1 Chapter 1</b>	<b>1</b>
<b>2 Chapter 2</b>	<b>3</b>
<b>3 Chapter 3</b>	<b>5</b>
<b>4 Chapter 4</b>	<b>7</b>
<b>5 Chapter 5</b>	<b>9</b>
<b>6 Chapter 6</b>	<b>11</b>
<b>7 Chapter 7</b>	<b>23</b>





# DAFTAR ISI

---

Foreword	xi
Kata Pengantar	xiii
Acknowledgments	xv
Acronyms	xvii
Glossary	xix
List of Symbols	xxi
Introduction	xxiii
<i>Rolly Maulana Awangga, S.T., M.T.</i>	
<b>1 Chapter 1</b>	<b>1</b>
<b>2 Chapter 2</b>	<b>3</b>
<b>3 Chapter 3</b>	<b>5</b>
<b>4 Chapter 4</b>	<b>7</b>
	<b>ix</b>

<b>5</b>	<b>Chapter 5</b>	<b>9</b>
<b>6</b>	<b>Chapter 6</b>	<b>11</b>
6.1	Damara Benedikta/ 1174012	11
6.1.1	Teori	11
6.1.2	Praktikum	14
<b>7</b>	<b>Chapter 7</b>	<b>23</b>
7.1	1174027 - Harun Ar - Rasyid	23
7.1.1	Teori	23
7.1.2	Praktek	28
7.1.3	Penanganan Error	36
7.1.4	Bukti Tidak Plagiat	37
7.2	1164013 - Ikrima Ningrumsari Mulyana	37
7.2.1	Teori	37
7.2.2	Praktek	42
7.2.3	Penanganan Error	50
7.2.4	Bukti Tidak Plagiat	51
	Daftar Pustaka	53
	Index	55

# FOREWORD

---

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa



# KATA PENGANTAR

---

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

*Bandung, Jawa Barat*  
*Februari, 2019*



# ACKNOWLEDGMENTS

---

Terima kasih atas semua masukan dari para mahasiswa agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk team IRC yang telah fokus untuk belajar dan memahami bagaimana buku ini mendampingi proses Intership.

R. M. A.





# ACRONYMS

---

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association



# GLOSSARY

---

git	Merupakan manajemen sumber kode yang dibuat oleh linus torvald.
bash	Merupakan bahasa sistem operasi berbasiskan *NIX.
linux	Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald



# SYMBOLS

---

$A$  Amplitude

$\&$  Propositional logic symbol

$a$  Filter Coefficient

$\mathcal{B}$  Number of Beats



# INTRODUCTION

---

ROLLY MAULANA AWANGGA, S.T., M.T.

Informatics Research Center  
Bandung, Jawa Barat, Indonesia

Pada era disruptif saat ini. git merupakan sebuah kebutuhan dalam sebuah organisasi pengembangan perangkat lunak. Buku ini diharapkan bisa menjadi penghantar para programmer, analis, IT Operation dan Project Manajer. Dalam melakukan implementasi git pada diri dan organisasinya.

Rumusnya cuman sebagai contoh aja biar keren[1].

$$ABCDEF\alpha\beta\Gamma\Delta\sum_{def}^{abc} \tag{I.1}$$





# BAB 1

---

# CHAPTER 1

---



## BAB 2

---

## CHAPTER 2

---



## BAB 3

---

## CHAPTER 3

---



## BAB 4

---

## CHAPTER 4

---





## BAB 5

---

## CHAPTER 5

---



## BAB 6

---

## CHAPTER 6

---

### 6.1 Damara Benedikta/ 1174012

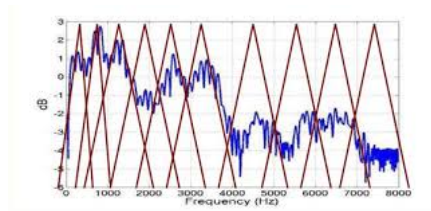
#### 6.1.1 Teori

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

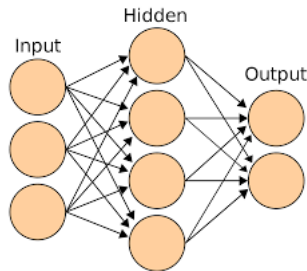
Karena MFCC digunakan untuk mengidentifikasi jenis suara misalkan jenis suara gendre lagu jes pop metal dan klasikal atau suara ultra sonic. Sehingga dibutuhkan penggunaan MFCC untuk memproses data tersebut agar dapat dibaca oleh manusia.

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

Konsep neural network itu sebenarnya mengadopsi dari kemampuan otak manusia yang mampu memberikan stimulasi/rangsangan, melakukan proses, dan memberikan output. Output diperoleh dari variasi stimulasi dan proses yang terjadi di dalam otak manusia. Neural Network



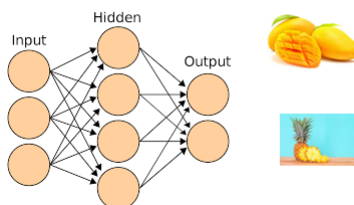
**Gambar 6.1** Ilustrasi gambar metode MFCC



**Gambar 6.2** Ilustrasi Konsep dasar neural network

3. Jelaskan konsep pembobotan dalam neural network. dilengkapidengan ilustrasi gambar.

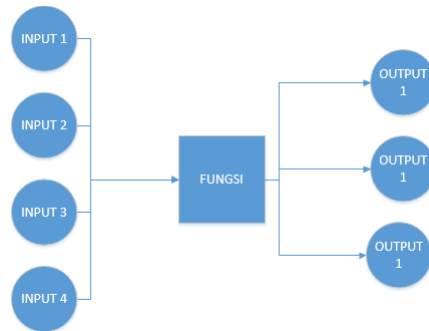
pembobotan dalam neural network yaitu digunakan untuk membedakan objek inputan atau variabel inputan untuk AI. Dimana data inputan yang masuk adalah 2 data "nanas" dan "mangga" yang diolah dengan proses membandingkan data dan diolah melalui pembobotan sehingga menampilkan hasil output.



**Gambar 6.3** Ilustrasi Konsep pembobotan pada neural network

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

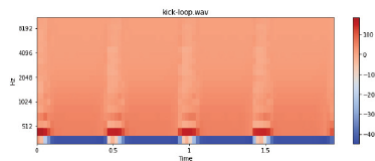
Dalam Neural Network cara aktifitas dilakukan terhadap input pada neural network inputan tersebut dimasukan kepada fungsi pada mesin sehingga di hasilkanlah output yang sesuai dengan fungsi tersebut.



**Gambar 6.4** 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.



**Gambar 6.5** 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 positif maka nilainya satu dan jika negatif maka nilainya nol.

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

fungsi dari NP.UNIQUE dalah untuk membuat data elemen menjadi nilai yang bersifat unik dalam artian (Array). Sedangkan perintah to\_categorical adalah

	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 6.6** Ilustrasi Konsep one-hot encoding

```
>>> 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** Ilustrasi np.unique

```
to_categorical

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

**Gambar 6.8** Ilustrasi to\_categorical

untuk membuat data integer yang terdeteksi untuk diubah menjadi data matrix biner.

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

fungsi dari Sequential dari code program adalah untuk membagi data - data agar dapat dianalisis oleh sistem lebih mudah, misalkan dari data 100 dibagi prosesnya menjadi 4 yaitu 25.

