

**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
8 Chapter 8	41
9 Chapter 9	85
10 Chapter 10	87
11 Chapter 11	89
12 Chapter 12	91
13 Chapter 13	93
14 Chapter 14	95



# DAFTAR ISI

---

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

<b>5</b>	<b>Chapter 5</b>	<b>9</b>
<b>6</b>	<b>Chapter 6</b>	<b>11</b>
<b>7</b>	<b>Chapter 7</b>	<b>13</b>
7.1	Muhammad Tomy Nur Maulidy 1174031	13
7.1.1	Teori	13
7.1.2	Praktek	19
7.1.3	Penanganan Error	40
<b>8</b>	<b>Chapter 8</b>	<b>41</b>
8.1	1174021 - Muhammad Fahmi	41
8.1.1	Soal Teori	41
8.1.2	Praktek Program	45
8.1.3	Penanganan Error	55
8.1.4	Bukti Tidak Plagiat	56
8.2	1174017 - Muh. Rifky Prananda	56
8.2.1	Soal Teori	56
8.2.2	Praktek Program	60
8.2.3	Penanganan Error	70
8.2.4	Bukti Tidak Plagiat	70
8.3	1174008 - Arjun Yuda Firwanda	71
8.3.1	Teori	71
8.3.2	Praktek	75
8.3.3	Penanganan Error	83
8.3.4	Bukti Tidak Plagiat	83
<b>9</b>	<b>Chapter 9</b>	<b>85</b>
<b>10</b>	<b>Chapter 10</b>	<b>87</b>
<b>11</b>	<b>Chapter 11</b>	<b>89</b>
<b>12</b>	<b>Chapter 12</b>	<b>91</b>
<b>13</b>	<b>Chapter 13</b>	<b>93</b>
<b>14</b>	<b>Chapter 14</b>	<b>95</b>

Daftar Pustaka	97
Index	99



# **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**

---



# BAB 7

---

# CHAPTER 7

---

## 7.1 Muhammad Tomy Nur Maulidy 1174031

### 7.1.1 Teori

**7.1.1.1 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:

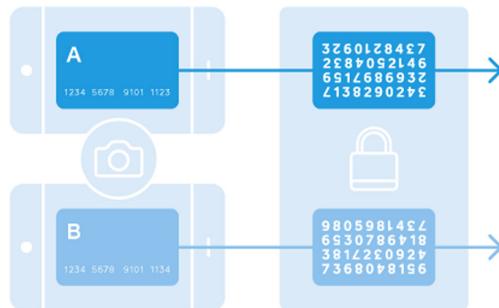
**7.1.1.2 Soal No. 2** Konsep dasar K Fold Cross Validation pada dataset komentar Youtube, dilengkapi dengan ilustrasi atau gambar.

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:

## Tokenization Simplified



Gambar 7.1 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
					Data Pengujian				
					Data Pelatihan				

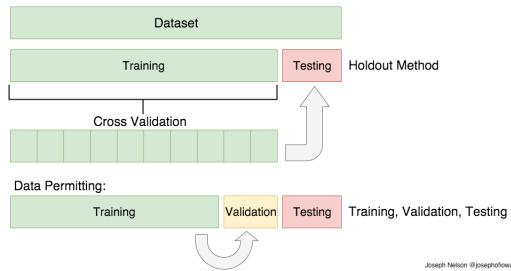
Gambar 1 – Skema 10 fold CV

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 token 2000 kata.

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

- Ilustrasi Gambar :

**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.

## ✓ 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.

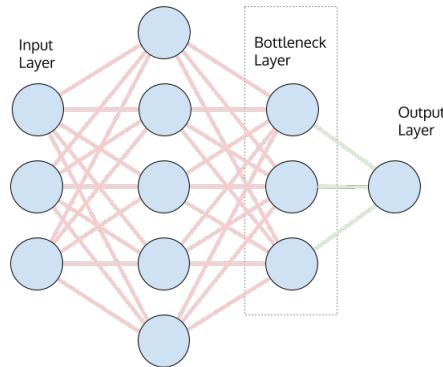
```
[1]: from nltk.tokenize import sent_tokenize
      sentence = "I love ice cream. I also like steak."
      sent_tokenize(sentence)

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

**Gambar 7.5** Tokenizer

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:

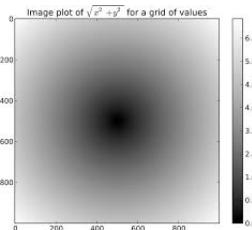


**Gambar 7.6** Train Inputs 1

**7.1.1.7 Soal No. 7** Maksud dari fungsi  $d_{\text{train inputs}} = \text{d train inputs}/\text{np.amax}(\text{np.absolute}(\text{d train inputs}))$  dan  $d_{\text{test inputs}} = \text{d test inputs}/\text{np.amax}(\text{np.absolute}(\text{d test inputs}))$ , 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 :

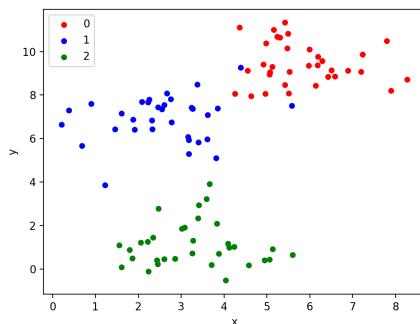


**Gambar 7.7** Train Inputs 2

**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 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  $2 \times 2$ . Misalkan  $f(x,y)$  yang digunakan berukuran  $3 \times 3$  dan kernel atau mask berukuran  $2 \times 2$  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 :

$$(2x1) + (3x0) + (1x2) + (0x4)$$

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

**Gambar 7.9** Perhitungan algoritma konvolusi

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 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 7.13** 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), 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 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 7.18 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:

**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

```

....: 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 #
<hr/>		
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
<hr/>		
max_pooling2d_1 (MaxPooling2D)	(None, 15, 15, 32)	0
<hr/>		
conv2d_2 (Conv2D)	(None, 13, 13, 32)	9248
<hr/>		
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 32)	0
<hr/>		
flatten_1 (Flatten)	(None, 1152)	0
<hr/>		
dense_1 (Dense)	(None, 1024)	1180672
<hr/>		
dropout_1 (Dropout)	(None, 1024)	0
<hr/>		
dense_2 (Dense)	(None, 369)	378225
<hr/>		
Total params: 1,569,041		
Trainable params: 1,569,041		
Non-trainable params: 0		
<hr/>		
None		

Gambar 7.20 10

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\_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 tensorflow 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 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),
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
45 f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" %
(conv2d_count
, dense_size, dropout, score[0], score[1], elapsed))
46 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:

**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 """
```

```

.....
....      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
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
input_shape=np.shape(train_input[0])))
10 model.add(MaxPooling2D(pool_size=(2, 2)))
11 model.add(Conv2D(32, (3, 3), activation='relu'))
12 model.add(MaxPooling2D(pool_size=(2, 2)))
13 model.add(Flatten())
14 model.add(Dense(128, activation='tanh'))
15 model.add(Dropout(0.5))
16 model.add(Dense(num_classes, activation='softmax'))
17 model.compile(loss='categorical_crossentropy', optimizer='adam',
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:

```
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 7.24 14

**7.1.2.15 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 7.25** 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 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 7.28** 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 # 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 7.29 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")
...
...: 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 7.30** 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.

# BAB 8

---

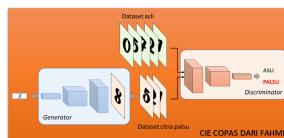
# CHAPTER 8

---

## 8.1 1174021 - Muhammad Fahmi

### 8.1.1 Soal Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.  
Tugas Generator sekarang sedang dibuat untuk membuat koleksi gambar palsu, yang saat ini dilihat oleh Diskriminator. Diskriminator tidak dapat membedakan antara yang asli dan yang palsu. Untuk ilustrasi, lihat gambar berikut:



Gambar 8.1 Teori 1

2. Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatorenya.

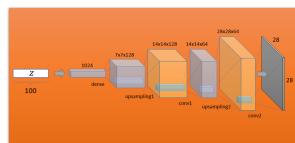
Diskriminator adalah CNN yang menerima input gambar yang dimiliki dan menghasilkan angka biner yang meminta input gambar, lalu menghasilkan gambar dari dataset asli atau menghasilkan gambar palsu. Untuk ilustrasi, lihat gambar berikut:



**Gambar 8.2** Teori 2

3. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur generator dibuat.

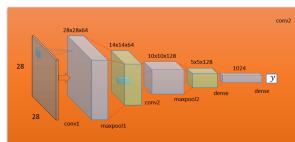
Aksitektur generator dibuat bisa dijelaskan pada gambar berikut :



**Gambar 8.3** Teori 3

4. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminator dibuat.

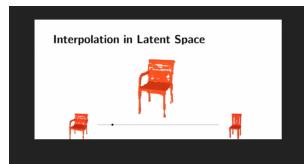
Aksitektur diskriminator dibuat bisa dijelaskan pada gambar berikut :



**Gambar 8.4** Teori 4

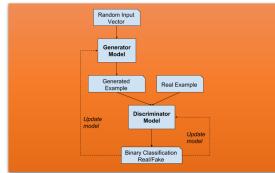
5. Jelaskan dengan ilustrasi gambar apa itu latent space.

Latent space dijelaskan pada gambar berikut :

**Gambar 8.5** Teori 5

6. Jelaskan dengan ilustrasi gambar apa itu adversarial play.

Adversarial play dijelaskan pada gambar berikut :

**Gambar 8.6** Teori 6

7. Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium.

Nash equilibrium adalah Teori permainan adalah studi tentang interaksi strategis antara agen rasional. Sederhananya itu berarti itu adalah studi interaksi ketika pihak-pihak yang terlibat mencoba dan melakukan yang terbaik dari sudut pandang mereka, detailnya dapat dijelaskan pada gambar berikut :

