

CERDAS MENGUASAI GIT

CERDAS MENGUASAI GIT

Dalam 24 Jam

Rolly M. Awangga
Informatics Research Center



Kreatif Industri Nusantara

Penulis:

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:

M. Yusril Helmi Setyawan

Penyunting:

Syafrial Fachrie Pane

Khaera Tunnisa

Diana Asri Wijayanti

Desain sampul dan Tata letak:

Deza Martha Akbar

Penerbit:

Kreatif Industri Nusantara

Redaksi:

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

Distributor:

Informatics Research Center

Jl. Sariasih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

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

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

CONTRIBUTORS

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

CONTENTS IN BRIEF

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	15
9 Chapter 9	31
10 Chapter 10	41
11 Chapter 11	43
12 Chapter 12	45
13 Chapter 13	47
14 Chapter 14	49

DAFTAR ISI

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

5	Chapter 5	9
6	Chapter 6	11
7	Chapter 7	13
8	Chapter 8	15
8.1	1174026 - Felix Lase	15
8.1.1	Soal Teori	15
8.1.2	Praktek Program	19
8.1.3	Penanganan Error	29
8.1.4	Bukti Tidak Plagiat	30
9	Chapter 9	31
9.1	1174026 Felix Setiawan Lase	31
9.1.1	Teori	31
9.1.2	Praktek	36
10	Chapter 10	41
11	Chapter 11	43
12	Chapter 12	45
13	Chapter 13	47
14	Chapter 14	49
	Daftar Pustaka	51
	Index	53

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

*Bandung, Jawa Barat
Februari, 2019*

ACKNOWLEDGMENTS

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

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

R. M. A.

ACRONYMS

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

GLOSSARY

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

SYMBOLS

- A Amplitude
- $\&$ Propositional logic symbol
- a Filter Coefficient

- \mathcal{B} Number of Beats

INTRODUCTION

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

Informatics Research Center
Bandung, Jawa Barat, Indonesia

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

Rumusnya cuman sebagai contoh aja biar keren[1].

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

BAB 1

CHAPTER 1

BAB 2

CHAPTER 2

BAB 3

CHAPTER 3

BAB 4

CHAPTER 4

BAB 5

CHAPTER 5

BAB 6

CHAPTER 6

BAB 7

CHAPTER 7

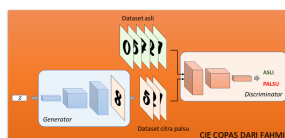
BAB 8

CHAPTER 8

8.1 1174026 - Felix Lase

8.1.1 Soal Teori

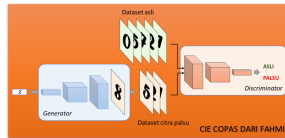
1. Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.
Tugas Generator sekarang sedang dibuat untuk membuat koleksi gambar palsu, yang saat ini dilihat oleh Diskriminator. Diskriminator tidak dapat membedakan antara yang asli dan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

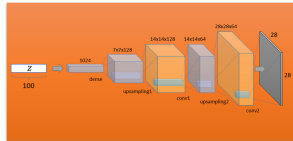
Diskriminator adalah CNN yang menerima input gambar yang dimiliki dan menghasilkan angka biner yang meminta input gambar, lalu menghasilkan gambar dari dataset asli atau menghasilkan gambar palsu. Untuk ilustrasi, lihat gambar berikut:



Gambar 8.2 Teori 2

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

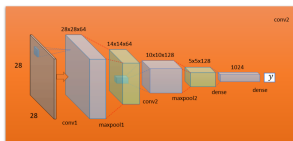
Aksitektur generator dibuat bisa dijelaskan pada gambar berikut :



Gambar 8.3 Teori 3

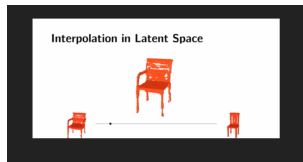
4. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminator dibuat.