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

**Gambar 6.9** Ilustrasi Konsep pembobotan pada neural network

### 6.1.2 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 satuke-las).

Isi data data merupakan datasets lagu atau suara yang tersiri dari 10 gendre 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. 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 featuse 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 ploting 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 flot 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 flot 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')

```

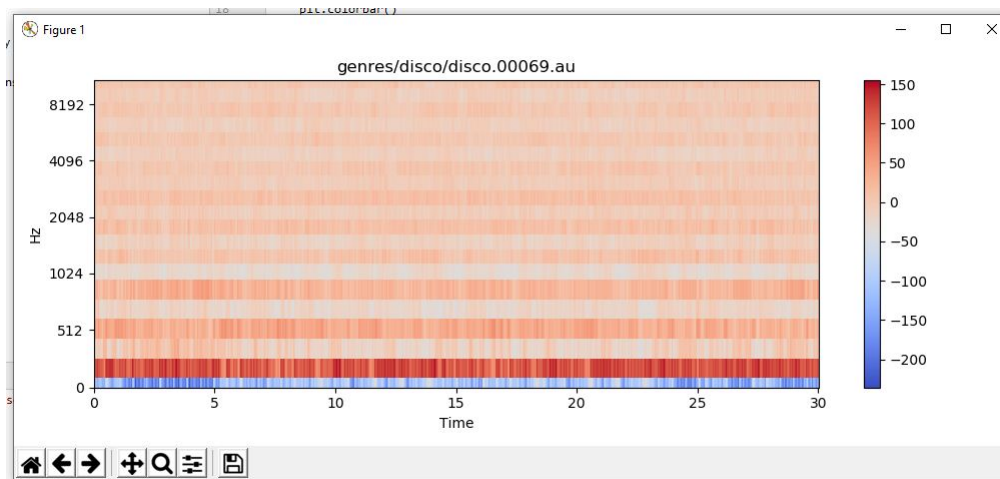


```

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 gelombang 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 peyimpanan data yang akan di tampilkan atau di eksekusi. untuk contoh hasilnya dapat dilihat pada gambar ?? berikut:



**Gambar 6.10** Ilustrasi gambar fungsi dari display\_mfcc()

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 yang diambil merupakan data 25.000 baris pertama ?

```

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.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 lainya 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 di 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']
7     for genre in genres:
8         sound_files = glob.glob('genres/'+genre+'/*.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
17     # blues : 1000000000
18     # classic 0100000000
19     label_uniq_ids, label_row_ids = np.unique(all_labels,
20        return_inverse=True)#ke integer
21     label_row_ids = label_row_ids.astype(np.int32, copy=False)
22     onehot_labels = to_categorical(label_row_ids, len(
23        label_uniq_ids))#ke one hot
24     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 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 clasiical 0100000000.

5. 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 mfcc
2 features , labels = generate_features_and_labels ()
```



**Gambar 6.11** Hasil dari fungsi generate features and labels

6. jelaskan kenapa harus dilakukan pemisahan data training dan data testing sebesar 80 persen praktekkan 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
```

7. praktekkan 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 kriteria.

```
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 ]

```

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

yaitu fungsi compile yang digunakan untuk mengetahui parameter yang digunakan dari data yang telah diolah untuk caranya dapat menggunakan codin-gan sebagai berikut, pada gambar tersebut memunculkan parameternya bera-pasaja 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())

```

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 6.12** Hasil fungsi compile

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

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 codin-gan berikut. pada gambar tersebut menunjukan 10 pengolahan data untuk menen-tukan nilai akurasi dan loss dari data tersebut dan selanjutnya dilakukan fngsi evaluasi.

```

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

```

```

Run: 1 x
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 6.13** Hasil fungsi fit

- 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 , batch_size
    =32)
3 # In[3]: fitur ekstraksi
4 print("Done!")
5 print("Loss: %.4f , accuracy : %.4f" % (loss , acc))

```

```

Run: 1 x
32/200 [===>.....] - ETA: 5s
200/200 [=====] - 1s 6ms/step
Done!
Loss: 2.3352, accuracy: 0.0800

Process finished with exit code 0
>>

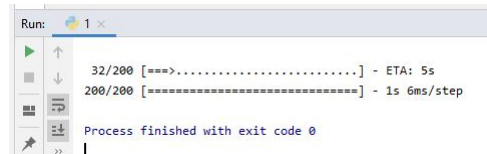
```

**Gambar 6.14** Hasil fungsi evaluasi

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

fungsi `predic` 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 6.15** Hasil fungsi prediksi



# BAB 7

---

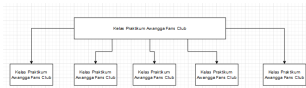
## CHAPTER 7

---

### 7.1 1174027 - Harun Ar - Rasyid

#### 7.1.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.



**Gambar 7.1** Ilustrasi Tokenizer

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.4.dilengkapi dengan ilustrasi atau gambar.

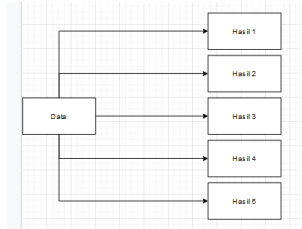
```
1 kfold = StratifiedKFold(n_splits=5)
```



```
2 splits = kfold.split(d, d['CLASS'])
3
```

### Listing 7.1 K Fold Cross Validation

Pada koding diatas terdapat variabel `kfold` yang didalamnya berisi parameter `split` yang diisikan nilai 5. hal tersebut dimaksudkan untuk membuat pengolahan data akan diulang setiap datanya sebanyak lima kali dengan atribut `class` sebagai acuan pengolahan datanya. Lalu kemudian akan di hasilkan akurasi dari pengulangan data tersebut sebesar sekian persen tergantung datanya



**Gambar 7.2** Ilustrasi K Fold Cross Validation

3. Jelaskan apa maksudnya kode program *for train, test in splits*. dilengkapi dengan ilustrasi atau gambar.

For train digunakan untuk melakukan training atau pelatihan pada data yang sudah dideklarasikan sebelumnya. Sedangkan test in split digunakan untuk membatasi jumlah data yang akan diinputkan atau data yang akan digunakan.

```
In [3]: import numpy as np
...: from sklearn.model_selection import train_test_split
...: x = np.arange(10).reshape((10, 2))
...: y = range(5)
...: print(x)
[[0 1]
 [1 2]
 [2 3]
 [3 4]
 [4 5]
 [5 6]
 [6 7]
 [7 8]
 [8 9]
 [9 0]]
In [4]: train_idx,
...: x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
...: print(x_train)
[[0 1]
 [1 2]
 [2 3]
 [3 4]
 [4 5]
 [5 6]
 [6 7]
 [7 8]
 [8 9]
 [9 0]]
```

**Gambar 7.3** Ilustrasi For train dan test in split

4. Jelaskan apa maksudnya kode program *train\_content = d['CONTENT'].iloc[train\_idx]* dan *test\_content = d['CONTENT'].iloc[test\_idx]*. dilengkapi dengan ilustrasi atau gambar.

Maksud dari kode program tersebut adalah membaca isian kolom pada field yang bernama `CONTENT` sebagai data training dan data testing untuk program

No	Nama	Content
1	Mobil	Kendaraan darat yang biasanya memiliki roda 4
2	Motor	Kendaraan darat yang biasanya memiliki roda 2
3	Traktor	Kendaraan darat yang dipakai untuk membajak sawah
4	Helikopter	Kendaraan udara yang memiliki baling-baling untuk terbang

**Gambar 7.4** Ilustrasi penggunaan kolom Content

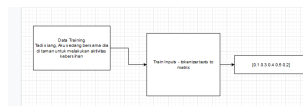
Jelaskan apa maksud dari fungsi `tokenizer = Tokenizer(num_words=2000)` dan `tokenizer.fit_on_texts(train_content)`, dilengkapi dengan ilustrasi atau gambar.

- `tokenizer = Tokenizer(num_words=2000)` digunakan untuk membaca kalimat yang telah dibuat menjadi token sebanyak 2000 kata
- `fit_on_texts` digunakan untuk membuat membaca data token teks yang telah dimasukan kedalam fungsi yaitu fungsi `train_konten`



**Gambar 7.5** Ilustrasi fit tokenizer dan num\_word=2000

5. Jelaskan apa maksud dari fungsi `d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')` dan `d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')`, dilengkapi dengan ilustrasi kode dan atau gambar. Untuk digunakan sebagai pengubah urutan teks yang tadi telah dilakukan tokenizer menjadi matriks yang berurutan seperti tf idf



**Gambar 7.6** Ilustrasi `d_train_inputs = tokenizer.texts to matrix`

6. Jelaskan apa maksud dari fungsi `d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))` dan `d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))`, dilengkapi dengan ilustrasi atau gambar. Fungsi tersebut digunakan untuk membagi matriks tfidf dengan penentuan maksimum array sepanjang sumbu sehingga akan menimbulkan garis ke bawah dan ke atas yang membentuk gambar v. Lalu hasil tersebut akan dimasukkan ke variabel `d_train_input` dan `d_test_input` dengan metode `absolute`. Yang berarti tanpa bilangan negatif.
7. Jelaskan apa maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx])` dan `d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])` dalam kode program, dilengkapi dengan ilustrasi atau gambar. Maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut `class` menjadi bentuk `matrix` dengan pengurutan berdasarkan data index training dan testing.
8. Jelaskan apa maksud dari fungsi di listing 7.5. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