```

In [4]: runfile('D:/FOLDER KHUSUS NGANPUS/SEMESTER 8/src/tugosa3/tugosa3.py', wdir='D:/FOLDER KHUSUS NGANPUS/SEMESTER 8/src/tugosa3')
PERTAMA KEGAIATAN KE 8/AI PULL 8/src/174021/tugosa3
B1: matrix game with payoff matrices:
Row player:
[[3 0]
 [4 1]]
Column player:
[[3 4]
 [0 1]]

```

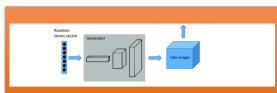
**Gambar 8.7** Teori 7

8. Sebutkan dan jelaskan contoh-contoh implementasi dari GAN.

Menurut saya implementasi 3DGAN yaitu pada MAPS dan juga IKEA. Karena pada maps dan juga ikea sudah menerapkan bentuk 3 dimensi yang bisa lebih menarik perhatikan pengguna.

9. Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer).

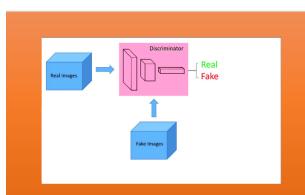
Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :



**Gambar 8.8** Teori 9

10. Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuah input layer, 3 buah hidden layer, dan satu output layer.

Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :



**Gambar 8.9** Teori 10

11. Jelaskan bagaimana kaitan output dan input antara generator dan diskriminatator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

Gambar dari Generator yang berhasil di deteksi oleh Diskriminatator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas untuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminatator, lalu, Diskriminatator tidak bisa membedakan fake dan real.

12. Jelaskan apa perbedaan antara Kullback-Leibler divergence (KL divergence)/relative entropy / Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to the average dalam mengukur kualitas dari model.

Perbedaan nya yaitu memiliki model dari rumus yang berbeda-beda sehingga mempengaruhi hasil train dan test

13. Jelaskan apa itu fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

Ukuran penting untuk menilai kualitas model. lalu kemudian akan melihat di keseimbangan Nash.

14. Jelaskan apa itu scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

The inception score adalah algoritma penilaian yang paling banyak digunakan untuk GAN. The Frechet Inception distance adalah Untuk mengatasi berbagai kekurangan Skor awal

15. Jelaskan kelebihan dan kekurangan GAN.

Kelebihan : GAN dapat memvisualisasikan bentuk model menjadi plot. Kemudian pada Kelemahan : model susah untuk implementasikan yang membuat data training menjadi lemah.

### **8.1.2 Praktek Program**

- ### 1. Soal 1

Konvolusi 3D. Konvolusi 3D menerapkan filter 3 dimensi ke kumpulan data dan filter 3 arah (x, y, z) untuk menghitung representasi fitur tingkat rendah. Bentuk outputnya adalah ruang volume 3 seperti kubus atau berbentuk kubus. 3D sangat membantu dalam pendekripsi peristiwa dalam video, gambar medis 3D, dll. Generative Adversarial Network adalah arsitektur jaringan saraf tiruan yang dimaksudkan untuk membuat atau membuat data yang benar-benar baru, dari nol hingga tidak ada sama sekali. Melihat target utama GAN adalah data gambar. Singkatnya, jaringan GAN berfungsi untuk memberikan gambar baru berdasarkan koleksi gambar yang telah ada sebelumnya selama proses training.

- ## 2. Soal 2

```
1 def build_generator():
2     """
3         Create a Generator Model with hyperparameters values defined
4         as follows
5     """
6
7     z_size = 200
8     gen_filters = [512, 256, 128, 64, 1]
9     gen_kernel_sizes = [4, 4, 4, 4, 4]
10    gen_strides = [1, 2, 2, 2, 2]
11    gen_input_shape = (1, 1, 1, z_size)
12    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
13    gen_convolutional_blocks = 5
14
15    input_layer = Input(shape=gen_input_shape)
16
17    # First 3D transpose convolution(or 3D deconvolution) block
18    a = Deconv3D(filters=gen_filters[0],
19                  kernel_size=gen_kernel_sizes[0],
20                  strides=gen_strides[0])(input_layer)
```

```

19     a = BatchNormalization()(a, training=True)
20     a = Activation(activation='relu')(a)
21
22     # Next 4 3D transpose convolution (or 3D deconvolution) blocks
23     for i in range(gen_convolutional_blocks - 1):
24         a = DeConv3D(filters=gen_filters[i + 1],
25                       kernel_size=gen_kernel_sizes[i + 1],
26                       strides=gen_strides[i + 1], padding='same')(a)
27
28         a = BatchNormalization()(a, training=True)
29         a = Activation(activation=gen_activations[i + 1])(a)
30
31     gen_model = Model(inputs=[input_layer], outputs=[a])
32     return gen_model

```

Kode di atas akan melakukan create generator ialah gloss, Bentuk jaringan Generator dapat dilihat berkebalikan dengan struktur jaringan saraf pada umumnya. Generator biasanya menerima input sebuah vektor z, yang kemudian mengubahnya menjadi sebuah output 3D atau 3 dimensi.

### 3. Soal 3

```

1 def build_discriminator():
2     """
3         Create a Discriminator Model using hyperparameters values
4         defined as follows
5     """
6
7     dis_input_shape = (64, 64, 64, 1)
8     dis_filters = [64, 128, 256, 512, 1]
9     dis_kernel_sizes = [4, 4, 4, 4, 4]
10    dis_strides = [2, 2, 2, 2, 1]
11    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
12    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
13    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
14                        'leaky_relu', 'sigmoid']
15    dis_convolutional_blocks = 5
16
17    dis_input_layer = Input(shape=dis_input_shape)
18
19    # The first 3D Convolutional block
20    a = Conv3D(filters=dis_filters[0],
21               kernel_size=dis_kernel_sizes[0],
22               strides=dis_strides[0],
23               padding=dis_paddings[0])(dis_input_layer)
24    # a = BatchNormalization()(a, training=True)
25    a = LeakyReLU(dis_alphas[0])(a)
26
27    # Next 4 3D Convolutional Blocks
28    for i in range(dis_convolutional_blocks - 1):
29        a = Conv3D(filters=dis_filters[i + 1],
30                   kernel_size=dis_kernel_sizes[i + 1],
31                   strides=dis_strides[i + 1],
32                   padding=dis_paddings[i + 1])(a)
33        a = BatchNormalization()(a, training=True)

```

```

33     if dis_activations[i + 1] == 'leaky_relu':
34         a = LeakyReLU(dis_alphas[i + 1])(a)
35     elif dis_activations[i + 1] == 'sigmoid':
36         a = Activation(activation='sigmoid')(a)
37
38     dis_model = Model(inputs=[dis_input_layer], outputs=[a])
39     return dis_model

```

Diskriminatator adalah d\_loss, Jaringan Discriminator merupakan jaringan klasifikasi biner yang menerima input gambar tiga dimensi dan mengeluarkan klasifikasi menyatakan input gambar adalah gambar asli dari dataset atau merupakan gambar buatan Generator. Diskriminatator dilatih dengan dataset yang diambil dari Generator, lalu di training untuk membedakan keduanya. Gambar dari Generator yang berhasil di deteksi oleh Diskriminatator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas untuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminatator, lalu, Diskriminatator tidak bisa membedakan fake dan real.

#### 4. Soal 4

Proses training 3D GAN yaitu dengan melakukan epoch sebanyak yang ditentukan.

#### 5. Soal 5

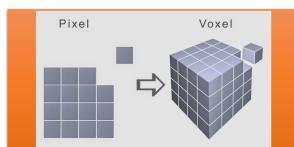
- Clone github
- Download dataset
- Buat folder baru logs dan results

#### 6. Soal 6

Dataset yang digunakan yaitu 3DShapeNets yang berisi model model bentuk benda dll, folder train berisi train dan folder test berisi data testing. dan semua data tersebut di simpan didalam folder volumetric\_data.

#### 7. Soal 7

Volume pixel atau voxel adalah titik dalam ruang tiga dimensi. Sebuah voxel mendefinisikan posisi dengan tiga koordinat dalam arah x, y, dan z. Sebuah voxel adalah unit dasar untuk mewakili gambar 3D. Untuk gambar nya ialah sebagai berikut :



**Gambar 8.10** Praktek Soal 7

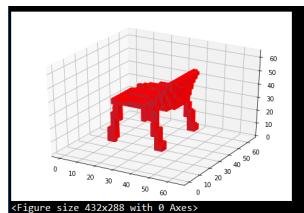
## 8. Soal 8

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sun May 10 18:10:55 2020
4
5 @author: FAHMI-PC
6
7 Ciee Copas :v
8 """
9
10 # In []
11
12 import scipy.io as io
13 voxels = io.loadmat("data/3DShapeNets/volumetric_data/chair/30/
14     test/chair_000000000_1.mat")['instance']
15 #%%%
16 import numpy as np
17 voxels = np.pad(voxels, (1, 1), 'constant', constant_values=(0,
18     0))
19 #%%%
20 import scipy.ndimage as nd
21 voxels = nd.zoom(voxels, (2, 2, 2), mode='constant', order=0)
22 #%%%
23 import matplotlib.pyplot as plt
24 from mpl_toolkits.mplot3d import Axes3D
25 fig = plt.figure()
26 ax = Axes3D(fig)
27 ax.voxels(voxels, edgecolor="red")
28 plt.show()
29 plt.savefig('data')

```

Kode di atas befungsi untuk visualisasidataset dalam tampilan plot. langkah-langkah seperti ini : import library, load data file .mat dan lakukan read memakai matplotlib, Hasilnya adalah sebagai berikut :



**Gambar 8.11** Hasil Soal 8

## 9. Soal 9

```

1 #%% soal9
2

```

```

3 def build_generator():
4     """
5         Create a Generator Model with hyperparameters values defined
6             as follows
7         """
8     z_size = 200
9     gen_filters = [512, 256, 128, 64, 1]
10    gen_kernel_sizes = [4, 4, 4, 4, 4]
11    gen_strides = [1, 2, 2, 2, 2]
12    gen_input_shape = (1, 1, 1, z_size)
13    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
14    gen_convolutional_blocks = 5
15
16    input_layer = Input(shape=gen_input_shape)
17
18    # First 3D transpose convolution(or 3D deconvolution) block
19    a = Deconv3D(filters=gen_filters[0],
20                  kernel_size=gen_kernel_sizes[0],
21                  strides=gen_strides[0])(input_layer)
22    a = BatchNormalization()(a, training=True)
23    a = Activation(activation='relu')(a)
24
25    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
26    for i in range(gen_convolutional_blocks - 1):
27        a = Deconv3D(filters=gen_filters[i + 1],
28                      kernel_size=gen_kernel_sizes[i + 1],
29                      strides=gen_strides[i + 1], padding='same')(a)
30        a = BatchNormalization()(a, training=True)
31        a = Activation(activation=gen_activations[i + 1])(a)
32
33    gen_model = Model(inputs=[input_layer], outputs=[a])
34    return gen_model

```

Kode di atas befungsi untuk membuat generator yaitu dengan ketentuan gen sebagai variabel dan membuat fungsi atau variabel genmodel lalu dilakukan return.