Aksitektur diskriminator dibuat bisa dijelaskan pada gambar berikut :



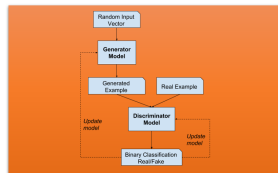
Gambar 8.4 Teori 4

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 KAMUS/MAMPU/SKIP/SEMESTER 6/
8/src/1174026/teori6.py', wdir='D:/FOLDER KAMUS
PERTEMUAN KE 8/AT PUL 8/src/1174026/teori6')
Bi matrix game with payoff matrices:

Row player:
[[3 0]
 [4 1]]

Column player:
[[3 4]
 [0 1]]
```

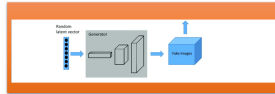
Gambar 8.7 Teori 7

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 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).

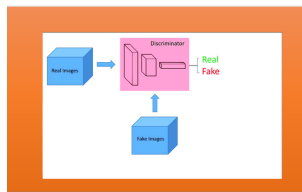
Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :



Gambar 8.8 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 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.

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.


```

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')(
27             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 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

```

Diskriminator 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. Diskriminator dilatih dengan dataset yang diambil dari Generator, lalu di training untuk membedakan keduanya. 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.

4. Soal 4

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

5. Soal 5

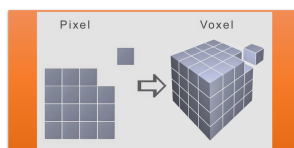
- Clone github
- Download dataset
- Buat folder baru logs dan results

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

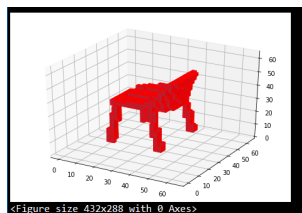
8. Soal 8

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Sun May 10 18:10:55 2020
4
5  @author: FAHMI-PC
6
7  Cieee 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
29 plt.show()
30 plt.savefig('data')

```

Kode di atas berfungsi untuk visualisasi dataset 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')(
30            a)
31        a = BatchNormalization()(a, training=True)
32        a = Activation(activation=gen_activations[i + 1])(a)
33
34    gen_model = Model(inputs=[input_layer], outputs=[a])
35    return gen_model

```

Kode di atas berfungsi 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 berfungsi 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 berfungsi 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=[
8  validity])
9  adversarial_model.compile(loss='binary_crossentropy',
10 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 berfungsi 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 berfungsi 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 berfungsi 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      print("Number of batches:", number_of_batches)
4      for index in range(number_of_batches):
5          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 berfungsi untuk membuat gambar bersih dari noise dan juga menyesuaikan shape.

21. Soal 21

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

Kode di atas berfungsi 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(
   volumes_batch, labels_real)
5             loss_fake = discriminator.train_on_batch(
   gen_volumes, labels_fake)
6
7             d_loss = 0.5 * np.add(loss_real, loss_fake)
8             print("d_loss:{0}".format(d_loss))
9
10            else:
11                d_loss = 0.0
```

Kode di atas berfungsi 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,
   1, 1, 1, z_size]).astype(np.float32)
7         g_loss = adversarial_model.train_on_batch(z,
   labels_real)
8         print("g_loss:{0}".format(g_loss))
9
10        gen_losses.append(g_loss)
11        dis_losses.append(d_loss)
```

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

24. Soal 24