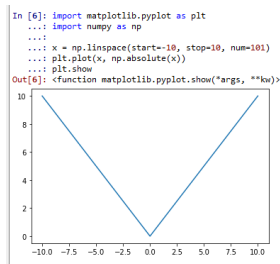
```

1 model = Sequential()
2 model.add(Dense(512, input_shape=(2000,)))
3 model.add(Activation('relu'))
4 model.add(Dropout(0.5))
5 model.add(Dense(2))
6 model.add(Activation('softmax'))
7

```

**Listing 7.2** Membuat model Neural Network

model = sequential berarti variabel model berisi method sequential yang berguna untuk searching data dengan menerima parameter atau argumen kunci dengan langkah tertentu untuk mencari data yang telah diolah. Kemudian model akan ditambahkan method add dengan dense yang berarti data - data yang diinputkan akan terhubung, dengan data 612 dan 2000 data kata atau word kemudian model tersebut di masukan fungsi activation dengan rumus atau metode relu. setelah itu data akan di dropout 0.5 atau dipangkas sebanyak 50 persen dikarenakan pada pohon bobot terlalu akurat terhadap data.



**Gambar 7.7** Ilustrasi d train inputs

```

In [10]: labels = [0,2,1,2,0,1]
...: keras.utils.to_categorical(labels)Out[11]:
array([[1., 0., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [0., 0., 1.],
       [1., 0., 0.],
       [0., 1., 0.]], dtype=float32)

```

**Gambar 7.8** Ilustrasi train outputs = np utils.to categorical

9. Jelaskan apa maksud dari fungsi di listing 7.6 dengan parameter tersebut.

```

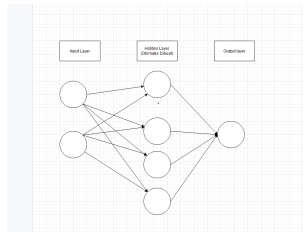
1 model.compile(loss='categorical_crossentropy', optimizer='
  adamax',
2               metrics=['accuracy'])
3

```

**Listing 7.3** Compile model

model tersebut kemudian di compile atau di kembalikan kembali fungsi nilainya yangmana akan mengembalikan fungsi nilai loss nya berapa yang diambil dari

fungsi adamax yang berguna untuk mengetahui nilai lossnya kemudian metrics = accuracy merupakan akurasi dari nilai matrixnya. kemudian terdiri atas beberapa layer atau hidden layer. perbedaan antara deep learning dan DNN atau Deep Neural Network yaitu deep learning merupakan pemakai algoritma dari DNN dan DNN merupakan algoritma yang ada pada deep learning.



**Gambar 7.9** Ilustrasi Neural Network

10. Jelaskan apa itu Deep Learning

Deep learning merupakan salah satu algoritma yang seperti Neural Network yang menggunakan meta data sebagai inputan dan mengolahnya menggunakan layer layer yang tersembunyi.

11. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning

Deep Neural Network merupakan algoritma jaringan syaraf yang melakukan pembobotan terhadap data yang sudah ada sebagai acuan untuk data inputan selanjutnya.

12. Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride  $(NPM \bmod 3 + 1) \times (NPM \bmod 3 + 1)$  yang terdapat max pooling.(nilai 30)

sebelum membuat ilustrasi perlu di ketahui apa itu stride, stride adalah acuan atau parameter yang menentukan pergeseran pada filter fixcel. sebagai contoh nilai stride 1 yang berarti filter akan bergeser sebanyak satu fixcel secara vertikal dan horizontal. selanjutnya apa itu max pooling contoh pada suatu gambar di tentukan Max Pooling dari  $3 \times 3$  dengan stride 1 yang berarti setiap pergeseran 1 pixel akan diambil nilai terbesar dari pixel  $3 \times 3$  tersebut.



```

13     # dengan ketentuan jika i lebih kecil daripada o
14     if i > 0:
15         # dibuat variabel img dengan isi keras untuk aktivasi
16         # neural network fungsi yang membaca data yang berada dalam
17         # folder HASYv2 dengan input nilai -1.0 dan 1.0
18         img = keras.preprocessing.image.img_to_array(
19             pil_image.open("HASYv2/" + row[0]))
20         # Pembagian data yang ada pada fungsi img sebanyak
21         255.0
22         img /= 255.0
23         # Penambahan nilai baru pada imgs pada row ke 1 2 dan
24         # dilanjutkan dengan variabel img
25         imgs.append((row[0], row[2], img))
26         # Penambahan nilai pada row ke 2 pada variabel
27         classes
28         classes.append(row[2])
29         # penambahan nilai satu pada variabel i
30         i += 1

```

3. Jelaskan kode program pada blok # In[3]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[3]: shuffle the data, split into 80% train, 20% test
2 # Melakukan import library random
3 import random
4 # melakukan random pada vungsi imgs
5 random.shuffle(imgs)
6 # Menginisiasi variabel split_idx dengan nilai integer 80 persen
7 # dikali dari pengembalian jumlah dari variabel imgs
8 split_idx = int(0.8*len(imgs))
9 # Menginisiasi variabel train dengan isi lebih besar split idx
10 train = imgs[:split_idx]
11 # Menginisiasi variabel test dengan isi lebih kecil split idx
12 test = imgs[split_idx:]

```

4. Jelaskan kode program pada blok # In[4]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[4]:
2 # Melakukan import library numpy dengan inisial np
3 import numpy as np
4 # Menginisiasi variabel train input dengan np method asarray yang
5 # mana membuat array dengan isi row 2 dari data train
6 train_input = np.asarray(list(map(lambda row: row[2], train)))
7 # membuat test input input dengan np method asarray yang mana
8 # membuat array dengan isi row 2 dari data test
9 test_input = np.asarray(list(map(lambda row: row[2], test)))
10 # Menginisiasi variabel train_output dengan np method asarray
11 # yang mana membuat array dengan isi row 1 dari data train
12 train_output = np.asarray(list(map(lambda row: row[1], train)))
13 # Menginisiasi variabel test_output dengan np method asarray yang
14 # mana membuat array dengan isi row 1 dari data test
15 test_output = np.asarray(list(map(lambda row: row[1], test)))

```

5. Jelaskan kode program pada blok # In[5]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[5]: import encoder and one hot
2 # Melakukan import library LabelEncode dari sklearn
3 from sklearn.preprocessing import LabelEncoder
4 # Melakukan import library OneHotEncoder dari sklearn
5 from sklearn.preprocessing import OneHotEncoder
```

6. Jelaskan kode program pada blok # In[6]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[6]:convert class names into one-hot encoding
2
3 # Menginisiasi variabel label_encoder dengan isi LabelEncoder
4 label_encoder = LabelEncoder()
5 # Menginisiasi variabel integer_encoded yang berfungsi untuk
  Menconvert variabel classes kedalam bentuk integer
6 integer_encoded = label_encoder.fit_transform(classes)
```

7. Jelaskan kode program pada blok # In[7]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[7]:then convert integers into one-hot encoding
2 # Menginisiasi variabel onehot_encoder dengan isi OneHotEncoder
3 onehot_encoder = OneHotEncoder(sparse=False)
4 # mengisi variabel integer_encoded dengan isi integer_encoded
  yang telah di convert pada fungsi sebelumnya
5 integer_encoded = integer_encoded.reshape(len(integer_encoded),
  1)
6 # Menconvert variabel integer_encoded kedalam onehot_encoder
7 onehot_encoder.fit(integer_encoded)
```

8. Jelaskan kode program pada blok # In[8]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[8]:convert train and test output to one-hot
2 # Menconvert data train output menggunakan variabel
  label_encoder kedalam variabel train_output_int
3 train_output_int = label_encoder.transform(train_output)
4 # Menconvert variabel train_output_int kedalam fungsi
  onehot_encoder
5 train_output = onehot_encoder.transform(train_output_int.reshape(
  len(train_output_int), 1))
6 # Menconvert data test-output menggunakan variabel label_encoder
  kedalam variabel test_output_int
7 test_output_int = label_encoder.transform(test_output)
8 # Menconvert variabel test_output_int kedalam fungsi
  onehot_encoder
```

```

9 test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
10 # Menginisiasi variabel num_classes dengan isi variabel
    label_encoder dan classess
11 num_classes = len(label_encoder.classes_)
12 # mencetak hasil dari nomer Class berupa persen
13 print("Number of classes: %d" % num_classes)