## 10. Soal 10

```

1 %% soal10
2
3
4 def build_discriminator():
5     """
6         Create a Discriminator Model using hyperparameters values
7             defined as follows
8         """
9
10    dis_input_shape = (64, 64, 64, 1)
11    dis_filters = [64, 128, 256, 512, 1]
12    dis_kernel_sizes = [4, 4, 4, 4, 4]
13    dis_strides = [2, 2, 2, 2, 1]
14    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
15    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]

```

```

15     dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
16                             'leaky_relu', 'sigmoid']
17     dis_convolutional_blocks = 5
18
19     dis_input_layer = Input(shape=dis_input_shape)
20
21     # The first 3D Convolutional block
22     a = Conv3D(filters=dis_filters[0],
23                 kernel_size=dis_kernel_sizes[0],
24                 strides=dis_strides[0],
25                 padding=dis_paddings[0])(dis_input_layer)
26     # a = BatchNormalization()(a, training=True)
27     a = LeakyReLU(dis_alphas[0])(a)
28
29     # Next 4 3D Convolutional Blocks
30     for i in range(dis_convolutional_blocks - 1):
31         a = Conv3D(filters=dis_filters[i + 1],
32                     kernel_size=dis_kernel_sizes[i + 1],
33                     strides=dis_strides[i + 1],
34                     padding=dis_paddings[i + 1])(a)
35         a = BatchNormalization()(a, training=True)
36         if dis_activations[i + 1] == 'leaky_relu':
37             a = LeakyReLU(dis_alphas[i + 1])(a)
38         elif dis_activations[i + 1] == 'sigmoid':
39             a = Activation(activation='sigmoid')(a)
40
41     dis_model = Model(inputs=[dis_input_layer], outputs=[a])
42     return dis_model

```

Kode di atas befungsi untuk membangun diskriminasi berfungsi untuk mendefinisikan seluruh gambar yang sudah di load generator sebagai gambar fake dan real.

## 11. Soal 11

Jika interpreter python menjalankan if name == main sebagai program utama, itu ialah menetapkan variabel name untuk memiliki nilai main. Jika file ini sedang di impor dari modul lain, name akan ditetapkan ke nama modul. Nama modul tersedia sebagai nilai untuk name variabel global.

## 12. Soal 12

```

1 #%% soal_12
2     object_name = "airplane"
3     data_dir = "data/3DShapeNets/volumetric_data/" \
4                 "{}/30/train/*.mat".format(object_name)
5     gen_learning_rate = 0.0025
6     dis_learning_rate = 10e-5
7     beta = 0.5
8     batch_size = 1
9     z_size = 200
10    epochs = 1
11    MODE = "train"

```

Kode di atas befungsi untuk melakukan load dataset dengan ketentuan data yang hanya dalam folder chair pada data train.

### 13. Soal 13

```

1 #%% soal_13
2 """
3 Create models
4 """
5 gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
6 dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)
7
8 discriminator = build_discriminator()
9 discriminator.compile(loss='binary_crossentropy', optimizer=
10   dis_optimizer)
11
12 generator = build_generator()
13 generator.compile(loss='binary_crossentropy', optimizer=
14   gen_optimizer)

```

Kode di atas menggunakan Adam sebagai algoritma pengoptimalan dan binary\_crossentropy sebagai kerugian loss.

### 14. Soal 14

```

1 #%% soal14
2 discriminator.trainable = False
3
4 input_layer = Input(shape=(1, 1, 1, z_size))
5 generated_volumes = generator(input_layer)
6 validity = discriminator(generated_volumes)
7 adversarial_model = Model(inputs=[input_layer], outputs=[validity])
8 adversarial_model.compile(loss='binary_crossentropy',
9   optimizer=gen_optimizer)

```

Kode di atas artinya ialah kita memasukkan random vector kedalam generator model lalu bagi 2 yaitu generated example dan real example, dan meneruskan ke diskriminator model sebagai real atau fake

### 15. Soal 15

```

1 #%% soal15
2 print("Loading data...")
3 volumes = get3DImages(data_dir=data_dir)
4 volumes = volumes [..., np.newaxis].astype(np.float)
5 print("Data loaded...")

```

Kode di atas befungsi untuk melakukan load data pada dataset.

### 16. Soal 16

```

1 #%% soal16
2     tensorboard = TensorBoard(log_dir="logs/{}".format(time.time()))
3     tensorboard.set_model(generator)
4     tensorboard.set_model(discriminator)

```

Kode di atas berfungsi untuk membuat tensorboard yang nantinya bisa diakses melalui localhost.

### 17. Soal 17

```

1 #%% soal17
2     labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1,
3         1))
3     labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1,
4         1))

```

Kode di atas befungsi untuk melakukan reshape agar shape yang dihasilkan tidak terlalu besar. Dengan membuat variabel real dan fake.

### 18. Soal 18

```

1 #%% soal18
2     if MODE == 'train':
3         for epoch in range(epochs):
4             print("Epoch:", epoch)
5
6             gen_losses = []
7             dis_losses = []

```

Kode di atas befungsi untuk melakukan training epoch, karena jika epoch semakin banyak maka kualitas training yang dihasilkan akan semakin baik.

### 19. Soal 19

```

1 #%% soal19
2             number_of_batches = int(volumes.shape[0] / batch_size
3         )
4             print("Number of batches:", number_of_batches)
5             for index in range(number_of_batches):
6                 print("Batch:", index + 1)

```

Batch adalah jumlah file yang akan di training.

### 20. Soal 20

```

1 #%% soal20
2             z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
3             volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]

```

Kode di atas befungsi untuk membuat gambar bersih dari noise dan juga menyesuaikan shape.

## 21. Soal 21

```

1 #%% soal21
2             # Next, generate volumes using the generate
3             network
4             gen_volumes = generator.predict_on_batch(z_sample
5         )

```

Kode di atas befungsi untuk membuat sample gambar palsu yang akan diteruskan ke diskriminator.

## 22. Soal 22

```

1 #%% soal22
2             discriminator.trainable = True
3             if index % 2 == 0:
4                 loss_real = discriminator.train_on_batch(
5                     volumes_batch, labels_real)
6                     loss_fake = discriminator.train_on_batch(
7                         gen_volumes, labels_fake)
8
9             d_loss = 0.5 * np.add(loss_real, loss_fake)
10            print("d_loss:{}".format(d_loss))
11

```

Kode di atas befungsi untuk membuat diskriminator bisa load gambar fake dan real dari generator, oleh karena itu ada generator loss dan diskriminator loss untuk melihat seberapa baik kualitas yang dihasilkan.

## 23. Soal 23

```

1 #%% soal23
2             discriminator.trainable = False
3             """
4             Train the generator network
5             """
6             z = np.random.normal(0, 0.33, size=[batch_size,
7                 1, 1, 1, z_size]).astype(np.float32)
8                 g_loss = adversarial_model.train_on_batch(z,
9                     labels_real)
10                print("g_loss:{}".format(g_loss))
11
12                gen_losses.append(g_loss)
13                dis_losses.append(d_loss)

```

Kode di atas befungsi untuk melakukan print gloss untuk generator dan juga dloss untuk diskriminator.

## 24. Soal 24

```

1 #%% soal24
2             # Every 10th mini-batch, generate volumes and
3             save them
4             if index % 10 == 0:
5                 z_sample2 = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
6                 generated_volumes = generator.predict(z_sample2, verbose=3)
7                 for i, generated_volume in enumerate(generated_volumes[:5]):
8                     voxels = np.squeeze(generated_volume)
9                     voxels[voxels < 0.5] = 0.
10                    voxels[voxels >= 0.5] = 1.
11                    saveFromVoxels(voxels, "results/img_{}-{}-{}".format(epoch, index, i))

```

Mengapa ada perulangan ? karena untuk melakukan perbandingan dari hasil yang sudah didapat.

## 25. Soal 25

```

1 #%% soal25
2             # Write losses to Tensorboard
3             write_log(tensorboard, 'g_loss', np.mean(gen_losses),
4             epoch)
5             write_log(tensorboard, 'd_loss', np.mean(dis_losses),
6             epoch)

```

TensorBoard adalah sebuah aplikasi web localhost untuk memeriksa dan menyelesaikan grafik dari hasil TensorFlow.

## 26. Soal 26

```

1 #%% soal26
2             generator.save_weights(os.path.join("models", "generator_weights.h5"))
3             discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))

```

File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah.

## 27. Soal 27

```

1 #%% soal27
2             if MODE == 'predict':
3                 # Create models
4                 generator = build_generator()
5                 discriminator = build_discriminator()

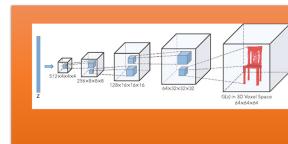
```

```

6      # Load model weights
7      generator.load_weights(os.path.join("models", "
8          generator_weights.h5"), True)
9      discriminator.load_weights(os.path.join("models", "
10         discriminator_weights.h5"), True)
11
12      # Generate 3D models
13      z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1,
14          z_size]).astype(np.float32)
15      generated_volumes = generator.predict(z_sample, verbose
16          =3)
17
18      for i, generated_volume in enumerate(generated_volumes
19          [:2]):
19          voxels = np.squeeze(generated_volume)
20          voxels[voxels < 0.5] = 0.
21          voxels[voxels >= 0.5] = 1.
22          saveFromVoxels(voxels, "results/gen_{}".format(i))

```

Ini adalah tahap akhir untuk melakukan testing dari model yang telah dibuat dan buat model dari yang sudah di create sebelumnya yaitu generator dan diskriminat. Untuk ilustrasi gambar sebagai berikut :



**Gambar 8.12** Praktek Soal 27

### 8.1.3 Penanganan Error

#### 1. ValueError

```

execfile(filename, namespace)
file = open('train.txt', 'r')
for line in file:
    exec(line, namespace)
exec(open(f'read', 'r').read(), namespace)
file = open('test.txt', 'r')
for line in file:
    exec(line, namespace)
    volumes = get10img(vols, dir_data_dir)
    all_imgs = np.array([np.reshape(img, (-1,)) for img in volumes])
    all_imgs = np.array([np.reshape(img, (-1,)) for img in volumes])
    file.write(str(np.argmax(all_imgs)))

```

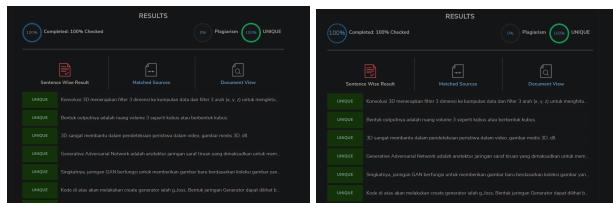
**Gambar 8.13** ValueError

#### 2. Cara Penanganan Error

- **ValueError**

Error tersebut karena disebabkan gagal load dataset karena salah penamaan direktori.

