

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 Tunnisia

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. Sariasisih 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

1 Chapter 1	1
2 Chapter 2	3
3 Chapter 3	5
4 Chapter 4	7
5 Chapter 5	9
6 Chapter 6	11
7 Chapter 7	13

DAFTAR ISI

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

5	Chapter 5	9
6	Chapter 6	11
7	Chapter 7	13
7.1	1174051 Evietania Charis Sujadi	13
7.1.1	Teori	13
7.1.2	Praktek	19
7.1.3	Penanganan Error	41
7.2	1174021 - Muhammad Fahmi	41
7.2.1	Soal Teori	41
7.2.2	Praktek Program	47
7.2.3	Penanganan Error	59
7.2.4	Bukti Tidak Plagiat	60
7.3	1174026 Felix Lase	60
7.3.1	Teori	60
7.3.2	Praktek	66
7.3.3	Penanganan Error	88
7.4	1174017 Muh. Rifky Prananda	89
7.4.1	Teori	89
7.4.2	Praktek	95
7.4.3	Penanganan Error	116
	Daftar Pustaka	119
	Index	121

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

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].

$$ABC\mathcal{DEF}\alpha\beta\Gamma\Delta \sum_{def}^{abc} \quad (I.1)$$

BAB 1

CHAPTER 1

BAB 2

CHAPTER 2

BAB 3

CHAPTER 3

BAB 4

CHAPTER 4

BAB 5

CHAPTER 5

BAB 6

CHAPTER 6

|||||| HEAD ===== 6666666 a5fe1da339fcb3569c1d26797b40cb9575ead78c

BAB 7

CHAPTER 7

7.1 1174051 Evietania Charis Sujadi

7.1.1 Teori

7.1.1.1 Soal No. 1 Kenapa file teks harus dilakukan tokenizer, dilengkapi dengan ilustrasi atau gambar.

Tokenizer adalah proses untuk membagi kalimat menjadi beberapa teks, hal ini sangat di perlukan dalam AI karena nanti setiap teks akan di hitung bobotnya dan akan memunculkan nilai vektor sehingga teks tersebut bisa di gunakan sebagai data untuk memprediksi teks yang muncul dalam satu kalimat sedangkan proses tkenizer merupakan caramembagi bagi teks dari suatu kalimat biasanya pembagi kalimat tersebut merupakan spasi dalam suatu kalimat.

- Ilustrasi Gambar:



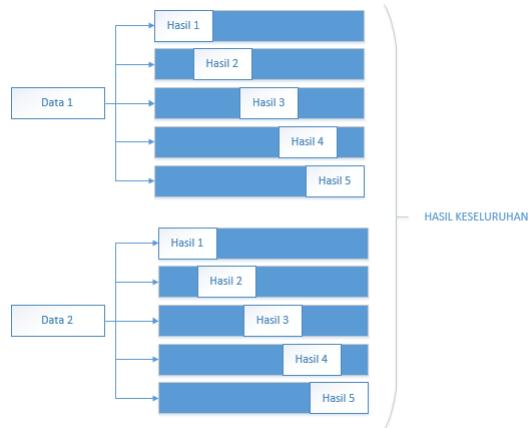
Gambar 7.1 Tokenizer

7.1.1.2 Soal No. 2 Konsep dasar K Fold Cross Validation pada dataset komentar Youtube, dilengkapi dengan ilustrasi atau gambar.

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

pada codingan tersebut terdapat kfold sebagai variabel yang didalamnya terdiri dari split 5 yang berarti pengulangan terhadap pengolahan masing lima kali pada kasus ini terdapat data sebanyak 5 berarti ke lima data tersebut di ulang sebanyak lima kali dengan atribut class sebagai acuan pengolahan datanya kemudian akan dihasilkan akurasi dari pengulangan data tersebut sebesar sekitar persen tergantung datanya.

- Ilustrasi Gambar:

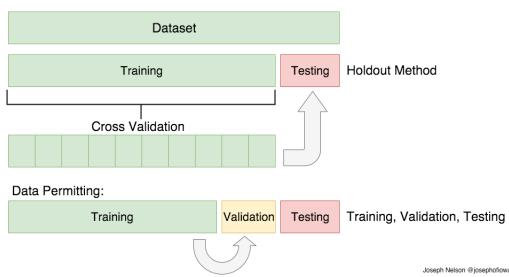


Gambar 7.2 Konsep dasar K Fold Cross Validation

7.1.1.3 Soal No. 3 Maksud kode program for train dan test in splits, dilengkapi dengan ilustrasi atau gambar.

Untuk melakukan pengujian atas data pada dataset sudah di tidak terjadi penumpukan dan split. Dimana di setiap class tidak akan muncul id yang sama.

- Ilustrasi Gambar :



Gambar 7.3 Train dan Test in Split

7.1.1.4 Soal No. 4 Maksud kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx], dilengkapi dengan ilustrasi atau gambar.

Untuk mengambil data pada kolom CONTENT yang merupakan bagian dari train idx dan test idx.

- Ilustrasi Gambar:



Gambar 7.4 Train content

7.1.1.5 Soal No. 5 Maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar.

Yang pertama, yaitu fungsi tokenizer ialah mem-vektorisasi jumlah kata yang ingin diubah kedalam bentuk 2000 kata.

Yang kedua, untuk melakukan fit tokenizer untuk dat trainnya dengan data test nya untuk kolom CONTENT saja.

- Ilustrasi Gambar :

✓ Tokenization

is a process of breaking up a piece of **text** into many pieces, such as sentences and words. It works by separating **words** using spaces and punctuation.

```
[1]: from nltk.tokenize import sent_tokenize
      sentence = "I love ice cream. I also like steak."
      sent_tokenize(sentence)

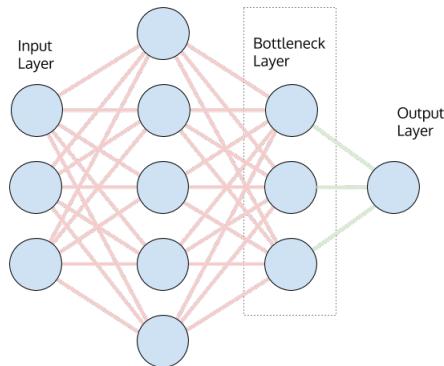
[1]: ['I love ice cream.', 'I also like steak.']}
```

Gambar 7.5 Tokenizer

7.1.1.6 Soal No. 6 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 atau gambar.

Variabel d train input untukmelakukan tokenizer dari bentuk teks ke matrix dari data train content dengan mode tfidf dan variabel d test inputs sama saja untuk data test.

- Ilustrasi Gambar:

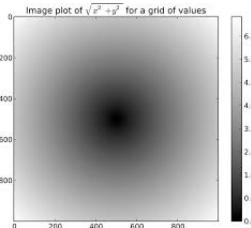


Gambar 7.6 Train Inputs 1

7.1.1.7 Soal No. 7 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.

Akan membagi matrix tfidf dengan amax untuk mengembalikan array atau maksimum array. Kemudian hasilnya dimasukan dalam variabel d train inputs untuk data train dan d test inputs untuk data test dengan nominal bilangan tanpa ada bilangan negatif dan koma.

- Ilustrasi Gambar :

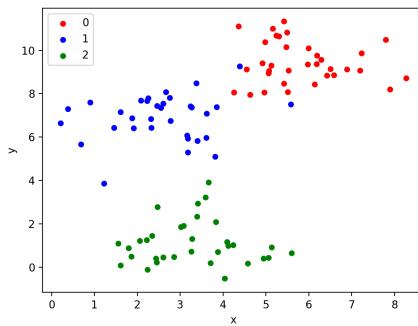


Gambar 7.7 Train Inputs 2

7.1.1.8 Soal No. 8 Maksud 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])

Fungsi pada kode program tersebut ditujukan untuk melakukan one-hot encoding supaya bisa masuk dan digunakan pada neural network. One-hot encoding diambil dari class yang berarti hanya terdapat 2 neuron, yaitu satu nol(1,0) atau nol satu(0,1) karena pilihannya hanya ada dua.

- Ilustrasi Gambar:



Gambar 7.8 Compile model

7.1.1.9 Soal No. 9 Maksud dari listing 7.2.

Fungsi kode program tersebut untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Terdapat 512 neuron inputan dengan input shape 2000 vektor yang sudah di-normalisasi. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer output terdapat 2 neuron outputan yaitu nol (1,0) atau nol satu (0,1). Kemudian outputan tersebut diaktivasi menggunakan fungsi softmax.

7.1.1.10 **Soal No. 10** Maksud dari listing 7.3.

Fungsi kode program tersebut untuk model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.

7.1.1.11 **Soal No. 11** Deep Learning

Deep learning, yang bisa diartikan sebagai rangkaian metode untuk melatih jaringan saraf buatan multi-lapisan. Ternyata, metode ini efektif dalam mengidentifikasi pola dari data. Manakala media membicarakan jaringan saraf, kemungkinan yang dimaksud adalah deep learning.

7.1.1.12 **Soal No. 12** Deep Neural Network, dan apa bedanya dengan Deep Learning

Algoritma DNN (Deep Neural Networks) adalah salah satu algoritma berbasis jaringan saraf yang dapat digunakan untuk pengambilan keputusan. Contoh yang dibahas kali ini adalah mengenai penentuan penerimaan pengajuan kredit sepeda motor baru berdasarkan kelompok data yang sudah ada.

Pembedannya dengan Deep Learning adalah terletak pada kedalaman model, deep learning adalah frasa yang digunakan untuk jaringan saraf yang kompleks.

7.1.1.13 **Soal No. 13** Jelaskan dengan ilustrasi gambar buatan sendiri, bagaimana perhitungan algoritma konvolusi dengan ukuran stride $(NPM \bmod 3 + 1) \times (NPM \bmod 3 + 1)$ yang terdapat max pooling.(nilai 30)

Karena NPM saya 1174051 dan hasil dari $(NPM \bmod 3) + 1 = 2$, maka saya menggunakan matrik kernel berukuran 2×2 . Misalkan $f(x,y)$ yang digunakan berukuran 3×3 dan kernel atau mask berukuran 2×2 adalah sebagai berikut:

Gambar Matriks:

$$F(x,y) = \begin{matrix} 2 & 3 & 5 \\ 1 & 0 & 8 \\ 7 & 2 & 0 \end{matrix} \quad \begin{matrix} 1 & 0 \\ 2 & 4 \end{matrix}$$

Gambar 7.9 Perhitungan algoritma konvolusi

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ adalah $f(x,y) * g(x,y)$

- Tempatkan matrik kernel di sebelah kiri atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(2 \times 1) + (3 \times 0) + (1 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

- Tempatkan matrik kernel di sebelah kanan atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(3 \times 1) + (5 \times 0) + (0 \times 2) + (8 \times 4)$$

Sehingga didapat hasil konvolusi = 35

- Tempatkan matrik kernel di sebelah kiri bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(1 \times 1) + (0 \times 0) + (7 \times 2) + (2 \times 4)$$

Sehingga didapat hasil konvolusi = 23

- Tempatkan matrik kernel di sebelah kanan bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(0 \times 1) + (8 \times 0) + (2 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

Hasil:

$$\begin{matrix} 4 & 35 \\ 23 & 4 \end{matrix}$$

Gambar 7.10 Hasil

7.1.2 Praktek

7.1.2.1 Soal No. 1 Jelaskan arti dari setiap baris kode program pada blok # In[1] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:46:02 2020
4
5 @author: User
6 """
7
8 import csv
9 from PIL import Image as pil_image
10 import keras.preprocessing.image

```

▪ Penjelasan:

Baris Code 1: Memasukkan atau mengimport file csv

Baris Code 2: Memasukkan module image sebagai pil_image dari library PIL

Baris Code 3: Memasukkan atau mengimport fungsi keras.preprocessing.image

▪ Hasil output:

```

In [1]: import csv
...: from PIL import Image as pil_image
...: import keras.preprocessing.image
Using TensorFlow backend.

```

Gambar 7.11 1

7.1.2.2 Soal No. 2 Jelaskan arti dari setiap baris kode program pada blok # In[2] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:25 2020
4
5 @author: User
6 """
7
8 imgs = []
9 classes = []
10 with open('HASYv2/hasy-data-labels.csv') as csvfile:
11     csvreader = csv.reader(csvfile)
12     i = 0
13     for row in csvreader:
14         if i > 0:
15             img = keras.preprocessing.image.img_to_array(
16                 pil_image.open("HASYv2/" + row[0]))
17             # neuron activation functions behave best when input
18             # values are between 0.0 and 1.0 (or -1.0 and 1.0),
19             # so we rescale each pixel value to be in the range
20             # 0.0 to 1.0 instead of 0-255
21             img /= 255.0
22             imgs.append((row[0], row[2], img))
23             classes.append(row[2])
24             i += 1

```

- Penjelasan:

Baris Code 1: Membuat variabel imgs tanpa parameter

Baris Code 2: Membuat variabel classes tanpa parameter

Baris Code 3: Membuka file HASYv2/hasy-data-labels.csv

Baris Code 4: Membuat variabel csvreader untuk pembacaan dari file csv yang dimasukkan

Baris Code 5: Membuat variabel i dengan parameter 0

Baris Code 6: Mengeksekusi baris dari pembacaan csv

Baris Code 7: Mengaplikasikan perintah "if" dengan ketentuan variabel i lebih besar dari angka 0, maka akan dilanjutkan ke perintah berikutnya

Baris Code 8: Membuat variabel img yang mengubah image menjadi bentuk array dari file HASYv2 yang dibuka dengan row berparameter 0.

Baris Code 9: Membuat variabel img atau dengan nilai 255.0

Baris Code 10: Mendefinisikan fungsi imgs.append dimana merupakan proses melampirkan atau menggabungkan data dengan file lain atau set data yang ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.

Baris Code 11: Mendefinisikan fungsi append kembali dari variabel classes dengan parameternya row[2].

Baris Code 12: Mendefinisikan fungsi dimana i variabel i akan ditambah nilainya sehingga akan bernilai 1.

- Hasil output:

```
In [2]: imgs = []
...: classes = []
...: with open('HASYv2/hasy-data-labels.csv') as csvfile:
...:     csvreader = csv.reader(csvfile)
...:     i = 0
...:     for row in csvreader:
...:         if i > 0:
...:             img =
...:             keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
...:             # neuron activation functions behave best when input values are
...:             # between 0.0 and 1.0 (or -1.0 and 1.0),
...:             # so we rescale each pixel value to be in the range 0.0 to 1.0
...:             instead of 0-255
...:             img /= 255.0
...:             imgs.append((row[0], row[2], img))
...:             classes.append(row[2])
...:             i += 1
```

Gambar 7.12 2

7.1.2.3 Soal No. 3 Jelaskan arti dari setiap baris kode program pada blok # In[3] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import random
9 random.shuffle(imgs)
10 split_idx = int(0.8*len(imgs))
11 train = imgs[:split_idx]
12 test = imgs[split_idx:]

```

▪ Penjelasan:

Baris Code 1: Memasukkan module random

Baris Code 2: Melakukan pengocokan atau pengacakan pada module random dengan parameter variabelnya imgs

Baris Code 3: Membagi dan memecah index dalam bentuk integer dengan mengkalikan nilai 0,8 dan fungsi len yang akan mengembalikan jumlah item dalam datanya dari variabel imgs

Baris Code 4: Membuat variabel train yang mengeksekusi imgs dengan pemecahan index awal pada data

Baris Code 5: Membuat variabel test yang mengeksekusi imgs dengan pemecahan index akhir pada data

▪ Hasil output:

```

In [3]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]

```

Gambar 7.13 3

7.1.2.4 Soal No. 4 Jelaskan arti dari setiap baris kode program pada blok # In[4] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import numpy as np
9

```

```

10 train_input = np.asarray(list(map(lambda row: row[2], train)))
11 test_input = np.asarray(list(map(lambda row: row[2], test)))
12
13 train_output = np.asarray(list(map(lambda row: row[1], train)))
14 test_output = np.asarray(list(map(lambda row: row[1], test)))

```

▪ Penjelasan:

Baris Code 1: Mengimport library numpy sebagai np

Baris Code 2: Membuat variabel train_input untuk input menjadi sebuah array dari np menggunakan fungsi list untuk mengkoleksikan data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungskan lamda pada row dengan parameter [2] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 3: Membuat variabel test_input dengan fungsi seperti train_input yang membedakan hanya datanya atau inputan yang diproses berasal dari variabel test

Baris Code 4: Membuat variabel train_output untuk mengubah keluaran menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengkoleksi data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungskan lamda pada row dengan parameter[1] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 5: Membuat variabel test_output dengan fungsi yang sama seperti train_output yang membedakan hanya datanya atau inputan yang diproses berasal dari variabel test

▪ Hasil output:

```

In [4]: import numpy as np
...
...: train_input = np.asarray(list(map(lambda row: row[2], train)))
...: test_input = np.asarray(list(map(lambda row: row[2], test)))
...
...: train_output = np.asarray(list(map(lambda row: row[1], train)))
...: test_output = np.asarray(list(map(lambda row: row[1], test)))

```

Gambar 7.14 4

7.1.2.5 Soal No. 5 Jelaskan arti dari setiap baris kode program pada blok # In[5] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020

```

```

4
5 @author: User
6 """
7
8 from sklearn.preprocessing import LabelEncoder
9 from sklearn.preprocessing import OneHotEncoder

```

▪ Penjelasan:

Baris Code 1: Memasukkan modul atau fungsi LabelEncoder dari sklearn.preprocessing untuk dapat digunakan menormalkan label dimana label encoder didefinisikan dengan nilai antara 0 dan n_classes-1.

Baris Code 2: Memasukkan modul atau fungsi OneHotEncoder dari sklearn.preprocessing untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)).

▪ Hasil output:

```
In [5]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.15 5

7.1.2.6 Soal No. 6 Jelaskan arti dari setiap baris kode program pada blok # In[6] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 label_encoder = LabelEncoder()
9 integer_encoded = label_encoder.fit_transform(classes)

```

▪ Penjelasan:

Baris Code 1: Membuat variabel label_encoder dengan modul atau fungsi dari LabelEncoder tanpa parameter

Baris Code 2: Membuat variabel integer_encoded dengan fungsi label_encoder.fit_transform() dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali dari variabel label_encoder.

▪ Hasil output:

```
In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.16 6

7.1.2.7 Soal No. 7 Jelaskan arti dari setiap baris kode program pada blok # In[7] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 onehot_encoder = OneHotEncoder(sparse=False)
9 integer_encoded = integer_encoded.reshape(len(integer_encoded),
10                                         1)
10 onehot_encoder.fit(integer_encoded)
```

- Penjelasan:

Baris 1: Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.

Baris 2: Membuat variabel integer_encoded memanggil variabel integer_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer_encoded.

Baris 3: Onehotencoding melakukan fitting pada integer_encoded.

- Hasil output:

```

In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
D:\Anaconda\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning:
The handling of integer data will change in version 0.22. Currently, the categories
are determined based on the range [0, max(values)], while in the future they will be
determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories
to integers, then you can now use the OneHotEncoder directly.
warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Gambar 7.17 7

7.1.2.8 Soal No. 8 Jelaskan arti dari setiap baris kode program pada blok # In[8] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
```

```

3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 train_output_int = label_encoder.transform(train_output)
9 train_output = onehot_encoder.transform(train_output_int.reshape(
10     len(train_output_int), 1))
11 test_output_int = label_encoder.transform(test_output)
12 test_output = onehot_encoder.transform(test_output_int.reshape(
13     len(test_output_int), 1))
14
15 num_classes = len(label_encoder.classes_)
16 print("Number of classes: %d" % num_classes)

```

▪ Penjelasan:

Baris 1: Membuat variabel train_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel train_output.

Baris 2: Membuat variabel train_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.

Baris 3: Membuat variabel test_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel test_output.

Baris 4: Membuat variabel test_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test_output_int telah dikembalikan.

Baris 5: Membuat variabel num_classes untuk mengetahui jumlah class dari label_encoder

Baris 6: Perintah print digunakan untuk memunculkan hasil dari variabel num_classes

```

In [8]: train_output_int = label_encoder.transform(train_output)
...: train_output =
...: onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
...: test_output_int = label_encoder.transform(test_output)
...: test_output =
...: onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
...:
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369

```

Gambar 7.18 8

▪ Hasil output:

7.1.2.9 Soal No. 9 Jelaskan arti dari setiap baris kode program pada blok # In[9] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 from keras.models import Sequential
9 from keras.layers import Dense, Dropout, Flatten
10 from keras.layers import Conv2D, MaxPooling2D

```

- Penjelasan:

Baris 1: Melakukan importing fungsi model sequential dari library keras.