```

1  ##### soal24
2          # Every 10th mini-batch, generate volumes and
   save them
3          if index % 10 == 0:
4              z_sample2 = np.random.normal(0, 0.33, size=[
   batch_size, 1, 1, 1, z_size]).astype(np.float32)
5              generated_volumes = generator.predict(
   z_sample2, verbose=3)
6              for i, generated_volume in enumerate(
   generated_volumes[:5]):
7                  voxels = np.squeeze(generated_volume)
8                  voxels[voxels < 0.5] = 0.
9                  voxels[voxels >= 0.5] = 1.
10                 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),
   epoch)
4          write_log(tensorboard, 'd_loss', np.mean(dis_losses),
   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", "
9     generator_weights.h5"), True)
10    discriminator.load_weights(os.path.join("models", "
11    discriminator_weights.h5"), True)
12
13    # Generate 3D models
14    z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1,
15    1, z_size]).astype(np.float32)
16    generated_volumes = generator.predict(z_sample, verbose
17    =3)
18
19    for i, generated_volume in enumerate(generated_volumes
20    [:2]):
21        voxels = np.squeeze(generated_volume)
22        voxels[voxels < 0.5] = 0.
23        voxels[voxels >= 0.5] = 1.
24        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 diskriminator. Untuk ilustrasi gambar sebagai berikut :



Gambar 8.12 Praktek Soal 27

8.1.3 Penanganan Error

1. ValueError

```

1  ValueError: cannot concatenate along axis 0
2
3  File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
4  ValueError: cannot concatenate along axis 0
5
6  File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
7  ValueError: cannot concatenate along axis 0
8
9  File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
10 ValueError: cannot concatenate along axis 0
11
12 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
13 ValueError: cannot concatenate along axis 0
14
15 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
16 ValueError: cannot concatenate along axis 0
17
18 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
19 ValueError: cannot concatenate along axis 0
20
21 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
22 ValueError: cannot concatenate along axis 0
23
24 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
25 ValueError: cannot concatenate along axis 0
26
27 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
28 ValueError: cannot concatenate along axis 0
29
30 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
31 ValueError: cannot concatenate along axis 0
32
33 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
34 ValueError: cannot concatenate along axis 0
35
36 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
37 ValueError: cannot concatenate along axis 0
38
39 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
40 ValueError: cannot concatenate along axis 0
41
42 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
43 ValueError: cannot concatenate along axis 0
44
45 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
46 ValueError: cannot concatenate along axis 0
47
48 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
49 ValueError: cannot concatenate along axis 0
50
51 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
52 ValueError: cannot concatenate along axis 0
53
54 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
55 ValueError: cannot concatenate along axis 0
56
57 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
58 ValueError: cannot concatenate along axis 0
59
60 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
61 ValueError: cannot concatenate along axis 0
62
63 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
64 ValueError: cannot concatenate along axis 0
65
66 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
67 ValueError: cannot concatenate along axis 0
68
69 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
70 ValueError: cannot concatenate along axis 0
71
72 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
73 ValueError: cannot concatenate along axis 0
74
75 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
76 ValueError: cannot concatenate along axis 0
77
78 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
79 ValueError: cannot concatenate along axis 0
80
81 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
82 ValueError: cannot concatenate along axis 0
83
84 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
85 ValueError: cannot concatenate along axis 0
86
87 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
88 ValueError: cannot concatenate along axis 0
89
90 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
91 ValueError: cannot concatenate along axis 0
92
93 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
94 ValueError: cannot concatenate along axis 0
95
96 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
97 ValueError: cannot concatenate along axis 0
98
99 File ~/anaconda3/lib/python3.7/site-packages/numpy/core/_add_newdocs.py:110:
100 ValueError: cannot concatenate along axis 0

```

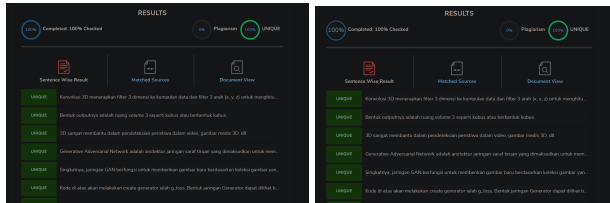
Gambar 8.13 ValueError

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

CHAPTER 9

9.1 1174026 Felix Setiawan Lase

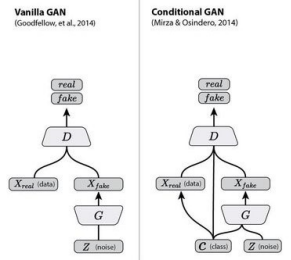
9.1.1 Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa perbedaan antara vanilla GAN dan cGAN.

