

Modul Praktikum Kecerdasan Buatan



Rolly Maulana Awangga
0410118609

Applied Bachelor of Informatics Engineering
Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering
Politeknik Pos Indonesia

Bandung 2019

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

Acknowledgements

Pertama-tama kami panjatkan puji dan syukur kepada Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga Buku Pedoman Tingkat Akhir ini dapat diselesaikan.

Abstract

Buku Pedoman ini dibuat dengan tujuan memberikan acuan, bagi mahasiswa Tingkat Akhir dan dosen Pembimbing. Pada intinya buku ini menjelaskan secara lengkap tentang Standar pengerjaan Intership dan Tugas Akhir di Program Studi D4 Teknik Informatika, dan juga mengatur mekanisme, teknik penulisan, serta penilaiannya. Dengan demikian diharapkan semua pihak yang terlibat dalam aktivitas Bimbingan Mahasiswa Tingkat Akhir berjalan lancar dan sesuai dengan standar.

Contents

List of Figures

Chapter 1

Judul Bagian Ketujuh

1.1 Annisa Fathoroni/1164067

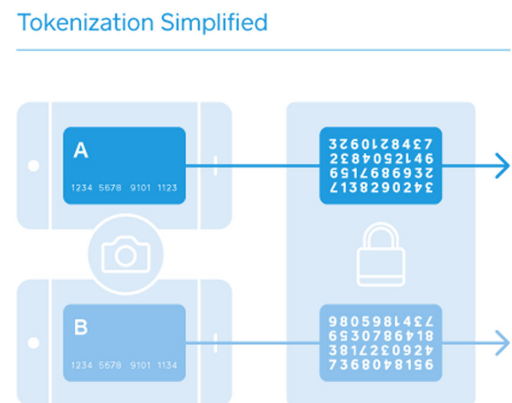
1.1.1 Teori - 1164067

1.1.1.1 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:



7/1164067/Teori/Chapter7AnnisaFathoroni1.jpg

Figure 1.1: Tokenizer - Annisa Fathoroni

1.1.1.2 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:



7/1164067/Teori/Chapter7AnnisaFathoroni4.png

Figure 1.4: Train content - Annisa Fathoroni

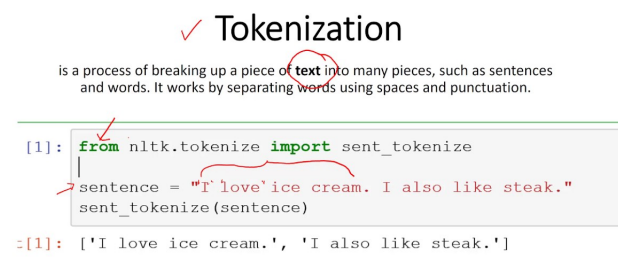
1.1.1.5 Soal No. 5

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 token 2000 kata.

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

- Ilustrasi Gambar :



7/1164067/Teori/Chapter7AnnisaFathoroni5.png

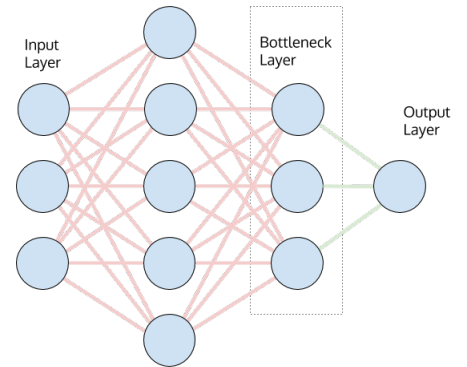
Figure 1.5: Tokenizer - Annisa Fathoroni

1.1.1.6 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` untuk melakukan 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:



7/1164067/Teori/Chapter7AnnisaFathoroni6.png

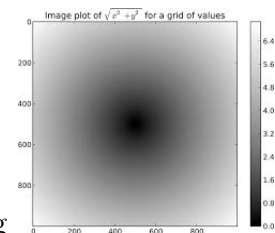
Figure 1.6: Train Inputs 1 - Annisa Fathoroni

1.1.1.7 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 :



7/1164067/Teori/Chapter7AnnisaFathoroni7.png

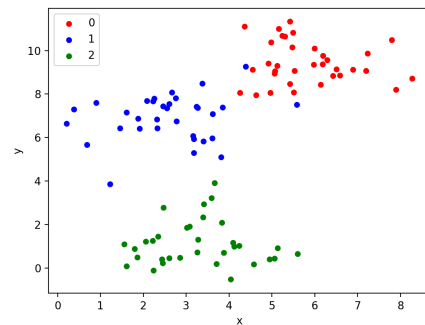
Figure 1.7: Train Inputs 2 - Annisa Fathoroni

1.1.1.8 Soal No. 8

Maksud dari $d \text{ train outputs} = \text{np utils.to categorical}(d[\text{'CLASS'}].\text{iloc}[\text{train idx}])$ dan $d \text{ test outputs} = \text{np utils.to categorical}(d[\text{'CLASS'}].\text{iloc}[\text{test idx}])$

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:



7/1164067/Teori/Chapter7AnnisaFathoroni8.png

Figure 1.8: Compile model - Annisa Fathoroni

1.1.1.9 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 dinormalisasi. 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.

1.1.1.10 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.

1.1.1.11 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.

1.1.1.12 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.

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

1.1.1.13 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 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:

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

7/1164067/Teori/Chapter7AnnisaFathoroni9.png

Figure 1.9: Perhitungan algoritma konvolusi - Annisa Fathoroni

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:



Figure 1.10: Hasil - Annisa Fathoroni

1.1.2 Praktek - 1164067

1.1.2.1 Soal No. 1

Jelaskan arti dari setiap baris kode program pada blok # In[1] dan hasil luarannya.

- Code:

```
import csv
from PIL import Image as pil_image
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:

7/1164067/Praktek/Chapter7AnnisaFathoronil.jpg

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

Figure 1.11: 1 - Annisa Fathoroni

1.1.2.2 Soal No. 2

Jelaskan arti dari setiap baris kode program pada blok # In[2] dan hasil luarannya.

- Code:

```
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)
            # neuron activation functions behave best when input va
            # so we rescale each pixel value to be in the range 0.0
            img /= 255.0
            imgs.append((row[0], row[2], img))
            classes.append(row[2])
```

- 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/"
...:             # neuron activation functions behave best w
between 0.0 and 1.0 (or -1.0 and 1.0),
...:             # so we rescale each pixel value to be in t
instead of 0-255
...:             img /= 255.0
...:             imgs.append((row[0], row[2], img))
...:             classes.append(row[2])
...:             i += 1
```

7/1164067/Praktek/Chapter7AnnisaFathoroni2.jpg

Figure 1.12: 2 - Annisa Fathoroni

1.1.2.3 Soal No. 3

Jelaskan arti dari setiap baris kode program pada blok # In[3] dan hasil luarannya.

- Code:

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

7/1164067/Praktek/Chapter7AnnisaFathoroni3.jpg

Figure 1.13: 3 - Annisa Fathoroni

1.1.2.4 Soal No. 4

Jelaskan arti dari setiap baris kode program pada blok # In[4] dan hasil luarannya.

- Code:

```
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)))
```

- 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 memfungsikan 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 memfungsikan 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:

7/1164067/Praktek/Chapter7AnnisaFathoroni4.jpg

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

Figure 1.14: 4 - Annisa Fathoroni

1.1.2.5 Soal No. 5

Jelaskan arti dari setiap baris kode program pada blok # In[5] dan hasil luarannya.