```

9. Jelaskan kode program pada blok # In[9]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[9]: import sequential
2 # Melakukan import library Sequential dari Keras
3 from keras.models import Sequential
4 # Melakukan import library Dense, Dropout, Flatten dari Keras
5 from keras.layers import Dense, Dropout, Flatten
6 # Melakukan import library Conv2D, MaxPooling2D dari Keras
7 from keras.layers import Conv2D, MaxPooling2D

```

10. Jelaskan kode program pada blok # In[10]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[10]: desain jaringan
2 # Menginisiasi variabel model dengan isian library Sequential
3 model = Sequential()
4 # variabel model di tambahkan library Conv2D tigapuluh dua bit
    dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang
    menggunakan data train_input
5 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
6    input_shape=np.shape(train_input[0])))
7 # variabel model di tambahkan dengan lib MaxPooling2D dengan
    ketentuan ukuran 2 x 2 pixel
8 model.add(MaxPooling2D(pool_size=(2, 2)))
9 # variabel model di tambahkan dengan library Conv2D 32bit dengan
    kernel 3 x 3
10 model.add(Conv2D(32, (3, 3), activation='relu'))
11 # variabel model di tambahkan dengan lib MaxPooling2D dengan
    ketentuan ukuran 2 x 2 pixel
12 model.add(MaxPooling2D(pool_size=(2, 2)))
13 # variabel model di tambahkan library Flatten
14 model.add(Flatten())
15 # variabel model di tambahkan library Dense dengan fungsi tanh
16 model.add(Dense(1024, activation='tanh'))
17 # variabel model di tambahkan library dropout untuk memangkas
    data tree sebesar 50 persen
18 model.add(Dropout(0.5))
19 # variabel model di tambahkan library Dense dengan data dari
    num_classes dan fungsi softmax
20 model.add(Dense(num_classes, activation='softmax'))
21 # mengcompile data model untuk mendapatkan data loss akurasi dan
    optimasi
22 model.compile(loss='categorical_crossentropy', optimizer='adam',
23    metrics=['accuracy'])

```



```

24 # mencetak variabel model kemudian memunculkan kesimpulan berupa
    data total parameter, trainable parameter dan bukan trainable
    parameter
25 print(model.summary())

```

11. Jelaskan kode program pada blok # In[11]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[11]: import sequential
2 # Melakukan import library keras callbacks
3 import keras.callbacks
4 # Menginisiasi variabel tensorboard dengan isi lib keras
5 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
    style')

```

12. Jelaskan kode program pada blok # In[12]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[12]: 5menit kali 10 epoch = 50 menit
2 # fungsi model titambahkan metod fit untuk mengetahui perhitungan
    dari train_input train_output
3 model.fit(train_input, train_output,
4 # dengan batch size 32 bit
5         batch_size=32,
6         epochs=10,
7         verbose=2,
8         validation_split=0.2,
9         callbacks=[tensorboard])
10
11 score = model.evaluate(test_input, test_output, verbose=2)
12 print('Test loss:', score[0])
13 print('Test accuracy:', score[1])

```

13. Jelaskan kode program pada blok # In[13]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[13]: try various model configurations and parameters to find
    the best
2 # Melakukan import library time
3 import time
4 #Menginisiasi variabel result dengan array kosong
5 results = []
6 # melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
7 for conv2d_count in [1, 2]:
8     # menentukan ukuran besaran fixcel dari data atau konvert 1
        fixcel mnjadi data yang berada pada codigan dibawah.
9     for dense_size in [128, 256, 512, 1024, 2048]:
10         # membuat looping untuk memangkas masing-masing data
            dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
11         for dropout in [0.0, 0.25, 0.50, 0.75]:
12             # Menginisiasi variabel model Sequential

```

```

13     model = Sequential()
14     #membuat looping untuk variabel i dengan jarak dari
    hasil konvolusi.
15     for i in range(conv2d_count):
16         # syarat jika i samadengan bobotnya 0
17         if i == 0:
18             # Penambahan method add pada variabel model
    dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat
    kernel dengan ukuran 3 x 3 dan rumus aktivasi relu dan data
    shape yang di hitung dari data train.
19             model.add(Conv2D(32, kernel_size=(3, 3),
    activation='relu', input_shape=np.shape(train_input[0])))
20             # jika tidak
21         else:
22             # Penambahan method add pada variabel model
    dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3
    dan fungsi aktivasi relu
23             model.add(Conv2D(32, kernel_size=(3, 3),
    activation='relu'))
24             # Penambahan method add pada variabel model
    dengan isian method Max pooling berdimensi 2 dengan ukuran
    fixcel 2 x 2.
25             model.add(MaxPooling2D(pool_size=(2, 2)))
26             # merubah feature gambar menjadi 1 dimensi vektor
27             model.add(Flatten())
28             # Penambahan method dense untuk pemadatan data dengan
    ukuran dense di tentukan dengan rumus fungsi tanh.
29             model.add(Dense(dense_size, activation='tanh'))
30             # membuat ketentuan jika pemangkasan lebih besar dari
    0 persen
31             if dropout > 0.0:
32                 # Penambahan method dropout pada model dengan
    nilai dari dropout
33                 model.add(Dropout(dropout))
34                 # Penambahan method dense dengan fungsi num
    classs dan rumus softmax
35                 model.add(Dense(num_classes, activation='softmax'))
36                 # mengcompile variabel model dengan hasi loss
    optimasi dan akurasi matrix
37                 model.compile(loss='categorical_crossentropy',
    optimizer='adam', metrics=['accuracy'])
38                 # melakukan log pada dir
39                 log_dir = './logs/conv2d-%d-dense-%d-dropout-%.2f' %
    (conv2d_count, dense_size, dropout)
40                 # Menginisiasi variabel tensorboard dengan isian dari
    library keras dan nilai dari log_dir
41                 tensorboard = keras.callbacks.TensorBoard(log_dir=
    log_dir)
42                 # Menginisiasi variabel start dengan isian dari
    library time menggunakan method time
43
44                 start = time.time()
45                 # Penambahan method fit pada model dengan data dari
    train input train output nilai batch nilai epoch verbose
    nilai 20 persen validation split dan callback dengan nilai
    tnsorboard.

```

```

46         model.fit(train_input , train_output , batch_size=32,
47                   epochs=10,
48                   verbose=0, validation_split=0.2, callbacks
49                   =[tensorboard ])
50         # Menginisiasi variabel score dengan nilai evaluasi
51         dari model menggunakan data tes input dan tes output
52         score = model.evaluate(test_input , test_output ,
53                                verbose=2)
54         # Menginisiasi variabel end
55         end = time.time()
56         # Menginisiasi variabel elapsed
57         elapsed = end - start
58         # mencetak hasil perhitungan
59         print("Conv2D count: %d, Dense size: %d, Dropout: %.2
60               f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count
61               , dense_size , dropout , score[0], score[1], elapsed))
62         results.append((conv2d_count , dense_size , dropout ,
63                         score[0], score[1], elapsed))

```

14. Jelaskan kode program pada blok # In[14]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1  # In[14]:rebuild/retrain a model with the best parameters (from
2  the search) and use all data
3  # Menginisiasi variabel model dengan isian library Sequential
4  model = Sequential()
5  # variabel model di tambahkan library Conv2D tigapuluh dua bit
6  dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang
7  menggunakan data train_input
8  model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
9                  input_shape=np.shape(train_input[0])))
10 # variabel model di tambahkan dengan lib MaxPooling2D dengan
11 ketentuan ukuran 2 x 2 pixel
12 model.add(MaxPooling2D(pool_size=(2, 2)))
13 # variabel model di tambahkan dengan library Conv2D 32bit dengan
14 kernel 3 x 3
15 model.add(Conv2D(32, (3, 3), activation='relu'))
16 # variabel model di tambahkan dengan lib MaxPooling2D dengan
17 ketentuan ukuran 2 x 2 pixel
18 model.add(MaxPooling2D(pool_size=(2, 2)))
19 # variabel model di tambahkan library Flatten
20 model.add(Flatten())
21 # variabel model di tambahkan library Dense dengan fungsi tanh
22 model.add(Dense(128, activation='tanh'))
23 # variabel model di tambahkan library dropout untuk memangkas
24 data tree sebesar 50 persen
25 model.add(Dropout(0.5))
26 # variabel model di tambahkan library Dense dengan data dari
27 num_classes dan fungsi softmax
28 model.add(Dense(num_classes , activation='softmax'))
29 # mengcompile data model untuk mendapatkan data loss akurasi dan
30 optimasi
31 model.compile(loss='categorical_crossentropy', optimizer='adam',
32               metrics=[ 'accuracy' ])