Baris 2: Melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.

Baris 3: Melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

- Hasil output:

```

In [9]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D

```

Gambar 7.19 9

7.1.2.10 Soal No. 10 Jelaskan arti dari setiap baris kode program pada blok # In[10] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                  input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(1024, activation='tanh'))

```

```
16 model.add(Dropout(0.5))
17 model.add(Dense(num_classes, activation='softmax'))
18
19 model.compile(loss='categorical_crossentropy', optimizer='adam',
20                 metrics=['accuracy'])
21
22 print(model.summary())
```

- Penjelasan:

Baris 1: Melakukan pemodelan Sequential.

Baris 2: Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train input mulai dari baris nol.

Baris 3: Menambahkan Max Pooling dengan matriks 2x2.

Baris 4: Penambahan Konvolusi 2D dengan 32 filter, konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.

Baris 5 Menambahkan Max Pooling dengan matriks 2x2.

Baris 6: Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.

Baris 7: Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .

Baris 8: Untuk output layer menggunakan data dari variabel num classes dengan fungsi activationnya softmax.

Baris 9: Konfigurasi proses pembelajaran, melalui metode compile, sebelum melatih suatu model.

Baris 10: Menampilkan model yang telah dibuat.

- Hasil output:

```

...: model.add(Flatten())
...: model.add(Dense(1024, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...:
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
...:                 metrics=['accuracy'])
...:
...: print(model.summary())
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\framework
\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\keras\backend
\tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with
keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.

Layer (type)          Output Shape         Param #
=====
conv2d_1 (Conv2D)     (None, 30, 30, 32)      896
max_pooling2d_1 (MaxPooling2D) (None, 15, 15, 32)    0
conv2d_2 (Conv2D)     (None, 13, 13, 32)      9248
max_pooling2d_2 (MaxPooling2D) (None, 6, 6, 32)      0
flatten_1 (Flatten)   (None, 1152)           0
dense_1 (Dense)       (None, 1024)          1180672
dropout_1 (Dropout)   (None, 1024)           0
dense_2 (Dense)       (None, 369)            378225
=====
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
=====

None

```

Gambar 7.20 10

7.1.2.11 Soal No. 11 Jelaskan arti dari setiap baris kode program pada blok # In[11] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8
9 import keras.callbacks
10 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
    style')

```

- Penjelasan:

Baris Code 1: Melakukan import library keras.callbacks yang digunakan pada penulisan log untuk TensorBoard, untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian.

Baris Code 2: Membuat variabel tensorboard untuk mendefinisikan fungsi TensorBoard pada keras.callbacks yang digunakan sebagai alat visualisasi yang disediakan dengan TensorFlow. Dan untuk fungsi log_dir memanggil data yaitu './logs/mnist-style'

- Hasil output:

```
In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

Gambar 7.21 11

7.1.2.12 Soal No. 12

Jelaskan arti dari setiap baris kode program pada blok # In[12] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8 model.fit(train_input, train_output,
9           batch_size=32,
10          epochs=10,
11          verbose=2,
12          validation_split=0.2,
13          callbacks=[tensorboard])
14
15 score = model.evaluate(test_input, test_output, verbose=2)
16 print('Test loss:', score[0])
17 print('Test accuracy:', score[1])
```

- Penjelasan:

Baris Code 1: Menerapkan fungsi model.fit yang didalamnya memproses train_input, train_output

Baris Code 2: Penerapan fungsi yang sama difungsikan batch_size apabila batch_size nya tidak ditemukan maka otomatis akan dijadikan nilai 32

Baris Code 3: Penerapan fungsi yang sama, difungsikan epochs dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10

Baris Code 4: Mendefinisikan fungsi verbose untuk digunakan sebagai opsi menghasilkan informasi logging dari data yang ditentukan dengan nilai 2

Baris Code 5: Mendefinisikan fungsi validation_split untuk memecah nilai dari perhitungan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi)

Baris Code 6: Mendefinisikan fungsi callbacks dengan parameteranya yang mengeksekusi tensorboard dimana digunakan untuk visualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses

Baris Code 7: Mendefinisikan variabel score dengan fungsi evaluate dari model dengan parameter test_input, tst_output dan verbose=2 untuk memprediksi output dan input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya

Baris Code 8: Mencetak score optimasi dari test dengan ketentuan nilai parameter 0

Baris Code 9: Mencetak score akurasi dari test dengan ketentuan nilai parameter 1

- Hasil output:

```
In [12]: model.fit(train_input, train_output,
...:         batch_size=32,
...:         epochs=10,
...:         verbose=2,
...:         validation_split=0.2,
...:         callbacks=[tensorboard])
...: score = model.evaluate(test_input, test_output, verbose=2)
...: print('Test loss:', score[0])
...: print('Test accuracy:', score[1])
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 2007s - loss: 1.5334 - acc: 0.6294 - val_loss: 0.9824 - val_acc: 0.7285
Epoch 2/10
- 918s - loss: 0.9694 - acc: 0.7321 - val_loss: 0.9162 - val_acc: 0.7494
Epoch 3/10
- 562s - loss: 0.8543 - acc: 0.7556 - val_loss: 0.8883 - val_acc: 0.7530
Epoch 4/10
- 503s - loss: 0.7854 - acc: 0.7699 - val_loss: 0.8441 - val_acc: 0.7666
Epoch 5/10
- 361s - loss: 0.7364 - acc: 0.7796 - val_loss: 0.8491 - val_acc: 0.7663
Epoch 6/10
- 366s - loss: 0.6936 - acc: 0.7899 - val_loss: 0.8522 - val_acc: 0.7692
Epoch 7/10
- 360s - loss: 0.6591 - acc: 0.7962 - val_loss: 0.8580 - val_acc: 0.7668
Epoch 8/10
- 389s - loss: 0.6344 - acc: 0.8017 - val_loss: 0.8496 - val_acc: 0.7654
Epoch 9/10
- 357s - loss: 0.6076 - acc: 0.8072 - val_loss: 0.8597 - val_acc: 0.7638
Epoch 10/10
- 359s - loss: 0.5905 - acc: 0.8116 - val_loss: 0.8889 - val_acc: 0.7637
Test loss: 0.8794204677150739
Test accuracy: 0.7632181175230488
```

Gambar 7.22 12

7.1.2.13 Soal No. 13 Jelaskan arti dari setiap baris kode program pada blok # In[13] dan hasil luarnya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 import time
9
10 results = []
11 for conv2d_count in [1, 2]:
12     for dense_size in [128, 256, 512, 1024, 2048]:
13         for dropout in [0.0, 0.25, 0.50, 0.75]:
14             model = Sequential()
15             for i in range(conv2d_count):
16                 if i == 0:
```

```

17         model.add(Conv2D(32, kernel_size=(3, 3),
18                         activation='relu', input_shape=np.shape(train_input[0])))
19                         else:
20                             model.add(Conv2D(32, kernel_size=(3, 3),
21                                         activation='relu'))
22                             model.add(MaxPooling2D(pool_size=(2, 2)))
23                             model.add(Flatten())
24                             model.add(Dense(dense_size, activation='tanh'))
25                             if dropout > 0.0:
26                                 model.add(Dropout(dropout))
27                             model.add(Dense(num_classes, activation='softmax'))
28
29         model.compile(loss='categorical_crossentropy',
30                         optimizer='adam', metrics=['accuracy'])
31
32         log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
33 (conv2d_count, dense_size, dropout)
34         tensorboard = keras.callbacks.TensorBoard(log_dir=
log_dir)
35
36         start = time.time()
37         model.fit(train_input, train_output, batch_size=32,
38 epochs=10,
39         verbose=0, validation_split=0.2, callbacks
=[tensorboard])
40         score = model.evaluate(test_input, test_output,
41 verbose=2)
42         end = time.time()
43         elapsed = end - start
44         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))
45         results.append((conv2d_count, dense_size, dropout,
46 score[0], score[1], elapsed))

```

▪ Penjelasan:

Baris 1: Import atau memasukkan modul time

Baris 2: Variabel result berisikan array kosong

Baris 3: Menggunakan convolution 2D yang berarti memiliki 1 atau 2 layer

Baris 4: Mendefinisikan dense size dengan ukuran 128, 256, 512, 1024, 2048

Baris 5: Mendefinsikan drop out dengan 0, 25%, 50%, dan 75%

Baris 6: Melakukan pemodelan Sequential

Baris 7: Jika ini adalah layer pertama, akan memasukkan bentuk input.

Baris 8: Kalau tidak kita hanya akan menambahkan layer.

Baris 9: Kemudian, setelah menambahkan layer konvolusi, lakukan hal yang sama dengan max pooling.

Baris 10: Lalu, ratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense size. Dimana akan selalu menggunakan algoritma tanh

Baris 11: Jika dropout digunakan, akan menambahkan layer dropout. Dropout ini berarti misalnya 50%, bahwa setiap kali memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui

Baris 12: Menempatkan di antara dua lapisan padat untuk dihindarkan dari melindunginya dari overfitting. Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.

Baris 13: Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.

Baris 14: Variabel start akan memanggil modul time

Baris 15: Melakukan fit atau compile

Baris 16: Melakukan scoring dengan evaluate yang akan menampilkan data loss dan accuracy dari model

Baris 17: 'End' merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.

Baris 18: Menampilkan hasil dari skrip diatas

- Hasil output:

```
...:
...:         model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...:
...:         log_dir = './logs/conv2d_%d-dense_%d-dropout_.2f' % (conv2d_count,
dense_size, dropout)
...:         tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
...:
...:         start = time.time()
...:         model.fit(train_input, train_output, batch_size=32, epochs=10,
...:                   verbose=0, validation_split=0.2, callbacks=[tensorboard])
...:         score = model.evaluate(test_input, test_output, verbose=2)
...:         end = time.time()
...:         elapsed = end - start
...:         print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.
2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
...:         results.append((conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.13, Accuracy: 0.74, Time: 1540
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.93, Accuracy: 0.76, Time: 1579
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.81, Accuracy: 0.77, Time: 1599
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.82, Accuracy: 0.77, Time: 1627
```

Gambar 7.23 13

7.1.2.14 Soal No. 14 Jelaskan arti dari setiap baris kode program pada blok # In[14] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                 input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(128, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.add(Dense(num_classes, activation='softmax'))
18 model.compile(loss='categorical_crossentropy', optimizer='adam',
19                 metrics=['accuracy'])
20 print(model.summary())
```

- Penjelasan:

Baris 1: Melakukan pemodelan Sequential

Baris 2: Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input shape

Baris 3: Melakukan Max Pooling 2D dengan ukuran matriks 2x2

Baris 4: Melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik

Baris 5: Flatten digunakan untuk meratakan inputan atau masukkan data

Baris 6: Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.

Baris 7: Dropout sebanyak 50% untuk menghindari overfitting

Baris 8: Menambahkan dense pada model untuk output dimana layerini akan menjadi jumlah dari class yang ada.

Baris 9: Mengcompile model yang didefinisikan diatas

Baris 10: Menampilkan ringkasan dari pemodelan yang dilakukan

- Hasil output:

```
In [14]: model = Sequential()
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
input_shape=np.shape(train_input[0])))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Conv2D(32, (3, 3), activation='relu'))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Flatten())
...: model.add(Dense(128, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...: print(model.summary())

Layer (type)          Output Shape         Param #
=====
```

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_3 (MaxPooling2)	(None, 15, 15, 32)	0
conv2d_4 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_4 (MaxPooling2)	(None, 6, 6, 32)	0
flatten_2 (Flatten)	(None, 1152)	0
dense_3 (Dense)	(None, 128)	147584
dropout_2 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 369)	47601

```
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
```

None

Gambar 7.24 14

7.1.2.15 Soal No. 15 Jelaskan arti dari setiap baris kode program pada blok # In[15] dan hasil luarnanya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.fit(np.concatenate((train_input, test_input)),
9           np.concatenate((train_output, test_output)),
10          batch_size=32, epochs=10, verbose=2)
```

- Penjelasan:

Baris 1: Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

- Hasil output:

```
In [26]: model.fit(np.concatenate((train_input, test_input)),
...:                 np.concatenate((train_output, test_output)),
...:                 batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 247s - loss: 1.7949 - acc: 0.5843
Epoch 2/10
- 236s - loss: 1.0797 - acc: 0.7064
Epoch 3/10
- 235s - loss: 0.9648 - acc: 0.7308
Epoch 4/10
- 237s - loss: 0.9060 - acc: 0.7434
Epoch 5/10
- 237s - loss: 0.8671 - acc: 0.7519
Epoch 6/10
- 236s - loss: 0.8362 - acc: 0.7573
Epoch 7/10
- 238s - loss: 0.8137 - acc: 0.7631
Epoch 8/10
- 239s - loss: 0.7980 - acc: 0.7652
Epoch 9/10
- 239s - loss: 0.7831 - acc: 0.7692
Epoch 10/10
- 239s - loss: 0.7695 - acc: 0.7724
Out[26]: <keras.callbacks.History at 0x14c1941f5f8>
```

Gambar 7.25 15

7.1.2.16 Soal No. 16 Jelaskan arti dari setiap baris kode program pada blok # In[16] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.save("mathsymbols.model")
```

- Penjelasan:

Baris 1: Menyimpan model yang telah di latih dengan nama mathsymbols.model

- Hasil output:

```
In [22]: model.save("mathsymbols.model")
```

Gambar 7.26 16

7.1.2.17 Soal No. 17 Jelaskan arti dari setiap baris kode program pada blok # In[17] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 np.save('classes.npy', label_encoder.classes_)
```

- Penjelasan:

Baris 1: Menyimpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

- Hasil output:

```
In [23]: np.save('classes.npy', label_encoder.classes_)
```

Gambar 7.27 17

7.1.2.18 Soal No. 18 Jelaskan arti dari setiap baris kode program pada blok # In[18] dan hasil luarnya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:35 2020
4
5 @author: User
6 """
7
8 import keras.models
9 model2 = keras.models.load_model("mathsymbols.model")
10 print(model2.summary())
```

- Penjelasan:

Baris 1: Memasukkan atau mengimport models dari librari Keras

Baris 2: Variabel model2 akan memanggil model yang telah disave sebelumnya

Baris 3: Menampilkan ringkasan dari hasil pemodelan

- Hasil output:

```
In [24]: import keras.models
...: model2 = keras.models.load_model("mathsymbols.model")
...: print(model2.summary())

```

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_3 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_4 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_4 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_2 (Flatten)	(None, 1152)	0
dense_3 (Dense)	(None, 128)	147584
dropout_2 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 369)	47601

Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0

None

Gambar 7.28 18

7.1.2.19 Soal No. 19 Jelaskan arti dari setiap baris kode program pada blok # In[19] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:36 2020
4
5 @author: User
6 """
7
8 label_encoder2 = LabelEncoder()
9 label_encoder2.classes_ = np.load('classes.npy')
10
11 def predict(img_path):
12     newimg = keras.preprocessing.image.img_to_array(pil_image.
13         open(img_path))
14     newimg /= 255.0
15
16     # do the prediction
17     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
18
19     # figure out which output neuron had the highest score, and
20     # reverse the one-hot encoding
21     inverted = label_encoder2.inverse_transform([np.argmax(
22         prediction)]) # argmax finds highest-scoring output
23     print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
24         max(prediction)))
```

- Penjelasan:

Baris 1: Memanggil fungsi LabelEncoder

Baris 2: Variabel label encoder akan memanggil class yang disimpan sebelumnya.

Baris 3: Function Predict akan mengubah gambar kedalam bentuk array

Baris 4: Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.

Baris 5: Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi

- Hasil output:

```
In [25]: label_encoder2 = LabelEncoder()
...: label_encoder2.classes_ = np.load('classes.npy')
...:
...: def predict(img_path):
...:     newimg =
keras.preprocessing.image.img_to_array(pil_image.open(img_path))
...:     newimg /= 255.0
...:
...:     # do the prediction
...:     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
...:
...:     # figure out which output neuron had the highest score, and reverse the
one-hot encoding
...:     inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # argmax finds highest-scoring output
...:     print("Prediction: %s, confidence: %.2f" % (inverted[0],
np.max(prediction)))
```

Gambar 7.29 19

7.1.2.20 Soal No. 20 Jelaskan arti dari setiap baris kode program pada blok # In[20] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:54:14 2020
4
5 @author: User
6 """
7
8 predict("HASYv2/hasy-data/v2-00010.png")
9
10 predict("HASYv2/hasy-data/v2-00500.png")
11
12 predict("HASYv2/hasy-data/v2-00700.png")
```

- Penjelasan:

- Baris 1: Melakukan prediksi dari pelatihan dari gambar v2-00010.png
 Baris 2: Melakukan prediksi dari pelatihan dari gambar v2-00500.png
 Baris 3: Melakukan prediksi dari pelatihan dari gambar v2-00700.png

- Hasil output:

```
In [26]: predict("HASYv2/hasy-data/v2-00010.png")
...
...
...
...
Prediction: \pitchfork, confidence: 0.00
Prediction: \copyright, confidence: 0.00
Prediction: J, confidence: 0.00
```

Gambar 7.30 20

7.1.3 Penanganan Error

7.1.3.1 Penanganan Error

- Screenshoot:

```
Traceback (most recent call last):
File "<ipython-input-3-fd9abfd31556>", line 1, in <module>
    model.fit(np.concatenate((train_input, test_input)),
NameError: name 'model' is not defined
```

Gambar 7.31 Error

- Code Error:

NameError: name 'model' is not defined

- Penanganan Error:

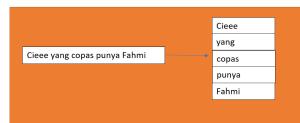
Berdasarkan error maka penyelesaiannya ialah melakukan pendefinisian variabel model sehingga code dapat dijalankan. Lakukan running pada code yang berada diatasnya dimana mendefinisikan variabel model.