- Code:

```
from sklearn.preprocessing import LabelEncoder
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:

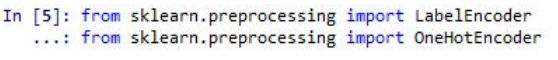
7/1164067/Praktek/Chapter7AnnisaFathoroni5.jpg  In [5]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder

Figure 1.15: 5 - Annisa Fathoroni

1.1.2.6 Soal No. 6

Jelaskan arti dari setiap baris kode program pada blok # In[6] dan hasil luarannya.

- Code:

```
label_encoder = LabelEncoder()  
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:

7/1164067/Praktek/Chapter7AnnisaFathoroni6.jpg  In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)

Figure 1.16: 6 - Annisa Fathoroni

1.1.2.7 Soal No. 7

Jelaskan arti dari setiap baris kode program pada blok # In[7] dan hasil luarannya.

- Code:

```
onehot_encoder = OneHotEncoder(sparse=False)  
integer_encoded = integer_encoded.reshape(len(integer_encoded), 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))
...: onehot_encoder.fit(integer_encoded)
D:\Anaconda\lib\site-packages\sklearn\preprocessing\_encoders.py:219: FutureWarning: The handling of integer data will change in version 0.22. Currently integer data is handled as a string, but this is not the case for all versions. Integers 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 use the parameter '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)
```

7/1164067/Praktek/Chapter7AnnisaFathoroni7.jpg

Figure 1.17: 6 - Annisa Fathoroni

1.1.2.8 Soal No. 8

Jelaskan arti dari setiap baris kode program pada blok # In[8] dan hasil luarannya.

- Code:

```
train_output_int = label_encoder.transform(train_output)
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int)))
test_output_int = label_encoder.transform(test_output)
test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int)))

num_classes = len(label_encoder.classes_)
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`.

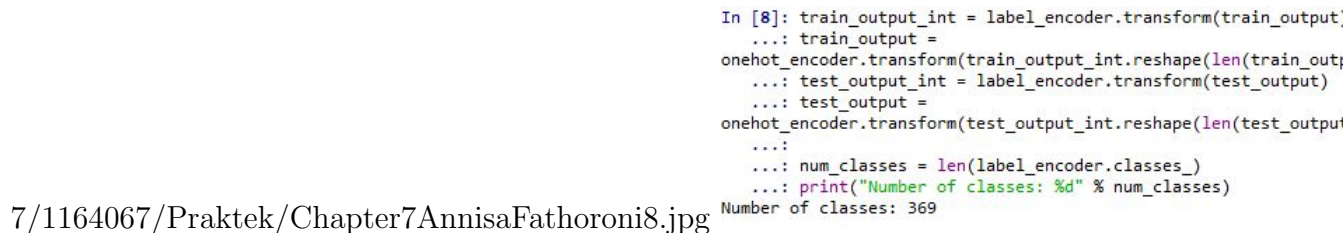
yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.

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 lebel_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)))
...: test_output_int = label_encoder.transform(test_output)
...: test_output =
onehot_encoder.transform(test_output_int.reshape(len(test_output)))
...:
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

7/1164067/Praktek/Chapter7AnnisaFathoroni8.jpg

Figure 1.18: 6 - Annisa Fathoroni

- Hasil output:

1.1.2.9 Soal No. 9

Jelaskan arti dari setiap baris kode program pada blok # In[9] dan hasil luarannya.

- Code:

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
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:

7/1164067/Praktek/Chapter7AnnisaFathoroni9.jpg

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

Figure 1.19: 6 - Annisa Fathoroni

1.1.2.10 Soal No. 10

Jelaskan arti dari setiap baris kode program pada blok # In[10] dan hasil luarannya.

- Code:

```
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(1024, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

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

- Penjelasan:

—

- Hasil output:

1.1.2.11 Soal No. 11

Jelaskan arti dari setiap baris kode program pada blok # In[11] dan hasil luarannya.

- Code:

```
import keras.callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-sty
```

- 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.

```

...: 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\
\op_def_library.py:263: colocate_with (from tensorflow.python
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\back
\tensorflow_backend.py:3445: calling dropout (from tensorflow
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
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		

7/1164067/Praktek/Chapter7AnnisaFathoroni10.jpg

Figure 1.20: 6 - Annisa Fathoroni

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:

7/1164067/Praktek/Chapter7AnnisaFathoroni11.jpg In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./

Figure 1.21: 6 - Annisa Fathoroni

1.1.2.12 Soal No. 12

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

- Code:

```
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])
```

- 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 parameter 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\
\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops)
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: 0.6294
Epoch 2/10
- 918s - loss: 0.9694 - acc: 0.7321 - val_loss: 0.9162 - val_acc: 0.7321
Epoch 3/10
- 562s - loss: 0.8543 - acc: 0.7556 - val_loss: 0.8883 - val_acc: 0.7556
Epoch 4/10
- 503s - loss: 0.7854 - acc: 0.7699 - val_loss: 0.8441 - val_acc: 0.7699
Epoch 5/10
- 361s - loss: 0.7364 - acc: 0.7796 - val_loss: 0.8491 - val_acc: 0.7796
Epoch 6/10
- 366s - loss: 0.6936 - acc: 0.7899 - val_loss: 0.8522 - val_acc: 0.7899
Epoch 7/10
- 360s - loss: 0.6591 - acc: 0.7962 - val_loss: 0.8580 - val_acc: 0.7962
Epoch 8/10
- 389s - loss: 0.6344 - acc: 0.8017 - val_loss: 0.8496 - val_acc: 0.8017
Epoch 9/10
- 357s - loss: 0.6076 - acc: 0.8072 - val_loss: 0.8597 - val_acc: 0.8072
Epoch 10/10
- 359s - loss: 0.5905 - acc: 0.8116 - val_loss: 0.8889 - val_acc: 0.8116
Test loss: 0.8794204677150739
Test accuracy: 0.7632181175230488
```

7/1164067/Praktek/Chapter7AnnisaFathoroni12.jpg

Figure 1.22: 6 - Annisa Fathoroni

1.1.2.13 Soal No. 13

Jelaskan arti dari setiap baris kode program pada blok # In[13] dan hasil luarannya.

- Code:

```
import time

results = []
for conv2d_count in [1, 2]:
    for dense_size in [128, 256, 512, 1024, 2048]:
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            model = Sequential()
            for i in range(conv2d_count):
                if i == 0:
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                else:
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
            model.add(MaxPooling2D(pool_size=(2, 2)))
```

- Penjelasan:

—

- Hasil output:

1.1.2.14 Soal No. 14

Jelaskan arti dari setiap baris kode program pada blok # In[14] dan hasil luarannya.

