

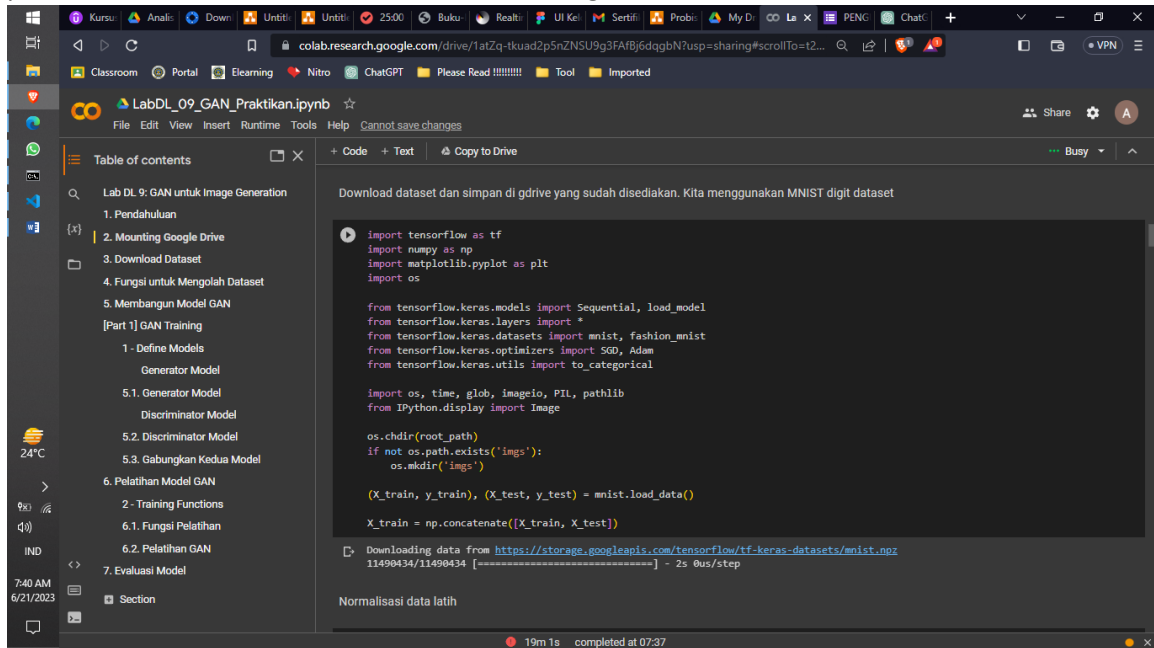
## LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

<b>Nama :</b> Agis Satria Mandala <b>NIM :</b> 2000018075	<b>Asisten:</b> <b>Paraf Asisten:</b>	<b>Tanggal:</b> <b>Nilai:</b>
--	--	----------------------------------

- 1.) Jelaskan bagaimana model GAN dapat membuat citra baru dari citra yang sudah dipelajarinya!  
GAN menggunakan generator untuk menghasilkan citra baru dari vektor latent acak, sementara discriminator membedakan citra asli dengan citra palsu yang dihasilkan oleh generator. Melalui proses pelatihan yang berulang, generator dan discriminator saling berlawanan, hingga generator dapat menghasilkan citra-citra baru yang sangat mirip dengan citra asli dari dataset pelatihan.
- 2.) Jelaskan arsitektur model generator dan discriminator dari model GAN!  
Generator dalam GAN menggunakan lapisan-lapisan konvolusi transpos untuk memperbesar citra dari vector latent, sementara discriminator menggunakan lapisan-lapisan konvolusi untuk mengklasifikasi citra asli dan palsu.