7.2 1174021 - Muhammad Fahmi

7.2.1 Soal Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
 Karena MTokenizer merupakan proses membagi teks yang dapat berupa kalimat, paragraf atau dokumen menjadi kata-kata atau bagian-bagian tertentu dalam

kalimat tersebut. Sebagai contoh dari kalimat "Cieeee yang copas punya fahmi", kalimat itu menjadi beberapa bagian yaitu "cieeee", "yang", "copas", "punya", "fahmi". Yang menjadi acuan yakni tanda baca dan spasi. Untuk ilustrasi, lihat gambar berikut:

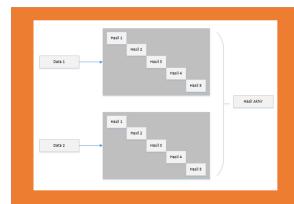


Gambar 7.32 Teori 1

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1. dilengkapi dengan ilustrasi atau gambar.

```
1 kfold = StratifiedKFold(n_splits=5)
2
```

Konsep sederhana dari K Fold Cross Validation ialah Pada code tersebut terdapat kfold yang bertujuan untuk melakukan split data menjadi 5 bagian dari dataset komentar Youtube tersebut. Sehingga dari setiap data yang sudah dibagi tersebut akan menghasilkan persentase dari setiap bagiannya, untuk menghasilkan hasil akhir dengan persentase yang cukup baik. Untuk ilustrasi, lihat gambar berikut:



Gambar 7.33 Teori 2

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

Untuk penjelasannya yaitu For train berfungsi untuk membagi data tersebut menjadi data training. Sedangkan test in splits berfungsi untuk menguji apakah dataset tersebut sudah dibagi menjadi beberapa bagian atau masih menumpuk. Untuk ilustrasi, lihat gambar berikut:

**Gambar 7.34** Teori 3

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.

Fungsi dalam kode tersebut berfungsi untuk mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Contoh sederhananya ketika data telah diubah menjadi data train dan data test maka kita dapat memilihnya untuk ditampilkan pada kolom yang di inginkan. Namun, untuk ilustrasi lihat gambar berikut:

```

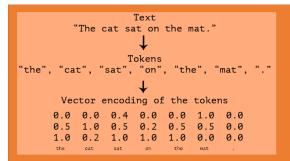
1 import pandas as pd
2
3 mydict = [{ 'a': 1, 'b': 2, 'c': 3, 'd': 4},
4           { 'a': 100, 'b': 200, 'c': 300, 'd': 400},
5           { 'a': 1000, 'b': 2000, 'c': 3000, 'd': 4000 }]
6 df = pd.DataFrame(mydict)
7 df

```

a	1
b	2
c	3
d	4
Name: 0, dtype: int64	

Gambar 7.35 Teori 4

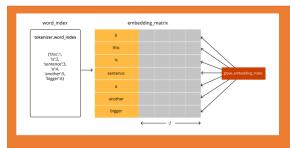
5. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar. Fungsi tokenizer ini berfungsi untuk melakukan vektorisasi data kedalam bentuk token sebanyak 2000 kata. Dan selanjutnya akan melakukan fit tokenizer hanya untuk data training saja tidak dengan data testingnya. Untuk ilustrasi lihat gambar berikut:



Gambar 7.36 Teori 5

6. 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.

Maksud dari baris diatas ialah untuk memasukkan text ke sebuah matrix dengan mode tfidf dan menginputkan data testing untuk di terjemahkan ke sebuah matriks. Untuk ilustrasi, lihat gambar berikut:



Gambar 7.37 Teori 6

7. 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 `np.amax` adalah nilai Maksimal. Jika sumbu tidak ada, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi `a.ndim - 1`. Untuk ilustrasi, lihat gambar berikut:

```

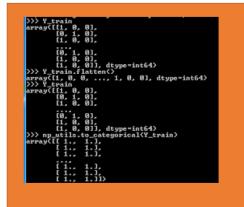
>>> a = np.arange(4).reshape((2,2))
>>> a
array([[0, 1],
       [2, 3]])
>>> np.amax(a)
# Maximum of the flattened array
3
>>> np.amax(a, axis=0) # Maxima along the first axis
array([1, 1])
>>> np.amax(a, axis=1) # Maxima along the second axis
array([1, 3])
>>> np.argmax(a, where=[False, True], initial=-1, axis=0)
array([-1, 3])
>>> b = np.arange(5, dtype=float)
>>> b[2] = np.nan
>>> b
array([ 0. ,  1. ,  nan,  3. ,  4. ])
>>> np.nanmax(b, initial=-1)
4.0
>>> np.nanmin(b)
4.0

```

Gambar 7.38 Teori 7

8. 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.

Fungsi dari baris kode tersebut ialah membuat train outputs dengan kategori dari class lalu dengan ketentuan iloc train idx. Kemudian membuat keluaran sebagai output. Untuk ilustrasi, lihat gambar berikut:



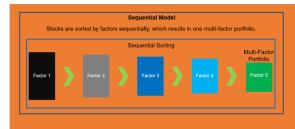
```
>>> Y_train
array([[ 1,  1,  0],
       [ 1,  1,  0],
       [ 1,  1,  0],
       [ 1,  1,  0],
       [ 1,  0,  0],
       [ 1,  0,  0], dtype=int64)
>>> Y_train[Y_train > 0]
array([ 1,  1,  0,  1,  0,  0], dtype=int64)
>>> np_utils.to_categorical(Y_train)
array([[ 1.,  1.,  1.],
       [ 1.,  1.,  1.],
       [ 1.,  1.,  1.],
       [ 1.,  1.,  1.],
       [ 1.,  1.,  1.],
       [ 1.,  1.,  1.]])
```

Gambar 7.39 Teori 8

9. Jelaskan apa maksud dari fungsi di listing 7.2. 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'))
```

Fungsi dari baris kode tersebut ialah model perlu mengetahui bentuk input apa yang harus diharapkan. Untuk alasan ini, lapisan pertama dalam model Sequential (dan hanya yang pertama, karena lapisan berikut dapat melakukan inferensi bentuk otomatis) perlu menerima informasi tentang bentuk inputnya.



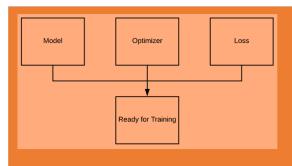
Gambar 7.40 Teori 9

10. Jelaskan apa maksud dari fungsi di listing 7.3 dengan parameter tersebut.

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

Fungsi dari baris kode tersebut ialah bisa meneruskan nama fungsi loss yang ada, atau melewati fungsi simbolis TensorFlow yang mengembalikan skalar untuk setiap titik data dan mengambil dua argumen y_true: True label. dan y_pred:

Prediksi. Tujuan yang dioptimalkan sebenarnya adalah rata-rata dari array output di semua titik data.



Gambar 7.41 Teori 10

11. Jelaskan apa itu Deep Learning.

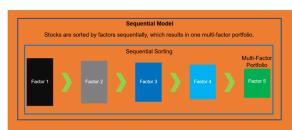
Deep Learning adalah salah satu cabang dari ilmu machine learning yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam machine learning dapat digunakan baik untuk supervised learning dan unsupervised learning.

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

DNN adalah salah satu algoritma berbasis jaringan saraf tiruan yang memiliki dari 1 lapisan saraf tersembunyi yang dapat digunakan untuk pengambilan keputusan. Perbedaannya dengan deep learning, yakni: DNN dapat menentukan dan mencerna karakteristik tertentu di suatu rangkaian data, kapabilitas lebih kompleks untuk mempelajari, mencerna, dan mengklasifikasikan data, serta dibagi ke dalam berbagai lapisan dengan fungsi yang berbeda-beda.

13. Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1) x (NPM mod3+1) yang terdapat max pooling.

1174021 mod3+1 x 1174021 mod3+1 = 2 x 2, adapun ilustrasi gambar nya sebagai berikut :



Gambar 7.42 Teori 9

7.2.2 Praktek Program

1. Soal 1

```

1 # In[1]:import lib
2 # mengimport library CSV untuk mengolah data ber ekstensi csv
3 import csv
4 #kemudian mengimport librari Image yang berguna untuk dari PIL
      atau Python Imaging Library yang berguna untuk mengolah data
      berupa gambar
5 from PIL import Image as pil_image
6 # kemudian mengimport librari keras yang menggunakan method
      preprocessing yang digunakan untuk membuat neutal network
7 import keras.preprocessing.image

```

Kode di atas menjelaskan tentang library yang akan di pakai yaitu import file csv, lalu load module pil image dan juga import library keras, hasilnya adalah sebagai berikut:

```

In [1]:import CSV
      #mengimport librari Image yang berguna untuk dari PIL atau Python
      #Imaging Library yang berguna untuk mengolah data berupa gambar
      # kemudian mengimport librari keras yang menggunakan method preprocessing
      #yang digunakan untuk membuat neutal network
      #import keras.preprocessing.image
Using TensorFlow backend.

```

Gambar 7.43 Hasil Soal 1.

2. Soal 2

```

1 # In[2]:load all images (as numpy arrays) and save their classes
2 #membuat variabel imgs dengan variabel kosong
3 imgs = []
4 #membuat variabel classes dengan variabel kosong
5 classes = []
6 #membuaka file hasy-data-labels.csv yang berada di foleder HASYv2
      yang di inisialisasi menjadi csvfile
7 with open('HASYv2/hasy-data-labels.csv') as csvfile:
8     #membuat variabel csvreader yang berisi method csv.reader
      yang membaca variabel csvfile
9     csvreader = csv.reader(csvfile)
10    # membuat variabel i dengan isi 0
11    i = 0
12    # membuat looping pada variabel csvreader
13    for row in csvreader:
14        # dengan ketentuan jika i lebihkecil daripada o
15        if i > 0:
16            # 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
17            img = keras.preprocessing.image.img_to_array(
18                pil_image.open("HASYv2/" + row[0]))
19                # neuron activation functions behave best when input
                  values are between 0.0 and 1.0 (or -1.0 and 1.0),
                  # so we rescale each pixel value to be in the range
                  0.0 to 1.0 instead of 0-255

```

```

20         #membagi data yang ada pada fungsi img sebanyak 255.0
21         img /= 255.0
22         # menambah nilai baru pada imgs pada row ke 1 2 dan
23         dilanjutkan dengan variabel img
24         imgs.append((row[0], row[2], img))
25         # menambahkan nilai pada row ke 2 pada variabel
26         classes
27             classes.append(row[2])
28             # penambahan nilai satu pada variabel i
29             i += 1

```

Kode di atas akan menampilkan hasil dari proses load dataset dari HASYv2 sebagai file csv, membuat variabel i dengan parameter 0, lalu perintah perulangan if dengan $i \neq 0$, variabel img dengan nilai 255.0, imgs.append yaitu proses melampirkan atau menggabungkan data dengan file. Berikut adalah hasil setelah saya lakukan running dan pembacaan file audio :



Gambar 7.44 Hasil Soal 2.

3. Soal 3

```

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

```

Perintah import random yaitu sebagai library untuk menghasilkan nilai acak, disini kita membuat data menjadi 2 bagian yaitu 80% sebagai training dan 20% sebagai data testing. Hasilnya adalah sebagai berikut :

```

In [4]: import random
.... random.shuffle(imgs)
.... split_idx = int(0.8*len(imgs))
.... train = imgs[:split_idx]
.... test = imgs[split_idx:]

```

Gambar 7.45 Hasil Soal 3.

4. Soal 4

```

1 # In[4]:
2 # mengimport librari numpy dengan inisial np
3 import numpy as np
4 # membuat 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 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 # membuat 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 # membuat 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)))

```

Kode di atas dapat digunakan untuk melakukan fungsi yang sebelumnya telah kita lakukan yaitu dengan membuat variabel baru untuk menampung data train input dan test input lalu keluarannya sebagai output. Hasilnya adalah sebagai berikut :

```

In [5]: import numpy as np
... train_input = np.asarray([list(map(lambda row: row[1], train))]
... test_input = np.asarray([list(map(lambda row: row[1], test))])
... train_output = np.asarray([list(map(lambda row: row[1], train))])
... test_output = np.asarray([list(map(lambda row: row[1], test))])

```

Gambar 7.46 Hasil Soal 4.

5. Soal 5

```

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

```

Kode diatas untuk melakukan import library yang berfungsi sebagai label encoder dan one hot encoder. Hasilnya adalah sebagai berikut :

```

In [6]: from sklearn.preprocessing import LabelEncoder
... from sklearn.preprocessing import OneHotEncoder

```

Gambar 7.47 Hasil Soal 5.

6. Soal 6

```

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

```

Kode diatas berfungsi untuk melakukan convert class names ke one-hot encoding. Hasilnya adalah sebagai berikut :

```
In [47]: label_encoder = LabelEncoder()
... # membuat variabel integer_encoded yang
berfungsi untuk mengkonvert variabel classes kedalam
bentuk integer
... integer_encoded =
label_encoder.fit_transform(classes)
```

Gambar 7.48 Hasil Soal 6.

7. Soal 7

```
1 # In[7]: then convert integers into one-hot encoding
2 # membuat 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 # mengkonvert variabel integer_encoded kedalam onehot_encoder
7 onehot_encoder.fit(integer_encoded)
```

Kode di atas befungsi untuk melakukan convert integers ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

```
In [8]: onehot_encoder = OneHotEncoder(sparse=False)
... integer_encoded = integer_encoded.reshape(-1, (integer_encoded, ))
... onehot_encoder.fit(integer_encoded)
Out[8]:
OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
handle_unknown='error', sparse=False)
```

Gambar 7.49 Hasil Soal 7.

8. Soal 8

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

Kode di atas befungsi untuk melakukan convert train dan test output ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

```
In [9]: train_output_int = label_encoder.transform(train_output)
train_output_int =
[0 0 0 ... 0 0 0] # array of integers where each integer represents a class index (0-399)
test_output_int = label_encoder.transform(test_output)
test_output_int =
[0 0 0 ... 0 0 0] # array of integers where each integer represents a class index (0-399)
# now_classes = len(label_encoder.classes_)
# now_classes
number_of_classes: 399
```

Gambar 7.50 Hasil Soal 8.

9. Soal 9

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

Kode di atas befungsi untuk melakukan import sequential dari library keras. Hasilnya adalah sebagai berikut :

```
In [10]: from keras.models import Sequential
... from keras.layers import Dense, Dropout, Flatten
... from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.51 Hasil Soal 9.

10. Soal 10

```
1 # In[10]: desain jaringan
2 # membuat variabel model dengan isian librari Sequential
3 model = Sequential()
4 # variabel model di tambahkan librari Conv2D tigapuluhan 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 pixcel
8 model.add(MaxPooling2D(pool_size=(2, 2)))
9 # variabel model di tambahkan dengan librari 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 pixcel
12 model.add(MaxPooling2D(pool_size=(2, 2)))
13 # variabel model di tambahkan librari Flatten
14 model.add(Flatten())
15 # variabel model di tambahkan librari Dense dengan fungsi tanh
16 model.add(Dense(1024, activation='tanh'))
```

```

17 # variabel model di tambahkan librari dropout untuk memangkas
    data tree sebesar 50 persen
18 model.add(Dropout(0.5))
19 # variabel model di tambahkan librari Dense dengan data dari
    num_classes dan fungsi softmax
20 model.add(Dense(num_classes, activation='softmax'))
21 # mengkompile data model untuk mendapatkan data loss akurasi dan
    optimasi
22 model.compile(loss='categorical_crossentropy', optimizer='adam',
    metrics=['accuracy'])
23 # mencetak variabel model kemudian memunculkan kesimpulan berupa
    data total parameter, trainable paremeter dan bukan trainable
    parameter
24 print(model.summary())
25

```

Kode di atas befungsi untuk melihat detail desain pada jaringan model yang telah di definisikan. Hasilnya adalah sebagai berikut :

```

        .....model.add(conv2d_0, kernel_size=(3, 3), activation="relu")
        .....model.add(conv2d_1, (3, 3), activation="relu"))
        .....model.add(maxPooling2d_0(pool_size=(2, 2)))
        .....model.add(conv2d_2, (3, 3), activation="relu"))
        .....model.add(maxPooling2d_1(pool_size=(2, 2)))
        .....model.addDense(1, activation="tanh"))
        .....model.add(Dense(10, activation="softmax"))
        .....model.compile(loss="categorical_crossentropy", optimizer="adam",
    metrics=[model.summary()])
Model: sequential_5

```

Layer (Type)	Output Shape	Param #
conv2d_0 (Conv2D)	(None, 10, 10, 32)	896
max_pooling2d_0 (MaxPooling2D)	(None, 5, 5, 32)	0
conv2d_1 (Conv2D)	(None, 5, 5, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 5, 5, 32)	0
flatten_0 (Flatten)	(None, 1562)	0
dense_0 (Dense)	(None, 28)	437584
dropout_4 (Dropout)	(None, 28)	0
dense_10 (Dense)	(None, 69)	47601

Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0

Gambar 7.52 Hasil Soal 10.

11. Soal 11

```

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

```

Kode di atas befungsi untuk melakukan load callbacks pada library keras. Hasilnya adalah sebagai berikut :

```

In [12]: import keras.callbacks
         tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')

```

Gambar 7.53 Hasil Soal 11.