- Code:

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(28, 28, 1)))
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())
```

- Penjelasan:

—

- Hasil output:

1.1.2.15 Soal No. 15

Jelaskan arti dari setiap baris kode program pada blok # In[15] dan hasil luarannya.

- Code:

```
model.fit(np.concatenate((train_input, test_input)),
          np.concatenate((train_output, test_output)),
          batch_size=32, epochs=10, verbose=2)
```

- Penjelasan:

—

- Hasil output:

1.1.2.16 Soal No. 16

Jelaskan arti dari setiap baris kode program pada blok # In[16] dan hasil luarannya.

- Code:

```
model.save("mathsymbols.model")
```

- Penjelasan:

—

- Hasil output:

1.1.2.17 Soal No. 17

Jelaskan arti dari setiap baris kode program pada blok # In[17] dan hasil luarannya.

- Code:

```
np.save('classes.npy', label_encoder.classes_)
```

- Penjelasan:

—

- Hasil output:

1.1.2.18 Soal No. 18

Jelaskan arti dari setiap baris kode program pada blok # In[18] dan hasil luarannya.

- Code:

```
import keras.models
model2 = keras.models.load_model("mathsymbols.model")
print(model2.summary())
```

- Penjelasan:

—

- Hasil output:

1.1.2.19 Soal No. 19

Jelaskan arti dari setiap baris kode program pada blok # In[19] dan hasil luarannya.

- Code:

```
label_encoder2 = LabelEncoder()
label_encoder2.classes_ = np.load('classes.npy')

def predict(img_path):
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(
        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
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction))))
```

- Penjelasan:

—

- Hasil output:

1.1.2.20 Soal No. 20

Jelaskan arti dari setiap baris kode program pada blok # In[20] dan hasil luarannya.

- Code:

```
predict("HASYv2/hasy-data/v2-00010.png")
```

```
predict("HASYv2/hasy-data/v2-00500.png")
```

```
predict("HASYv2/hasy-data/v2-00700.png")
```

- Penjelasan:

—

- Hasil output:

1.1.3 Penanganan Error

1.2 Tasya Wiendhyra / 1164086

1.2.1 Teori

1.2.1.1 Jelaskan kenapa

teks harus di lakukan tokenizer dilengkapi dengan ilustrasi atau gambar

Untuk memudahkan mesin memahami maksud dari apa yang kita inginkan dalam machine learning, kata pada teks disebut token, dan proses vektorisasi dari bentuk kata ke dalam token tersebut disebut tokenizer dan tokenizer akan merubah sebuah teks menjadi simbol, kata, ataupun biner dan bentuk lainnya kedalam token. Untuk lebih jelasnya perhatikan ilustrasi berikut. Disini saya mempunyai sebuah kalimat yaitu "Nama Saya Tasya Wiendhyra" maka ketika kita lakukan proses tokenizer maka akan berubah menjadi ['Nama', 'Saya', 'Tasya', 'Wiendhyra'].

1.2.1.2 Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing ??dilengkapi dengan ilustrasi atau gambar

Listing 1.1: K Fold Cross Validation

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

StratifiedKFold berisikan presentasi sampel untuk setiap kelas. Dimana dalam ilustrasi ini sampel dibagi menjadi 5 dalam setiap class nya. Kemudian sampel tadi akan dimasukan kedalam class dari dataset youtube tadi.

Untuk ilustrasi lebih jelasnya, ada pada gambar berikut :

```
>>> from sklearn.model_selection import StratifiedKFold
>>> X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
>>> y = np.array([0, 0, 1, 1])
>>> skf = StratifiedKFold(n_splits=2)
>>> skf.get_n_splits(X, y)
2
>>> print(skf)
StratifiedKFold(n_splits=2, random_state=None, shuffle=False)
>>> for train_index, test_index in skf.split(X, y):
...     print("TRAIN:", train_index, "TEST:", test_index)
...     X_train, X_test = X[train_index], X[test_index]
...     y_train, y_test = y[train_index], y[test_index]
TRAIN: [1 3] TEST: [0 2]
TRAIN: [0 2] TEST: [1 3]
```

7/1164086/Teori/chapter7tasya1.png

Figure 1.23: Ilustrasi KFold Cross Tasya

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

Maksudnya yaitu untuk menguji apakah setiap data pada dataset sudah di split dan tidak terjadi penumpukan. Yang dimana maksudnya di setiap class tidak akan muncul

id yang sama. Ilustrasinya misalkan kita memiliki 4 baju dengan model yang berbeda. Kemudian kita bagikan kedua anak, tentunya setiap anak yang menerima baju tidak memiliki baju yang sama modelnya.

1.2.1.4 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

Maksudnya yaitu mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memilihnya untuk ditampilkan pada kolom yang diinginkan.

1.2.1.5 Soal No. 5 Jelaskan apa maksud dari fungsi `tokenizer = Tokenizer(num_words=2000)` dan `tokenizer.fit_on_texts(train_content)`, dilengkapi dengan ilustrasi atau gambar

Dimana variabel tokenizer akan melakukan vektorisasi kata menggunakan fungsi Tokenizer yang dimana jumlah kata yang ingin diubah kedalam bentuk token adalah 2000 kata. Dan untuk `tokenizer.fit_on_texts(train_content)` maksudnya kita akan melakukan fit tokenizer hanya untuk data trainnya saja tidak dengan data test nya untuk kolom CONTENT. Ilustrasinya, Jadi, jika Anda memberikannya sesuatu seperti, "Kucing itu duduk di atas tikar." Ini akan membuat kamus s.t. `word_index["the"] = 0`; `word_index["cat"] = 1` itu adalah kata -i kamus indeks sehingga setiap kata mendapatkan nilai integer yang unik.

1.2.1.6 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

Maksudnya yaitu untuk variabel d_train_inputs akan melakukan tokenizer dari bentuk teks ke matrix dari data train_content dengan mode matriksnya yaitu tfidf begitu juga dengan variabel d_test_inputs untuk data test. Berikut gambar ilustrasinya

1.2.1.7 Jelaskan apa maksud dari fungsi `d_train_inputs = d_train_inputs/np.amax(np.abs(d_train_inputs))` dan `d_test_inputs = d_test_inputs/np.amax(np.abs(d_test_inputs))`, dilengkapi dengan ilustrasi atau gambar

Fungsi tersebut akan membagi matrix tfidf tadi dengan amax yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Yang hasilnya akan dimasukkan

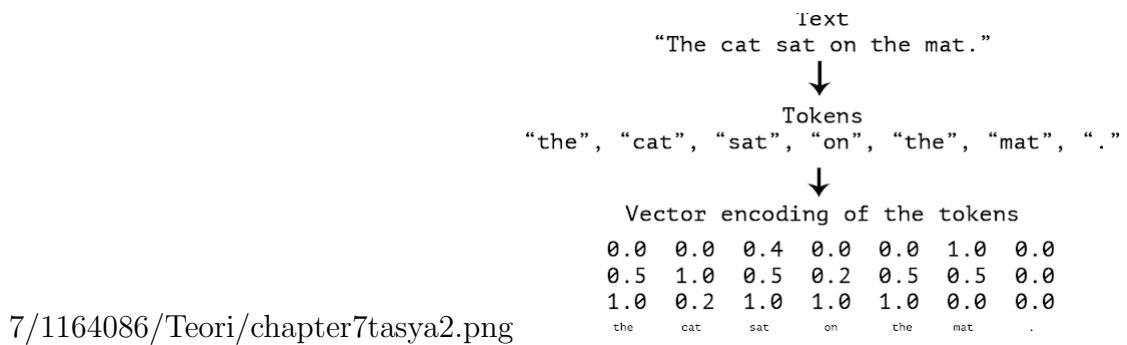


Figure 1.24: Ilustrasi Text To Matrix Tasya

kedalam variabel `d_train_inputs` untuk data train dan `d_test_inputs` untuk data test dengan nominal absolut atau tanpa ada bilangan negatif dan koma.