Vanilla GANs biasanya tidak memiliki convolutional Neural Jaringan (CNNs) di jaringan mereka. Conditional GANs (cGANs) adalah perpanjangan dari model GAN. Mereka memungkinkan untuk generasi gambar yang memiliki kondisi tertentu atau atribut dan telah terbukti menjadi lebih baik dari Vanilla GANs sebagai hasilnya.

cGANs adalah jenis GAN yang dikondisikan pada beberapa informasi tambahan. informasi tambahan y ke Generator sebagai lapisan input tambahan. Dalam Vanilla GANs, tidak ada kontrol atas Kategori gambar yang dihasilkan. Ketika kita menambahkan kondisi y ke Generator, kita dapat menghasilkan gambar dari kategori tertentu, menggunakan y , yang mungkin jenis data, seperti label kelas atau data integer. Vanilla GANs bisa belajar hanya satu kategori dan sangat sulit untuk arsitek GANs untuk beberapa kategori. Sebuah cGAN,

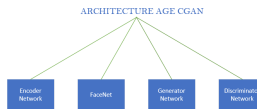
bagaimanapun, dapat digunakan untuk menghasilkan model multi-modal dengan kondisi yang berbeda untuk kategori yang berbeda.



Gambar 9.1 Ilustrasi Vanilla GAN dan cGAN

2. Jelaskan dengan ilustrasi gambar sendiri arsitektur dari Age-cGAN.

Arsitektur cGAN untuk penuaan wajah sedikit lebih rumit. AgecGan terdiri dari empat jaringan: Encoder, FaceNet, Jaringan Generator, dan jaringan diskriminator. Dengan Encoder, kita belajar pemetaan invers gambar wajah masukan dan kondisi usia dengan vektor laten. FaceNet adalah jaringan pengenalan wajah yang mempelajari perbedaan antara gambar input x dan gambar yang direkonstruksi. Kami memiliki jaringan Generator, yang mengambil representasi tersembunyi yang terdiri dari gambar wajah dan vektor kondisi dan menghasilkan gambar. Jaringan diskriminator adalah untuk mendiskriminasi antara gambar nyata dan gambar palsu. Masalah dengan cGANs adalah bahwa mereka tidak dapat mempelajari tugas pemetaan terbalik masukan gambar x dengan atribut y ke vektor laten z . Solusi untuk masalah ini adalah dengan menggunakan jaringan Encoder. Kita dapat melatih jaringan encoder untuk memperkirakan pemetaan terbalik dari input Images x .

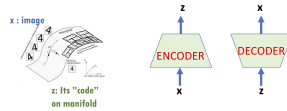


Gambar 9.2 Ilustrasi Arsitektur cGAN

3. Jelaskan dengan ilustrasi gambar sendiri arsitektur encoder network dari AgecGAN.

Tujuan utama dari jaringan Encoder adalah untuk menghasilkan vektor laten dari gambar yang disediakan. Pada dasarnya, dibutuhkan gambar dimensi (64,

64, 3) dan mengubahnya menjadi vektor 100-dimensi. Jaringan Encoder adalah jaringan syaraf convolutional yang dalam. Jaringan berisi empat convolutional blok dan dua lapisan padat. Setiap blok convolutional berisi lapisan convolutional, lapisan normalisasi batch, dan fungsi aktivasi. Di setiap blok convolutional, setiap lapisan convolutional diikuti oleh lapisan normalisasi batch, kecuali lapisan convolutional pertama.