```

```

22 # mencetak variabel model kemudian memunculkan kesimpulan berupa
    data total parameter, trainable parameter dan bukan trainable
    parameter
23 print(model.summary())

```

15. Jelaskan kode program pada blok # In[15]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[15]:join train and test data so we train the network on all
    data we have available to us
2 # melakukan join numpy menggunakan data train_input test_input
3 model.fit(np.concatenate((train_input, test_input)),
4           # kelanjutan data yang di gunakan pada join
            train_output test_output
5           np.concatenate((train_output, test_output)),
6           #menggunakan ukuran 32 bit dan epoch 10
7           batch_size=32, epochs=10, verbose=2)

```

16. Jelaskan kode program pada blok # In[16]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[16]:save the trained model
2 #menyimpan model atau mengeksport model yang telah di jalantadi
3 model.save("mathsymbols.model")

```

17. Jelaskan kode program pada blok # In[17]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[17]:save label encoder (to reverse one-hot encoding)
2 # menyompan label encoder dengan nama classes.npy
3 np.save('classes.npy', label_encoder.classes_)

```

18. Jelaskan kode program pada blok # In[18]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[18]:load the pre-trained model and predict the math symbol
    for an arbitrary image;
2 # the code below could be placed in a separate file
3 # mengimpor library keras model
4 import keras.models
5 # Menginisiasi variabel model2 untuk meload model yang telah di
    simpan tadi
6 model2 = keras.models.load_model("mathsymbols.model")
7 # mencetak hasil model2
8 print(model2.summary())

```

19. Jelaskan kode program pada blok # In[19]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[19]: restore the class name to integer encoder
2 # Menginisiasi variabel label encoder ke 2 dengan isian fungsi
   label_encoder.
3 label_encoder2 = LabelEncoder()
4 # Penambahan method classess dengan data classess yang di eksport
   tadi
5 label_encoder2.classes_ = np.load('classes.npy')
6 # membuat fungsi predict dengan path img
7 def predict(img_path):
8     # Menginisiasi variabel newimg dengan membuaay image menjadi
       array dan membuka data berdasarkan img path
9     newimg = keras.preprocessing.image.img_to_array(pil_image.
       open(img_path))
10    # membagi data yang terdapat pada variabel newimg sebanyak
       255
11    newimg /= 255.0
12
13    # do the prediction
14    # Menginisiasi variabel predivtion dengan isian variabel
       model2 menggunakan fungsi predic dengan syarat variabel
       newimg dengan data reshape
15    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
16
17    # figure out which output neuron had the highest score, and
       reverse the one-hot encoding
18    # Menginisiasi variabel inverted denagan label encoder2 dan
       menggunakan argmax untuk mencari skor luaran tertinggi
19    inverted = label_encoder2.inverse_transform([np.argmax(
       prediction)])
20    # mencetak prediksi gambar dan confidence dari gambar.
21    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
       max(prediction)))

```

20. Jelaskan kode program pada blok # In[20]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[20]: grab an image (we'll just use a random training image
   for demonstration purposes)
2 # mencari prediksi menggunakan fungsi prediksi yang di buat tadi
   dari data di HASYv2/hasy-data/v2-00010.png
3 predict("HASYv2/hasy-data/v2-00010.png")
4 # mencari prediksi menggunakan fungsi prediksi yang di buat tadi
   dari data di HASYv2/hasy-data/v2-00500.png
5 predict("HASYv2/hasy-data/v2-00500.png")
6 # mencari prediksi menggunakan fungsi prediksi yang di buat tadi
   dari data di HASYv2/hasy-data/v2-00700.png
7 predict("HASYv2/hasy-data/v2-00700.png")

```

### 7.1.3 Penanganan Error

1. SS Error

```
File "D:\Kuliah\Semester 4\Kecerdasan Buatan\Bahan/urc/1174027/6/1174027.py", line 9,
in <module>
  import librosa
ModuleNotFoundError: No module named 'librosa'
```

**Gambar 7.11** No Module Name error

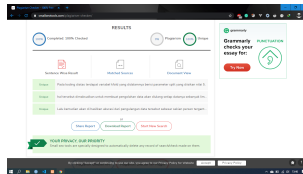
## 2. Jenis Error

- No Module

## 3. Cara Penanganan

Dengan cara melakukan instalasi module yang bersangkutan / menginstal library yang digunakan

### 7.1.4 Bukti Tidak Plagiat

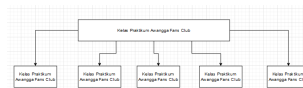


**Gambar 7.12** Tidak Melakukan Plagiat Pada Ch 7

## 7.2 1164013 - Ikrima Ningrumsari Mulyana

### 7.2.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.



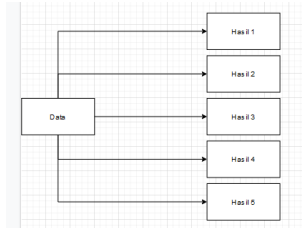
**Gambar 7.13** Ilustrasi Tokenizer

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.4. dilengkapi dengan ilustrasi atau gambar.

```
1 kfold = StratifiedKFold(n_splits=5)
2 splits = kfold.split(d, d['CLASS'])
3
```

**Listing 7.4** K Fold Cross Validation

Pada koding diatas terdapat variabel `kfold` yang didalamnya berisi parameter `split` yang diisikan nilai 5. hal tersebut dimaksudkan untuk membuat pengolahan data akan diulang setiap datanya sebanyak lima kali dengan atribut class sebagai acuan pengolahan datanya. Lalu kemudian akan di hasilkan akurasi dari pengulangan data tersebut sebesar sekian persen tergantung datanya



**Gambar 7.14** Ilustrasi K Fold Cross Validation

3. Jelaskan apa maksudnya kode program *for train, test in splits*. dilengkapi dengan ilustrasi atau gambar.

For train digunakan untuk melakukan training atau pelatihan pada data yang sudah dideklarasikan sebelumnya. Sedangkan test in split digunakan untuk membatasi jumlah data yang akan diinputkan atau data yang akan digunakan.

```
In [3]: import numpy as np
...     from sklearn.model_selection import train_test_split
...     x = np.arange(10).reshape(1, 1)
...     y = np.arange(1)
In [4]: splits = []
...     for i in range(5):
...         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=i)
...         splits.append((x_train, x_test, y_train, y_test))
In [5]: splits
```

**Gambar 7.15** Ilustrasi For train dan test in split

4. Jelaskan apa maksudnya kode program *train\_content = d['CONTENT'].iloc[train\_idx]* dan *test\_content = d['CONTENT'].iloc[test\_idx]*. dilengkapi dengan ilustrasi atau gambar.

Maksud dari kode program tersebut adalah membaca isian kolom pada field yang bernama CONTENT sebagai data training dan data testing untuk program

No	Nama	Content
1	Motor	Kendaraan darat yang biasanya memiliki roda 4
2	Motor	Kendaraan darat yang biasanya memiliki roda 2
3	Traktor	Kendaraan darat yang dipakai untuk membajak sawah
4	Helikopter	Kendaraan udara yang memiliki baling-baling untuk terbang

**Gambar 7.16** Ilustrasi penggunaan kolom Content

Jelaskan apa maksud dari fungsi *tokenizer = Tokenizer(num\_words=2000)* dan *tokenizer.fit\_on\_texts(train\_content)*, dilengkapi dengan ilustrasi atau gambar.