### 8.1.4 Bukti Tidak Plagiat



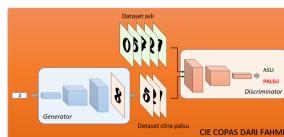
**Gambar 8.14** Bukti Tidak Melakukan Plagiat Chapter 8

## 8.2 1174017 - Muh. Rifky Prananda

### 8.2.1 Soal Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.

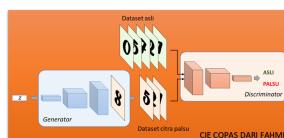
Tugas Generator sekarang sedang dibuat untuk membuat koleksi gambar palsu, yang saat ini dilihat oleh Diskriminator. Diskriminator tidak dapat membedakan antara yang asli dan yang palsu. Untuk ilustrasi, lihat gambar berikut:



**Gambar 8.15** Teori 1

2. Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatornya.

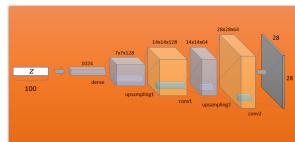
Diskriminator adalah CNN yang menerima input gambar yang dimiliki dan menghasilkan angka biner yang meminta input gambar, lalu menghasilkan gambar dari dataset asli atau menghasilkan gambar palsu. Untuk ilustrasi, lihat gambar berikut:



**Gambar 8.16** Teori 2

3. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur generator dibuat.

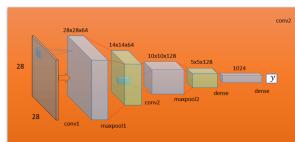
Aksitektur generator dibuat bisa dijelaskan pada gambar berikut :



**Gambar 8.17** Teori 3

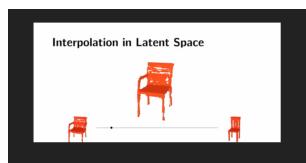
4. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminator dibuat.

Aksitektur diskriminator dibuat bisa dijelaskan pada gambar berikut :



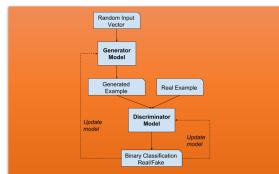
**Gambar 8.18** Teori 4

5. Jelaskan dengan ilustrasi gambar apa itu latent space.  
Latent space dijelaskan pada gambar berikut :



**Gambar 8.19** Teori 5

6. Jelaskan dengan ilustrasi gambar apa itu adversarial play.  
Adversarial play dijelaskan pada gambar berikut :

**Gambar 8.20** Teori 6

7. Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium.

Nash equilibrium adalah Teori permainan adalah studi tentang interaksi strategis antara agen rasional. Sederhananya itu berarti itu adalah studi interaksi ketika pihak-pihak yang terlibat mencoba dan melakukan yang terbaik dari sudut pandang mereka, detailnya dapat dijelaskan pada gambar berikut :

```

In [4]: runfile('D:/FOLDER KRSUS_NGAMPUS/SEMESTER 6/KONSEP/1174021/tugas8/tugas8.py', wdir='D:/FOLDER KRSUS_NGAMPUS/SEMESTER 6/KONSEP/1174021/tugas8')
Bi matrix game with payoff matrices:
Row player:
[[3 0]
 [4 1]]
Column player:
[[3 4]
 [0 1]]
  
```

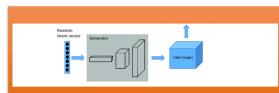
**Gambar 8.21** Teori 7

8. Sebutkan dan jelaskan contoh-contoh implementasi dari GAN.

Menurut saya implementasi 3DGAN yaitu pada MAPS dan juga IKEA. Karena pada maps dan juga ikea sudah menerapkan bentuk 3 dimensi yang bisa lebih menarik perhatian pengguna.

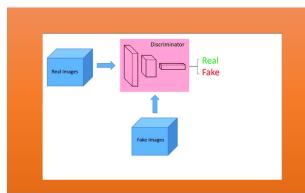
9. Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer).

Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :

**Gambar 8.22** Teori 9

10. Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuah input layer, 3 buah hidden layer, dan satu output layer.

Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :



**Gambar 8.23** Teori 10

- Jelaskan bagaimana kaitan output dan input antara generator dan diskriminator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

Gambar dari Generator yang berhasil di deteksi oleh Diskriminator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas untuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminator, lalu, Diskriminator tidak bisa membedakan fake dan real.

- Jelaskan apa perbedaan antara Kullback-Leibler divergence (KL divergence)/relative entropy Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to the average dalam mengukur kualitas dari model.

Perbedaan nya yaitu memiliki model dari rumus yang berbeda-beda sehingga mempengaruhi hasil train dan test

- Jelaskan apa itu fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

Ukuran penting untuk menilai kualitas model. lalu kemudian akan melihat di keseimbangan Nash.

- Jelaskan apa itu scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

The inception score adalah algoritma penilaian yang paling banyak digunakan untuk GAN. The Frechet Inception distance adalah Untuk mengatasi berbagai kekurangan Skor awal

- Jelaskan kelebihan dan kekurangan GAN.

Kelebihan : GAN dapat memvisualisasikan bentuk model menjadi plot. Kelemahan pada Kelemahan : model susah untuk implementasikan yang membuat data training menjadi lemah.

## 8.2.2 Praktek Program

### 1. Soal 1

Konvolusi 3D. Konvolusi 3D menerapkan filter 3 dimensi ke kumpulan data dan filter 3 arah (x, y, z) untuk menghitung representasi fitur tingkat rendah. Bentuk outputnya adalah ruang volume 3 seperti kubus atau berbentuk kubus. 3D sangat membantu dalam pendekripsi peristiwa dalam video, gambar medis 3D, dll. Generative Adversarial Network adalah arsitektur jaringan saraf tiruan yang dimaksudkan untuk membuat atau membuat data yang benar-benar baru, dari nol hingga tidak ada sama sekali. Melihat target utama GAN adalah data gambar. Singkatnya, jaringan GAN berfungsi untuk memberikan gambar baru berdasarkan koleksi gambar yang telah ada sebelumnya selama proses training.

### 2. Soal 2

```

1 def build_generator():
2     """
3         Create a Generator Model with hyperparameters values defined
4         as follows
5     """
6     z_size = 200
7     gen_filters = [512, 256, 128, 64, 1]
8     gen_kernel_sizes = [4, 4, 4, 4, 4]
9     gen_strides = [1, 2, 2, 2, 2]
10    gen_input_shape = (1, 1, 1, z_size)
11    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
12    gen_convolutional_blocks = 5
13
14    input_layer = Input(shape=gen_input_shape)
15
16    # First 3D transpose convolution(or 3D deconvolution) block
17    a = Deconv3D(filters=gen_filters[0],
18                  kernel_size=gen_kernel_sizes[0],
19                  strides=gen_strides[0])(input_layer)
20    a = BatchNormalization()(a, training=True)
21    a = Activation(activation='relu')(a)
22
23    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
24    for i in range(gen_convolutional_blocks - 1):
25        a = Deconv3D(filters=gen_filters[i + 1],
26                      kernel_size=gen_kernel_sizes[i + 1],
27                      strides=gen_strides[i + 1], padding='same')(a)
28        a = BatchNormalization()(a, training=True)
29        a = Activation(activation=gen_activations[i + 1])(a)
30
31    gen_model = Model(inputs=[input_layer], outputs=[a])
32    return gen_model

```

Kode di atas akan melakukan create generator ialah gloss, Bentuk jaringan Generator dapat dilihat berkebalikan dengan struktur jaringan saraf pada umum-

nya. Generator biasanya menerima input sebuah vektor z, yang kemudian mengubahnya menjadi sebuah output 3D atau 3 dimensi.

### 3. Soal 3

```

1 def build_discriminator():
2     """
3         Create a Discriminator Model using hyperparameters values
4             defined as follows
5         """
6
7     dis_input_shape = (64, 64, 64, 1)
8     dis_filters = [64, 128, 256, 512, 1]
9     dis_kernel_sizes = [4, 4, 4, 4, 4]
10    dis_strides = [2, 2, 2, 2, 1]
11    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
12    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
13    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
14                        'leaky_relu', 'sigmoid']
15    dis_convolutional_blocks = 5
16
17    dis_input_layer = Input(shape=dis_input_shape)
18
19    # The first 3D Convolutional block
20    a = Conv3D(filters=dis_filters[0],
21                kernel_size=dis_kernel_sizes[0],
22                strides=dis_strides[0],
23                padding=dis_paddings[0])(dis_input_layer)
24    # a = BatchNormalization()(a, training=True)
25    a = LeakyReLU(dis_alphas[0])(a)
26
27    # Next 4 3D Convolutional Blocks
28    for i in range(dis_convolutional_blocks - 1):
29        a = Conv3D(filters=dis_filters[i + 1],
30                    kernel_size=dis_kernel_sizes[i + 1],
31                    strides=dis_strides[i + 1],
32                    padding=dis_paddings[i + 1])(a)
33        a = BatchNormalization()(a, training=True)
34        if dis_activations[i + 1] == 'leaky_relu':
35            a = LeakyReLU(dis_alphas[i + 1])(a)
36        elif dis_activations[i + 1] == 'sigmoid':
37            a = Activation(activation='sigmoid')(a)
38
39    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
40    return dis_model

```

Diskriminatator adalah d\_loss, Jaringan Discriminator merupakan jaringan klasifikasi biner yang menerima input gambar tiga dimensi dan mengeluarkan klasifikasi menyatakan input gambar adalah gambar asli dari dataset atau merupakan gambar buatan Generator. Diskriminatator dilatih dengan dataset yang diambil dari Generator, lalu di training untuk membedakan keduanya. Gambar dari Generator yang berhasil di deteksi oleh Diskriminatator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas un-

tuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminator, lalu, Diskriminator tidak bisa membedakan fake dan real.