```

>>> x = np.array([-1.2, 1.2])
>>> np.absolute(x)
array([ 1.2,  1.2])

```

Figure 1.25: Ilustrasi np Absolute Tasya

1.2.1.8 Jelaskan apa maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d_train_outputs)` dan `d_test_outputs = np_utils.to_categorical(d_test_outputs)` dalam kode program, dilengkapi dengan ilustrasi atau gambar

Dalam variabel `d_train_output` dan `d_test_outputs` akan dilakukan one hot encoding, dimana `np_utils` akan mengubah vektor dengan bentuk integer ke matriks kelas biner untuk kolom CLASS dimana nantinya hanya akan ada dua pilihan yaitu 1 atau 0. 1 untuk spam 0 untuk non spam atau sebaliknya. Berikut gambar ilustrasinya :

```

> labels
array([0, 2, 1, 2, 0])
# 'to_categorical' converts this into a matrix with as many
# columns as there are classes. The number of rows
# stays the same.
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]])

```

Figure 1.26: Ilustrasi One Hot Encoding Tasya

1.2.1.9 Jelaskan apa maksud dari fungsi di listing ???. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

Listing 1.2: Membuat model Neural Network

```

model = Sequential()

```



```

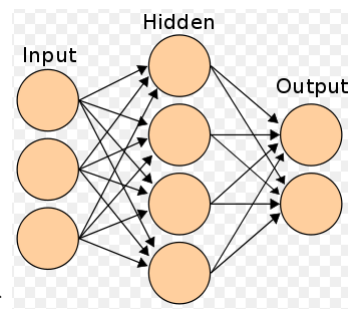
model.add(Dense(512, input_shape=(2000,)))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(2))
model.add(Activation('softmax'))

```

Penjelasannya sebagai berikut :

- Melakukan pemodelan Sequential
- Layer pertama dense dari 512 neuron untuk inputan dengan inputan tadi yang sudah dijadikan matriks sebanyak 2000
- Activationnya menggunakan fungsi relu yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.
- Dropout ini untuk melakukan pembobotan, dimana pembobotan hanya dilakukan 50% saja agar tidak terjadi penumpukan data dari dense inputan tadi
- Dense 2 mengkategorikan 2 neuron untuk output nya yaitu 1 dan 0.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax.

Ilustrasinya seperti berikut :



7/1164086/Teori/chapter7tasya6.png

Figure 1.27: Ilustrasi Neural Network Pemodelan Tasya

1.2.1.10 Jelaskan apa maksud dari fungsi di listing ?? dengan parameter tersebut

Listing 1.3: Compile model

```

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

```

Melakukan peng compile-an dari model Sequential tadi dengan Loss yang merupakan fungsi optimisasi skor menggunakan categorical_crossentropy , dan menggunakan algoritma adam sebagai optimizer. Adam yaitu algoritma pengoptimalan yang dapat digunakan sebagai ganti dari prosedur penurunan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training. Dengan metrik yaitu fungsi yang digunakan untuk menilai kinerja mode Anda disini menggunakan fungsi accuracy.

1.2.1.11 Jelaskan apa itu Deep Learning

Deep Learning adalah subbidang machine learning yang berkaitan dengan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan atau Artificial Neural Networks. Jaringan saraf tiruan, algoritma yang terinspirasi oleh otak manusia, belajar dari sejumlah besar data. Demikian pula dengan bagaimana kita belajar dari pengalaman, algoritma pembelajaran yang mendalam akan melakukan tugas berulang kali, setiap kali sedikit mengubahnya untuk meningkatkan hasilnya.

1.2.1.12 Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning

Deep Neural Network adalah jaringan syaraf tiruan (JST) dengan beberapa lapisan antara lapisan input dan output. DNN menemukan manipulasi matematis yang benar untuk mengubah input menjadi output, apakah itu hubungan linear atau hubungan non-linear. Merupakan jaringan syaraf dengan tingkat kompleksitas tertentu, jaringan syaraf dengan lebih dari dua lapisan. Deep Neural Network menggunakan pemodelan matematika yang canggih untuk memproses data dengan cara yang kompleks.

DNN hanya terdiri dari dua laipsan yaitu input dan output, sedangkan dalam Deep learning kita dapat mendefinisikan layer sebanyak yang kita inginkan atau butuhkan.

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

Stridenya 3

- terdapat data seperti berikut
- Kemudian hitung konvolusi untuk setiap matriksnya seperti berikut :
 - pertama

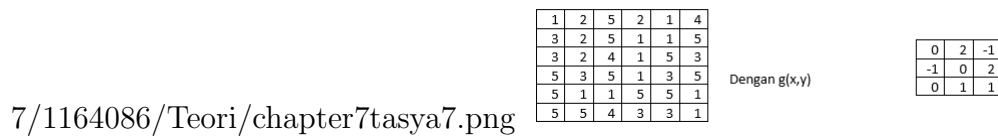


Figure 1.28: Algoritma Konvolusi Tasya

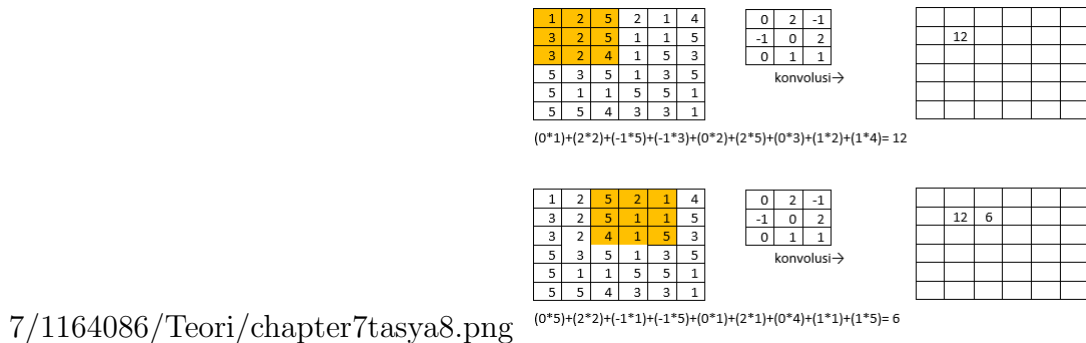


Figure 1.29: Algoritma Konvolusi Tasya

– Kedua

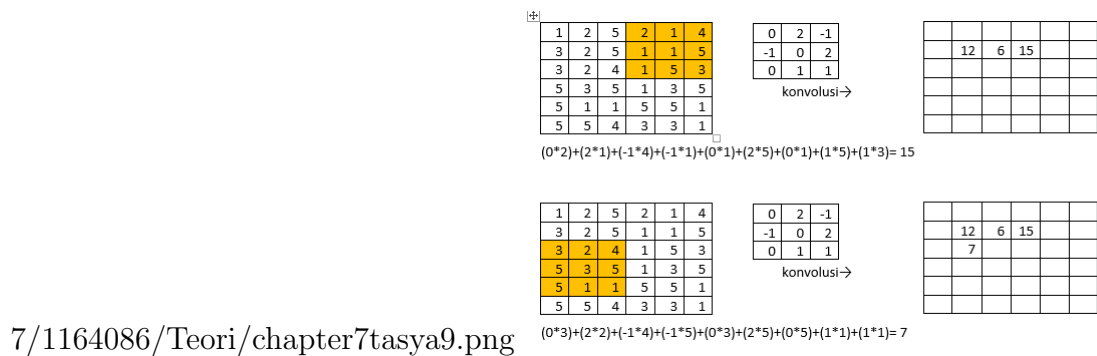


Figure 1.30: Algoritma Konvolusi Tasya

– Ketiga

– Keempat

– Kelima

- Didapatkan hasil akhir nilai konvolusi dan juga max poolingnya seperti berikut

1.2.2 Praktek

1.2.2.1 No.1 Kode Program Blok # In 1

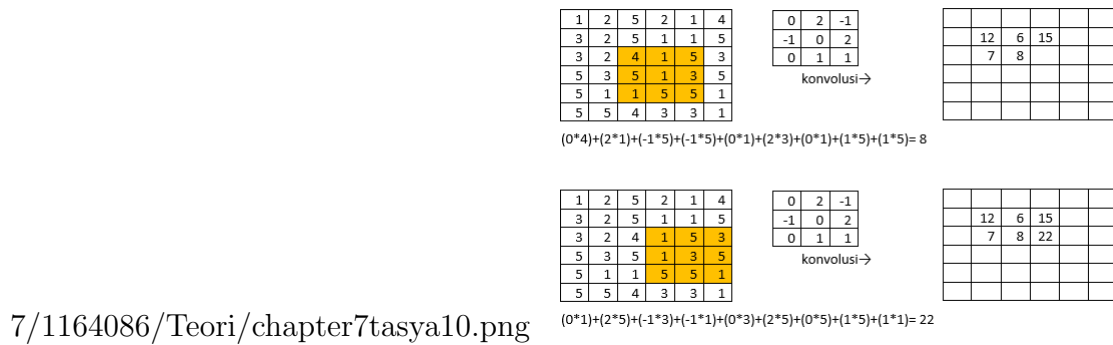


Figure 1.31: Algoritma Konvolusi Tasya

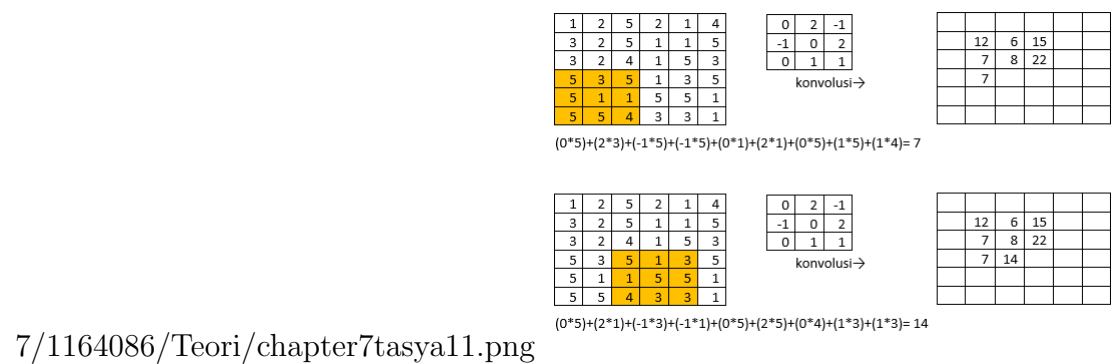
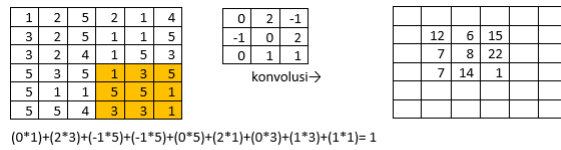


Figure 1.32: Algoritma Konvolusi Tasya

Keterangannya sebagai berikut :

- Pertama kita akan mengimpor librari csv
- Dimana dari librai PIL atau Pillow atau Python Imaging Library akan diimpor modul Image yang di inisiasikan sebagain pil.image. Modul Image menyediakan kelas dengan nama yang sama yang digunakan untuk mewakili gambar PIL. Modul ini juga menyediakan sejumlah fungsi pabrik, termasuk fungsi untuk memuat image dari file, dan untuk membuat image baru.
- mengimpor librari image dari keras .Yang menghasilkan kumpulan data gambar tensor dengan augmentasi data waktu nyata. Data akan diulang (dalam batch).
- Berikut Hasilnya :

1.2.2.2 No.2 Kode Program Blok # In 2



7/1164086/Teori/chapter7tasya12.png

Figure 1.33: Algoritma Konvulusi Tasya



7/1164086/Teori/chapter7tasya13.png

Figure 1.34: Algoritma Konvulusi Tasya