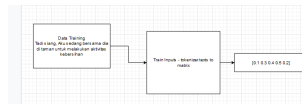
- `tokenizer = Tokenizer(num_words=2000)` digunakan untuk membaca kalimat yang telah dibuat menjadi token sebanyak 2000 kata

- `fit_on_texts` digunakan untuk membuat membaca data token teks yang telah dimasukan kedalam fungsi yaitu fungsi `train_konten`



**Gambar 7.17** Ilustrasi fit tokenizer dan `num_word=2000`

5. Jelaskan apa maksud dari fungsi `d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')` dan `d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')`, dilengkapi dengan ilustrasi kode dan atau gambar. Untuk digunakan sebagai pengubah urutan teks yang tadi telah dilakukan tokenizer menjadi matriks yang berurutan seperti tf idf



**Gambar 7.18** Ilustrasi `d_train_inputs = tokenizer.texts to matrix`

6. Jelaskan apa maksud dari fungsi `d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))` dan `d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))`, dilengkapi dengan ilustrasi atau gambar. Fungsi tersebut digunakan untuk membagi matriks tfidf dengan penentuan maksimum array sepanjang sumbu sehingga akan menimbulkan garis ke bawah dan ke atas yang membentuk gambar v. Lalu hasil tersebut akan dimasukkan ke variabel `d_train_input` dan `d_test_input` dengan metode absolute. Yang berarti tanpa bilangan negatif.
7. Jelaskan apa maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx])` dan `d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])` dalam kode program, dilengkapi dengan ilustrasi atau gambar. Maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing.
8. Jelaskan apa maksud dari fungsi di listing 7.5. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

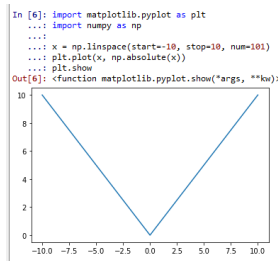
```

1 model = Sequential()
2 model.add(Dense(512, input_shape=(2000,)))
3 model.add(Activation('relu'))
4 model.add(Dropout(0.5))
5 model.add(Dense(2))
6 model.add(Activation('softmax'))
  
```



**Listing 7.5** Membuat model Neural Network

model = sequential berarti variabel model berisi method sequential yang berguna untuk searching data dengan menerima parameter atau argumen kunci dengan langkah tertentu untuk mencari data yang telah diolah. Kemudian model akan ditambahkan method add dengan dense yang berarti data - data yang diinputkan akan terhubung, dengan data 612 dan 2000 data kata atau word kemudian model tersebut di masukan fungsi activation dengan rumus atau metode relu. setelah itu data akan di dropout 0.5 atau dipangkas sebanyak 50 persen dikarenakan pada pohon bobot terlalu akurat terhadap data.

**Gambar 7.19** Ilustrasi d train inputs

```
In [10]: labels = [0,2,1,2,0,1]
...: keras.utils.to_categorical(labels)Out[11]:
array([[1., 0., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [0., 0., 1.],
       [1., 0., 0.],
       [0., 1., 0.]]) dtype=float32)
```

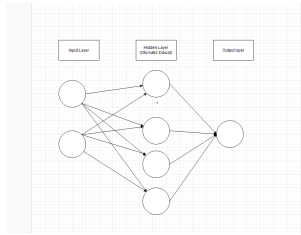
**Gambar 7.20** Ilustrasi train outputs = np utils.to categorical

9. Jelaskan apa maksud dari fungsi di listing 7.6 dengan parameter tersebut.

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

**Listing 7.6** Compile model

model tersebut kemudian di compile atau di kembalikan kembali fungsi nilainya yangmana akan mengembalikan fungsi nilai loss nya berapa yang diambil dari fungsi adamax yang berguna untuk mengetahui nilai lossnya kemudian metrics = accuracy merupakan akurasi dari nilai matrixnya. kemudian terdiri atas beberapa layer atau hidden layer. perbedaan antara deep learning dan DNN atau Deep Neural Network yaitu deep learning merupakan pemakai algoritma dari DNN dan DNN merupakan algoritma yang ada pada deep learning.



**Gambar 7.21** Ilustrasi Neural Network

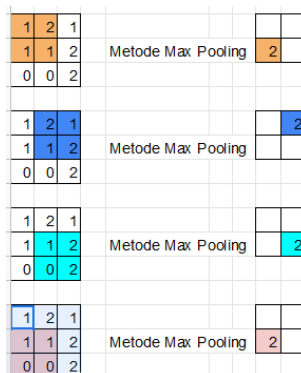
10. Jelaskan apa itu Deep Learning

Deep learning merupakan salah satu algoritma yang seperti Neural Network yang menggunakan meta data sebagai inputan dan mengolahnya menggunakan layer layer yang tersembunyi.

11. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning  
Deep Neural Network merupakan algoritma jaringan syaraf yang melakukan pembobotan terhadap data yang sudah ada sebagai acuan untuk data inputan selanjutnya.

12. Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride  $(NPM \bmod 3 + 1) \times (NPM \bmod 3 + 1)$  yang terdapat max pooling.(nilai 30)

sebelum membuat ilustrasi perlu di ketahui apa itu stride, stride adalah acuan atau parameter yang menentukan pergeseran pada filter fixcel. sebagai contoh nilai stride 1 yang berarti filter akan bergeser sebanyak satu fixcel secara vertikal dan horizontal. selanjutnya apa itu max pooling contoh pada suatu gambar di tentukan Max Pooling dari  $3 \times 3$  dengan stride 1 yang berarti setiap pergeseran 1 pixel akan diambil nilai terbesar dari pixel  $3 \times 3$  tersebut.



**Gambar 7.22** Ilustrasi perhitungan stride 1 max pooling

## 7.2.2 Praktek

1. Jelaskan kode program pada blok # In[1]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[1]:import lib
2 # menimport library CSV untuk mengolah data ber ekstensi csv
3 import csv
4 #lalu import library Image dari PIL atau Python Imaging Library
   untuk mengolah data berupa gambar
5 from PIL import Image as pil_image
6 # lalu import library keras dengan menggunakan method
   preprocessing untuk membuat neural network
7 import keras.preprocessing.image

```

2. Jelaskan kode program pada blok # In[2]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[2]:load all images (as numpy arrays) and save their classes
2 #Menginisiasi variabel imgs dan classes dengan variabel array
   kosong
3 imgs = []
4 classes = []
5 #membuka file hasy-data-labels.csv yang berada di folode HASYv2
   (yang di inisialisasi menjadi csvfile)
6 with open('HASYv2/hasy-data-labels.csv') as csvfile:
7     #Menginisiasi variabel csvreader yang berisi method csv.
   reader yang membaca variabel csvfile
8     csvreader = csv.reader(csvfile)
9     # Menginisiasi variabel i dengan isi 0
10    i = 0
11    # membuat looping pada variabel csvreader
12    for row in csvreader:
13        # dengan ketentuan jika i lebihkecil daripada o
14        if i > 0:
15            # dibuat variabel img dengan isi keras untuk aktivasi
   neural network fungsi yang membaca data yang berada dalam
   folder HASYv2 dengan input nilai -1.0 dan 1.0
16            img = keras.preprocessing.image.img_to_array(
   pil_image.open("HASYv2/" + row[0]))
17            #Pembagian data yang ada pada fungsi img sebanyak
   255.0
18            img /= 255.0
19            # Penambahan nilai baru pada imgs pada row ke 1 2,
   dilanjutkan dengan variabel img
20            imgs.append((row[0], row[2], img))
21            # Penambahan nilai pada row ke 2 pada variabel
   classes
22            classes.append(row[2])
23            # penambahan nilai satu pada variabel i
24            i += 1