#### 4. Soal 4

Proses training 3D GAN yaitu dengan melakukan epoch sebanyak yang ditentukan.

#### 5. Soal 5

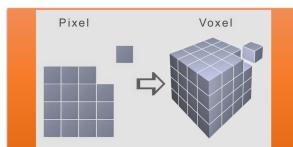
- Clone github
- Download dataset
- Buat folder baru logs dan results

#### 6. Soal 6

Dataset yang digunakan yaitu 3DShapeNets yang berisi model model bentuk benda dll, folder train berisi train dan folder test berisi data testing. dan semua data tersebut di simpan didalam folder volumetric\_data.

#### 7. Soal 7

Volume pixel atau voxel adalah titik dalam ruang tiga dimensi. Sebuah voxel mendefinisikan posisi dengan tiga koordinat dalam arah x, y, dan z. Sebuah voxel adalah unit dasar untuk mewakili gambar 3D. Untuk gambar nya ialah sebagai berikut :



**Gambar 8.24** Praktek Soal 7

#### 8. Soal 8

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue May 12 01:51:34 2020
4
5 @author: Rifky
6
7 """
8
9
10 # In []
11
12 import scipy.io as io
13 voxels = io.loadmat("data/3DShapeNets/volumetric_data/chair/30/
14     test/chair_000000000_1.mat")['instance']

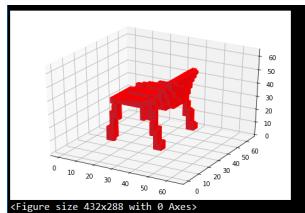
```

```

14 #%%
15 import numpy as np
16 voxels = np.pad(voxels, (1, 1), 'constant', constant_values=(0,
17 0))
18 #%%
19 import scipy.ndimage as nd
20 voxels = nd.zoom(voxels, (2, 2, 2), mode='constant', order=0)
21 #%%
22 import matplotlib.pyplot as plt
23 from mpl_toolkits.mplot3d import Axes3D
24 fig = plt.figure()
25 ax = Axes3D(fig)
26 ax.voxels(voxels, edgecolor="red")
27 plt.show()
28 plt.savefig('data')

```

Kode di atas befungsi untuk visualisasidataset dalam tampilan plot. langkah-langkah seperti ini : import library, load data file .mat dan lakukan read memakai matplotlib, Hasilnya adalah sebagai berikut :



**Gambar 8.25** Hasil Soal 8

## 9. Soal 9

```

1 #%% soal9
2
3 def build_generator():
4     """
5         Create a Generator Model with hyperparameters values defined
6         as follows
7     """
8     z_size = 200
9     gen_filters = [512, 256, 128, 64, 1]
10    gen_kernel_sizes = [4, 4, 4, 4, 4]
11    gen_strides = [1, 2, 2, 2, 2]
12    gen_input_shape = (1, 1, 1, z_size)
13    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
14    gen_convolutional_blocks = 5
15
16    input_layer = Input(shape=gen_input_shape)
17
18    # First 3D transpose convolution(or 3D deconvolution) block
19    a = Deconv3D(filters=gen_filters[0],

```

```

19         kernel_size=gen_kernel_sizes[0],
20         strides=gen_strides[0])(input_layer)
21     a = BatchNormalization()(a, training=True)
22     a = Activation(activation='relu')(a)
23
24     # Next 4 3D transpose convolution (or 3D deconvolution) blocks
25     for i in range(gen_convolutional_blocks - 1):
26         a = Deconv3D(filters=gen_filters[i + 1],
27                       kernel_size=gen_kernel_sizes[i + 1],
28                       strides=gen_strides[i + 1], padding='same')(a)
29
30         a = BatchNormalization()(a, training=True)
31         a = Activation(activation=gen_activations[i + 1])(a)
32
33     gen_model = Model(inputs=[input_layer], outputs=[a])
34     return gen_model

```

Kode di atas befungsi untuk membuat generator yaitu dengan ketentuan gen sebagai variabel dan membuat fungsi atau variabel genmodel lalu dilakukan return.

## 10. Soal 10

```

1 %% soal10
2
3
4 def build_discriminator():
5     """
6         Create a Discriminator Model using hyperparameters values
7         defined as follows
8     """
9
10    dis_input_shape = (64, 64, 64, 1)
11    dis_filters = [64, 128, 256, 512, 1]
12    dis_kernel_sizes = [4, 4, 4, 4, 4]
13    dis_strides = [2, 2, 2, 2, 1]
14    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
15    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
16    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
17                        'leaky_relu', 'sigmoid']
18    dis_convolutional_blocks = 5
19
20    dis_input_layer = Input(shape=dis_input_shape)
21
22    # The first 3D Convolutional block
23    a = Conv3D(filters=dis_filters[0],
24               kernel_size=dis_kernel_sizes[0],
25               strides=dis_strides[0],
26               padding=dis_paddings[0])(dis_input_layer)
27    # a = BatchNormalization()(a, training=True)
28    a = LeakyReLU(dis_alphas[0])(a)
29
30    # Next 4 3D Convolutional Blocks
31    for i in range(dis_convolutional_blocks - 1):
32        a = Conv3D(filters=dis_filters[i + 1],
33

```

```

32         kernel_size=dis_kernel_sizes[i + 1],
33         strides=dis_strides[i + 1],
34         padding=dis_paddings[i + 1])(a)
35     a = BatchNormalization()(a, training=True)
36     if dis_activations[i + 1] == 'leaky_relu':
37         a = LeakyReLU(dis_alphas[i + 1])(a)
38     elif dis_activations[i + 1] == 'sigmoid':
39         a = Activation(activation='sigmoid')(a)
40
41     dis_model = Model(inputs=[dis_input_layer], outputs=[a])
42
43     return dis_model

```

Kode di atas befungsi untuk membangun diskriminator berfungsi untuk mendefinisikan seluruh gambar yang sudah di load generator sebagai gambar fake dan real.

### 11. Soal 11

Jika interpreter python menjalankan if name == main sebagai program utama, itu ialah menetapkan variabel name untuk memiliki nilai main. Jika file ini sedang di impor dari modul lain, name akan ditetapkan ke nama modul. Nama modul tersedia sebagai nilai untuk name variabel global.

### 12. Soal 12

```

1 #%% soal 12
2 object_name = "airplane"
3 data_dir = "data/3DShapeNets/volumetric_data/" \
4             "{}/30/train/*/*.mat".format(object_name)
5 gen_learning_rate = 0.0025
6 dis_learning_rate = 10e-5
7 beta = 0.5
8 batch_size = 1
9 z_size = 200
10 epochs = 1
11 MODE = "train"

```

Kode di atas befungsi untuk melakukan load dataset dengan ketentuan data yang hanya dalam folder chair pada data train.

### 13. Soal 13

```

1 #%% soal 13
2 """
3 Create models
4 """
5 gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
6 dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)
7
8 discriminator = build_discriminator()
9 discriminator.compile(loss='binary_crossentropy', optimizer=
10 dis_optimizer)

```

```

11 generator = build_generator()
12 generator.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)

```

Kode di atas menggunakan Adam sebagai algoritma pengoptimalan dan binary\_crossentropy sebagai kerugian loss.

#### 14. Soal 14

```

1 #%% soal14
2 discriminator.trainable = False
3
4 input_layer = Input(shape=(1, 1, 1, z_size))
5 generated_volumes = generator(input_layer)
6 validity = discriminator(generated_volumes)
7 adversarial_model = Model(inputs=[input_layer], outputs=[validity])
8 adversarial_model.compile(loss='binary_crossentropy',
    optimizer=gen_optimizer)

```

Kode di atas artinya ialah kita memasukkan random vector kedalam generator model lalu membagi 2 yaitu generated example dan real example, dan meneruskan ke diskriminator model sebagai real atau fake

#### 15. Soal 15

```

1 #%% soal15
2 print("Loading data ...")
3 volumes = get3DImages(data_dir=data_dir)
4 volumes = volumes[..., np.newaxis].astype(np.float)
5 print("Data loaded ...")

```

Kode di atas befungsi untuk melakukan load data pada dataset.

#### 16. Soal 16

```

1 #%% soal16
2 tensorboard = TensorBoard(log_dir="logs/{}".format(time.time()))
3 tensorboard.set_model(generator)
4 tensorboard.set_model(discriminator)

```

Kode di atas berfungsi untuk membuat tensorboard yang nantinya bisa diakses melalui localhost.

#### 17. Soal 17

```

1 #%% soal17
2 labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1,
    1, 1))
3 labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1,
    1, 1))

```

Kode di atas befungsi untuk melakukan reshape agar shape yang dihasilkan tidak terlalu besar. Dengan membuat variabel real dan fake.

#### 18. Soal 18

```

1 #%% soal18
2     if MODE == 'train':
3         for epoch in range(epochs):
4             print("Epoch:", epoch)
5
6             gen_losses = []
7             dis_losses = []

```

Kode di atas befungsi untuk melakukan training epoch, karena jika epoch semakin banyak maka kualitas training yang dihasilkan akan semakin baik.

#### 19. Soal 19

```

1 #%% soal19
2             number_of_batches = int(volumes.shape[0] / batch_size
3         )
4             print("Number of batches:", number_of_batches)
5             for index in range(number_of_batches):
6                 print("Batch:", index + 1)

```

Batch adalah jumlah file yang akan di training.

#### 20. Soal 20

```

1 #%% soal20
2             z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
3             volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]

```

Kode di atas befungsi untuk untuk membuat gambar bersih dari noise dan juga menyesuaikan shape.

#### 21. Soal 21