```

for row in csvreader:
    if i > 0:
        img = keras.preprocessing.image.img_to_array(pil_image.open)
        # neuron activation functions behave best when input values
        # so we rescale each pixel value to be in the range 0.0 to
        img /= 255.0
        imgs.append((row[0], row[2], img))
        classes.append(row[2])
    i += 1

```

Keterangannya sebagai berikut :

- variabel imgs berisikan array kosong
- Variabel classes berisikan array kosong
- Membuka file csv dari Folder HSYv2 dengan nama file hasy-data-labels.csv sebagai csvfile
- Variabel csvreader akan menggunakan fungsi reader pada library csv untuk membaca file csv tadi yang disimpan di csvfile.
- Dimana variabel i dimuali dari nol.
- Untuk setiap baris pada csvreader

7/1164086/Praktek/chapter7tasya14.png

```

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

```

Figure 1.35: Kode Program Blok In 1 Tasya

- Jika i lebih besar dari 0
- Jadi itu akan mengambil contoh Gambar PIL dan mengubahnya menjadi array numpy dengan mengambil data dari HSYv2 dan dimulai dari baris ke nol.
- Hasil dari variabel img akan dibagi dengan 255.0
- .append akan membuat list array baru untuk baris 0 baris 2 pada img.
- Menyimpan setiap class nya pada baris 2
- Penambahan i sebanyak 1.
- Hasilnya seperti berikut :

7/1164086/Praktek/chapter7tasya15.png

```

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

```

Figure 1.36: Kode Program Blok In 2 Tasya

1.2.2.3 No.3 Kode Program Blok # In 3

Keterangannya sebagai berikut :

- Impor librari Random dari Python
- Melakukan pengacakan untuk imgs dengan Metode Shuffle untuk mengocok urutan di tempat. yaitu, mengubah posisi item dalam daftar.
- Membagi data dari imgs dengan cara mengalikan 80% dengan jumlah data dari imgs.

- Untuk data train mengambil hasil dari perhitungan sebelumnya.
- Untuk data test mengambil sisa dari jumlah yang telah dijadikan data train
- Hasilnya seperti berikut :

7/1164086/Praktek/chapter7tasya16.png

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

Figure 1.37: Kode Program Blok In 3 Tasya

1.2.2.4 No.4 Kode Program Blok # In 4

```
train_output = np.asarray(list(map(lambda row: row[1], train)))
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Keterangannya sebagai berikut :

- Impor librari Numpy yang di inisiasikan sebagai np
- Variabel train_input mengubah input menjadi sebuah array yang diambil dari baris 2, data train.
- Variabel test_input mengubah input menjadi sebuah array yang diambil dari baris 2, data test.
- Variabel train_output mengubah input menjadi sebuah array yang diambil dari baris 1, data train.
- Variabel train_output mengubah input menjadi sebuah array yang diambil dari baris 1, data test.
- Hasilnya seperti berikut

7/1164086/Praktek/chapter7tasya17.png

```
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)))
```

Figure 1.38: Kode Program Blok In 4 Tasya

1.2.2.5 No.5 Kode Program Blok # In 5

Keterangannya sebagai berikut :

- Impor Fungsi LabelEncoder
- Impor Fungsi OneHotEncoder
- Berikut hasilnya :

7/1164086/Praktek/chapter7tasya18.png In [5]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder

Figure 1.39: Kode Program Blok In 5 Tasya

1.2.2.6 No.6 Kode Program Blok # In 6

Keterangannya sebagai berikut :

- Variabel label_encoder akan memanggil fungsi LabelEncoder tadi.
- variabel integer_encoded akan menggunakan labelencoder untuk melakukan fit pada classes agar berubah datanya menjadi integer.
- Berikut hasilnya :

7/1164086/Praktek/chapter7tasya19.png In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)

Figure 1.40: Kode Program Blok In 6 Tasya

1.2.2.7 No.7 Kode Program Blok # In 7

Keterangannya sebagai berikut :

- Variabel onehot_encoder akan memanggil fungsi OneHotEncoder dimana tidak berisikan matriks sparse.
- Pada variabel integer_encoded akan diubah bentuknya dimana setiap nilai integer akan direpresentasikan sebagai vektor binari dengan nilai 0 kecuali index dari integer tersebut ditandai dengan 1.