12. Soal 12

```

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])

```

Kode di atas befungsi untuk melakukan load epoch yang akan di training dan diberikan hasil selama 10 kali epoch. Hasilnya adalah sebagai berikut :

```

Epoch 0/10
1/10 [====] loss: 1.5816 - val_loss: 1.4932 - val_accuracy: 0.7182
WARNING:tensorflow:From C:\Users\Arafat\PycharmCond3\white package\keras\utils\batch_
processing.py:57: The name tf.summary is deprecated. Please use tf.compat.v1.summary
instead.
Epoch 1/10
1/10 [====] loss: 0.7481 - accuracy: 0.7248 - val_loss: 0.4669 - val_accuracy: 0.7523
Epoch 2/10
1/10 [====] loss: 0.7481 - accuracy: 0.7477 - val_loss: 0.4852 - val_accuracy: 0.7562
Epoch 3/10
1/10 [====] loss: 0.6228 - accuracy: 0.7632 - val_loss: 0.4736 - val_accuracy: 0.7629
Epoch 4/10
1/10 [====] loss: 0.7251 - accuracy: 0.7716 - val_loss: 0.4829 - val_accuracy: 0.7618
Epoch 5/10
1/10 [====] loss: 0.7356 - accuracy: 0.7804 - val_loss: 0.4869 - val_accuracy: 0.7642
Epoch 6/10
1/10 [====] loss: 0.7364 - accuracy: 0.7819 - val_loss: 0.4866 - val_accuracy: 0.7654
Epoch 7/10
1/10 [====] loss: 0.4765 - accuracy: 0.7924 - val_loss: 0.4849 - val_accuracy: 0.7679
Epoch 8/10
1/10 [====] loss: 0.4532 - accuracy: 0.7975 - val_loss: 0.4735 - val_accuracy: 0.7681
Epoch 9/10
1/10 [====] loss: 0.4513 - accuracy: 0.8021 - val_loss: 0.4792 - val_accuracy: 0.7651
test loss: 0.7094515808903445
test accuracy: 0.7094515808903445

```

Gambar 7.54 Hasil Soal 12.

13. Soal 13

```

1 # In[13]: try various model configurations and parameters to find
   the best
2 # mengimport librari time
3 import time
4 #membuat 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            # membuat 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                    # menambahkan method add pada variabel model
           dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat

```

```
kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data
shape yang di hitung dari data train.
    model.add(Conv2D(32, kernel_size=(3, 3),
activation='relu', input_shape=np.shape(train_input[0])))
        # jika tidak
    else:
        # menambahkan method add pada variabel model
dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3
dan fungsi aktivasi relu
    model.add(Conv2D(32, kernel_size=(3, 3),
activation='relu'))
        # menambahkan method add pada variabel model
dengan isian method Max pooling berdimensi 2 dengan ukuran
fixcel 2 x 2.
    model.add(MaxPooling2D(pool_size=(2, 2)))
        # merubah feature gambar menjadi 1 dimensi vektor
    model.add(Flatten())
        # menambahkan method dense untuk pematatan data
dengan ukuran dense di tentukan dengan rumus fungsi tanh.
    model.add(Dense(dense_size, activation='tanh'))
        # membuat ketentuan jika pemangkasan lebih besar dari
0 persen
    if dropout > 0.0:
        # menambahkan method dropout pada model dengan
nilai dari dropout
        model.add(Dropout(dropout))
        # menambahkan method dense dengan fungsi num
classs dan rumus softmax
        model.add(Dense(num_classes, activation='softmax'))
        # mongkompile variabel model dengan hasi loss
optimasi dan akurasi matrix
        model.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])
        # melakukan log pada dir
        log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
(conv2d_count, dense_size, dropout)
        # membuat variabel tensorflow dengan isian dari
librari keras dan nilai dari lig dir
        tensorboard = keras.callbacks.TensorBoard(log_dir=
log_dir)
        # membuat variabel start dengan isian dari librari
time menggunakan method time

        start = time.time()
        # menambahkan 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.
        model.fit(train_input, train_output, batch_size=32,
epochs=10,
            verbose=0, validation_split=0.2, callbacks
=[tensorboard])
        # membuat variabel score dengan nilai evaluasi dari
model menggunakan data tes input dan tes output
        score = model.evaluate(test_input, test_output,
verbose=2)
```

```

50     # membuat variabel end
51     end = time.time()
52     # membuat variabel elapsed
53     elapsed = end - start
54     # mencetak hasil perhitungan
55     print("Conv2D count: %d, Dense size: %d, Dropout: %.2f
      f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count
      , dense_size, dropout, score[0], score[1], elapsed))
      results.append((conv2d_count, dense_size, dropout,
      score[0], score[1], elapsed))

```

Kode di atas befungsi untuk melakukan konfigurasi model dengan parameter yang ditentukan dan mencoba mencari data yang terbaik. Hasilnya adalah sebagai berikut :

```

conv2d count: 1, dense size: 128, Dropout: 0.00 Loss: 1.19, Accuracy: 0.74, Time: 419 sec
conv2d count: 1, dense size: 128, Dropout: 0.50 Loss: 0.86, Accuracy: 0.74, Time: 450 sec
conv2d count: 1, dense size: 128, Dropout: 0.75 Loss: 0.82, Accuracy: 0.74, Time: 451 sec
conv2d count: 1, dense size: 256, Dropout: 0.00 Loss: 1.19, Accuracy: 0.74, Time: 450 sec
conv2d count: 1, dense size: 256, Dropout: 0.50 Loss: 0.86, Accuracy: 0.74, Time: 451 sec
conv2d count: 1, dense size: 256, Dropout: 0.75 Loss: 0.82, Accuracy: 0.74, Time: 451 sec
conv2d count: 1, dense size: 512, Dropout: 0.00 Loss: 1.09, Accuracy: 0.74, Time: 461 sec
conv2d count: 1, dense size: 512, Dropout: 0.50 Loss: 0.86, Accuracy: 0.74, Time: 461 sec
conv2d count: 1, dense size: 512, Dropout: 0.75 Loss: 0.82, Accuracy: 0.74, Time: 461 sec
conv2d count: 1, dense size: 1024, Dropout: 0.00 Loss: 1.09, Accuracy: 0.74, Time: 471 sec
conv2d count: 1, dense size: 1024, Dropout: 0.50 Loss: 0.86, Accuracy: 0.74, Time: 471 sec
conv2d count: 1, dense size: 1024, Dropout: 0.75 Loss: 0.82, Accuracy: 0.74, Time: 471 sec
conv2d count: 1, dense size: 2048, Dropout: 0.00 Loss: 2.19, Accuracy: 0.72, Time: 512 sec
conv2d count: 1, dense size: 2048, Dropout: 0.50 Loss: 1.09, Accuracy: 0.72, Time: 512 sec
conv2d count: 1, dense size: 2048, Dropout: 0.75 Loss: 0.86, Accuracy: 0.72, Time: 512 sec
conv2d count: 2, dense size: 128, Dropout: 0.25 Loss: 0.79, Accuracy: 0.77, Time: 579 sec
conv2d count: 2, dense size: 128, Dropout: 0.50 Loss: 0.77, Accuracy: 0.77, Time: 579 sec
conv2d count: 2, dense size: 128, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 579 sec
conv2d count: 2, dense size: 256, Dropout: 0.25 Loss: 0.79, Accuracy: 0.77, Time: 579 sec
conv2d count: 2, dense size: 256, Dropout: 0.50 Loss: 0.77, Accuracy: 0.77, Time: 579 sec
conv2d count: 2, dense size: 256, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 579 sec
conv2d count: 2, dense size: 512, Dropout: 0.25 Loss: 0.86, Accuracy: 0.77, Time: 581 sec
conv2d count: 2, dense size: 512, Dropout: 0.50 Loss: 0.79, Accuracy: 0.77, Time: 581 sec
conv2d count: 2, dense size: 512, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 581 sec
conv2d count: 2, dense size: 1024, Dropout: 0.25 Loss: 1.09, Accuracy: 0.75, Time: 581 sec
conv2d count: 2, dense size: 1024, Dropout: 0.50 Loss: 0.86, Accuracy: 0.75, Time: 581 sec
conv2d count: 2, dense size: 1024, Dropout: 0.75 Loss: 0.79, Accuracy: 0.75, Time: 581 sec
conv2d count: 2, dense size: 2048, Dropout: 0.25 Loss: 0.79, Accuracy: 0.77, Time: 587 sec
conv2d count: 2, dense size: 2048, Dropout: 0.50 Loss: 0.77, Accuracy: 0.77, Time: 587 sec
conv2d count: 2, dense size: 2048, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 587 sec
conv2d count: 3, dense size: 128, Dropout: 0.25 Loss: 0.79, Accuracy: 0.77, Time: 589 sec
conv2d count: 3, dense size: 128, Dropout: 0.50 Loss: 0.77, Accuracy: 0.77, Time: 589 sec
conv2d count: 3, dense size: 128, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 589 sec
conv2d count: 3, dense size: 256, Dropout: 0.25 Loss: 0.79, Accuracy: 0.77, Time: 589 sec
conv2d count: 3, dense size: 256, Dropout: 0.50 Loss: 0.77, Accuracy: 0.77, Time: 589 sec
conv2d count: 3, dense size: 256, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 589 sec
conv2d count: 3, dense size: 512, Dropout: 0.25 Loss: 0.86, Accuracy: 0.77, Time: 591 sec
conv2d count: 3, dense size: 512, Dropout: 0.50 Loss: 0.79, Accuracy: 0.77, Time: 591 sec
conv2d count: 3, dense size: 512, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 591 sec
conv2d count: 3, dense size: 1024, Dropout: 0.25 Loss: 1.09, Accuracy: 0.75, Time: 591 sec
conv2d count: 3, dense size: 1024, Dropout: 0.50 Loss: 0.86, Accuracy: 0.75, Time: 591 sec
conv2d count: 3, dense size: 1024, Dropout: 0.75 Loss: 0.79, Accuracy: 0.75, Time: 591 sec
conv2d count: 3, dense size: 2048, Dropout: 0.25 Loss: 0.79, Accuracy: 0.77, Time: 597 sec
conv2d count: 3, dense size: 2048, Dropout: 0.50 Loss: 0.77, Accuracy: 0.77, Time: 597 sec
conv2d count: 3, dense size: 2048, Dropout: 0.75 Loss: 0.79, Accuracy: 0.77, Time: 597 sec

```

Gambar 7.55 Hasil Soal 13.

14. Soal 14

```

1 # In[14]: rebuild/retrain a model with the best parameters (from
      the search) and use all data
2 # membuat variabel model dengan isian librari Sequential
3 model = Sequential()
4 # variabel model di tambahkan librari Conv2D tigapuluhan 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 pixcel
7 model.add(MaxPooling2D(pool_size=(2, 2)))
8 # variabel model di tambahkan dengan librari 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 pixcel
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 # variabel model di tambahkan librari Flatten
13 model.add(Flatten())
14 # variabel model di tambahkan librari Dense dengan fungsi tanh
15 model.add(Dense(128, activation='tanh'))

```

```

16 # variabel model di tambahkan librari dropout untuk memangkas
    data tree sebesar 50 persen
17 model.add(Dropout(0.5))
18 # variabel model di tambahkan librari Dense dengan data dari
    num_classes dan fungsi softmax
19 model.add(Dense(num_classes, activation='softmax'))
20 # mengkompile 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 paremeter dan bukan trainable
    parameter
23 print(model.summary())

```

Kode di atas befungsi untuk membuat data training kembali dengan model yang sudah di tentukan dan mencari yang terbaik dari hasil training tersebut. Hasilnya adalah sebagai berikut :

Layer (Type)	Output Shape	Param #
conv2d_0 (Conv2D)	(None, 36, 36, 32)	896
max_pooling2d_0 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_1 (Conv2D)	(None, 11, 11, 32)	9248
max_pooling2d_1 (MaxPooling2D)	(None, 6, 6, 32)	0
Flatten_5 (Flatten)	(None, 1152)	0
dense_9 (Dense)	(None, 128)	147584
dropout_4 (Dropout)	(None, 128)	0
dense_10 (Dense)	(None, 36)	47681

Total params: 265,129
Trainable params: 205,129
Non-trainable params: 60,000

Gambar 7.56 Hasil Soal 14.

15. Soal 15

```

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)

```

Kode di atas befungsi untuk melakukan join terhadap data train dan test dengan seluruh konfigurasi yang telah di tentukan sebelumnya. Hasilnya adalah sebagai berikut :

```

1 [16]: model.fit(ep.concatenate((train_input, test_input)),
...                np.concatenate((train_output, test_output)),
...                batch_size=1, epochs=10, verbose=1)
Epoch 1/10
  126s - loss: 1.7795 - accuracy: 0.5871
Epoch 2/10
  126s - loss: 1.6744 - accuracy: 0.7074
Epoch 3/10
  138s - loss: 0.9564 - accuracy: 0.7320
Epoch 4/10
  115s - loss: 0.8977 - accuracy: 0.7458
Epoch 5/10
  118s - loss: 0.8573 - accuracy: 0.7527
Epoch 6/10
  116s - loss: 0.8297 - accuracy: 0.7586
Epoch 7/10
  126s - loss: 0.8075 - accuracy: 0.7632
Epoch 8/10
  126s - loss: 0.7917 - accuracy: 0.7663
Epoch 9/10
  129s - loss: 0.7765 - accuracy: 0.7797
Epoch 10/10
  118s - loss: 0.7637 - accuracy: 0.7718
<keras.callbacks.History at 0x21637ebc3d8>

```

Gambar 7.57 Hasil Soal 15.**16. Soal 16**

```

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

```

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai mathsymbols.model. Hasilnya adalah sebagai berikut :

```

# save the trained model
model.save("mathsymbols.model")

```

Gambar 7.58 Hasil Soal 16.**17. Soal 17**

```

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

```

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai classes.npy dengan reverse ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

```

# save Label encoder (to reverse one-hot encoding)
np.save('classes.npy', label_encoder.classes_)

```

Gambar 7.59 Hasil Soal 17.**18. Soal 18**

```

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 # mengimpport librari keras model
4 import keras.models

```

```

5 # membuat 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())

```

Kode di atas befungsi untuk melakukan load data yang sudah di train dan juga memprediksikan hasil. Hasilnya adalah sebagai berikut :

Layer (Type)	Output Shape	Param #
conv2d_68 (Conv2D)	(None, 10, 10, 32)	896
max_pooling2d_68 (MaxPooling2D)	(None, 5, 5, 32)	0
conv2d_69 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_69 (MaxPooling2D)	(None, 6, 6, 32)	0
Flatten_45 (Flatten)	(None, 312)	0
dense_89 (Dense)	(None, 128)	147584
dropout_34 (Dropout)	(None, 128)	0
dense_90 (Dense)	(None, 369)	47681

Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
None

Gambar 7.60 Hasil Soal 18.

19. Soal 19

```

1 # In[19]: restore the class name to integer encoder
2 # membuat variabel label encoder ke 2 dengan isian fungsi label
  encoder.
3 label_encoder2 = LabelEncoder()
4 # menambahkan 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     # membuat variabel newimg dengan membuat immage 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    # membuat 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    # membuat variabel inverted dengan 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)))
```

Kode di atas befungsi untuk melakukan restore terhadap nama class pada fungsi integer encoder, dengan membuat fungsi predict. Hasilnya adalah sebagai berikut :

Layer (Type)	Output Shape	Param #
conv2d_68 (Conv2D)	(None, 10, 10, 32)	896
max_pooling2d_68 (MaxPooling)	(None, 5, 5, 32)	0
conv2d_69 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_69 (MaxPooling)	(None, 6, 6, 32)	0
flatten_45 (Flatten)	(None, 1152)	0
dense_89 (Dense)	(None, 128)	147584
dropout_34 (Dropout)	(None, 128)	0
dense_90 (Dense)	(None, 369)	47601
Total params: 205,229		
Trainable params: 205,229		
Non-trainable params: 0		
None		

Gambar 7.61 Hasil Soal 19.

20. Soal 20

```
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")
```

Kode di atas befungsi untuk menunjukkan hasil akhir prediski. Hasilnya adalah sebagai berikut :

```
# grab an image (we'll just use a random training image for demonstration purposes)
predict("HASYv2/hasy-data/v2-00010.png")
prediction: A_4, confidence: 0.47
predict("HASYv2/hasy-data/v2-00500.png")
prediction: V_2, confidence: 0.56
predict("HASYv2/hasy-data/v2-00700.png")
prediction: V_alpha, confidence: 0.88
```

Gambar 7.62 Hasil Soal 20.

7.2.3 Penanganan Error

1. KeyboardInterrupt

```
file = open('input/00-distracteddriving_1.jpg', 'rb')
img = keras.preprocessing.image.load_img(file, target_size=(256, 256))
x = img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)
fp = multilabel_confusion_matrix(y_true, y_pred)
KeyboardInterrupt
```

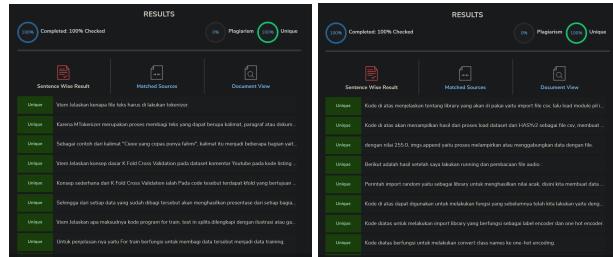
Gambar 7.63 KeyboardInterrupt

2. Cara Penanganan Error

- KeyboardInterrupt

Error tersebut karena disebabkan oleh human typo yang melakukan klik pada keyboard saat running.

7.2.4 Bukti Tidak Plagiat



Gambar 7.64 Bukti Tidak Melakukan Plagiat Chapter 7

7.3 1174026 Felix Lase

7.3.1 Teori

7.3.1.1 Soal No. 1

Kenapa file teks harus dilakukan tokenizer, dilengkapi dengan ilustrasi atau gambar.

Karena tokenizer ini berfungsi untuk mengkonversi teks menjadi urutan integer indeks kata atau vektor binary, word count atau tf-idf. Text harus dilakukan tokenizer agar dapat dirubah menjadi vektor. Dari perubahan ke vektor tersebut maka data/textnya dapat dibaca oleh komputer (terkomputerisasi).

- Ilustrasi Gambar:

7.3.1.2 Soal No. 2

Konsep dasar K Fold Cross Validation pada dataset komentar Youtube, dilengkapi dengan ilustrasi atau gambar.

Pada Starti

edKFold memiliki input untuk setiap class yang terbagi menjadi 5 class pada setiap class-nya. Lalu dimasukkan kedalam class dataset youtube.