Gambar 9.3 Ilustrasi Network Encoder

4. Jelaskan dengan ilustrasi gambar sendiri arsitektur generator network dari AgecGAN.

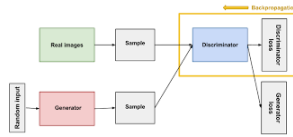
Tujuan utama dari generator adalah untuk menghasilkan gambar dari dimensi (64, 64, 3). Dibutuhkan vektor laten 100 dimensi dan beberapa informasi tambahan, y , dan mencoba untuk menghasilkan gambar yang realistis. Jaringan Generator adalah jaringan neural yang mendalam convolutional juga. Hal ini terdiri dari lapisan padat, upsampling, dan convolutional. Dibutuhkan dua nilai input: vektor kebisingan dan nilai pengkondisian. Nilai pengkondisian adalah informasi tambahan yang diberikan ke jaringan. Untuk Age-cGAN, ini akan menjadi usia.



Gambar 9.4 Ilustrasi Network Generator

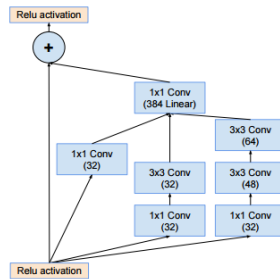
Jelaskan dengan ilustrasi gambar sendiri arsitektur discriminator network dari Age-cGAN.

Tujuan utama dari jaringan diskriminator adalah untuk mengidentifikasi apakah gambar yang disediakan adalah palsu atau nyata. Hal ini dilakukan dengan melewati gambar melalui serangkaian lapisan sampling bawah dan beberapa lapisan klasifikasi. Dengan kata lain, ini memprediksi Apakah gambar itu nyata atau palsu. Seperti jaringan lain, Jaringan diskriminator lain dalam jaringan convolutional. Ini berisi beberapa blok convolutional. Setiap blok convolutional berisi lapisan convolutional, lapisan normalisasi batch, dan fungsi aktivasi, selain blok convolutional pertama, yang tidak memiliki lapisan normalisasi batch.



Gambar 9.5 Ilustrasi Discriminator Network

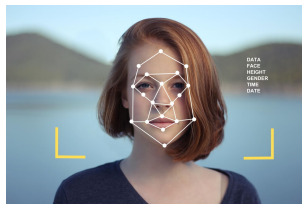
5. Jelaskan dengan ilustrasi gambar apa itu pretrained Inception-ResNet-2 Model. pre-trained Inception-ResNet-2 network, sekali disediakan dengan gambar, mengembalikan yang sesuai embedding. Tertanam yang diekstrak untuk gambar asli dan gambar direkonstruksi dapat dihitung dengan menghitung jarak Euclidean dari yang tertanam.



Gambar 9.6 Ilustrasi Inception-ResNet-2 Model.

6. Jelaskan dengan ilustrasi gambar sendiri arsitektur Face recognition network Age-cGAN.

Tujuan utama dari jaringan pengenalan wajah adalah untuk mengenali identitas seseorang dalam gambar yang diberikan.



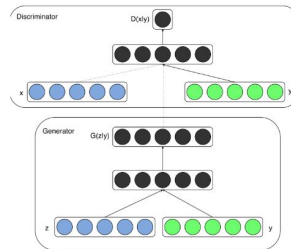
Gambar 9.7 Ilustrasi Face recognition network Age-cGAN.

7. . Sebutkan dan jelaskan serta di sertai contoh-contoh tahapan dari Age-cGAN

Age-cGAN memiliki beberapa tahapan pelatihan. Seperti disebutkan di bagian

sebelumnya, Age-cGAN memiliki empat jaringan, yang dilatih dalam tiga tahap. Pelatihan AgecGAN terdiri dari tiga tahap:

- pelatihan GAN bersyarat: pada tahap ini, kita melatih jaringan Generator dan jaringan diskriminator.
- awal pendekatan vektor laten: pada tahap ini, kami melatih jaringan Encoder.
- optimasi vektor laten: pada tahap ini, kami mengoptimalkan kedua encoder dan jaringan generator.