- Melakukan fit untuk one hot encoder kedalam integer_encoder.
- Berikut hasilnya :

7/1164086/Praktek/chapter7tasya20.png

```
In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
C:\Users\DidinIrfandi\Anaconda3\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)
```

Figure 1.41: Kode Program Blok In 7 Tasya

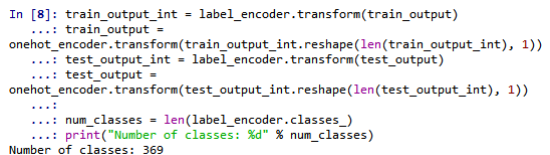
1.2.2.8 No.8 Kode Program Blok # In 8

```
print ("Number of classes : %d" % num_classes)
```

Keterangannya sebagai berikut :

- Variabel train_output_int akan mengubah data dari train_output menjadi LabeEncoder
- Dimana pada train_output setelah diubah labelnya menjadi integer dilakukan one hot encoding diambil dari train_output_int dan menggunakan .reshape untuk memberikan bentuk baru ke array tanpa mengubah datanya dengan keterangan jika index dari integer tersebut ditandai dengan 1 dan sisanya yang bukan nol.
- Variabel test_output_int akan mengubah data dari test_output menjadi LabeEncoder
- Dimana pada train_output setelah diubah labelnya menjadi integer dilakukan one hot encoding diambil dari test_output_int dan menggunakan .reshape untuk memberikan bentuk baru ke array tanpa mengubah datanya dengan keterangan jika index dari integer tersebut ditandai dengan 1 dan sisanya yang bukan nol.
- Variabel num_classes akan menampilkan jumlah data dari classes yang telah dilakukan label encoder

- Menampilkan tulisan "Number of classes : %d" dimana mengembalikan nilai integer dari num_classes.
- Hasilnya sebagai berikut :



```
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
```

7/1164086/Praktek/chapter7tasya21.png

Figure 1.42: Kode Program Blok In 8 Tasya

1.2.2.9 No.9 Kode Program Blok # In 9

Keterangannya sebagai berikut :

- Impor Sequential dari model pada librari Keras.
- Impor Dense, Dropout, Flatten dari modul Layers pada librari Keras.
- Impor Conv2D, MaxPooling2D dari modul Layers pada librari Keras.
- Hasilnya seperti berikut :



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

7/1164086/Praktek/chapter7tasya22.png

Figure 1.43: Kode Program Blok In 9 Tasya

1.2.2.10 No.10 Kode Program Blok # In 10

```
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())
```

Keterangannya sebagai berikut :

- Melakukan pemodelan Sequential.
- Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train_input mulai dari baris nol.
- Menambahkan Max Pooling dengan matriks 2x2.
- Dilakukan lagi penambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.
- Menambahkan lagi Max Pooling dengan matriks 2x2.
- Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.
- Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .
- Untuk output layer menggunakan data dari variabel num_classes dengan fungsi activationnya softmax.
- Mengonfigurasi proses pembelajaran, yang dilakukan melalui metode compile, sebelum melatih suatu model.
- Menampilkan atau mencetak representasi ringkasan model yang telah dibuat.
- Hasilnya sebagai berikut :

1.2.2.11 No.11 Kode Program Blok # In 11

Keterangannya sebagai berikut :

- Impor Modul Callbacks dari Librari Keras.

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		

7/1164086/Praktek/chapter7tasya23.png

Figure 1.44: Kode Program Blok In 10 Tasya

- Variabel callback mendefinisikan Callback ini untuk menulis log untuk TensorBoard, yang memungkinkan Anda untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian Anda, serta histogram aktivasi untuk berbagai lapisan dalam model Anda.
- Hasilnya sebagai berikut :

7/1164086/Praktek/chapter7tasya24.png

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

Figure 1.45: Kode Program Blok In 11 Tasya

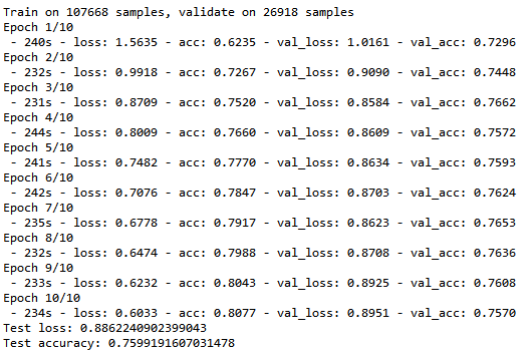
1.2.2.12 No.12 Kode Program Blok # In 12

```
score = model.evaluate(test_input , test_output , verbose=2)
print('Test_loss: ', score[0])
print('Test_accuracy: ', score[1])
```

Keterangannya sebagai berikut :

- Melakukan fit model dengan 32 ukuran subset dari sampel pelatihan Anda
- Epoch sebanyak 10 kali
- Vebrose=2 maksudnya menampilkan nomor dari epoch yang sedang berjalan atau yang sudah dijalankan.
- Validasi plit sebanyak 20% sebagai fraksi data pelatihan untuk digunakan sebagai data validasi.

- Menggunakan TensorBoard sebagai callback untuk diterapkan selama pelatihan dan validasi.
- Variabel score mengembalikan nilai evaluate untuk menampilkan data lost dan data accuracy dari test
- Menampilkan data loss dengan menghitung jumlah kemunculan nol .
- Menampilkan data accuracy dengan menghitung jumlah kemunculan 1.
- Berikut hasilnya :



```

Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 240s - loss: 1.5635 - acc: 0.6235 - val_loss: 1.0161 - val_acc: 0.7296
Epoch 2/10
- 232s - loss: 0.9918 - acc: 0.7267 - val_loss: 0.9090 - val_acc: 0.7448
Epoch 3/10
- 231s - loss: 0.8709 - acc: 0.7520 - val_loss: 0.8584 - val_acc: 0.7662
Epoch 4/10
- 244s - loss: 0.8009 - acc: 0.7660 - val_loss: 0.8609 - val_acc: 0.7572
Epoch 5/10
- 241s - loss: 0.7482 - acc: 0.7770 - val_loss: 0.8634 - val_acc: 0.7593
Epoch 6/10
- 242s - loss: 0.7076 - acc: 0.7847 - val_loss: 0.8703 - val_acc: 0.7624
Epoch 7/10
- 235s - loss: 0.6778 - acc: 0.7917 - val_loss: 0.8623 - val_acc: 0.7653
Epoch 8/10
- 232s - loss: 0.6474 - acc: 0.7988 - val_loss: 0.8708 - val_acc: 0.7636
Epoch 9/10
- 233s - loss: 0.6232 - acc: 0.8043 - val_loss: 0.8925 - val_acc: 0.7608
Epoch 10/10
- 234s - loss: 0.6033 - acc: 0.8077 - val_loss: 0.8951 - val_acc: 0.7570
Test loss: 0.8862240902399043
Test accuracy: 0.7599191607031478

```

7/1164086/Praktek/chapter7tasya25.png

Figure 1.46: Kode Program Blok In 12 Tasya

1.2.2.13 No.13 Kode Program Blok # In 13

```

for dropout in [0.0, 0.25, 0.50, 0.75]:
    model = Sequential()
    for i in range(conv2d_count):
        if i == 0:
            model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
        else:
            model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
            model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Flatten())
    model.add(Dense(dense_size, activation='tanh'))
    if dropout > 0.0:
        model.add(Dropout(dropout))
    model.add(Dense(num_classes, activation='softmax'))

```

Keterangannya sebagai berikut :

- impor modul time dari python anaconda

- Variabel result berisikan array kosong.
- Menggunakan convolution 2D yang dimana akan memiliki 1 atau 2 layer.
- Mendefinisikan dense_size dengan ukuran 128, 256, 512, 1024, 2048
- Mendefinisikan drop_out dengan 0, 25%, 50%, dan 75%
- Melakukan pemodelan Sequential
- Jika ini adalah layer pertama, kita perlu memasukkan bentuk input.
- Kalau tidak kita hanya akan menambahkan layer.
- Kemudian, setelah menambahkan layer konvolusi, kita akan melakukan hal yang sama dengan max pooling.
- Lalu, kita akan meratakan atau flatten dan menambahkan dense size ukuran apapun yang berasal dari dense_size. Dimana akan selalu menggunakan algoritma tanh
- Jika dropout digunakan, kita akan menambahkan layer dropout. Menyebut dropout ini berarti, katakanlah 50%, bahwa setiap kali ia memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui
- menempatkan ini di antara dua lapisan padat untuk dihidupkan dari melindunginya dari overfitting.
- Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.
- Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.
- Variabel start akan memanggil modul time atau waktu
- Melakukan fit atau compile
- Melakukan scoring dengan .evaluate yang akan menampilkan data loss dan accuracy dari model
- end merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.