- Ilustrasi Gambar:

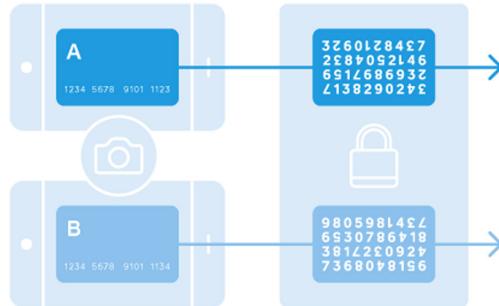
7.3.1.3 Soal No. 3

Maksud kode program for train dan test in splits, dilengkapi dengan ilustrasi atau gambar.

Untuk melakukan pengujian atas data pada dataset sudah di tidak terjadi penumpukan dan split. Dimana di setiap class tidak akan muncul id yang sama.

- Ilustrasi Gambar :

Tokenization Simplified

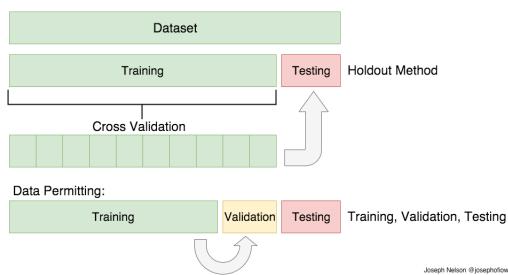


Gambar 7.65 Tokenizer

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
				Data Pengujian					
				Data Pelatihan					

Gambar 1 – Skema 10 fold CV

Gambar 7.66 Konsep dasar K Fold Cross Validation

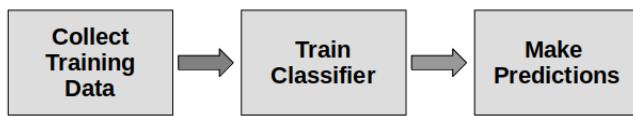


Gambar 7.67 Train dan Test in Split

7.3.1.4 Soal No. 4 Maksud kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx], dilengkapi dengan ilustrasi atau gambar.

Untuk mengambil data pada kolom CONTENT yang merupakan bagian dari train idx dan test idx.

- Ilustrasi Gambar:



Gambar 7.68 Train content

7.3.1.5 Soal No. 5 Maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar.

Yang pertama, yaitu fungsi tokenizer ialah mem-vektorisasi jumlah kata yang ingin diubah kedalam bentuk 2000 kata.

Yang kedua, untuk melakukan fit tokenizer untuk dat其实nya dengan data test nya untuk kolom CONTENT saja.

- Ilustrasi Gambar :

7.3.1.6 Soal No. 6 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 atau gambar.

Variabel d train input untukmelakukan tokenizer dari bentuk teks ke matrix dari data train content dengan mode tfidf dan variabel d test inputs sama saja untuk data test.

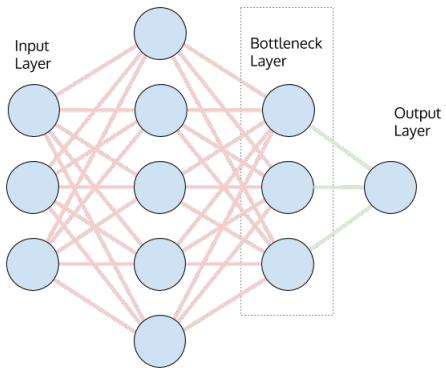
- Ilustrasi Gambar:

✓ Tokenization

is a process of breaking up a piece of `text` into many pieces, such as sentences and words. It works by separating `words` using spaces and punctuation.

```
[1]: from nltk.tokenize import sent_tokenize  
      sentence = "I love ice cream. I also like steak."  
      sent_tokenize(sentence)  
[1]: ['I love ice cream.', 'I also like steak.']}
```

Gambar 7.69 Tokenizer

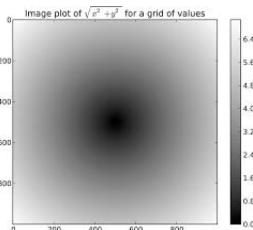


Gambar 7.70 Train Inputs 1

7.3.1.7 Soal No. 7 Maksud dari fungsi $d\text{ train inputs} = d\text{ train inputs}/np.\text{amax}(np.\text{absolute}(d\text{ train inputs}))$ dan $d\text{ test inputs} = d\text{ test inputs}/np.\text{amax}(np.\text{absolute}(d\text{ test inputs}))$, dilengkapi dengan ilustrasi atau gambar.

Akan membagi matrix tfidf dengan amax untuk mengembalikan array atau maksimum array. Kemudian hasilnya dimasukan dalam variabel $d\text{ train inputs}$ untuk data train dan $d\text{ test inputs}$ untuk data test dengan nominal bilangan tanpa ada bilangan negatif dan koma.

- Ilustrasi Gambar :

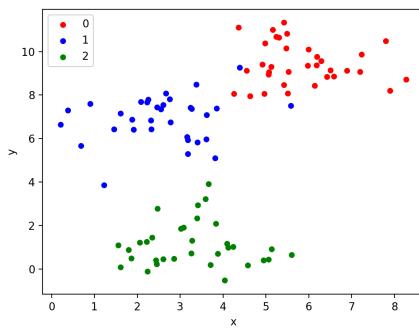


Gambar 7.71 Train Inputs 2

7.3.1.8 Soal No. 8 Maksud dari $d\text{ train outputs} = np.\text{utils.to_categorical}(d[\text{CLASS}].iloc[\text{train idx}])$ dan $d\text{ test outputs} = np.\text{utils.to_categorical}(d[\text{CLASS}].iloc[\text{test idx}])$

Fungsi pada kode program tersebut ditujukan untuk melakukan one-hot encoding supaya bisa masuk dan digunakan pada neural network. One-hot encoding diambil dari class yang berarti hanya terdapat 2 neuron, yaitu satu nol(1,0) atau nol satu(0,1) karena pilihannya hanya ada dua.

- Ilustrasi Gambar:



Gambar 7.72 Compile model

7.3.1.9 Soal No. 9 Maksud dari listing 7.2.

Fungsi kode program tersebut untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Terdapat 512 neuron inputan dengan input shape 2000 vektor yang sudah di-normalisasi. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer output terdapat 2 neuron outputan yaitu nol (1,0) atau nol satu (0,1). Kemudian outputan tersebut diaktivasi menggunakan fungsi softmax.

7.3.1.10 Soal No. 10 Maksud dari listing 7.3.

Fungsi kode program tersebut untuk model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.

7.3.1.11 Soal No. 11 Deep Learning

Deep learning, yang bisa diartikan sebagai rangkaian metode untuk melatih jaringan saraf buatan multi-lapisan. Ternyata, metode ini efektif dalam mengidentifikasi pola dari data. Manakala media membicarakan jaringan saraf, kemungkinan yang dimaksud adalah deep learning.

7.3.1.12 Soal No. 12 Deep Neural Network, dan apa bedanya dengan Deep Learning

Algoritma DNN (Deep Neural Networks) adalah salah satu algoritma berbasis jaringan saraf yang dapat digunakan untuk pengambilan keputusan. Contoh yang dibahas kali ini adalah mengenai penentuan penerimaan pengajuan kredit sepeda motor baru berdasarkan kelompok data yang sudah ada.

Pebedaannya dengan Deep Learning adalah terletak pada kedalaman model, deep learning adalah frasa yang digunakan untuk jaringan saraf yang kompleks.

7.3.1.13 Soal No. 13 Jelaskan dengan ilustrasi gambar buatan sendiri, bagaimana perhitungan algoritma konvolusi dengan ukuran stride ($NPM \bmod 3 + 1$)x($NPM \bmod 3 + 1$) yang terdapat max pooling.(nilai 30)

Karena NPM saya 1164067 dan hasil dari $(NPM \bmod 3) + 1 = 2$, maka saya menggunakan matrik kernel berukuran 2×2 . Misalkan $f(x,y)$ yang digunakan berukuran 3×3 dan kernel atau mask berukuran 2×2 adalah sebagai berikut:

Gambar Matriks:

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ adalah $f(x,y) * g(x,y)$

- Tempatkan matrik kernel di sebelah kiri atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$F(x,y) = \begin{matrix} 2 & 3 & 5 \\ 1 & 0 & 8 \\ 7 & 2 & 0 \end{matrix} \quad \begin{matrix} 1 & 0 \\ 2 & 4 \end{matrix}$$

Gambar 7.73 Perhitungan algoritma konvolusi

$$(2 \times 1) + (3 \times 0) + (1 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

- Tempatkan matrik kernel di sebelah kanan atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(3 \times 1) + (5 \times 0) + (0 \times 2) + (8 \times 4)$$

Sehingga didapat hasil konvolusi = 35

- Tempatkan matrik kernel di sebelah kiri bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(1 \times 1) + (0 \times 0) + (7 \times 2) + (2 \times 4)$$

Sehingga didapat hasil konvolusi = 23

- Tempatkan matrik kernel di sebelah kanan bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(0 \times 1) + (8 \times 0) + (2 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

Hasil:

$$\begin{matrix} 4 & 35 \\ 23 & 4 \end{matrix}$$

Gambar 7.74 Hasil

7.3.2 Praktek

7.3.2.1 Soal No. 1 Jelaskan arti dari setiap baris kode program pada blok # In[1] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:46:02 2020
4
5 @author: User

```

```

6 """
7
8 import csv
9 from PIL import Image as pil_image
10 import keras.preprocessing.image

```

- Penjelasan:

Baris Code 1: Memasukkan atau mengimport file csv

Baris Code 2: Memasukkan module image sebagai pil_image dari library PIL

Baris Code 3: Memasukkan atau mengimport fungsi keras.preprocessing.image

- Hasil output:

```

In [1]: import csv
...: from PIL import Image as pil_image
...: import keras.preprocessing.image
Using TensorFlow backend.

```

Gambar 7.75 1

7.3.2.2 Soal No. 2 Jelaskan arti dari setiap baris kode program pada blok # In[2] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:25 2020
4
5 @author: User
6 """
7
8 imgs = []
9 classes = []
10 with open('HASYv2/hasy-data-labels.csv') as csvfile:
11     csvreader = csv.reader(csvfile)
12     i = 0
13     for row in csvreader:
14         if i > 0:
15             img = keras.preprocessing.image.img_to_array(
16                 pil_image.open("HASYv2/" + row[0]))
17             # neuron activation functions behave best when input
18             # values are between 0.0 and 1.0 (or -1.0 and 1.0),
19             # so we rescale each pixel value to be in the range
20             # 0.0 to 1.0 instead of 0-255
21             img /= 255.0
22             imgs.append((row[0], row[2], img))
23             classes.append(row[2])
24         i += 1

```

- Penjelasan:

Baris Code 1: Membuat variabel imgs tanpa parameter

Baris Code 2: Membuat variabel classes tanpa parameter

Baris Code 3: Membuka file HASYv2/hasy-data-labels.csv

Baris Code 4: Membuat variabel csvreader untuk pembacaan dari file csv yang dimasukkan

Baris Code 5: Membuat variabel i dengan parameter 0

Baris Code 6: Mengeksekusi baris dari pembacaan csv

Baris Code 7: Mengaplikasikan perintah "if" dengan ketentuan variabel i lebih besar dari angka 0, maka akan dilanjutkan ke perintah berikutnya

Baris Code 8: Membuat variabel img yang mengubah image menjadi bentuk array dari file HASYv2 yang dibuka dengan row berparameter 0.

Baris Code 9: Membuat variabel img atau dengan nilai 255.0

Baris Code 10: Mendefinisikan fungsi imgs.append dimana merupakan proses melampirkan atau menggabungkan data dengan file lain atau set data yang ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.

Baris Code 11: Mendefinisikan fungsi append kembali dari variabel classes dengan parameternya row[2].

Baris Code 12: Mendefinisikan fungsi dimana i variabel i akan ditambah nilainya sehingga akan bernilai 1.

- Hasil output:

```
In [2]: imgs = []
...: classes = []
...: with open('HASYv2/hasy-data-labels.csv') as csvfile:
...:     csvreader = csv.reader(csvfile)
...:     i = 0
...:     for row in csvreader:
...:         if i > 0:
...:             img =
keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
...:
# neuron activation functions behave best when input values are
between 0.0 and 1.0 (or -1.0 and 1.0),
...:
# so we rescale each pixel value to be in the range 0.0 to 1.0
instead of 0-255
...:             img /= 255.0
...:             imgs.append((row[0], row[2], img))
...:             classes.append(row[2])
...:             i += 1
```

Gambar 7.76 2

7.3.2.3 Soal No. 3 Jelaskan arti dari setiap baris kode program pada blok # In[3] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import random
9 random.shuffle(imgs)
10 split_idx = int(0.8*len(imgs))
11 train = imgs[:split_idx]
12 test = imgs[split_idx:]

```

- Penjelasan:

Baris Code 1: Memasukkan module random

Baris Code 2: Melakukan pengocokan atau pengacakan pada module random dengan parameter variabelnya imgs

Baris Code 3: Membagi dan memecah index dalam bentuk integer dengan mengkalikan nilai 0,8 dan fungsi len yang akan mengembalikan jumlah item dalam datanya dari variabel imgs

Baris Code 4: Membuat variabel train yang mengeksekusi imgs dengan pemecahan index awal pada data

Baris Code 5: Membuat variabel test yang mengeksekusi imgs dengan pemecahan index akhir pada data

- Hasil output:

```

In [3]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]

```

Gambar 7.77 3

7.3.2.4 Soal No. 4 Jelaskan arti dari setiap baris kode program pada blok # In[4] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import numpy as np
9

```

```

10 train_input = np.asarray(list(map(lambda row: row[2], train)))
11 test_input = np.asarray(list(map(lambda row: row[2], test)))
12
13 train_output = np.asarray(list(map(lambda row: row[1], train)))
14 test_output = np.asarray(list(map(lambda row: row[1], test)))

```

▪ Penjelasan:

Baris Code 1: Mengimport library numpy sebagai np

Baris Code 2: Membuat variabel train_input untuk input menjadi sebuah array dari np menggunakan fungsi list untuk mengkoleksikan data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungskikan lamda pada row dengan parameter [2] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 3: Membuat variabel test_input dengan fungsi seperti train_input yang membedakan hanya datanya atau inputan yang diproses berasal dari variabel test

Baris Code 4: Membuat variabel train_output untuk mengubah keluaran menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengkoleksi data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungskikan lamda pada row dengan parameter[1] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 5: Membuat variabel test_output dengan fungsi yang sama seperti train_output yang membedakan hanya datanya atau inputan yang diproses berasal dari variabel test

▪ Hasil output:

```

In [4]: import numpy as np
...
...: train_input = np.asarray(list(map(lambda row: row[2], train)))
...: test_input = np.asarray(list(map(lambda row: row[2], test)))
...
...: train_output = np.asarray(list(map(lambda row: row[1], train)))
...: test_output = np.asarray(list(map(lambda row: row[1], test)))

```

Gambar 7.78 4

7.3.2.5 Soal No. 5 Jelaskan arti dari setiap baris kode program pada blok # In[5] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020

```

```

4
5 @author: User
6 """
7
8 from sklearn.preprocessing import LabelEncoder
9 from sklearn.preprocessing import OneHotEncoder

```

- Penjelasan:

Baris Code 1: Memasukkan modul atau fungsi LabelEncoder dari sklearn.preprocessing untuk dapat digunakan menormalkan label dimana label encoder didefinisikan dengan nilai antara 0 dan n_classes-1.

Baris Code 2: Memasukkan modul atau fungsi OneHotEncoder dari sklearn.preprocessing untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)).

- Hasil output:

```
In [5]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.79 5

7.3.2.6 Soal No. 6 Jelaskan arti dari setiap baris kode program pada blok # In[6] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 label_encoder = LabelEncoder()
9 integer_encoded = label_encoder.fit_transform(classes)

```

- Penjelasan:

Baris Code 1: Membuat variabel label_encoder dengan modul atau fungsi dari LabelEncoder tanpa parameter

Baris Code 2: Membuat variabel integer_encoded dengan fungsi label_encoder.fit_transform dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali dari variabel label_encoder.

- Hasil output:

```
In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.80 6

7.3.2.7 Soal No. 7 Jelaskan arti dari setiap baris kode program pada blok # In[7] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 onehot_encoder = OneHotEncoder(sparse=False)
9 integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
10 onehot_encoder.fit(integer_encoded)
```

- Penjelasan:

Baris 1: Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.

Baris 2: Membuat variabel integer_encoded memanggil variabel integer_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer_encoded.

Baris 3: Onehotencoding melakukan fitting pada integer_encoded.

- Hasil output:

```
In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
D:\Anaconda\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning:
The handling of integer data will change in version 0.22. Currently, the categories
are determined based on the range [0, max(values)], while in the future they will be
determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories
to integers, then you can now use the OneHotEncoder directly.
    warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Gambar 7.81 7

7.3.2.8 Soal No. 8 Jelaskan arti dari setiap baris kode program pada blok # In[8] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 train_output_int = label_encoder.transform(train_output)
9 train_output = onehot_encoder.transform(train_output_int.reshape(
10     len(train_output_int), 1))
11 test_output_int = label_encoder.transform(test_output)
12 test_output = onehot_encoder.transform(test_output_int.reshape(
13     len(test_output_int), 1))
14 num_classes = len(label_encoder.classes_)
15 print("Number of classes: %d" % num_classes)

```

- Penjelasan:

Baris 1: Membuat variabel `train_output_int` yang mengeksekusi `label_encoder` dengan mengubah nilai dari parameter variabel `train_output`.

Baris 2: Membuat variabel `train_output` yang mengeksekusi variabel `onehot_encoder` dari kode program 7 dengan mengubah nilai dari variabel parameter `train_output_int` yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari `train_output_int` telah dikembalikan.

Baris 3: Membuat variabel `test_output_int` yang mengeksekusi `label_encoder` dengan mengubah nilai dari parameter variabel `test_output`.

Baris 4: Membuat variabel `test_output` yang mengeksekusi variabel `onehot_encoder` dari kode program 7 dengan mengubah nilai dari variabel parameter `test_output_int` yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari `test_output_int` telah dikembalikan.

Baris 5: Membuat variabel `num_classes` untuk mengetahui jumlah class dari `label_encoder`

Baris 6: Perintah `print` digunakan untuk memunculkan hasil dari variabel `num_classes`

- Hasil output:

7.3.2.9 Soal No. 9 Jelaskan arti dari setiap baris kode program pada blok # In[9] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020

```

```
In [8]: train_output_int = label_encoder.transform(train_output)
...: train_output =
onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
...: test_output_int = label_encoder.transform(test_output)
...: test_output =
onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
...
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

Gambar 7.82 8