Gambar 9.8 Age-cGAN.

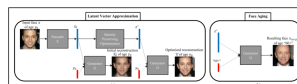
8. Berikan contoh perhitungan fungsi training objektif

$$L(\theta) = - \sum_{\{\mathbf{x}, \mathbf{y}\} \in \mathbf{D}} \log p(\mathbf{y} | \mathbf{x}, \theta)$$

Gambar 9.9 training objektif

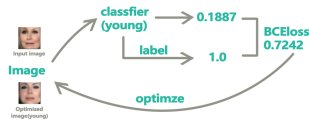
9. Berikan contoh dengan ilustrasi penjelasan dari Initial latent vector approximation

Perkiraan vektor laten awal adalah metode untuk memperkirakan vektor laten untuk mengoptimalkan rekonstruksi gambar wajah. Untuk memperkirakan vektor laten, kami memiliki jaringan Encoder. Kami melatih jaringan Encoder pada gambar yang dihasilkan dan gambar nyata. Setelah dilatih, Jaringan Encoder akan mulai menghasilkan vektor laten dari Distribusi. Tujuan pelatihan fungsi untuk pelatihan jaringan Encoder adalah kehilangan jarak Euclidean.



Gambar 9.10 Ilustrasi Initial latent vector approximation

10. Berikan contoh perhitungan latent vector optimization



Gambar 9.11 Contoh Perhitungan Latent vector optimization

9.1.2 Praktek

1. Jelaskan bagaimana cara ekstrak le dataset Age-cGAN menggunakan google colab. Menggunakan Google Colab, dimana membuat notebooks baru, kemudian membuat ekstraksi file dari link dataset.

```

1 # In[1. Ekstrak File]:
2 import tarfile
3 tf = tarfile.open("/content/drive/My Drive/Colab Notebooks/
4   wiki_crop.tar")
5 tf.extractall(path="/content/drive/My Drive/Colab Notebooks")

```

2. Jelaskan bagaimana kode program bekerja untuk melakukan load terhadap dataset yang sudah di ekstrak, termasuk bagaimana penjelasan kode program perhitungan usia. Dibawah ini merupakan code untuk melakukan fungsi perhitungan usia.

```

1 # In[2. Load Data]:
2 def load_data(wiki_dir, dataset='wiki'):
3     # Load the wiki.mat file
4     meta = loadmat(os.path.join(wiki_dir, "{}.mat".format(dataset)
5     ))
6
7     # Load the list of all files
8     full_path = meta[dataset][0, 0]["full_path"][0]
9
10    # List of Matlab serial date numbers
11    dob = meta[dataset][0, 0]["dob"][0]
12
13    # List of years when photo was taken
14    photo_taken = meta[dataset][0, 0]["photo_taken"][0] # year
15
16    # Calculate age for all dobs
17    age = [calculate_age(photo_taken[i], dob[i]) for i in range(
18    len(dob))]
19
20    # Create a list of tuples containing a pair of an image path
21    and age
22    images = []
23    age_list = []
24    for index, image_path in enumerate(full_path):
25        images.append(image_path[0])

```

```

23     age_list.append(age[index])
24
25     # Return a list of all images and respective age
26     return images, age_list