- Menampilkan hasil dari run skrip diatas
- Hasilnya sebagai berikut :

```

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: 1572
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.91, Accuracy: 0.76, Time: 1632
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.80, Accuracy: 0.78, Time: 1662
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.80, Accuracy: 0.78, Time: 1735
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.28, Accuracy: 0.74, Time: 2153
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.08, Accuracy: 0.76, Time: 2212
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.91, Accuracy: 0.78, Time: 2184
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.75 - Loss: 0.78, Accuracy: 0.78, Time: 2184
sec

```

7/1164086/Praktek/chapter7tasya26.png

Figure 1.47: Kode Program Blok In 13 Tasya

1.2.2.14 No.14 Kode Program Blok # In 14

```

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', metric=
print(model.summary()))

```

Keterangannya sebagai berikut :

- Melakukan pemodelan Sequential
- Untuk layer pertama, Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input_shape
- Dilakukan Max Pooling 2D dengan ukuran matriks 2x2
- Untuk layer kedua, melakukan Convulasi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik
- Flatten digubakan ntuk meratakan inputan
- Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.
- Dropout sebanyak 50% untuk menghindari overfitting

- Menambahkan dense pada model untuk output dimana layer ini akan menjadi jumlah dari class yang ada.
- Mengcompile model yang didefinisikan diatas
- Menampilkan ringkasan dari pemodelan yang dilakukan
- Gambarnya seperti berikut :

```
.... print(model.summary())
```

Layer (type)	Output Shape	Param #
conv2d_12 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_12 (MaxPooling)	(None, 15, 15, 32)	0
conv2d_13 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_13 (MaxPooling)	(None, 6, 6, 32)	0
flatten_11 (Flatten)	(None, 1152)	0
dense_21 (Dense)	(None, 128)	147584
dropout_8 (Dropout)	(None, 128)	0
dense_22 (Dense)	(None, 369)	47601
Total params: 205,329		
Trainable params: 205,329		
Non-trainable params: 0		

7/1164086/Praktek/chapter7tasya27.png

Figure 1.48: Kode Program Blok In 14 Tasya

1.2.2.15 No.15 Kode Program Blok # In 15

Keterangannya sebagai berikut :

- Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.
- Hasilnya sebagai berikut :

1.2.2.16 No.16 Kode Program Blok # In 16

Keterangannya sebagai berikut :

- Menyimpan atau save model yang telah di latih dengan nama mathsymbols.model
- Hasilnya seperti berikut :


```

In [15]: model.fit(np.concatenate((train_input, test_input)),
...:              np.concatenate((train_output, test_output)),
...:              batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 267s - loss: 1.7763 - acc: 0.5881
Epoch 2/10
- 249s - loss: 1.0767 - acc: 0.7061
Epoch 3/10
- 248s - loss: 0.9666 - acc: 0.7297
Epoch 4/10
- 249s - loss: 0.9096 - acc: 0.7411
Epoch 5/10
- 248s - loss: 0.8699 - acc: 0.7518
Epoch 6/10
- 253s - loss: 0.8428 - acc: 0.7551
Epoch 7/10
- 250s - loss: 0.8211 - acc: 0.7608
Epoch 8/10
- 247s - loss: 0.8050 - acc: 0.7639
Epoch 9/10
- 252s - loss: 0.7880 - acc: 0.7679
Epoch 10/10
- 251s - loss: 0.7751 - acc: 0.7697
Out[15]: <keras.callbacks.History at 0x22f13d79a20>

```

7/1164086/Praktek/chapter7tasya29.png

Figure 1.49: Kode Program Blok In 15 Tasya

```

In [16]: model.save("mathsymbols.model")
mathsymbols.model 13/04/2019 03.45 MODEL File 2.443 KB

```

7/1164086/Praktek/chapter7tasya30.png

Figure 1.50: Kode Program Blok In 16 Tasya

1.2.2.17 No.17 Kode Program Blok # In 17

Keterangannya sebagai berikut :

- Simpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy
- Hasilnya seperti berikut :

```

In [17]: np.save('classes.npy', label_encoder.classes_)
classes.npy 13/04/2019 03.45 NPY File 28 KB

```

7/1164086/Praktek/chapter7tasya31.png

Figure 1.51: Kode Program Blok In 17 Tasya

1.2.2.18 No.18 Kode Program Blok # In 18

Keterangannya sebagai berikut :

- Impor models dari librari Keras
- Variabel model2 akan memanggil model yang telah disave tadi
- Menampilkan ringkasan dari hasil pemodelan
- Hasilnya sebagai berikut :

```

...: model2.summary()
...: print(model2.summary())

```

Layer (type)	Output Shape	Param #
conv2d_12 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_12 (MaxPooling)	(None, 15, 15, 32)	0
conv2d_13 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_13 (MaxPooling)	(None, 6, 6, 32)	0
flatten_11 (Flatten)	(None, 1152)	0
dense_21 (Dense)	(None, 128)	147584
dropout_8 (Dropout)	(None, 128)	0
dense_22 (Dense)	(None, 369)	47601
Total params: 205,329		
Trainable params: 205,329		
Non-trainable params: 0		

7/1164086/Praktek/chapter7tasya32.png

Figure 1.52: Kode Program Blok In 18 Tasya

1.2.2.19 No.19 Kode Program Blok # In 19

```

# do the prediction
prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

# figure out which output neuron had the highest score, and reverse
inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
print(" Prediction: %s, confidence: %.2f" % (inverted[0], np.max(pre

```

Keterangannya sebagai berikut :

- Memanggil fungsi LabelEncoder
- Variabel label_encoder akan memanggil class yang disave sebelumnya.
- Function Predict akan mengubah gambar kedalam bentuk array
- Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.
- Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi
- Menampilkan hasil dari variabel prediction dan inverted
- Hasilnya sebagai berikut :

```

In [19]: 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)))

```

7/1164086/Praktek/chapter7tasya33.png

Figure 1.53: Kode Program Blok In 19 Tasya

1.2.2.20 No.20 Kode Program Blok # In 20

```

# do the prediction
prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

# figure out which output neuron had the highest score, and reverse
inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
print(" Prediction: %s, confidence: %.2f" % (inverted[0], np.max(pre

```

Keterangannya sebagai berikut :

- Melakukan prediksi dari pelatihan dari gambar v2-00010.png
- Melakukan prediksi dari pelatihan dari gambar v2-00500.png
- Melakukan prediksi dari pelatihan dari gambar v2-00700.png
- Hasilnya sebagai berikut :

```

In [20]: predict("HASYv2/hasy-data/v2-00010.png")
...:
...: predict("HASYv2/hasy-data/v2-00500.png")
...:
...: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: A, confidence: 0.85
Prediction: \pi, confidence: 0.81
Prediction: \alpha, confidence: 0.90