```

3. Jelaskan kode program pada blok # In[3]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[3]: shuffle the data, split into 80% train, 20% test
2 # Melakukan import library random
3 import random
4 # melakukan random pada vungsi imgs
5 random.shuffle(imgs)
6 # Menginisiasi variabel split_idx dengan nilai integer 80% dikali
   dari pengembalian jumlah dari variabel imgs
7 split_idx = int(0.8*len(imgs))
8 # Menginisiasi variabel train dengan isi lebih besar split idx
9 train = imgs[:split_idx]
10 # Menginisiasi variabel test dengan isi lebih kecil split idx
11 test = imgs[split_idx:]

```

4. Jelaskan kode program pada blok # In[4]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[4]:
2 # Melakukan import library numpy dengan inisial np
3 import numpy as np
4 # Menginisiasi variabel train input dengan np method asarray yang
   mana membuat array dengan isi row 2 dari data train
5 train_input = np.asarray(list(map(lambda row: row[2], train)))
6 # membuat test input input dengan np method asarray yang mana
   membuat array dengan isi row 2 dari data test
7 test_input = np.asarray(list(map(lambda row: row[2], test)))
8 # Menginisiasi variabel train_output dengan np method asarray
   yang mana membuat array dengan isi row 1 dari data train
9 train_output = np.asarray(list(map(lambda row: row[1], train)))
10 # Menginisiasi variabel test_output dengan np method asarray yang
   mana membuat array dengan isi row 1 dari data test
11 test_output = np.asarray(list(map(lambda row: row[1], test)))

```

5. Jelaskan kode program pada blok # In[5]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[5]: import encoder and one hot
2 # Melakukan import library LabelEncode dari sklearn
3 from sklearn.preprocessing import LabelEncoder
4 # Melakukan import library OneHotEncoder dari sklearn
5 from sklearn.preprocessing import OneHotEncoder

```

6. Jelaskan kode program pada blok # In[6]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[6]: convert class names into one-hot encoding
2
3 # Menginisiasi variabel label_encoder dengan isi LabelEncoder

```

```

4 label_encoder = LabelEncoder()
5 # Menginisiasi variabel integer_encoded yang berfungsi untuk
  Menconvert variabel classes kedalam bentuk integer
6 integer_encoded = label_encoder.fit_transform(classes)

```

7. Jelaskan kode program pada blok # In[7]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[7]:then convert integers into one-hot encoding
2 # Menginisiasi variabel onehot_encoder dengan isi OneHotEncoder
3 onehot_encoder = OneHotEncoder(sparse=False)
4 # mengisi variabel integer_encoded dengan isi integer_encoded
  yang telah di convert pada fungsi sebelumnya
5 integer_encoded = integer_encoded.reshape(len(integer_encoded),
  1)
6 # Menconvert variabel integer_encoded kedalam onehot_encoder
7 onehot_encoder.fit(integer_encoded)

```

8. Jelaskan kode program pada blok # In[8]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[8]:convert train and test output to one-hot
2 # Menconvert data train output menggunakan variabel
  label_encoder kedalam variabel train_output_int
3 train_output_int = label_encoder.transform(train_output)
4 # Menconvert variabel train_output_int kedalam fungsi
  onehot_encoder
5 train_output = onehot_encoder.transform(train_output_int.reshape(
  len(train_output_int), 1))
6 # Menconvert data test_output menggunakan variabel label_encoder
  kedalam variabel test_output_int
7 test_output_int = label_encoder.transform(test_output)
8 # Menconvert variabel test_output_int kedalam fungsi
  onehot_encoder
9 test_output = onehot_encoder.transform(test_output_int.reshape(
  len(test_output_int), 1))
10 # Menginisiasi variabel num_classes dengan isi variabel
  label_encoder dan classess
11 num_classes = len(label_encoder.classes_)
12 # mencetak hasil dari nomer Class berupa persen
13 print("Number of classes: %d" % num_classes)

```

9. Jelaskan kode program pada blok # In[9]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[9]: import sequential
2 # Melakukan import library Sequential dari Keras
3 from keras.models import Sequential
4 # Melakukan import library Dense, Dropout, Flatten dari Keras
5 from keras.layers import Dense, Dropout, Flatten
6 # Melakukan import library Conv2D, MaxPooling2D dari Keras
7 from keras.layers import Conv2D, MaxPooling2D

```

10. Jelaskan kode program pada blok # In[10]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[10]: desain jaringan
2 # Menginisiasi variabel model dengan isian library Sequential
3 model = Sequential()
4 # variabel model di tambahkan library Conv2D tigapuluh dua bit
   dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang
   menggunakan data train_input
5 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
6                 input_shape=np.shape(train_input[0])))
7 # variabel model di tambahkan dengan lib MaxPooling2D dengan
   ketentuan ukuran 2 x 2 pixel
8 model.add(MaxPooling2D(pool_size=(2, 2)))
9 # variabel model di tambahkan dengan library Conv2D 32bit dengan
   kernel 3 x 3
10 model.add(Conv2D(32, (3, 3), activation='relu'))
11 # variabel model di tambahkan dengan lib MaxPooling2D dengan
   ketentuan ukuran 2 x 2 pixel
12 model.add(MaxPooling2D(pool_size=(2, 2)))
13 # variabel model di tambahkan library Flatten
14 model.add(Flatten())
15 # variabel model di tambahkan library Dense dengan fungsi tanh
16 model.add(Dense(1024, activation='tanh'))
17 # variabel model di tambahkan library dropout untuk memangkas
   data tree sebesar 50 persen
18 model.add(Dropout(0.5))
19 # variabel model di tambahkan library Dense dengan data dari
   num_classes dan fungsi softmax
20 model.add(Dense(num_classes, activation='softmax'))
21 # mengkompilasi data model untuk mendapatkan data loss akurasi dan
   optimasi
22 model.compile(loss='categorical_crossentropy', optimizer='adam',
23               metrics=['accuracy'])
24 # mencetak variabel model kemudian memunculkan kesimpulan berupa
   data total parameter, trainable parameter dan bukan trainable
   parameter
25 print(model.summary())

```

11. Jelaskan kode program pada blok # In[11]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[11]: import sequential
2 # Melakukan import library keras callbacks
3 import keras.callbacks
4 # Menginisiasi variabel tensorboard dengan isi lib keras
5 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
   style')

```

12. Jelaskan kode program pada blok # In[12]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[12]: 5menit kali 10 epoch = 50 menit
2 # fungsi model titambahkan metod fit untuk mengetahui perhitungan
   dari train_input train_output
3 model.fit(train_input , train_output ,
4 # dengan batch size 32 bit
5         batch_size=32,
6         epochs=10,
7         verbose=2,
8         validation_split=0.2,
9         callbacks=[tensorboard])
10
11 score = model.evaluate(test_input , test_output , verbose=2)
12 print('Test loss:', score[0])
13 print('Test accuracy:', score[1])

```

13. Jelaskan kode program pada blok # In[13]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[13]: try various model configurations and parameters to find
   the best
2 # Melakukan import library time
3 import time
4 #Menginisiasi variabel result dengan array kosong
5 results = []
6 # melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
7 for conv2d_count in [1, 2]:
8     # menentukan ukuran besaran fixcel dari data atau konvert 1
       fixcel mnjadi data yang berada pada codigan dibawah.
9     for dense_size in [128, 256, 512, 1024, 2048]:
10         # membuat looping untuk memangkas masing-masing data
           dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
11         for dropout in [0.0, 0.25, 0.50, 0.75]:
12             # Menginisiasi variabel model Sequential
13             model = Sequential()
14             #membuat looping untuk variabel i dengan jarak dari
               hasil konvolusi.
15             for i in range(conv2d_count):
16                 # syarat jika i samadengan bobotnya 0
17                 if i == 0:
18                     # Penambahan method add pada variabel model
                       dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat
                       kernel dengan ukuran 3 x 3 dan rumus aktivasi relu dan data
                       shape yang di hitung dari data train.
19                     model.add(Conv2D(32, kernel_size=(3, 3),
                       activation='relu', input_shape=np.shape(train_input[0])))
20                     # jika tidak
21                     else:
22                         # Penambahan method add pada variabel model
                           dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3
                           dan fungsi aktivasi relu
23                         model.add(Conv2D(32, kernel_size=(3, 3),
                           activation='relu'))
24                         # Penambahan method add pada variabel model
                           dengan isian method Max pooling berdimensi 2 dengan ukuran
                           fixcel 2 x 2.

```