```

1 #%% soal21
2             # Next, generate volumes using the generate
3             network
4             gen_volumes = generator.predict_on_batch(z_sample)

```

Kode di atas befungsi untuk membuat sample gambar palsu yang akan diteruskan ke diskriminatot.

#### 22. Soal 22

```

1 #%% soal22
2
3     discriminator.trainable = True
4     if index % 2 == 0:
5         loss_real = discriminator.train_on_batch(
6             volumes_batch, labels_real)
7         loss_fake = discriminator.train_on_batch(
8             gen_volumes, labels_fake)
9
10    d_loss = 0.5 * np.add(loss_real, loss_fake)
11    print("d_loss:{}".format(d_loss))
12
13 else:
14     d_loss = 0.0

```

Kode di atas befungsi untuk membuat diskriminator bisa load gambar fake dan real dari generator, oleh karena itu ada generator loss dan diskriminator loss untuk melihat seberapa baik kualitas yang dihasilkan.

### 23. Soal 23

```

1 #%% soal23
2
3     discriminator.trainable = False
4     """
5     Train the generator network
6     """
7     z = np.random.normal(0, 0.33, size=[batch_size,
8         1, 1, 1, z_size]).astype(np.float32)
9     g_loss = adversarial_model.train_on_batch(z,
10        labels_real)
11    print("g_loss:{}".format(g_loss))
12
13    gen_losses.append(g_loss)
14    dis_losses.append(d_loss)

```

Kode di atas befungsi untuk melakukan print gloss untuk generator dan juga dloss untuk diskriminator.

### 24. Soal 24

```

1 #%% soal24
2
3     # Every 10th mini-batch, generate volumes and
4     save them
5     if index % 10 == 0:
6         z_sample2 = np.random.normal(0, 0.33, size=[batch_size,
7             1, 1, 1, z_size]).astype(np.float32)
8         generated_volumes = generator.predict(
9             z_sample2, verbose=3)
10        for i, generated_volume in enumerate(
11            generated_volumes[:5]):
12            voxels = np.squeeze(generated_volume)
13            voxels[voxels < 0.5] = 0.
14            voxels[voxels >= 0.5] = 1.
15            saveFromVoxels(voxels, "results/img_{}_{}"
16            -{}).format(epoch, index, i))

```

Mengapa ada perulangan ? karena untuk melakukan perbandingan dari hasil yang sudah didapat.

## 25. Soal 25

```

1 #%% soal25
2         # Write losses to Tensorboard
3         write_log(tensorboard, 'g_loss', np.mean(gen_losses),
4 epoch)
5         write_log(tensorboard, 'd_loss', np.mean(dis_losses),
6 epoch)

```

TensorBoard adalah sebuah aplikasi web localhost untuk memeriksa dan menyelesaikan grafik dari hasil TensorFlow.

## 26. Soal 26

```

1 #%% soal26
2     generator.save_weights(os.path.join("models", "generator_weights.h5"))
3     discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))

```

File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah.

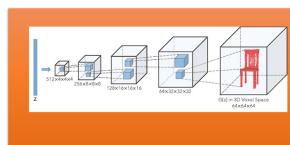
## 27. Soal 27

```

1 #%% soal27
2 if MODE == 'predict':
3     # Create models
4     generator = build_generator()
5     discriminator = build_discriminator()
6
7     # Load model weights
8     generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
9     discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)
10
11    # Generate 3D models
12    z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1,
13    1, z_size]).astype(np.float32)
14    generated_volumes = generator.predict(z_sample, verbose
15    =3)
16
17    for i, generated_volume in enumerate(generated_volumes
18    [:2]):
19        voxels = np.squeeze(generated_volume)

```

Ini adalah tahap akhir untuk melakukan testing dari model yang telah dibuat dan buat model dari yang sudah di create sebelumnya yaitu generator dan diskriminator. Untuk ilustrasi gambar sebagai berikut :



**Gambar 8.26** Praktek Soal 27

### 8.2.3 Penanganan Error

#### 1. ValueError

```

execfile(filename, namespace)
title = "ValueError", line = 1, in execfile
execfile(f.read(), filename, nspace)
title = "ValueError", line = 1, in execfile
values = pReLU(data_d,data_d)
title = "ValueError", line = 1, in execfile
values = pReLU(data_d,data_d)
title = "ValueError", line = 1, in pReLU
all_titles = np.random.choice(glob.glob('*.npy'), size=10)
title = "ValueError", line = 1, in numpy.random.choice
raise ValueError("a" cannot be empty unless no samples are taken)
  
```

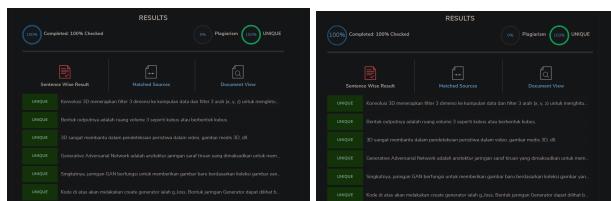
**Gambar 8.27** ValueError

#### 2. Cara Penanganan Error

- **ValueError**

Error tersebut karena disebabkan gagal load dataset karena salah penamaan direktori.

### 8.2.4 Bukti Tidak Plagiat



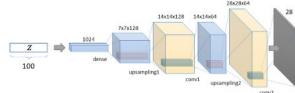
**Gambar 8.28** Bukti Tidak Melakukan Plagiat Chapter 8

## 8.3 1174008 - Arjun Yuda Firwanda

### 8.3.1 Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.

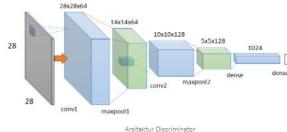
Generator merupakan generator yang biasanya menggunakan data yang ada untuk menghasilkan data baru, serta generatator sendiri bertujuan untuk menghasilkan data berupa video,gambar,audio dan teks.



**Gambar 8.29** Generator

2. Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminatot dengan Dosen sebagai Diskriminator.

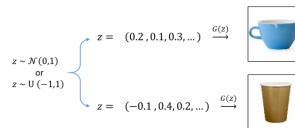
Diskriminatot merupakan untuk membedakan antara data nyata dan data yang dihasilkan oleh generator. pada jaringan diskriminatot sendiri mencoba memasukan data ke dalam kategori yang sudah ditentukan.



**Gambar 8.30** Diskriminatot

3. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur generator dibuat

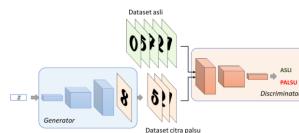
Dilihat berkebalikan dengan struktur jaringan saraf pada umumnya. Pada Generator dinyatakan menerima input vektor z dan kemudian mengubahnya menjadi gambar tiga dimensi.



**Gambar 8.31** Arsitektuk Generator

4. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminatot dibuat

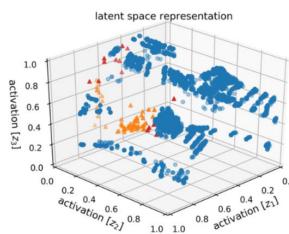
Diskriminatör merupakan jaringan klasifikasi biner yang menerima input gambar tiga dimensi dan mengeluarkan klasifikasi gambar asli. Diskriminatör dilatih dengan sekumpulan data, dan sekumpulan dataset, dan dilatih untuk bisa membedakan keduanya.



**Gambar 8.32** Arsitektur Diskriminatör

5. Jelaskan dengan ilustrasi gambar apa itu latent space.

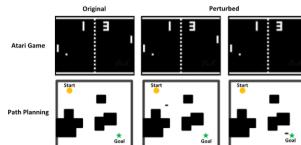
Latent space merupakan vektor angka yang dihasilkan secara acak.



**Gambar 8.33** Arsitektur Diskriminatör

6. Jelaskan dengan ilustrasi gambar apa itu adversarial play

Adversarial Play, berbagi data pelestarian privasi, dan vaksin rancangan. Saya menjelaskan bagaimana aspek konseptual kedua dari game keamanan menawarkan pemodelan alami paradigm untuk ini.



**Gambar 8.34** Adversarial Play

7. Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium

Nash Equilibrium sendiri menggambarkan keadaan tertentu dalam teori permainannya. Sehingga dalam permainan non kooperatif ini dimana setiap pemainnya memiliki strategi untuk hasil yang terbaik berdasarkan apa yang mereka harapkan apa yang dilakukan oleh pemain lainnya.

		Perusahaan B	
		Diferensiasi	Homogen
Perusahaan A	Diferensiasi	100	100
	Homogen	200	0
Perusahaan B	Diferensiasi	200	0
	Homogen	0	0

**Gambar 8.35** Nash equilibrium

### 8. Sebutkan dan jelaskan contoh-contoh implementasi dari GAN

Cara membuat GAN dengan citra beresolusi tinggi. Kita mulai dengan data sederhana yang jauh lebih mudah, yaitu dataset citra angka tulisan tangan MNIST. Membangun GAN dengan library Keras dan Tensorflow.

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from keras.models import Sequential, load_model
from keras.layers import Dense, Conv2D, Attent
from keras.layers import Activation, Dropout, Input
from keras.layers import Reshape, UpSampling2D
from keras.layers import MaxPooling2D, Activation
from keras.datasets import mnist
from keras.optimizers import SGD
from keras.utils import to_categorical
from PIL import Image
```

**Gambar 8.36** GAN

### 9. Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator neural network dengan sebuah input layer, tiga hidden layer dense layer, dan satu output layer reshape layer

Inputan seed 1x100. Generator akan mengubah menjadi sebuah gambar 28x28 yang menggunakan Convolutional Neural Network.

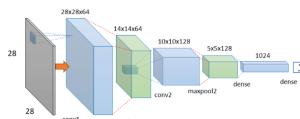
```
def generator_model():
    model = Sequential([
        Dense(1024, input_dim=100, activation='tanh'),
        Dense(128*7*7),
        Reshape((7, 7, 128)),
        UpSampling3D(size=(2, 2)),
        Conv2D(64, (5, 5), padding='same', activation='tanh'),
        UpSampling3D(size=(2, 2)),
        Conv2D(1, (5, 5), padding='same', activation='tanh')
    ])
    return model

generator_model().summary()
```

**Gambar 8.37** Hasil Arsitektuk Generator

### 10. Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuah input layer, 3 buah hidden layer, dan satu output layer.

Dikriminator merupakan CNN yang menerima input 28,28 dan menghasilkan angka biner, jika kelas 1 maka gambar asli, jika kelas 0 maka gambar palsu.