```
4
5 @author: User
6 """
7
8 from keras.models import Sequential
9 from keras.layers import Dense, Dropout, Flatten
10 from keras.layers import Conv2D, MaxPooling2D
```

▪ Penjelasan:

Baris 1: Melakukan importing fungsi model sequential dari library keras.

Baris 2: Melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.

Baris 3: Melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

▪ Hasil output:

```
In [9]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.83 9

7.3.2.10 Soal No. 10 Jelaskan arti dari setiap baris kode program pada blok # In[10] dan hasil luarannya.

▪ Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10 input_shape=np.shape(train_input[0])))
```

```

11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(1024, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.add(Dense(num_classes, activation='softmax'))
18
19 model.compile(loss='categorical_crossentropy', optimizer='adam',
20                 metrics=['accuracy'])
21
22 print(model.summary())

```

▪ Penjelasan:

Baris 1: Melakukan pemodelan Sequential.

Baris 2: Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train input mulai dari baris nol.

Baris 3: Menambahkan Max Pooling dengan matriks 2x2.

Baris 4: Penambahan Konvolusi 2D dengan 32 filter, konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.

Baris 5 Menambahkan Max Pooling dengan matriks 2x2.

Baris 6: Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.

Baris 7: Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .

Baris 8: Untuk output layer menggunakan data dari variabel num classes dengan fungsi activationnya softmax.

Baris 9: Konfigurasi proses pembelajaran, melalui metode compile, sebelum melatih suatu model.

Baris 10: Menampilkan model yang telah dibuat.

▪ Hasil output:

7.3.2.11 Soal No. 11 Jelaskan arti dari setiap baris kode program pada blok # In[11] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """

```

```

....: model.add(Flatten())
....: model.add(Dense(1024, activation='tanh'))
....: model.add(Dropout(0.5))
....: model.add(Dense(num_classes, activation='softmax'))
....:
....: model.compile(loss='categorical_crossentropy', optimizer='adam',
....:                 metrics=['accuracy'])
....:
....: print(model.summary())
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\keras\backend\tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.

Layer (type)          Output Shape         Param #
=====
conv2d_1 (Conv2D)      (None, 30, 30, 32)     896
max_pooling2d_1 (MaxPooling2D) (None, 15, 15, 32)   0
conv2d_2 (Conv2D)      (None, 13, 13, 32)    9248
max_pooling2d_2 (MaxPooling2D) (None, 6, 6, 32)    0
flatten_1 (Flatten)    (None, 1152)           0
dense_1 (Dense)        (None, 1024)          1180672
dropout_1 (Dropout)    (None, 1024)           0
dense_2 (Dense)        (None, 369)            378225
=====
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
=====

None

```

Gambar 7.84 10

```

7
8
9 import keras.callbacks
10 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
    style')

```

- Penjelasan:

Baris Code 1: Melakukan import library keras.callbacks yang digunakan pada penulisan log untuk TensorBoard, untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian.

Baris Code 2: Membuat variabel tensorboard untuk mendefinisikan fungsi TensorBoard pada keras.callbacks yang digunakan sebagai alat visualisasi yang disediakan dengan TensorFlow. Dan untuk fungsi log_dir memanggil data yaitu './logs/mnist-style'

- Hasil output:

```
In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

Gambar 7.85 11

7.3.2.12 Soal No. 12 Jelaskan arti dari setiap baris kode program pada blok # In[12] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8 model.fit(train_input, train_output,
9            batch_size=32,
10           epochs=10,
11           verbose=2,
12           validation_split=0.2,
13           callbacks=[tensorboard])
14
15 score = model.evaluate(test_input, test_output, verbose=2)
16 print('Test loss:', score[0])
17 print('Test accuracy:', score[1])

```

- Penjelasan:

Baris Code 1: Menerapkan fungsi model.fit yang didalamnya memproses train_input, train_output

Baris Code 2: Penerapan fungsi yang sama difungsikan batch_size apabila batch_sizenya tidak ditemukan maka otomatis akan dijadikan nilai 32

Baris Code 3: Penerapan fungsi yang sama, difungsikan epochs dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10

Baris Code 4: Mendefinisikan fungsi verbose untuk digunakan sebagai opsi menghasilkan informasi logging dari data yang ditentukan dengan nilai 2

Baris Code 5: Mendefinisikan fungsi validation_split untuk memecah nilai dari perhitungan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi)

Baris Code 6: Mendefinisikan fungsi callbacks dengan parameteranya yang mengeksekusi tensorboard dimana digunakan untuk visualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses

Baris Code 7: Mendefinisikan variabel score dengan fungsi evaluate dari model dengan parameter test_input, test_output dan verbose=2 untuk memprediksi output dan input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya

Baris Code 8: Mencetak score optimasi dari test dengan ketentuan nilai parameter 0

Baris Code 9: Mencetak score akurasi dari test dengan ketentuan nilai parameter 1

- Hasil output:

7.3.2.13 Soal No. 13 Jelaskan arti dari setiap baris kode program pada blok # In[13] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 import time
9
10 results = []
11 for conv2d_count in [1, 2]:
12     for dense_size in [128, 256, 512, 1024, 2048]:
13         for dropout in [0.0, 0.25, 0.50, 0.75]:
14             model = Sequential()
15             for i in range(conv2d_count):
16                 if i == 0:
17                     model.add(Conv2D(32, kernel_size=(3, 3),
activation='relu', input_shape=np.shape(train_input[0])))
18                 else:

```

```
In [12]: model.fit(train_input, train_output,
...:     batch_size=32,
...:     epochs=10,
...:     verbose=2,
...:     validation_split=0.2,
...:     callbacks=[tensorboard])
...:
...: score = model.evaluate(test_input, test_output, verbose=2)
...: print('Test loss:', score[0])
...: print('Test accuracy:', score[1])
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\ops
\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and
will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 2007s - loss: 1.5334 - acc: 0.6294 - val_loss: 0.9824 - val_acc: 0.7285
Epoch 2/10
- 918s - loss: 0.9694 - acc: 0.7321 - val_loss: 0.9162 - val_acc: 0.7494
Epoch 3/10
- 562s - loss: 0.8543 - acc: 0.7556 - val_loss: 0.8883 - val_acc: 0.7530
Epoch 4/10
- 503s - loss: 0.7854 - acc: 0.7699 - val_loss: 0.8441 - val_acc: 0.7666
Epoch 5/10
- 361s - loss: 0.7364 - acc: 0.7796 - val_loss: 0.8491 - val_acc: 0.7663
Epoch 6/10
- 366s - loss: 0.6936 - acc: 0.7899 - val_loss: 0.8522 - val_acc: 0.7692
Epoch 7/10
- 360s - loss: 0.6591 - acc: 0.7962 - val_loss: 0.8580 - val_acc: 0.7668
Epoch 8/10
- 389s - loss: 0.6344 - acc: 0.8017 - val_loss: 0.8496 - val_acc: 0.7654
Epoch 9/10
- 357s - loss: 0.6076 - acc: 0.8072 - val_loss: 0.8597 - val_acc: 0.7638
Epoch 10/10
- 359s - loss: 0.5905 - acc: 0.8116 - val_loss: 0.8889 - val_acc: 0.7637
Test loss: 0.8794204677150739
Test accuracy: 0.7632181175230488
```

Gambar 7.86 12

```

19         model.add(Conv2D(32, kernel_size=(3, 3),
20                           activation='relu'))
21         model.add(MaxPooling2D(pool_size=(2, 2)))
22         model.add(Flatten())
23         model.add(Dense(dense_size, activation='tanh'))
24         if dropout > 0.0:
25             model.add(Dropout(dropout))
26         model.add(Dense(num_classes, activation='softmax'))
27
28         model.compile(loss='categorical_crossentropy',
29                         optimizer='adam', metrics=['accuracy'])
30
31         log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
32         (conv2d_count, dense_size, dropout)
33         tensorboard = keras.callbacks.TensorBoard(log_dir=
34         log_dir)
35
36         start = time.time()
37         model.fit(train_input, train_output, batch_size=32,
38         epochs=10,
39                     verbose=0, validation_split=0.2, callbacks
=[tensorboard])
40         score = model.evaluate(test_input, test_output,
41         verbose=2)
42         end = time.time()
43         elapsed = end - start
44         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))
45         results.append((conv2d_count, dense_size, dropout,
46         score[0], score[1], elapsed))

```

▪ Penjelasan:

Baris 1: Import atau memasukkan modul time

Baris 2: Variabel result berisikan array kosong

Baris 3: Menggunakan convolution 2D yang berarti memiliki 1 atau 2 layer

Baris 4: Mendefinisikan dense size dengan ukuran 128, 256, 512, 1024, 2048

Baris 5: Mendefinsikan drop out dengan 0, 25%, 50%, dan 75%

Baris 6: Melakukan pemodelan Sequential

Baris 7: Jika ini adalah layer pertama, akan memasukkan bentuk input.

Baris 8: Kalau tidak kita hanya akan menambahkan layer.

Baris 9: Kemudian, setelah menambahkan layer konvolusi, lakukan hal yang sama dengan max pooling.

Baris 10: Lalu, ratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense size. Dimana akan selalu menggunakan algoritma tanh

Baris 11: Jika dropout digunakan, akan menambahkan layer dropout. Dropout ini berarti misalnya 50%, bahwa setiap kali memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui

Baris 12: Menempatkan di antara dua lapisan padat untuk dihindarkan dari melindunginya dari overfitting. Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.

Baris 13: Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.

Baris 14: Variabel start akan memanggil modul time

Baris 15: Melakukan fit atau compile

Baris 16: Melakukan scoring dengan evaluate yang akan menampilkan data loss dan accuracy dari model

Baris 17: 'End' merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.

Baris 18: Menampilkan hasil dari skrip diatas

- Hasil output:

```
...:
...:           model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...:
...:           log_dir = './logs/conv2d_%d-dense_%d-dropout_.2f' % (conv2d_count,
dense_size, dropout)
...:           tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
...:
...:           start = time.time()
...:           model.fit(train_input, train_output, batch_size=32, epochs=10,
...:                      verbose=0, validation_split=0.2, callbacks=[tensorboard])
...:           score = model.evaluate(test_input, test_output, verbose=2)
...:           end = time.time()
...:           elapsed = end - start
...:           print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.
2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
...:           results.append((conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.13, Accuracy: 0.74, Time: 1540
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.93, Accuracy: 0.76, Time: 1579
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.81, Accuracy: 0.77, Time: 1599
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.82, Accuracy: 0.77, Time: 1627
```

Gambar 7.87 13

7.3.2.14 Soal No. 14 Jelaskan arti dari setiap baris kode program pada blok # In[14] dan hasil luarnya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                 input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(128, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.compile(loss='categorical_crossentropy', optimizer='adam',
18                 metrics=['accuracy'])
18 print(model.summary())

```

▪ Penjelasan:

Baris 1: Melakukan pemodelan Sequential

Baris 2: Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input shape

Baris 3: Melakukan Max Pooling 2D dengan ukuran matriks 2x2

Baris 4: Melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik

Baris 5: Flatten digunakan untuk meratakan inputan atau masukkan data

Baris 6: Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.

Baris 7: Dropout sebanyak 50% untuk menghindari overfitting

Baris 8: Menambahkan dense pada model untuk output dimana layerini akan menjadi jumlah dari class yang ada.

Baris 9: Mengcompile model yang didefinisikan diatas

Baris 10: Menampilkan ringkasan dari pemodelan yang dilakukan

▪ Hasil output:

7.3.2.15 Soal No. 15 Jelaskan arti dari setiap baris kode program pada blok # In[15] dan hasil luarannya.

▪ Code:

```
In [14]: model = Sequential()
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
input_shape=np.shape(train_input[0])))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Conv2D(32, (3, 3), activation='relu'))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Flatten())
...: model.add(Dense(128, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...: print(model.summary())
```

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_3 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_4 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_4 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_2 (Flatten)	(None, 1152)	0
dense_3 (Dense)	(None, 128)	147584
dropout_2 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 369)	47601
Total params:	205,329	
Trainable params:	205,329	
Non-trainable params:	0	

```
None
```

Gambar 7.88 14

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.fit(np.concatenate((train_input, test_input)),
9         np.concatenate((train_output, test_output)),
10        batch_size=32, epochs=10, verbose=2)

```

- Penjelasan:

Baris 1: Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

- Hasil output:

```

In [26]: model.fit(np.concatenate((train_input, test_input)),
...:             np.concatenate((train_output, test_output)),
...:             batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 247s - loss: 1.7949 - acc: 0.5843
Epoch 2/10
- 236s - loss: 1.0797 - acc: 0.7064
Epoch 3/10
- 235s - loss: 0.9648 - acc: 0.7308
Epoch 4/10
- 237s - loss: 0.9060 - acc: 0.7434
Epoch 5/10
- 237s - loss: 0.8671 - acc: 0.7519
Epoch 6/10
- 236s - loss: 0.8362 - acc: 0.7573
Epoch 7/10
- 238s - loss: 0.8137 - acc: 0.7631
Epoch 8/10
- 239s - loss: 0.7980 - acc: 0.7652
Epoch 9/10
- 239s - loss: 0.7831 - acc: 0.7692
Epoch 10/10
- 239s - loss: 0.7695 - acc: 0.7724
Out[26]: <keras.callbacks.History at 0x14c1941f5f8>

```

Gambar 7.89 15

7.3.2.16 Soal No. 16 Jelaskan arti dari setiap baris kode program pada blok # In[16] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.save("mathsymbols.model")

```

- Penjelasan:

Baris 1: Menyimpan model yang telah di latih dengan nama mathsymbols.model

- Hasil output:

```
In [22]: model.save("mathsymbols.model")
```

Gambar 7.90 16

7.3.2.17 Soal No. 17 Jelaskan arti dari setiap baris kode program pada blok # In[17] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 np.save('classes.npy', label_encoder.classes_)
```

- Penjelasan:

Baris 1: Menyimpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

- Hasil output:

```
In [23]: np.save('classes.npy', label_encoder.classes_)
```

Gambar 7.91 17

7.3.2.18 Soal No. 18 Jelaskan arti dari setiap baris kode program pada blok # In[18] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:35 2020
4
5 @author: User
6 """
7
8 import keras.models
9 model2 = keras.models.load_model("mathsymbols.model")
10 print(model2.summary())
```

- Penjelasan:

Baris 1: Memasukkan atau mengimport models dari librari Keras

Baris 2: Variabel model2 akan memanggil model yang telah disave sebelumnya

Baris 3: Menampilkan ringkasan dari hasil pemodelan

- Hasil output:

```
In [24]: import keras.models
...: model2 = keras.models.load_model("mathsymbols.model")
...: print(model2.summary())

Layer (type)                 Output Shape              Param #
=====
conv2d_3 (Conv2D)            (None, 30, 30, 32)      896
max_pooling2d_3 (MaxPooling2 (None, 15, 15, 32)      0
conv2d_4 (Conv2D)            (None, 13, 13, 32)      9248
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 32)        0
flatten_2 (Flatten)          (None, 1152)             0
dense_3 (Dense)              (None, 128)              147584
dropout_2 (Dropout)          (None, 128)              0
dense_4 (Dense)              (None, 369)              47601
=====
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
=====
None
```

Gambar 7.92 18

7.3.2.19 Soal No. 19 Jelaskan arti dari setiap baris kode program pada blok # In[19] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:36 2020
4
5 @author: User
6 """
7
8 label_encoder2 = LabelEncoder()
9 label_encoder2.classes_ = np.load('classes.npy')
10
11 def predict(img_path):
12     newimg = keras.preprocessing.image.img_to_array(pil_image.
13         open(img_path))
14     newimg /= 255.0
```

```

15 # do the prediction
16 prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
17
18 # figure out which output neuron had the highest score, and
19 # reverse the one-hot encoding
20 inverted = label_encoder2.inverse_transform([np.argmax(
    prediction)]) # argmax finds highest-scoring output
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
        max(prediction)))

```

▪ Penjelasan:

Baris 1: Memanggil fungsi LabelEncoder

Baris 2: Variabel label encoder akan memanggil class yang disimpan sebelumnya.

Baris 3: Function Predict akan mengubah gambar kedalam bentuk array

Baris 4: Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.

Baris 5: Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi

▪ Hasil output:

```

In [25]: label_encoder2 = LabelEncoder()
...: label_encoder2.classes_ = np.load('classes.npy')
...:
...: def predict(img_path):
...:     newimg =
keras.preprocessing.image.img_to_array(pil_image.open(img_path))
...:     newimg /= 255.0
...:
...:     # do the prediction
...:     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
...:
...:     # figure out which output neuron had the highest score, and reverse the
one-hot encoding
...:     inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # #
argmax finds highest-scoring output
...:     print("Prediction: %s, confidence: %.2f" % (inverted[0],
np.max(prediction)))

```

Gambar 7.93 19

7.3.2.20 Soal No. 20 Jelaskan arti dari setiap baris kode program pada blok # In[20] dan hasil luarannya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:54:14 2020
4
5 @author: User

```

```

6 """
7
8 predict("HASYv2/hasy-data/v2-00010.png")
9
10 predict("HASYv2/hasy-data/v2-00500.png")
11
12 predict("HASYv2/hasy-data/v2-00700.png")

```

- Penjelasan:

Baris 1: Melakukan prediksi dari pelatihan dari gambar v2-00010.png

Baris 2: Melakukan prediksi dari pelatihan dari gambar v2-00500.png

Baris 3: Melakukan prediksi dari pelatihan dari gambar v2-00700.png

- Hasil output:

```

In [26]: predict("HASYv2/hasy-data/v2-00010.png")
...
...: predict("HASYv2/hasy-data/v2-00500.png")
...
...: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: \pitchfork, confidence: 0.00
Prediction: \copyright, confidence: 0.00
Prediction: J, confidence: 0.00

```

Gambar 7.94 20

7.3.3 Penanganan Error

7.3.3.1 Penanganan Error

- Screenshoot:

```

Traceback (most recent call last):
File "<ipython-input-3-fd9abfd31556>", line 1, in <module>
    model.fit(np.concatenate((train_input, test_input)),
NameError: name 'model' is not defined

```

Gambar 7.95 Error

- Code Error:

NameError: name 'model' is not defined

- Penanganan Error:

Berdasarkan error maka penyelesaiannya ialah melakukan pendefinisian variabel model sehingga code dapat dijalankan. Lakukan running pada code yang berada diatasnya dimana mendefinisikan variabel model.