```

25         model.add(MaxPooling2D(pool_size=(2, 2)))
26         # merubah feature gambar menjadi 1 dimensi vektor
27         model.add(Flatten())
28         # Penambahan method dense untuk pemadatan data dengan
    ukuran dense di tentukan dengan rumus fungsi tanh.
29         model.add(Dense(dense_size , activation='tanh'))
30         # membuat ketentuan jika pemangkasan lebih besar dari
    0 persen
31         if dropout > 0.0:
32             # Penambahan method dropout pada model dengan
    nilai dari dropout
33             model.add(Dropout(dropout))
34             # Penambahan method dense dengan fungsi num
    classs dan rumus softmax
35             model.add(Dense(num_classes , activation='softmax'))
36             # mongkompile variabel model dengan hasil loss
    optimasi dan akurasi matrix
37             model.compile(loss='categorical_crossentropy',
    optimizer='adam', metrics=['accuracy'])
38             # melakukan log pada dir
39             log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
    (conv2d_count, dense_size, dropout)
40             # Menginisiasi variabel tensorboard dengan isian dari
    library keras dan nilai dari log_dir
41             tensorboard = keras.callbacks.TensorBoard(log_dir=
    log_dir)
42             # Menginisiasi variabel start dengan isian dari
    library time menggunakan method time
43
44             start = time.time()
45             # Penambahan method fit pada model dengan data dari
    train input train output nilai batch nilai epoch verbose
    nilai 20 persen validation split dan callback dengan nilai
    tnsorboard.
46             model.fit(train_input , train_output , batch_size=32,
    epochs=10,
47                             verbose=0, validation_split=0.2, callbacks
    =[tensorboard])
48             # Menginisiasi variabel score dengan nilai evaluasi
    dari model menggunakan data tes input dan tes output
49             score = model.evaluate(test_input , test_output ,
    verbose=2)
50             # Menginisiasi variabel end
51             end = time.time()
52             # Menginisiasi variabel elapsed
53             elapsed = end - start
54             # mencetak hasil perhitungan
55             print("Conv2D count: %d, Dense size: %d, Dropout: %.2
    f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count
    , dense_size, dropout, score[0], score[1], elapsed))
56             results.append((conv2d_count, dense_size, dropout,
    score[0], score[1], elapsed))

```



14. Jelaskan kode program pada blok # In[14]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[14]:rebuild/retrain a model with the best parameters (from
   the search) and use all data
2 # Menginisiasi variabel model dengan isian library Sequential
3 model = Sequential()
4 # variabel model di tambahkan library Conv2D tigapuluh dua bit
   dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang
   menggunakan data train_input
5 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
   input_shape=np.shape(train_input[0])))
6 # variabel model di tambahkan dengan lib MaxPooling2D dengan
   ketentuan ukuran 2 x 2 pixel
7 model.add(MaxPooling2D(pool_size=(2, 2)))
8 # variabel model di tambahkan dengan library Conv2D 32bit dengan
   kernel 3 x 3
9 model.add(Conv2D(32, (3, 3), activation='relu'))
10 # variabel model di tambahkan dengan lib MaxPooling2D dengan
   ketentuan ukuran 2 x 2 pixel
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 # variabel model di tambahkan library Flatten
13 model.add(Flatten())
14 # variabel model di tambahkan library Dense dengan fungsi tanh
15 model.add(Dense(128, activation='tanh'))
16 # variabel model di tambahkan library dropout untuk memangkas
   data tree sebesar 50 persen
17 model.add(Dropout(0.5))
18 # variabel model di tambahkan library Dense dengan data dari
   num_classes dan fungsi softmax
19 model.add(Dense(num_classes, activation='softmax'))
20 # mengkompilasi data model untuk mendapatkan data loss akurasi dan
   optimasi
21 model.compile(loss='categorical_crossentropy', optimizer='adam',
   metrics=['accuracy'])
22 # mencetak variabel model kemudian memunculkan kesimpulan berupa
   data total parameter, trainable parameter dan bukan trainable
   parameter
23 print(model.summary())

```

15. Jelaskan kode program pada blok # In[15]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[15]:join train and test data so we train the network on all
   data we have available to us
2 # melakukan join numpy menggunakan data train_input test_input
3 model.fit(np.concatenate((train_input, test_input)),
4           # kelanjutan data yang di gunakan pada join
           train_output_test_output
5           np.concatenate((train_output, test_output)),
6           #menggunakan ukuran 32 bit dan epoch 10
7           batch_size=32, epochs=10, verbose=2)

```

16. Jelaskan kode program pada blok # In[16]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[16]:save the trained model
2 #menyimpan model atau mengekspor model yang telah di jalantadi
3 model.save("mathsymbols.model")
```

17. Jelaskan kode program pada blok # In[17]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[17]:save label encoder (to reverse one-hot encoding)
2 # menyompan label encoder dengan nama classes.npy
3 np.save('classes.npy', label_encoder.classes_)
```

18. Jelaskan kode program pada blok # In[18]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[18]:load the pre-trained model and predict the math symbol
   for an arbitrary image;
2 # the code below could be placed in a separate file
3 # mengimpor library keras model
4 import keras.models
5 # Menginisiasi variabel model2 untuk meload model yang telah di
   simpan tadi
6 model2 = keras.models.load_model("mathsymbols.model")
7 # mencetak hasil model2
8 print(model2.summary())
```

19. Jelaskan kode program pada blok # In[19]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
1 # In[19]:restore the class name to integer encoder
2 # Menginisiasi variabel label encoder ke 2 dengan isian fungsi
   label encoder.
3 label_encoder2 = LabelEncoder()
4 # Penambahan method classess dengan data classess yang di ekspor
   tadi
5 label_encoder2.classes_ = np.load('classes.npy')
6 # membuat fungsi predict dengan path img
7 def predict(img_path):
8     # Menginisiasi variabel newimg dengan membuay image menjadi
   array dan membuka data berdasarkan img path
9     newimg = keras.preprocessing.image.img_to_array(pil_image.
   open(img_path))
10    # membagi data yang terdapat pada variabel newimg sebanyak
   255
11    newimg /= 255.0
12
13    # do the prediction
```

```

14 # Menginisiasi variabel predivtion dengan isian variabel
    model2 menggunakan fungsi predic dengan syarat variabel
    newimg dengan data reshape
15 prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
16
17 # figure out which output neuron had the highest score, and
    reverse the one-hot encoding
18 # Menginisiasi variabel inverted denagan label encoder2 dan
    menggunakan argmax untuk mencari skor luaran tertinggi
19 inverted = label_encoder2.inverse_transform([np.argmax(
    prediction)])
20 # mencetak prediksi gambar dan confidence dari gambar.
21 print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
    max(prediction)))

```

20. Jelaskan kode program pada blok # In[20]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```

1 # In[20]: grab an image (we'll just use a random training image
    for demonstration purposes)
2 # mencari prediksi menggunakan fungsi prediksi yang di buat tadi
    dari data di HASyV2/hasy-data/v2-00010.png
3 predict("HASYv2/hasy-data/v2-00010.png")
4 # mencari prediksi menggunakan fungsi prediksi yang di buat tadi
    dari data di HASyV2/hasy-data/v2-00500.png
5 predict("HASYv2/hasy-data/v2-00500.png")
6 # mencari prediksi menggunakan fungsi prediksi yang di buat tadi
    dari data di HASyV2/hasy-data/v2-00700.png
7 predict("HASYv2/hasy-data/v2-00700.png")

```

## 7.2.3 Penanganan Error

### 1. SS Error

```

File "C:\Users\Budianto\Documents\Semester 6\Kecerdasan Buatan\Solusi\src\1174827\8\1174827.py", line 9,
in <module>
    import librosa
ModuleNotFoundError: No module named 'librosa'

```

**Gambar 7.23** No Module Name error

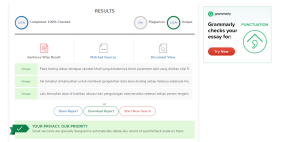
### 2. Jenis Error

- No Module

### 3. Cara Penanganan

Dengan cara melakukan instalasi module yang bersangkutan / menginstal library yang digunakan

## 7.2.4 Bukti Tidak Plagiat



**Gambar 7.24** Tidak Melakukan Plagiat Pada Ch 7



# DAFTAR PUSTAKA

---

- [1] R. Awangga, "Sampeu: Servicing web map tile service over web map service to increase computation performance," in *IOP Conference Series: Earth and Environmental Science*, vol. 145, no. 1. IOP Publishing, 2018, p. 012057.



# Index

---

disruptif, **xxiii**  
modern, **xxiii**