**Gambar 8.38** Hasil Arsitektur Generator

11. Jelaskan bagaimana kaitan output dan input antara generator dan diskriminatator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

Generator, Di sini kami tetapkan ukuran input seed adalah 1x100. Diskriminatator, yang menerima input 28,28 dan menghasilkan angka biner, jika kelas 1 maka gambar asli, jika kelas 0 maka gambar palsu.

12. Jelaskan apa perbedaan antara KullbackLeibler divergence KL divergence relative entropy, Jensen-Shannon JS divergence information radius iRaD total divergence to the average dalam mengukur kualitas dari model.

Kullback-Leibler divergence untuk menghitung skor yang mengukur divergensi satu distribusi probabilitas dari yang lain. Jensen-Shannon divergence metode untuk mengukur kesamaan antara dua distribusi probabilitas. Ia juga dikenal sebagai radius informasi IRad atau total divergence ke rata-rata.

13. Jelaskan apa itu fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

Fungsi obyektif digunakan untuk membuat jaringan gerator yang menghasilkan gambar yang mirip dengan gambar yang asli nya, serta meningkatkan kesamaan data yang dihasilkan oleh generator ke data aslinya.

$$\begin{aligned} \min_{\theta_G} \max_{\theta_D} v(\theta_G, \theta_D) = & \mathbf{E}_{x, y \sim p_{data}} [\log D(x, y)] \\ & + \mathbf{E}_{z \sim p_z(z), \hat{y} \sim p_{\hat{y}}} [\log (1 - D(G(z), \hat{y}))] \end{aligned} \quad (1)$$

**Gambar 8.39** Hasil Arsitektuk Generator

14. Jelaskan apa itu scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

The Inception Score memperkirakan kualitas koleksi gambar sintetis berdasarkan seberapa baik model klasifikasi gambar berkinerja terbaik Inception v3 mengklasifikasikannya sebagai salah satu dari 1.000 objek yang dikenal. Skor menggabungkan kepercayaan prediksi kelas bersyarat untuk setiap gambar sintetis kualitas dan integral dari probabilitas marginal dari kelas prediksi keragaman. The Frechet Inception Distance meringkas jarak antara vektor fitur Inception untuk gambar nyata dan yang dihasilkan dalam domain yang sama.

15. Jelaskan kelebihan dan kekurangan GAN

Kelebihan GAN yang pertama dapat dijadikan sebagai pembelajaran tanpa pengawasan, kemudian yang kedua gan menghasilkan data yang mirip dengan aslinya, dan selanjutnya yang ketiga sebagai belajar distribusi kepadatan data.

Kekurangannya sulit dilatih, tidak stabil, serta terdapat masalah pada mode tutup.

### 8.3.2 Praktek

1. Jelaskan apa itu 3D convolutions Konvolusi 3D menerapkan filter 3 dimensi ke dataset dan filter bergerak 3 arah x y z untuk menghitung representasi fitur level rendah. Bentuk output mereka adalah ruang volume 3 dimensi seperti kubus.
2. Jelaskan dengan kode program arsitektur dari generator networknya, beserta penjelasan input dan output dari generator network. Pada kode program ini, menyebutkan seed atau ketetapan ukuran input. Yakni 1x100 dan mengubah outputnya menjadi 28x28 yang menggunakan Convolution Neural Network.

```

1 def build_discriminator():
2     """
3         Create a Discriminator Model using hyperparameters values
4         defined as follows
5     """
6
7     dis_input_shape = (64, 64, 64, 1)
8     dis_filters = [64, 128, 256, 512, 1]
9     dis_kernel_sizes = [4, 4, 4, 4, 4]
10    dis_strides = [2, 2, 2, 2, 1]
11    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
12    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
13    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
14                        'leaky_relu', 'sigmoid']
15    dis_convolutional_blocks = 5
16
17    dis_input_layer = Input(shape=dis_input_shape)
18
19    # The first 3D Convolutional block
20    a = Conv3D(filters=dis_filters[0],
21                kernel_size=dis_kernel_sizes[0],
22                strides=dis_strides[0],
23                padding=dis_paddings[0])(dis_input_layer)
24    # a = BatchNormalization()(a, training=True)
25    a = LeakyReLU(dis_alphas[0])(a)
26
27    # Next 4 3D Convolutional Blocks
28    for i in range(dis_convolutional_blocks - 1):
29        a = Conv3D(filters=dis_filters[i + 1],
30                    kernel_size=dis_kernel_sizes[i + 1],
31                    strides=dis_strides[i + 1],
32                    padding=dis_paddings[i + 1])(a)
33        a = BatchNormalization()(a, training=True)
34        if dis_activations[i + 1] == 'leaky_relu':
35            a = LeakyReLU(dis_alphas[i + 1])(a)
36        elif dis_activations[i + 1] == 'sigmoid':
37            a = Activation('sigmoid')(a)
```

```

36         a = Activation(activation='sigmoid')(a)
37
38     dis_model = Model(inputs=[dis_input_layer], outputs=[a])
39     return dis_model

```

3. Jelaskan dengan kode program arsitektur dari diskriminator network, beserta penjelasan input dan outputnya. Pada program ini, input gambar berupa 28x28 dan menghasilkan angka biner yang menyatakan bahwa kelas 1 merupakan data asli, dan jika kelas 0 maka gambar palsu.

```

1 def build_generator():
2     """
3         Create a Generator Model with hyperparameters values defined
4         as follows
5     """
6     z_size = 200
7     gen_filters = [512, 256, 128, 64, 1]
8     gen_kernel_sizes = [4, 4, 4, 4, 4]
9     gen_strides = [1, 2, 2, 2, 2]
10    gen_input_shape = (1, 1, 1, z_size)
11    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
12    gen_convolutional_blocks = 5
13
14    input_layer = Input(shape=gen_input_shape)
15
16    # First 3D transpose convolution(or 3D deconvolution) block
17    a = Deconv3D(filters=gen_filters[0],
18                  kernel_size=gen_kernel_sizes[0],
19                  strides=gen_strides[0])(input_layer)
20    a = BatchNormalization()(a, training=True)
21    a = Activation(activation='relu')(a)
22
23    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
24    for i in range(gen_convolutional_blocks - 1):
25        a = Deconv3D(filters=gen_filters[i + 1],
26                      kernel_size=gen_kernel_sizes[i + 1],
27                      strides=gen_strides[i + 1], padding='same')(a)
28        a = BatchNormalization()(a, training=True)
29        a = Activation(activation=gen_activations[i + 1])(a)
30
31    gen_model = Model(inputs=[input_layer], outputs=[a])
32    return gen_model

```

4. Jelaskan proses training 3D-GANs Proses Training disini melatih data gambar menggunakan Generator Bangkitkan data citra palsu menggunakan Generator sejumlah dataset citra asli. Latih Discriminator untuk bisa membedakan dataset citra asli dari dataset citra palsu. Gunakan Discriminator yang sudah dilatih untuk melatih Generator agar bisa membangkitkan dataset citra palsu yang dinilai asli oleh Discriminator.
5. Jelaskan bagaimana melakukan settingan awal chapter 02 untuk memenuhi semua kebutuhan sebelum melanjutkan ke tahapan persiapan data. load dataset

file csv. menghasilkan label biner lulus gagal berdasarkan G1 G2 G3 nilai tes, masing-masing 0 20 poin ambang batas untuk kelulusan adalah jumlah 30. gunakan one-hot encodin pada kolom kategorikal. gunakan shuffle row. menggunakan decision tree. menggunakan visualisasi pohon.

6. Jelaskan tentang dataset yang digunakan, dari mulai tempat unduh, cara membuka dan melihat data. Sampai deskripsi dari isi dataset dengan detail penjelasan setiap folder yang membuat orang awam paham. Isi dari dataset 3DShapeNets ada 6 folder. Dan masing-masing folder terdapat isi yang berbeda. Folder 3D untuk gambar 3D Folder bp untuk mendeskripsikan dan meneruskan program. Folder generative untuk mengenerate data gambar. Folder util untuk input json. Folder volumetric data untuk mengetahui metrik volume data. Folder voxelization untuk menampilkan hasil gambar berupa plot.
7. Jelaskan apa itu voxel dengan ilustrasi dan bahasa paling awam Voxel diibaratkan seperti pixel di bidang tiga dimensi, memiliki panjang, lebar dan tinggi. Volume pixel atau voxel adalah titik dalam ruang tiga dimensi. Sebuah voxel mendefinisikan posisi dengan tiga koordinat dalam arah x, y, dan z. Sebuah voxel adalah unit dasar untuk mewakili gambar 3D
8. Visualisasikan dataset tersebut dalam tampilan visual plot, jelaskan cara melakukan visualisasinya import library, load data file .mat, lalu read memakai matplotlib.
9. buka le run.py jelaskan perbaris kode pada fungsi untuk membuat generator yaitu build generator. Membuat generator yaitu dengan ketentuan gen sebagai variabel dan membuat fungsi atau variabel gen model lalu dilakukan return
10. jelaskan juga fungsi untuk membangun diskriminator pada fungsi build discriminator. Membangun diskriminator berfungsi untuk mendefenisikan seluruh gambar yang sudah di load generator sebagai gambar fake dan real
11. jelaskan apa maksud dari kode program name main itu menetapkan beberapa variabel khusus seperti name itu mengeksekusi semua kode yang ditemukan dalam file. Jika interpreter python menjalankan if name main sebagai program utama, itu ialah menetapkan variabel name untuk memiliki nilai main. Jika file ini sedang diimpor dari modul lain, name akan ditetapkan ke nama modul. Nama modul tersedia sebagai nilai untuk name variabel global.

```
| if __name__ == '__main__':
```

```

    ...
    saveFromVoxels(voxels,
Loading data...
Data loaded...
Epoch: 0
Number of batches: 10
Batch: 1
d_loss:0.6931722164154053
g_loss:0.6931138038635254
Batch: 2
g_loss:0.6931138038635254
Batch: 3
d_loss:0.6931579113006592
g_loss:0.6931090354919434
Batch: 4
g_loss:0.6931090354919434
Batch: 5
d_loss:0.6931560039520264
g_loss:0.693108081817627