```

3. Jelaskan bagaimana kode program The Encoder Network bekerja dijelaskan dengan bahasa awam dengan ilustrasi sederhana. Proses Encoder berfungsi untuk mempelajari pemetaan terbalik dari gambar wajah dan kondisi usia dengan vector latent Z .

```

1  # In[3. Encoder Bekerja]:
2  def build_encoder():
3      """
4      Encoder Network
5      """
6      input_layer = Input(shape=(64, 64, 3))
7
8      # 1st Convolutional Block
9      enc = Conv2D(filters=32, kernel_size=5, strides=2, padding='
10     same')(input_layer)
11     # enc = BatchNormalization()(enc)
12     enc = LeakyReLU(alpha=0.2)(enc)
13
14     # 2nd Convolutional Block
15     enc = Conv2D(filters=64, kernel_size=5, strides=2, padding='
16     same')(enc)
17     enc = BatchNormalization()(enc)
18     enc = LeakyReLU(alpha=0.2)(enc)
19
20     # 3rd Convolutional Block
21     enc = Conv2D(filters=128, kernel_size=5, strides=2, padding='
22     same')(enc)
23     enc = BatchNormalization()(enc)
24     enc = LeakyReLU(alpha=0.2)(enc)
25
26     # 4th Convolutional Block
27     enc = Conv2D(filters=256, kernel_size=5, strides=2, padding='
28     same')(enc)
29     enc = BatchNormalization()(enc)
30     enc = LeakyReLU(alpha=0.2)(enc)
31
32     # Flatten layer
33     enc = Flatten()(enc)
34
35     # 1st Fully Connected Layer
36     enc = Dense(4096)(enc)
37     enc = BatchNormalization()(enc)
38     enc = LeakyReLU(alpha=0.2)(enc)
39
40     # Second Fully Connected Layer
41     enc = Dense(100)(enc)
42
43     # Create a model
44     model = Model(inputs=[input_layer], outputs=[enc])
45     return model

```

4. Jelaskan bagaimana kode program The Generator Network bekerja dijelaskan dengan bahasa awam dengan ilustrasi sederhana. Proses Generator agar bekerja dengan baik dibutuhkan representasi dari gambar wajah dan vector kondisi sebagai inputan yang menghasilkan sebuah gambar.

```

1 # In[4. Generator Network Bekerja]:
2 def build_generator():
3     """
4     Create a Generator Model with hyperparameters values defined
5     as follows
6     """
7     latent_dims = 100
8     num_classes = 6
9
10    input_z_noise = Input(shape=(latent_dims,))
11    input_label = Input(shape=(num_classes,))
12
13    x = concatenate([input_z_noise, input_label])
14
15    x = Dense(2048, input_dim=latent_dims + num_classes)(x)
16    x = LeakyReLU(alpha=0.2)(x)
17    x = Dropout(0.2)(x)
18
19    x = Dense(256 * 8 * 8)(x)
20    x = BatchNormalization()(x)
21    x = LeakyReLU(alpha=0.2)(x)
22    x = Dropout(0.2)(x)
23
24    x = Reshape((8, 8, 256))(x)
25
26    x = UpSampling2D(size=(2, 2))(x)
27    x = Conv2D(filters=128, kernel_size=5, padding='same')(x)
28    x = BatchNormalization(momentum=0.8)(x)
29    x = LeakyReLU(alpha=0.2)(x)
30
31    x = UpSampling2D(size=(2, 2))(x)

```

5. Jelaskan bagaimana kode program The Discriminator Network bekerja dijelaskan dengan bahasa awam dengan ilustrasi sederhana. Proses Discriminator untuk membedakan antara gambar asli dan gambar palsu.

```

1 # In[5. Discriminator Network Bekerja]:
2 def build_discriminator():
3     """
4     Create a Discriminator Model with hyperparameters values
5     defined as follows
6     """
7     input_shape = (64, 64, 3)
8     label_shape = (6,)
9     image_input = Input(shape=input_shape)
10    label_input = Input(shape=label_shape)
11
12    x = Conv2D(64, kernel_size=3, strides=2, padding='same')(
13        image_input)
14    x = LeakyReLU(alpha=0.2)(x)