## LANGKAH PRAKTIKUM

1. Jalankan tahap **3. Download Dataset**. Tahapan ini akan mengimport library yang diperlukan untuk praktikum ini dan mendownload dataset MIST digit.



```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
import os

from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import *
from tensorflow.keras.datasets import mnist, fashion_mnist
from tensorflow.keras.optimizers import SGD, Adam
from tensorflow.keras.utils import to_categorical

import os, time, glob, imageio, PIL, pathlib
from IPython.display import Image

os.chdir(root_path)
if not os.path.exists('imgs'):
    os.mkdir('imgs')

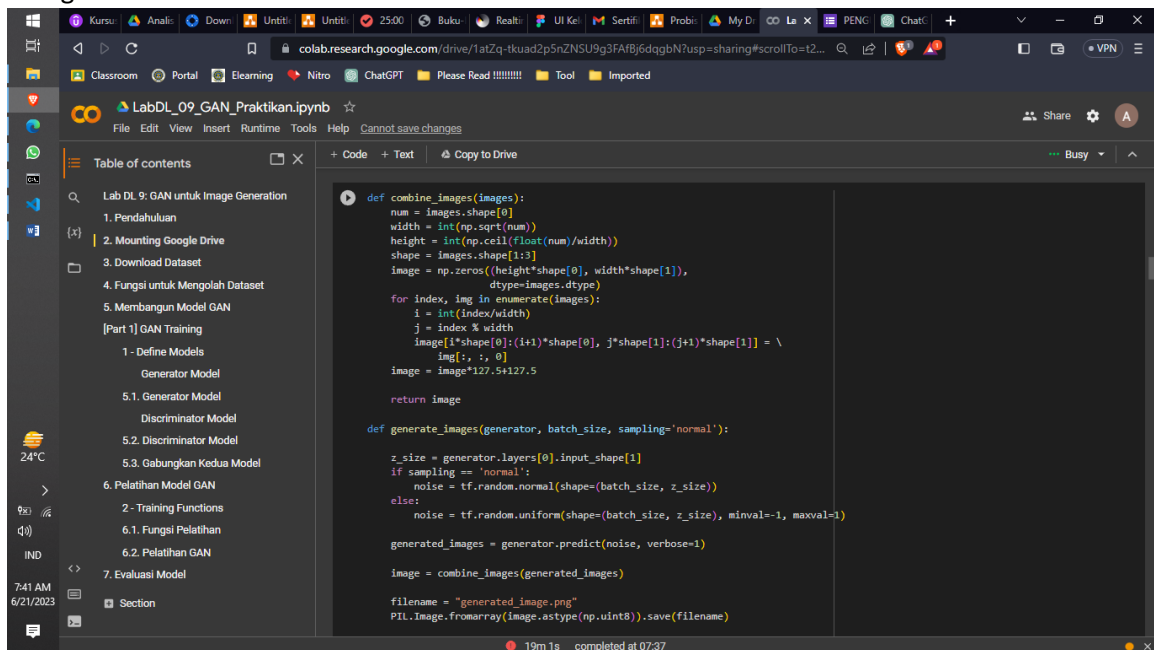
(X_train, y_train), (X_test, y_test) = mnist.load_data()

X_train = np.concatenate([X_train, X_test])
```

Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist-np11490434/11490434> [=====] - 2s 0us/step