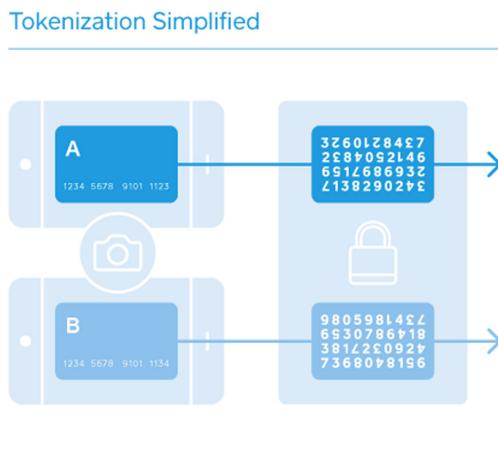
7.4 1174017 Muh. Rifky Prananda

7.4.1 Teori

7.4.1.1 Soal No. 1 Kenapa file teks harus dilakukan tokenizer, dilengkapi dengan ilustrasi atau gambar.

Karena tokenizer ini berfungsi untuk mengkonversi teks menjadi urutan integer indeks kata atau vektor binary, word count atau tf-idf. Text harus dilakukan tokenizer agar dapat dirubah menjadi vektor. Dari perubahan ke vektor tersebut maka data/textnya dapat dibaca oleh komputer (terkomputerisasi).

- Ilustrasi Gambar:



Gambar 7.96 Tokenizer

7.4.1.2 Soal No. 2 Konsep dasar K Fold Cross Validation pada dataset komentar Youtube, dilengkapi dengan ilustrasi atau gambar.

Pada Starti

edKFold memiliki input untuk setiap class yang terbagi menjadi 5 class pada setiap class-nya. Lalu dimasukkan kedalam class dataset youtube.

- Ilustrasi Gambar:

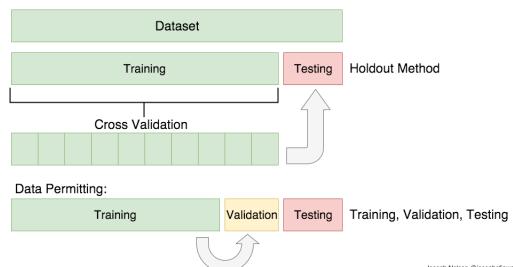
7.4.1.3 Soal No. 3 Maksud kode program for train dan test in splits, dilengkapi dengan ilustrasi atau gambar.

Untuk melakukan pengujian atas data pada dataset sudah di tidak terjadi penumpukan dan split. Dimana di setiap class tidak akan muncul id yang sama.

- Ilustrasi Gambar :

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
				Data Pengujian					
				Data Pelatihan					

Gambar 1 – Skema 10 fold CV

Gambar 7.97 Konsep dasar K Fold Cross Validation**Gambar 7.98** Train dan Test in Split

7.4.1.4 Soal No. 4 Maksud kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx], dilengkapi dengan ilustrasi atau gambar.

Untuk mengambil data pada kolom CONTENT yang merupakan bagian dari train idx dan test idx.

- Ilustrasi Gambar:



Gambar 7.99 Train content

7.4.1.5 Soal No. 5 Maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar.

Yang pertama, yaitu fungsi tokenizer ialah mem-vektorisasi jumlah kata yang ingin diubah kedalam bentuk token 2000 kata.

Yang kedua, untuk melakukan fit tokenizer untuk dat trainnya dengan data test nya untuk kolom CONTENT saja.

- Ilustrasi Gambar :

Tokenization

is a process of breaking up a piece of **text** into many pieces, such as sentences and words. It works by separating **words** using spaces and punctuation.

```

[1]: from nltk.tokenize import sent_tokenize
      sentence = "I love ice cream. I also like steak."
      sent_tokenize(sentence)
  
```

```

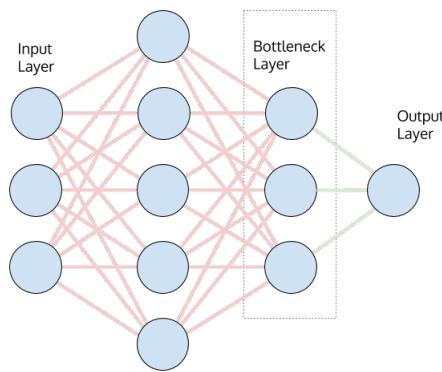
:[1]: ['I love ice cream.', 'I also like steak.']
  
```

Gambar 7.100 Tokenizer

7.4.1.6 Soal No. 6 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 atau gambar.

Variabel d train input untukmelakukan tokenizer dari bentuk teks ke matrix dari data train content dengan mode tfidf dan variabel d test inputs sama saja untuk data test.

- Ilustrasi Gambar:

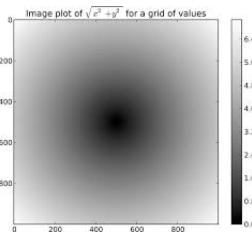


Gambar 7.101 Train Inputs 1

7.4.1.7 Soal No. 7 Maksud dari fungsi $d\text{ train inputs} = d\text{ train inputs}/\text{np.amax}(\text{np.absolute}(d\text{ train inputs}))$ dan $d\text{ test inputs} = d\text{ test inputs}/\text{np.amax}(\text{np.absolute}(d\text{ test inputs}))$, dilengkapi dengan ilustrasi atau gambar.

Akan membagi matrix tfidf dengan amax untuk mengembalikan array atau maksimum array. Kemudian hasilnya dimasukan dalam variabel $d\text{ train inputs}$ untuk data train dan $d\text{ test inputs}$ untuk data test dengan nominal bilangan tanpa ada bilangan negatif dan koma.

- Ilustrasi Gambar :

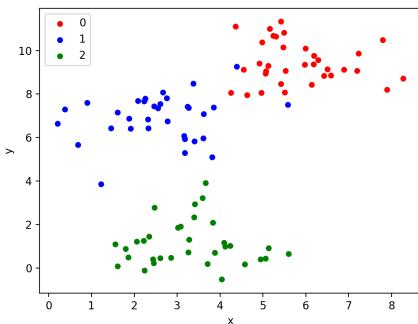


Gambar 7.102 Train Inputs 2

7.4.1.8 Soal No. 8 Maksud dari $d\text{ train outputs} = \text{np utils.to categorical}(d[\text{CLASS}].iloc[train idx])$ dan $d\text{ test outputs} = \text{np utils.to categorical}(d[\text{CLASS}].iloc[test idx])$

Fungsi pada kode program tersebut ditujukan untuk melakukan one-hot encoding supaya bisa masuk dan digunakan pada neural network. One-hot encoding diambil dari class yang berarti hanya terdapat 2 neuron, yaitu satu nol(1,0) atau nol satu(0,1) karena pilihannya hanya ada dua.

- Ilustrasi Gambar:



Gambar 7.103 Compile model

7.4.1.9 **Soal No. 9** Maksud dari listing 7.2.

Fungsi kode program tersebut untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Terdapat 512 neuron inputan dengan input shape 2000 vektor yang sudah di-normalisasi. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer output terdapat 2 neuron outputan yaitu nol (1,0) atau nol satu (0,1). Kemudian outputan tersebut diaktifasi menggunakan fungsi softmax.

7.4.1.10 **Soal No. 10** Maksud dari listing 7.3.

Fungsi kode program tersebut untuk model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.

7.4.1.11 **Soal No. 11** Deep Learning

Deep learning, yang bisa diartikan sebagai rangkaian metode untuk melatih jaringan saraf buatan multi-lapisan. Ternyata, metode ini efektif dalam mengidentifikasi pola dari data. Manakala media membicarakan jaringan saraf, kemungkinan yang dimaksud adalah deep learning.

7.4.1.12 Soal No. 12 Deep Neural Network, dan apa bedanya dengan Deep Learning

Algoritma DNN (Deep Neural Networks) adalah salah satu algoritma berbasis jaringan saraf yang dapat digunakan untuk pengambilan keputusan. Contoh yang dibahas kali ini adalah mengenai penentuan penerimaan pengajuan kredit sepeda motor baru berdasarkan kelompok data yang sudah ada.

Pembedannya dengan Deep Learning adalah terletak pada kedalaman model, deep learning adalah frasa yang digunakan untuk jaringan saraf yang kompleks.

7.4.1.13 Soal No. 13 Jelaskan dengan ilustrasi gambar buatan sendiri, bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1)x(NPM mod3+1) yang terdapat max pooling.(nilai 30)

Karena NPM saya 1164067 dan hasil dari $(NPM \bmod 3) + 1 = 2$, maka saya menggunakan matrik kernel berukuran 2×2 . Misalkan $f(x,y)$ yang digunakan berukuran 3×3 dan kernel atau mask berukuran 2×2 adalah sebagai berikut:

Gambar Matriks:

$$F(x,y) = \begin{matrix} 2 & 3 & 5 \\ 1 & 0 & 8 \\ 7 & 2 & 0 \end{matrix} \quad \begin{matrix} 1 & 0 \\ 2 & 4 \end{matrix}$$

Gambar 7.104 Perhitungan algoritma konvolusi

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ adalah $f(x,y) * g(x,y)$

- Tempatkan matrik kernel di sebelah kiri atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(2 \times 1) + (3 \times 0) + (1 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

- Tempatkan matrik kernel di sebelah kanan atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(3 \times 1) + (5 \times 0) + (0 \times 2) + (8 \times 4)$$

Sehingga didapat hasil konvolusi = 35

- Tempatkan matrik kernel di sebelah kiri bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(1 \times 1) + (0 \times 0) + (7 \times 2) + (2 \times 4)$$

Sehingga didapat hasil konvolusi = 23

- Tempatkan matrik kernel di sebelah kanan bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :
 $(0 \times 1) + (8 \times 0) + (2 \times 2) + (0 \times 4)$
Sehingga didapat hasil konvolusi = 4
Hasil:

$$\begin{matrix} & 4 & 35 \\ 23 & & 4 \end{matrix}$$

Gambar 7.105 Hasil

7.4.2 Praktek

- 7.4.2.1 Soal No. 1** Jelaskan arti dari setiap baris kode program pada blok # In[1] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:46:02 2020
4
5 @author: User
6 """
7
8 import csv
9 from PIL import Image as pil_image
10 import keras.preprocessing.image

```

- Penjelasan:

Baris Code 1: Memasukkan atau mengimport file csv

Baris Code 2: Memasukkan module image sebagai pil_image dari library PIL

Baris Code 3: Memasukkan atau mengimport fungsi keras.preprocessing.image

- Hasil output:

```

In [1]: import csv
...: from PIL import Image as pil_image
...: import keras.preprocessing.image
Using TensorFlow backend.

```

Gambar 7.106 1

7.4.2.2 Soal No. 2 Jelaskan arti dari setiap baris kode program pada blok # In[2] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:25 2020
4
5 @author: User
6 """
7
8 imgs = []
9 classes = []
10 with open('HASYv2/hasy-data-labels.csv') as csvfile:
11     csvreader = csv.reader(csvfile)
12     i = 0
13     for row in csvreader:
14         if i > 0:
15             img = keras.preprocessing.image.img_to_array(
16                 pil_image.open("HASYv2/" + row[0]))
17                 # neuron activation functions behave best when input
18                 # values are between 0.0 and 1.0 (or -1.0 and 1.0),
19                 # so we rescale each pixel value to be in the range
20                 # 0.0 to 1.0 instead of 0-255
21                 img /= 255.0
22             imgs.append((row[0], row[2], img))
23             classes.append(row[2])
24         i += 1

```

- Penjelasan:

Baris Code 1: Membuat variabel imgs tanpa parameter

Baris Code 2: Membuat variabel classes tanpa parameter

Baris Code 3: Membuka file HASYv2/hasy-data-labels.csv

Baris Code 4: Membuat variabel csvreader untuk pembacaan dari file csv yang dimasukkan

Baris Code 5: Membuat variabel i dengan parameter 0

Baris Code 6: Mengeksekusi baris dari pembacaan csv

Baris Code 7: Mengaplikasikan perintah "if" dengan ketentuan variabel i lebih besar dari angka 0, maka akan dilanjutkan ke perintah berikutnya

Baris Code 8: Membuat variabel img yang mengubah image menjadi bentuk array dari file HASYv2 yang dibuka dengan row berparameter 0.

Baris Code 9: Membuat variabel img atau dengan nilai 255.0

Baris Code 10: Mendefinisikan fungsi imgs.append dimana merupakan proses melampirkan atau menggabungkan data dengan file lain atau set data yang ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.

Baris Code 11: Mendefinisikan fungsi append kembali dari variabel classes dengan parameternya row[2].

Baris Code 12: Mendefinisikan fungsi dimana i variabel i akan ditambah nilainya sehingga akan bernilai 1.

- Hasil output:

```
In [2]: imgs = []
...: classes = []
...: with open('HASYv2/hasy-data-labels.csv') as csvfile:
...:     csvreader = csv.reader(csvfile)
...:     i = 0
...:     for row in csvreader:
...:         if i > 0:
...:             img =
keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
...:             # neuron activation functions behave best when input values are
between 0.0 and 1.0 (or -1.0 and 1.0),
...:             # so we rescale each pixel value to be in the range 0.0 to 1.0
instead of 0-255
...:             img /= 255.0
...:             imgs.append((row[0], row[2], img))
...:             classes.append(row[2])
...:             i += 1
```

Gambar 7.107 2

7.4.2.3 Soal No. 3 Jelaskan arti dari setiap baris kode program pada blok # In[3] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import random
9 random.shuffle(imgs)
10 split_idx = int(0.8*len(imgs))
11 train = imgs[:split_idx]
12 test = imgs[split_idx:]
```

- Penjelasan:

Baris Code 1: Memasukkan module random

Baris Code 2: Melakukan pengocokan atau pengacakkan pada module random dengan parameter variabelnya imgs

Baris Code 3: Membagi dan memecah index dalam bentuk integer dengan mengkalikan nilai 0,8 dan fungsi len yang akan mengembalikan jumlah item dalam datanya dari variabel imgs

Baris Code 4: Membuat variabel train yang mengeksekusi imgs dengan pemecahan index awal pada data

Baris Code 5: Membuat variabel test yang mengeksekusi imgs dengan pemecahan index akhir pada data

- Hasil output:

```
In [3]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]
```

Gambar 7.108 3

7.4.2.4 Soal No. 4 Jelaskan arti dari setiap baris kode program pada blok # In[4] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import numpy as np
9
10 train_input = np.asarray(list(map(lambda row: row[2], train)))
11 test_input = np.asarray(list(map(lambda row: row[2], test)))
12
13 train_output = np.asarray(list(map(lambda row: row[1], train)))
14 test_output = np.asarray(list(map(lambda row: row[1], test)))
```

- Penjelasan:

Baris Code 1: Mengimport library numpy sebagai np

Baris Code 2: Membuat variabel train_input untuk input menjadi sebuah array dari np menggunakan fungsi list untuk mengkoleksikan data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungksikan lamda pada row dengan parameter [2] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 3: Membuat variabel test_input dengan fungsi seperti train_input yang membedakan hanya datanya atau inputan yang diproses berasal dari variabel test

Baris Code 4: Membuat variabel train_output untuk mengubah keluaran menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengkoleksi

data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungskan lamda pada row dengan parameter[1] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 5: Membuat variabel test_output dengan fungsi yang sama seperti train_output yang membedakan hanya datanya atau inputan yang diproses berdasarkan dari variabel test

- Hasil output:

```
In [4]: import numpy as np
...
...: train_input = np.asarray(list(map(lambda row: row[2], train)))
...: test_input = np.asarray(list(map(lambda row: row[2], test)))
...
...: train_output = np.asarray(list(map(lambda row: row[1], train)))
...: test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Gambar 7.109 4

7.4.2.5 Soal No. 5 Jelaskan arti dari setiap baris kode program pada blok # In[5] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 from sklearn.preprocessing import LabelEncoder
9 from sklearn.preprocessing import OneHotEncoder
```

- Penjelasan:

Baris Code 1: Memasukkan modul atau fungsi LabelEncoder dari sklearn.preprocessing untuk dapat digunakan menormalkan label dimana label encoder didefinisikan dengan nilai antara 0 dan n_classes-1.

Baris Code 2: Memasukkan modul atau fungsi OneHotEncoder dari sklearn.preprocessing untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)).

- Hasil output:

```
In [5]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.110 5

7.4.2.6 Soal No. 6 Jelaskan arti dari setiap baris kode program pada blok # In[6] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 label_encoder = LabelEncoder()
9 integer_encoded = label_encoder.fit_transform(classes)

```

- Penjelasan:

Baris Code 1: Membuat variabel label_encoder dengan modul atau fungsi dari LabelEncoder tanpa parameter

Baris Code 2: Membuat variabel integer_encoded dengan fungsi label_encoder.fit_transform dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali dari variabel label_encoder.

- Hasil output:

```
In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.111 6

7.4.2.7 Soal No. 7 Jelaskan arti dari setiap baris kode program pada blok # In[7] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 onehot_encoder = OneHotEncoder(sparse=False)
9 integer_encoded = integer_encoded.reshape(len(integer_encoded),
10 1)
onehot_encoder.fit(integer_encoded)

```

- Penjelasan:

Baris 1: Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.

Baris 2: Membuat variabel integer_encoded memanggil variabel integer_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer_encoded.

Baris 3: Onehotencoding melakukan fitting pada integer_encoded.

- Hasil output:

```
In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
D:\Anaconda\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning:
The handling of integer data will change in version 0.22. Currently, the categories
are determined based on the range [0, max(values)], while in the future they will be
determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories
to integers, then you can now use the OneHotEncoder directly.
    warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Gambar 7.112 7

7.4.2.8 Soal No. 8 Jelaskan arti dari setiap baris kode program pada blok # In[8] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 train_output_int = label_encoder.transform(train_output)
9 train_output = onehot_encoder.transform(train_output_int.reshape(
10     len(train_output_int), 1))
11 test_output_int = label_encoder.transform(test_output)
12 test_output = onehot_encoder.transform(test_output_int.reshape(
13     len(test_output_int), 1))
14 num_classes = len(label_encoder.classes_)
15 print("Number of classes: %d" % num_classes)
```

- Penjelasan:

Baris 1: Membuat variabel train_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel train_output.

Baris 2: Membuat variabel train_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train_output_int

yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.

Baris 3: Membuat variabel test_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel test_output.

Baris 4: Membuat variabel test_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test_output_int telah dikembalikan.