```

```

13
14     label_input1 = Lambda(expand_label_input)(label_input)
15     x = concatenate([x, label_input1], axis=3)
16
17     x = Conv2D(128, kernel_size=3, strides=2, padding='same')(x)
18     x = BatchNormalization()(x)
19     x = LeakyReLU(alpha=0.2)(x)
20
21     x = Conv2D(256, kernel_size=3, strides=2, padding='same')(x)
22     x = BatchNormalization()(x)
23     x = LeakyReLU(alpha=0.2)(x)
24
25     x = Conv2D(512, kernel_size=3, strides=2, padding='same')(x)
26     x = BatchNormalization()(x)
27     x = LeakyReLU(alpha=0.2)(x)
28
29     x = Flatten()(x)
30     x = Dense(1, activation='sigmoid')(x)
31
32     model = Model(inputs=[image_input, label_input], outputs=[x])
33     return model

```

6. Jelaskan bagaimana kode program Training cGAN bekerja dijelaskan dengan bahasa awam dengan ilustrasi sederhana. Proses Training cGAN ini dengan load file .mat pada dataset lalu epoch sebanuak 500 kali.

```

1 # In[6. Training cGAN]:
2     if __name__ == '__main__':
3         # Define hyperparameters
4         data_dir = "data"
5         wiki_dir = os.path.join(data_dir, "wiki_crop1")
6         epochs = 500
7         batch_size = 2
8         image_shape = (64, 64, 3)
9         z_shape = 100
10        TRAIN_GAN = True
11        TRAIN_ENCODER = False
12        TRAIN_GAN_WITH_FR = False
13        fr_image_shape = (192, 192, 3)
14
15        # Define optimizers
16        dis_optimizer = Adam(lr=0.0002, beta_1=0.5, beta_2=0.999,
17                               epsilon=10e-8)
18        gen_optimizer = Adam(lr=0.0002, beta_1=0.5, beta_2=0.999,
19                               epsilon=10e-8)
20        adversarial_optimizer = Adam(lr=0.0002, beta_1=0.5, beta_2=
21                                     0.999, epsilon=10e-8)

```

7. Jelaskan bagaimana kode program Initial dan latent vector approximation bekerja dijelaskan dengan bahasa awam dengan ilustrasi sederhana. Initial dan Latent Vector Approximation bekerja melakukan prediksi epoch yang telah di buat sebanyak 500 kali, dan nanti hasilnya ada di folder result.

```

1 # In[7. Laten Vector]:
2

```



```

3     Train encoder
4     """
5
6     if TRAIN_ENCODER:
7         # Build and compile encoder
8         encoder = build_encoder()
9         encoder.compile(loss=euclidean_distance_loss, optimizer='
adam')
10
11        # Load the generator network's weights
12        try:
13            generator.load_weights("generator.h5")
14        except Exception as e:
15            print("Error:", e)
16
17        z_i = np.random.normal(0, 1, size=(5000, z_shape))
18
19        y = np.random.randint(low=0, high=6, size=(5000,), dtype=
np.int64)
20        num_classes = len(set(y))
21        y = np.reshape(np.array(y), [len(y), 1])
22        y = to_categorical(y, num_classes=num_classes)
23
24        for epoch in range(epochs):
25            print("Epoch:", epoch)
26
27            encoder.losses = []
28
29            number_of_batches = int(z_i.shape[0] / batch_size)
30            print("Number of batches:", number_of_batches)
31            for index in range(number_of_batches):
32                print("Batch:", index + 1)
33
34                z_batch = z_i[index * batch_size:(index + 1) *
batch_size]
35                y_batch = y[index * batch_size:(index + 1) *
batch_size]
36
37                generated_images = generator.predict_on_batch([
z_batch, y_batch])
38
39                # Train the encoder model
40                encoder_loss = encoder.train_on_batch(
generated_images, z_batch)
41                print("Encoder loss:", encoder_loss)
42
43                encoder.losses.append(encoder_loss)
44
45                # Write the encoder loss to Tensorboard
46                write_log(tensorboard, "encoder_loss", np.mean(
encoder.losses), epoch)
47
48                # Save the encoder model
49                encoder.save_weights("encoder.h5")

```

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