```

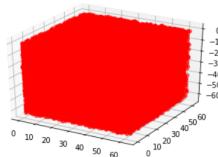
**Gambar 8.40** File

12. jelaskan secara detil perbaris dan per parameter apa arti dari kode program :  
Artinya adalah load dataset yang hanya dalam folder chair data train

```

1 """
2     Specify Hyperparameters
3 """
4 object_name = "chair"
5 data_dir = "3DShapeNets/volumetric_data/" \
6             "{}/train/*.mat".format(object_name)
7 gen_learning_rate = 0.0025
8 dis_learning_rate = 10e-5
9 beta = 0.5
10 batch_size = 1
11 z_size = 200
12 epochs = 10
13 MODE = "train"

```

**Gambar 8.41** Parameter

13. Jelaskan secara detil dari kode program pembuatan dan kompilasi arsitektur berikut Disini menggunakan Adam sebagai algoritma pengoptimalan dan binary crossentropy sebagai kerugian loss.

```

1 """
2     Create models
3 """
4 gen_optimizer = Adam(lr=gen_learning_rate , beta_1=beta)
5 dis_optimizer = Adam(lr=dis_learning_rate , beta_1=beta)
6
7 discriminator = build_discriminator()

```

```

8     discriminator.compile(loss='binary_crossentropy', optimizer=
9         dis_optimizer)
10
11     generator = build_generator()
12     generator.compile(loss='binary_crossentropy', optimizer=
13         gen_optimizer)

```



**Gambar 8.42** Kompilasi Arsitektur

14. Jelaskan secara detil kode program untuk membuat dan melakukan kompilasi model adversarial berikut. Ini artinya ialah kita memasukkan random vector kedalam generator model lalu membagi 2 yaitu generated example dan real example, dan meneruskan ke diskriminator model sebagai real atau fake.

```

1     input_layer = Input(shape=(1, 1, 1, z_size))
2     generated_volumes = generator(input_layer)
3     validity = discriminator(generated_volumes)
4     adversarial_model = Model(inputs=[input_layer], outputs=[validity])
5     adversarial_model.compile(loss='binary_crossentropy',
6         optimizer=gen_optimizer)

```

File	Name	Size	Date modified	Type
discriminator_weights.H5		42,200 KB	5/8/2020 5:27 PM	H5 File
generator_weights.H5		68,000 KB	5/8/2020 5:27 PM	H5 File

**Gambar 8.43** Kompilasi Model Adversarial

15. Jelaskan Ekstrak dan load data kursi dengan menggunakan fungsi getVoxels-Format dan get3DImages yang digunakan pada kode program berikut : Ini melakukan load data pada dataset.

```

1     print("Loading data...")
2     volumes = get3DImages(data_dir=data_dir)
3     volumes = volumes [..., np.newaxis].astype(np.float)
4     print("Data loaded...")

```

16. Jelaskan maksud dari kode program instansiasi TensorBoard yang menambahkan generator dan diskriminator pada program berikut: Ini berfungsi untuk membuat tensorboard yang nantinya bisa diakses melalui localhost

```

1     tensorboard = TensorBoard(log_dir="logs/{}".format(time.time()))
2         )
3     tensorboard.set_model(generator)
4     tensorboard.set_model(discriminator)

```

17. Jelaskan apa fungsi dari np reshape ones zeros pada kode program berikut dengan parameternya: Fungsi ini ialah untuk melakukan reshape agar shape yang dihasilkan tidak terlalu besar.

```

1     labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1,
2     1, 1))
2     labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1,
3     1, 1))

```

18. Jelaskan kenapa harus ada perulangan dalam meraih epoch. Dan jelaskan apa itu epoch terkait kode program berikut: Karena jika epoch semakin banyak maka kualitas training yang dihasilkan akan semakin baik

```

1     if MODE == 'train':
2         for epoch in range(epochs):
3             print("Epoch:", epoch)
4
5             gen_losses = []
6             dis_losses = []

```

19. Jelaskan apa itu batches dan kaitannya dengan kode program berikut, dan kenapa berada di dalam epoch: Batch adalah jumlah file yang akan di training

```

1             number_of_batches = int(volumes.shape[0] / batch_size
2         )
3             print("Number of batches:", number_of_batches)
4             for index in range(number_of_batches):
5                 print("Batch:", index + 1)

```

20. Berikut adalah kode program pengambilan gambar dan noise. Jelaskan apa fungsi np.random.normal serta astype, serta jelaskan apa arti parameter titik dua dan jelaskan isi dari z sample dan volumes batch: Ini adalah untuk membuat gambar bersih dari noise dan juga menyesuaikan shape

```

1     z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
2     volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]

```

21. Berikut adalah kode program generator gambar palsu. Jelaskan apa fungsi generator.predict on batch, serta jelaskan apa arti parameter z sample: Ialah membuat sample gambar palsu yang akan diteruskan ke diskriminator.

```

1             # Next, generate volumes using the generate
2             network
3             gen_volumes = generator.predict_on_batch(z_sample)

```

22. Berikut adalah kode program training diskriminator dengan gambar palsu dari generator dan gambar asil. Jelaskan apa maksudnya harus dilakukan training diskriminator secara demikian dan jelaskan apa isi loss fake dan loss real serta d loss dan fungsi train on batch. Diskriminator bisa load gambar fake dan real dari generator, oleh karena itu ada generator loss dan diskriminator loss untuk melihat seberapa baik kualitas yang dihasilkan.

```

1      """
2          Train the discriminator network
3      """
4          discriminator.trainable = True
5          if index % 2 == 0:
6              loss_real = discriminator.train_on_batch(
7                  volumes_batch, labels_real)
7                  loss_fake = discriminator.train_on_batch(
8                      gen_volumes, labels_fake)
9
9          d_loss = 0.5 * np.add(loss_real, loss_fake)
10         print("d_loss:{}".format(d_loss))
11
12     else:
13         d_loss = 0.0

```

23. Berikut adalah kode program training model adversarial yang terdapat generator dan diskriminator. Jelaskan apa bagaimana proses terbentuknya parameter z dan g loss: Dengan melakukan print g loss untuk generator dan juga d loss untuk diskriminator

```

1      z = np.random.normal(0, 0.33, size=[batch_size,
2          1, 1, 1, z_size]).astype(np.float32)
3          g_loss = adversarial_model.train_on_batch(z,
4              labels_real)
4          print("g_loss:{}".format(g_loss))
5
5          gen_losses.append(g_loss)
6          dis_losses.append(d_loss)

```

24. Berikut adalah kode program generate dan menyimpan gambar 3D setelah beberapa saat setiap epoch. Jelaskan mengapa ada perulangan dengan parameter tersebut, serta jelaskan arti setiap variabel beserta perlihatkan isinya dan artikan isinya: Mengapa ada perulangan ? karena untuk melakukan perbandingan dari hasil yang sudah didapat.

```

1      # Every 10th mini-batch, generate volumes and
2      save them
3          if index % 10 == 0:
4              z_sample2 = np.random.normal(0, 0.33, size=[
5                  batch_size, 1, 1, 1, z_size]).astype(np.float32)
6                  generated_volumes = generator.predict(
7                      z_sample2, verbose=3)
8                  for i, generated_volume in enumerate(
9                      generated_volumes[:5]):
10                     voxels = np.squeeze(generated_volume)
11                     voxels[voxels < 0.5] = 0.
12                     voxels[voxels >= 0.5] = 1.
13                     saveFromVoxels(voxels, "results/img_{}-{}"
14 -{}".format(epoch, index, i))

```

25. Berikut adalah kode program menyimpan average losses setiap epoch. Jelaskan apa itu tensorboard dan setiap parameter yang digunakan pada kode program

ini: TensorBoard adalah sebuah aplikasi web localhost untuk memeriksa dan menyelesaikan grafik dari hasil TensorFlow.

```

1      # Write losses to Tensorboard
2      write_log(tensorboard, 'g_loss', np.mean(gen_losses),
3 epoch)
        write_log(tensorboard, 'd_loss', np.mean(dis_losses),
epoch)
```

26. Berikut adalah kode program menyimpan model. Jelaskan apa itu format h5 dan penjelasan dari kode program berikut: File H5 adalah file data yang disimpan dalam Format Data Hirarki. Ini berisi array multidimensi data ilmiah.

```

1 """
2     Save models
3 """
4     generator.save_weights(os.path.join("models", "generator_weights.h5"))
5     discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))
```

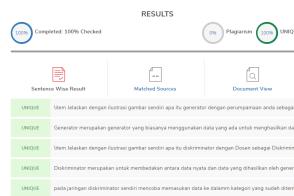
27. Berikut adalah kode program testing model. Jelaskan dengan ilustrasi gambar dari mulai meload hingga membuat gambar 3D dengan menggunakan z sample, bisakah parameter z sample tersebut diubah2: Ini adalah tahap akhir untuk melakukan testing dari model yang telah dibuat dan buat model dari yang sudah di create sebelumnya yaitu generator dan diskriminator.

```

1 if MODE == 'predict':
2     # Create models
3     generator = build_generator()
4     discriminator = build_discriminator()
5
6     # Load model weights
7     generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
8     discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)
9
10    # Generate 3D models
11    z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1,
12    1, z_size]).astype(np.float32)
13    generated_volumes = generator.predict(z_sample, verbose
14    =3)
15
16    for i, generated_volume in enumerate(generated_volumes
17    [:2]):
18        voxels = np.squeeze(generated_volume)
19        voxels[voxels < 0.5] = 0.
20        voxels[voxels >= 0.5] = 1.
21        saveFromVoxels(voxels, "results/gen_{}".format(i))
```

### 8.3.3 Penanganan Error

#### 8.3.4 Bukti Tidak Plagiat



**Gambar 8.44**   Bukti Tidak Melakukan Plagiat Chapter 8



## **BAB 9**

---

## **CHAPTER 9**

---



## **BAB 10**

---

## **CHAPTER 10**

---



## **BAB 11**

---

## **CHAPTER 11**

---



## **BAB 12**

---

## **CHAPTER 12**

---



## **BAB 13**

---

## **CHAPTER 13**

---



## **BAB 14**

---

## **CHAPTER 14**

---



## 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, [xxv](#)  
modern, [xxv](#)