Baris 5: Membuat variabel num_classes untuk mengetahui jumlah class dari label_encoder

Baris 6: Perintah print digunakan untuk memunculkan hasil dari variabel num_classes

```
In [8]: train_output_int = label_encoder.transform(train_output)
...: train_output =
onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
...: test_output_int = label_encoder.transform(test_output)
...: test_output =
onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
...:
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

Gambar 7.113 8

- Hasil output:

7.4.2.9 Soal No. 9 Jelaskan arti dari setiap baris kode program pada blok # In[9] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 from keras.models import Sequential
9 from keras.layers import Dense, Dropout, Flatten
10 from keras.layers import Conv2D, MaxPooling2D
```

- Penjelasan:

Baris 1: Melakukan importing fungsi model sequential dari library keras.

Baris 2: Melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.

Baris 3: Melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

- Hasil output:

```
In [9]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.114 9

7.4.2.10 Soal No. 10 Jelaskan arti dari setiap baris kode program pada blok # In[10] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                 input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(1024, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.add(Dense(num_classes, activation='softmax'))
18
19 model.compile(loss='categorical_crossentropy', optimizer='adam',
20                 metrics=['accuracy'])
21
22 print(model.summary())
```

- Penjelasan:

Baris 1: Melakukan pemodelan Sequential.

Baris 2: Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train input mulai dari baris nol.

Baris 3: Menambahkan Max Pooling dengan matriks 2x2.

Baris 4: Penambahan Konvolusi 2D dengan 32 filter, konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.

Baris 5 Menambahkan Max Pooling dengan matriks 2x2.

Baris 6: Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.

Baris 7: Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .

Baris 8: Untuk output layer menggunakan data dari variabel num classes dengan fungsi activationnya softmax.

Baris 9: Konfigurasi proses pembelajaran, melalui metode compile, sebelum melatih suatu model.

Baris 10: Menampilkan model yang telah dibuat.

- Hasil output:

```
...: model.add(Flatten())
...: model.add(Dense(1024, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
...: metrics=['accuracy'])
...
...: print(model.summary())
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\framework
\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\keras\backend
\backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with
keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 -
keep_prob`.
-----
```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_1 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_1 (Flatten)	(None, 1152)	0
dense_1 (Dense)	(None, 1024)	1180672
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 369)	378225

```
=====
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
-----
```

None

Gambar 7.115 10

7.4.2.11 Soal No. 11 Jelaskan arti dari setiap baris kode pada blok # In[11] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8
9 import keras.callbacks
10 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
    style')

```

- Penjelasan:

Baris Code 1: Melakukan import library keras.callbacks yang digunakan pada penulisan log untuk TensorBoard, untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian.

Baris Code 2: Membuat variabel tensorboard untuk mendefinisikan fungsi TensorBoard pada keras.callbacks yang digunakan sebagai alat visualisasi yang disediakan dengan TensorFlow. Dan untuk fungsi log_dir memanggil data yaitu './logs/mnist-style'

- Hasil output:

```
In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

Gambar 7.116 11

7.4.2.12 Soal No. 12 Jelaskan arti dari setiap baris kode program pada blok # In[12] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8 model.fit(train_input, train_output,
9            batch_size=32,
10           epochs=10,
11           verbose=2,
12           validation_split=0.2,
13           callbacks=[tensorboard])
14
15 score = model.evaluate(test_input, test_output, verbose=2)
16 print('Test loss:', score[0])
17 print('Test accuracy:', score[1])

```

- Penjelasan:

Baris Code 1: Menerapkan fungsi model.fit yang didalamnya memproses train_input, train_output

Baris Code 2: Penerapan fungsi yang sama difungsikan batch_size apabila batch_sizenya tidak ditemukan maka otomatis akan dijadikan nilai 32

Baris Code 3: Penerapan fungsi yang sama, difungsikan epochs dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10

Baris Code 4: Mendefinisikan fungsi verbose untuk digunakan sebagai opsi menghasilkan informasi logging dari data yang ditentukan dengan nilai 2

Baris Code 5: Mendefinisikan fungsi validation_split untuk memecah nilai dari perhitungan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi)

Baris Code 6: Mendefinisikan fungsi callbacks dengan parameternya yang mengeksekusi tensorboard dimana digunakan untuk visualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses

Baris Code 7: Mendefinisikan variabel score dengan fungsi evaluate dari model dengan parameter test_input, tst_output dan verbose=2 untuk memprediksi output dan input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya

Baris Code 8: Mencetak score optimasi dari test dengan ketentuan nilai parameter 0

Baris Code 9: Mencetak score akurasi dari test dengan ketentuan nilai parameter 1

- Hasil output:

7.4.2.13 Soal No. 13 Jelaskan arti dari setiap baris kode program pada blok # In[13] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 import time
9
10 results = []
11 for conv2d_count in [1, 2]:
12     for dense_size in [128, 256, 512, 1024, 2048]:
13         for dropout in [0.0, 0.25, 0.50, 0.75]:
14             model = Sequential()

```

```
In [12]: model.fit(train_input, train_output,
...:     batch_size=32,
...:     epochs=10,
...:     verbose=2,
...:     validation_split=0.2,
...:     callbacks=[tensorboard])
...:
...: score = model.evaluate(test_input, test_output, verbose=2)
...: print('Test loss:', score[0])
...: print('Test accuracy:', score[1])
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\ops\numpy_ops.py:3066: to_int32 (from tensorflow.python.ops.numpy_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 2007s - loss: 1.5334 - acc: 0.6294 - val_loss: 0.9824 - val_acc: 0.7285
Epoch 2/10
- 918s - loss: 0.9694 - acc: 0.7321 - val_loss: 0.9162 - val_acc: 0.7494
Epoch 3/10
- 562s - loss: 0.8543 - acc: 0.7556 - val_loss: 0.8883 - val_acc: 0.7530
Epoch 4/10
- 503s - loss: 0.7854 - acc: 0.7699 - val_loss: 0.8441 - val_acc: 0.7666
Epoch 5/10
- 361s - loss: 0.7364 - acc: 0.7796 - val_loss: 0.8491 - val_acc: 0.7663
Epoch 6/10
- 366s - loss: 0.6936 - acc: 0.7899 - val_loss: 0.8522 - val_acc: 0.7692
Epoch 7/10
- 360s - loss: 0.6591 - acc: 0.7962 - val_loss: 0.8580 - val_acc: 0.7668
Epoch 8/10
- 389s - loss: 0.6344 - acc: 0.8017 - val_loss: 0.8496 - val_acc: 0.7654
Epoch 9/10
- 357s - loss: 0.6076 - acc: 0.8072 - val_loss: 0.8597 - val_acc: 0.7638
Epoch 10/10
- 359s - loss: 0.5905 - acc: 0.8116 - val_loss: 0.8889 - val_acc: 0.7637
Test loss: 0.8794204677150739
Test accuracy: 0.7632181175230488
```

Gambar 7.117 12

```

15         for i in range(conv2d_count):
16             if i == 0:
17                 model.add(Conv2D(32, kernel_size=(3, 3),
18 activation='relu', input_shape=np.shape(train_input[0])))
19             else:
20                 model.add(Conv2D(32, kernel_size=(3, 3),
21 activation='relu'))
22                 model.add(MaxPooling2D(pool_size=(2, 2)))
23                 model.add(Flatten())
24                 model.add(Dense(dense_size, activation='tanh'))
25                 if dropout > 0.0:
26                     model.add(Dropout(dropout))
27                 model.add(Dense(num_classes, activation='softmax'))
28
29                 model.compile(loss='categorical_crossentropy',
30 optimizer='adam', metrics=['accuracy'])
31
32                 log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
33 (conv2d_count, dense_size, dropout)
34                 tensorboard = keras.callbacks.TensorBoard(log_dir=
log_dir)
35
36                 start = time.time()
37                 model.fit(train_input, train_output, batch_size=32,
38 epochs=10,
39 verbose=0, validation_split=0.2, callbacks
=[tensorboard])
40                 score = model.evaluate(test_input, test_output,
verbose=2)
41                 end = time.time()
42                 elapsed = end - start
43                 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))
44                 results.append((conv2d_count, dense_size, dropout,
score[0], score[1], elapsed))

```

▪ Penjelasan:

Baris 1: Import atau memasukkan modul time

Baris 2: Variabel result berisikan array kosong

Baris 3: Menggunakan convolution 2D yang berarti memiliki 1 atau 2 layer

Baris 4: Mendefinisikan dense size dengan ukuran 128, 256, 512, 1024, 2048

Baris 5: Mendefinsikan drop out dengan 0, 25%, 50%, dan 75%

Baris 6: Melakukan pemodelan Sequential

Baris 7: Jika ini adalah layer pertama, akan memasukkan bentuk input.

Baris 8: Kalau tidak kita hanya akan menambahkan layer.

Baris 9: Kemudian, setelah menambahkan layer konvolusi, lakukan hal yang sama dengan max pooling.

Baris 10: Lalu, ratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense size. Dimana akan selalu menggunakan algoritma tanh

Baris 11: Jika dropout digunakan, akan menambahkan layer dropout. Dropout ini berarti misalnya 50%, bahwa setiap kali memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui

Baris 12: Menempatkan di antara dua lapisan padat untuk dihindarkan dari melindunginya dari overfitting. Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.

Baris 13: Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.

Baris 14: Variabel start akan memanggil modul time

Baris 15: Melakukan fit atau compile

Baris 16: Melakukan scoring dengan evaluate yang akan menampilkan data loss dan accuracy dari model

Baris 17: 'End' merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.

Baris 18: Menampilkan hasil dari skrip diatas

- Hasil output:

```
....  
....      model.compile(loss='categorical_crossentropy', optimizer='adam',  
metrics=['accuracy'])  
....  
....      log_dir = './logs/conv2d_%d-dense_%d-dropout_.2f' % (conv2d_count,  
dense_size, dropout)  
....      tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)  
....  
....      start = time.time()  
....      model.fit(train_input, train_output, batch_size=32, epochs=10,  
....                  verbose=0, validation_split=0.2, callbacks=[tensorboard])  
....      score = model.evaluate(test_input, test_output, verbose=2)  
....      end = time.time()  
....      elapsed = end - start  
....      print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.  
2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count, dense_size, dropout, score[0],  
score[1], elapsed))  
....      results.append((conv2d_count, dense_size, dropout, score[0],  
score[1], elapsed))  
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.13, Accuracy: 0.74, Time: 1540  
sec  
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.93, Accuracy: 0.76, Time: 1579  
sec  
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.81, Accuracy: 0.77, Time: 1599  
sec  
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.82, Accuracy: 0.77, Time: 1627
```

7.4.2.14 Soal No. 14 Jelaskan arti dari setiap baris kode program pada blok # In[14] dan hasil luarnya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                 input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(128, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.add(Dense(num_classes, activation='softmax'))
18 model.compile(loss='categorical_crossentropy', optimizer='adam',
19                 metrics=['accuracy'])
20 print(model.summary())

```

- Penjelasan:

Baris 1: Melakukan pemodelan Sequential

Baris 2: Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input shape

Baris 3: Melakukan Max Pooling 2D dengan ukuran matriks 2x2

Baris 4: Melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik

Baris 5: Flatten digunakan untuk meratakan inputan atau masukkan data

Baris 6: Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.

Baris 7: Dropout sebanyak 50% untuk menghindari overfitting

Baris 8: Menambahkan dense pada model untuk output dimana layerini akan menjadi jumlah dari class yang ada.

Baris 9: Mengcompile model yang didefinisikan diatas

Baris 10: Menampilkan ringkasan dari pemodelan yang dilakukan

- Hasil output:

```
In [14]: model = Sequential()
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
input_shape=np.shape(train_input[0])))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Conv2D(32, (3, 3), activation='relu'))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Flatten())
...: model.add(Dense(128, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...: print(model.summary())

Layer (type)                 Output Shape              Param #
=====
conv2d_3 (Conv2D)            (None, 30, 30, 32)      896
max_pooling2d_3 (MaxPooling2 (None, 15, 15, 32)      0
conv2d_4 (Conv2D)            (None, 13, 13, 32)      9248
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 32)        0
flatten_2 (Flatten)          (None, 1152)             0
dense_3 (Dense)              (None, 128)              147584
dropout_2 (Dropout)          (None, 128)              0
dense_4 (Dense)              (None, 369)              47601
=====
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
```

None

Gambar 7.119 14

7.4.2.15 Soal No. 15 Jelaskan arti dari setiap baris kode program pada blok # In[15] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.fit(np.concatenate((train_input, test_input)),
9         np.concatenate((train_output, test_output)),
10        batch_size=32, epochs=10, verbose=2)

```

- Penjelasan:

Baris 1: Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

- Hasil output:

```

In [26]: model.fit(np.concatenate((train_input, test_input)),
...:                 np.concatenate((train_output, test_output)),
...:                 batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 247s - loss: 1.7949 - acc: 0.5843
Epoch 2/10
- 236s - loss: 1.0797 - acc: 0.7064
Epoch 3/10
- 235s - loss: 0.9648 - acc: 0.7308
Epoch 4/10
- 237s - loss: 0.9060 - acc: 0.7434
Epoch 5/10
- 237s - loss: 0.8671 - acc: 0.7519
Epoch 6/10
- 236s - loss: 0.8362 - acc: 0.7573
Epoch 7/10
- 238s - loss: 0.8137 - acc: 0.7631
Epoch 8/10
- 239s - loss: 0.7980 - acc: 0.7652
Epoch 9/10
- 239s - loss: 0.7831 - acc: 0.7692
Epoch 10/10
- 239s - loss: 0.7695 - acc: 0.7724
Out[26]: <keras.callbacks.History at 0x14c1941f5f8>

```

Gambar 7.120 15

7.4.2.16 Soal No. 16 Jelaskan arti dari setiap baris kode program pada blok # In[16] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """

```

```

3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.save("mathsymbols.model")

```

- Penjelasan:

Baris 1: Menyimpan model yang telah di latih dengan nama mathsymbols.model

- Hasil output:

```
In [22]: model.save("mathsymbols.model")
```

Gambar 7.121 16

7.4.2.17 Soal No. 17 Jelaskan arti dari setiap baris kode program pada blok # In[17] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 np.save('classes.npy', label_encoder.classes_)

```

- Penjelasan:

Baris 1: Menyimpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

- Hasil output:

```
In [23]: np.save('classes.npy', label_encoder.classes_)
```

Gambar 7.122 17

7.4.2.18 Soal No. 18 Jelaskan arti dari setiap baris kode program pada blok # In[18] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:35 2020
4

```

```

5 @author: User
6 """
7
8 import keras.models
9 model2 = keras.models.load_model("mathsymbols.model")
10 print(model2.summary())

```

▪ Penjelasan:

Baris 1: Memasukkan atau mengimport models dari librari Keras

Baris 2: Variabel model2 akan memanggil model yang telah disave sebelumnya

Baris 3: Menampilkan ringkasan dari hasil pemodelan

▪ Hasil output:

```

In [24]: import keras.models
...: model2 = keras.models.load_model("mathsymbols.model")
...: print(model2.summary())

Layer (type)                 Output Shape              Param #
=====
conv2d_3 (Conv2D)            (None, 30, 30, 32)      896
max_pooling2d_3 (MaxPooling2 (None, 15, 15, 32)      0
conv2d_4 (Conv2D)            (None, 13, 13, 32)      9248
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 32)        0
flatten_2 (Flatten)          (None, 1152)           0
dense_3 (Dense)              (None, 128)            147584
dropout_2 (Dropout)          (None, 128)           0
dense_4 (Dense)              (None, 369)           47601
=====
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
=====
None

```

Gambar 7.123 18

7.4.2.19 Soal No. 19 Jelaskan arti dari setiap baris kode program pada blok # In[19] dan hasil luarnya.

▪ Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:36 2020
4
5 @author: User
6 """
7

```

```

8 label_encoder2 = LabelEncoder()
9 label_encoder2.classes_ = np.load('classes.npy')
10
11 def predict(img_path):
12     newimg = keras.preprocessing.image.img_to_array(pil_image.
13         open(img_path))
14     newimg /= 255.0
15
16     # do the prediction
17     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
18
19     # figure out which output neuron had the highest score, and
20     # reverse the one-hot encoding
21     inverted = label_encoder2.inverse_transform([np.argmax(
22         prediction)]) # argmax finds highest-scoring output
23     print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
24         max(prediction)))

```

▪ Penjelasan:

Baris 1: Memanggil fungsi LabelEncoder

Baris 2: Variabel label encoder akan memanggil class yang disimpan sebelumnya.

Baris 3: Function Predict akan mengubah gambar kedalam bentuk array

Baris 4: Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.

Baris 5: Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi

▪ Hasil output:

```

In [25]: label_encoder2 = LabelEncoder()
...: label_encoder2.classes_ = np.load('classes.npy')
...:
...: def predict(img_path):
...:     newimg =
keras.preprocessing.image.img_to_array(pil_image.open(img_path))
...:     newimg /= 255.0
...:
...:     # do the prediction
...:     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
...:
...:     # figure out which output neuron had the highest score, and reverse the
one-hot encoding
...:     inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # argmax finds highest-scoring output
...:     print("Prediction: %s, confidence: %.2f" % (inverted[0],
np.max(prediction)))

```

Gambar 7.124 19

7.4.2.20 Soal No. 20 Jelaskan arti dari setiap baris kode program pada blok # In[20] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:54:14 2020
4
5 @author: User
6 """
7
8 predict("HASYv2/hasy-data/v2-00010.png")
9
10 predict("HASYv2/hasy-data/v2-00500.png")
11
12 predict("HASYv2/hasy-data/v2-00700.png")

```

- Penjelasan:

- Baris 1: Melakukan prediksi dari pelatihan dari gambar v2-00010.png
 Baris 2: Melakukan prediksi dari pelatihan dari gambar v2-00500.png
 Baris 3: Melakukan prediksi dari pelatihan dari gambar v2-00700.png

- Hasil output:

```

In [26]: predict("HASYv2/hasy-data/v2-00010.png")
...:
...: predict("HASYv2/hasy-data/v2-00500.png")
...:
...: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: \pitchfork, confidence: 0.00
Prediction: \copyright, confidence: 0.00
Prediction: J, confidence: 0.00

```

Gambar 7.125 20

7.4.3 Penanganan Error

7.4.3.1 Penanganan Error

- Screenshoot:

```

Traceback (most recent call last):
  File "<ipython-input-3-fd9abfd31556>", line 1, in <module>
    model.fit(np.concatenate((train_input, test_input)),
NameError: name 'model' is not defined

```

Gambar 7.126 Error

- Code Error:

NameError: name 'model' is not defined

- Penanganan Error:

Berdasarkan error maka penyelesaiannya ialah melakukan pendefinisian variabel model sehingga code dapat dijalankan. Lakukan running pada code yang berada diatasnya dimana mendefinisikan variabel model.

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](#)