```

7/1164086/Praktek/chapter7tasya34.png

Figure 1.54: Kode Program Blok In 20 Tasya

1.2.3 Penanganan Error

1.2.3.1 Error Starting Kernel

- Berikut merupakan screenshot error

7/1164086/Praktek/chapter7error1.png

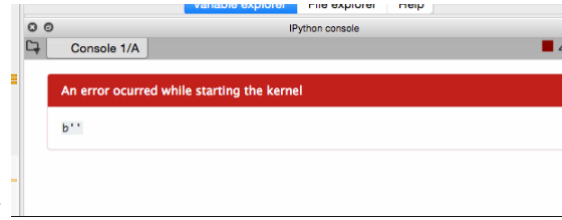


Figure 1.55: Error Tasya

- Error tersebut merupakan error yang terjadi dan membuat kita tidak dapat mengakses dan menggunakan kernel atau konsol pada spyder.
- Untuk penanganannya sebagai berikut :
 1. Tutup spyder yang sedang dijalankan
 2. Kemudian buka kembali spyder
 3. Atau jika tidak berhasil, buka anaconda prompt dan ketikan "conda update spyder"
 4. Jika tidak berhasil juga bisa menginstall ulang anaconda
 5. Maka ketika dijaalankan lagi hasilnya seperti berikut :

7/1164086/Praktek/chapter7error2.png

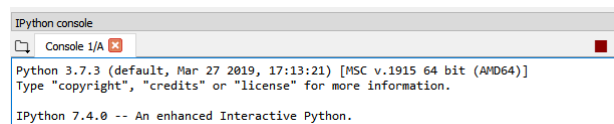


Figure 1.56: Penanganan Error Kernel Tasya

Appendix A

Form Penilaian Jurnal

gambar ?? dan ?? merupakan contoh bagaimana reviewer menilai jurnal kita.

NO	UNSUR	KETERANGAN	MAKS	KETERANGAN
1	Keefektifan Judul Artikel	Maksimal 12 (dua belas) kata dalam Bahasa Indonesia atau 10 (sepuluh) kata dalam Bahasa Inggris	2	a. Tidak lugas dan tidak ringkas (0) b. Kurang lugas dan kurang ringkas (1) c. Ringkas dan lugas (2)
2	Pencantuman Nama Penulis dan Lembaga Penulis		1	a. Tidak lengkap dan tidak konsisten (0) b. Lengkap tetapi tidak konsisten (0,5) c. Lengkap dan konsisten (1)
3	Abstrak	Dalam Bahasa Indonesia dan Bahasa Inggris yang baik, jumlah 150-200 kata. Isi terdiri dari latar belakang, metode, hasil, dan kesimpulan. Isi tertuang dengan kalimat yang jelas.	2	a. Tidak dalam Bahasa Indonesia dan Bahasa Inggris (0) b. Abstrak kurang jelas dan ringkas, atau hanya dalam Bahasa Inggris, atau dalam Bahasa Indonesia saja (1) c. Abstrak yang jelas dan ringkas dalam Bahasa Indonesia dan Bahasa Inggris (2)
4	Kata Kunci	Maksimal 5 kata kunci terpenting dalam paper	1	a. Tidak ada (0) b. Ada tetapi kurang mencerminkan konsep penting dalam artikel (0,5) c. Ada dan mencerminkan konsep penting dalam artikel (1)
5	Sistematika Pembahasan	Terdiri dari pendahuluan, tinjauan pustaka, metode penelitian, hasil dan pembahasan, kesimpulan dan saran, daftar pustaka	1	a. Tidak lengkap (0) b. Lengkap tetapi tidak sesuai sistematika (0,5) c. Lengkap dan bersistem (1)
6	Pemanfaatan Instrumen Pendukung	Pemanfaatan Instrumen Pendukung seperti gambar dan tabel	1	a. Tidak dimanfaatkan (0) b. Kurang informatif atau komplementer (0,5) c. Informatif dan komplementer (1)
7	Cara Pengacuan dan Pengutipan		1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
8	Penyusunan Daftar Pustaka	Penyusunan Daftar Pustaka	1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
9	Peristilahan dan Kebahasaan		2	a. Buruk (0) b. Baik (1) c. Cukup (2)
10	Makna Sumbangan bagi Kemajuan		4	a. Tidak ada (0) b. Kurang (1) c. Sedang (2) d. Cukup (3) e. Tinggi (4)

Figure A.1: Form nilai bagian 1.

11	Dampak Ilmiah		7	a. Tidak ada (0) b. Kurang (1) c. Sedang (3) d. Cukup (5) e. Besar (7)
12	Nisbah Sumber Acuan Primer berbanding Sumber lainnya	Sumber acuan yang langsung merujuk pada bidang ilmiah tertentu, sesuai topik penelitian dan sudah teruji.	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
13	Derajat Kemutakhiran Pustaka Acuan	Derajat Kemutakhiran Pustaka Acuan	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
14	Analisis dan Sintesis	Analisis dan Sintesis	4	a. Sedang (2) b. Cukup (3) c. Baik (4)
15	Penyimpulan	Sangat jelas relevasinya dengan latar belakang dan pembahasan, dirumuskan dengan singkat	3	a. Kurang (1) b. Cukup (2) c. Baik (3)
16	Unsur Plagiat		0	a. Tidak mengandung plagiat (0) b. Terdapat bagian-bagian yang merupakan plagiat (-5) c. Keseluruhannya merupakan plagiat (-20)
TOTAL			36	
Catatan : Nilai minimal untuk diterima 25				

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

M : Kalo Intership II atau TA harus buat aplikasi ? D : Ga harus buat aplikasi tapi harus ngoding

M : Pa saya bingung mau ngapain, saya juga bingung mau presentasi apa? D : Makanya baca de, buka jurnal topik ‘ganteng’ nah kamu baca dulu sehari 5 kali ya, 4 hari udah 20 tuh. Bingung itu tanda kurang wawasan alias kurang baca.

M : Pa saya sudah cari jurnal terindeks scopus tapi ga nemu. D : Kamu punya mata de? coba dicolok dulu. Kamu udah lakuin apa aja? tolong di list laporkan ke grup Tingkat Akhir. Tinggal buka google scholar klik dari tahun 2014, cek nama jurnalnya di scimagojr.com beres.

M : Pa saya belum dapat tempat intership, jadi ga tau mau presentasi apa? D : kamu kok ga nyambung, yang dipresentasikan itu yang kamu baca bukan yang akan kamu lakukan.

M : Pa ini jurnal harus yang terindex scopus ga bisa yang lain ? D : Index scopus menandakan artikel tersebut dalam standar semantik yang mudah dipahami dan dibaca serta bukan artikel asal jadi. Jika diluar scopus biasanya lebih sukar untuk dibaca dan dipahami karena tidak adanya proses review yang baik dan benar terhadap artikel.

M : Pa saya tidak mengerti D : Coba lihat standar alasan

M : Pa saya bingung D : Coba lihat standar alasan

M : Pa saya sibuk D : Mbahmu....

M : Pa saya ganteng D : Ndasmu....

M : Pa saya kece D : wes karepmu lah....

Biasanya anda memiliki alasan tertentu jika menghadapi kendala saat proses bimbingan, disini saya akan melakukan standar alasan agar persepsi yang diterima sama dan tidak salah kaprah. Penggunaan kata alasan tersebut antara lain :

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.

2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.