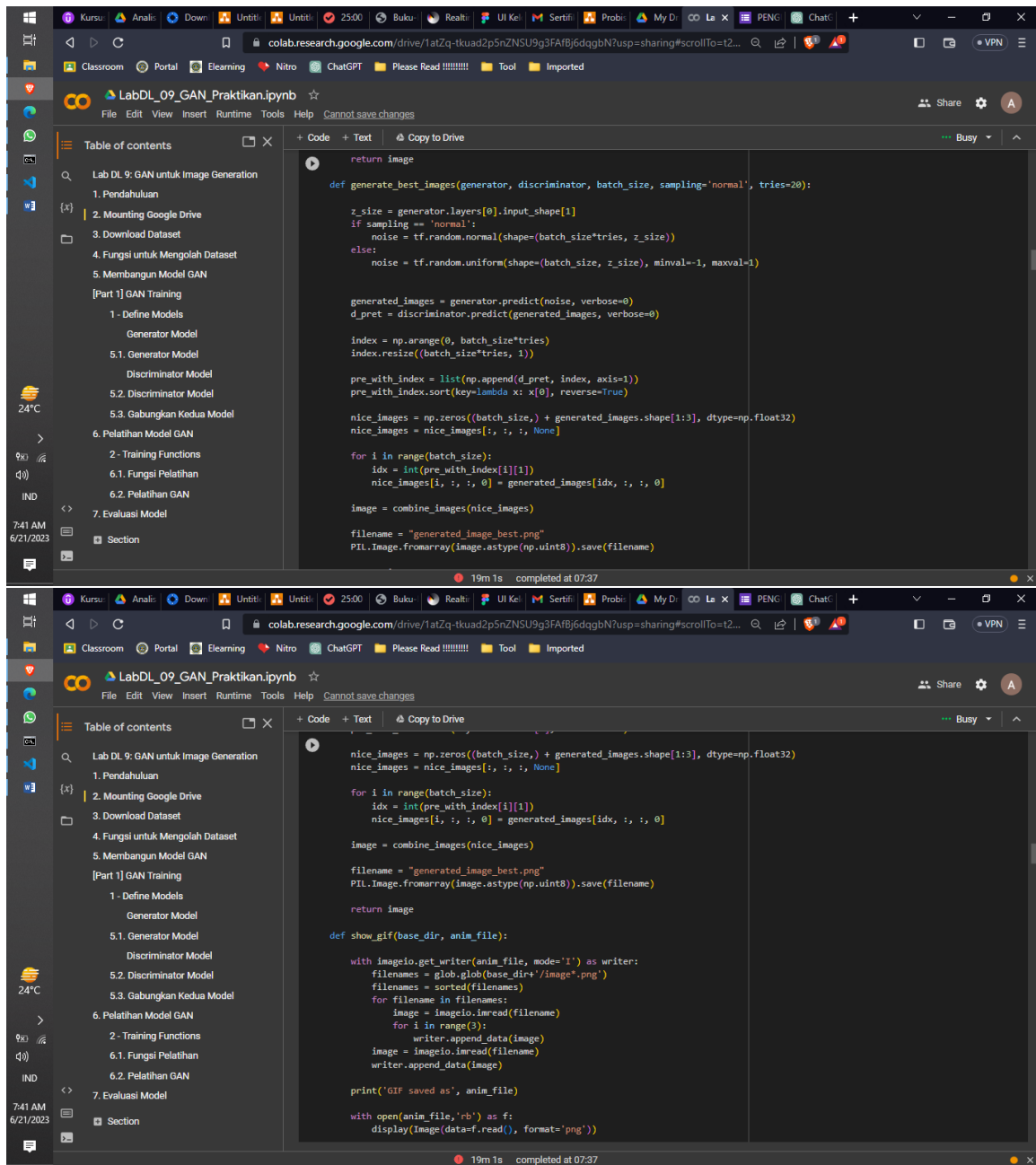
Normalisasi data latih

2. Jalankan tahap **4. Fungsi untuk Mengolah Dataset**. Tahapan ini akan membuat fungsi untuk mengolah dataset.



```
def combine_images(images):
    num = images.shape[0]
    width = int(np.sqrt(num))
    height = int(np.ceil(float(num)/width))
    shape = images.shape[1:]
    image = np.zeros((height*shape[0], width*shape[1]),
                     dtype=images.dtype)
    for index, img in enumerate(images):
        i = int(index/width)
        j = index % width
        image[i*shape[0]:(i+1)*shape[0], j*shape[1]:(j+1)*shape[1]] = \
            img[:, :, 0]
    image = image*127.5+127.5
    return image

def generate_images(generator, batch_size, sampling='normal'):
    z_size = generator.layers[0].input_shape[1]
    if sampling == 'normal':
        noise = tf.random.normal(shape=(batch_size, z_size))
    else:
        noise = tf.random.uniform(shape=(batch_size, z_size), minval=-1, maxval=1)
    generated_images = generator.predict(noise, verbose=1)
    image = combine_images(generated_images)
    filename = "generated_image.png"
    PIL.Image.fromarray(image.astype(np.uint8)).save(filename)
```



- Jalankan tahap **5. Membangun Model GAN**. Tahapan ini terdiri dari membangun model generator dan discriminator lalu digabungkan menjadi satu model. Jalankan code di **5.1. Generator Model**, kode di **5.2. Discriminator Model**, dan kode di **5.3. Gabungkan Kedua Model**.

colab.research.google.com/drive/1atZq-4kuad2p5nZNSU9g3FAfBj6dggBN?usp=sharing#scrollTo=t2...

LabDL\_09\_GAN\_Praktikan.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

Table of contents

- Lab DL 9: GAN untuk Image Generation
- 1. Pendahuluan
- 2. Mounting Google Drive
- 3. Download Dataset
- 4. Fungsi untuk Mengolah Dataset
- 5. Membangun Model GAN
- [Part 1] GAN Training
  - 1. Define Models
    - Generator Model
    - Discriminator Model
  - 5.1. Generator Model
  - 5.2. Discriminator Model
  - 5.3. Gabungkan Kedua Model
- 6. Pelatihan Model GAN
- 2. Training Functions
  - 6.1. Fungsi Pelatihan
  - 6.2. Pelatihan GAN
- 7. Evaluasi Model
- Section

```
[ ] def generator_model(latent_dim = 128):
    model = Sequential([
        Dense(1024, input_dim=latent_dim, activation='relu'),
        BatchNormalization(),
        Dense(7 * 7 * 128, activation='relu'),
        Reshape((7, 7, 128)),
        Conv2DTranspose(64, (5, 5), strides=(2,2), padding='same', activation='relu'),
        BatchNormalization(),
        Conv2DTranspose(1, (5, 5), strides=(2,2), padding='same', activation='tanh'),
    ])
    return model

generator_model().summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 1024)	132096
batch_normalization (Batch Normalization)	(None, 1024)	4096
dense_1 (Dense)	(None, 6272)	6428800
reshape (Reshape)	(None, 7, 7, 128)	0
conv2d_transpose (Conv2DTranspose)	(None, 14, 14, 64)	204864
batch_normalization_1 (Batch Normalization)	(None, 14, 14, 64)	256

19m 1s completed at 07:37

colab.research.google.com/drive/1atZq-4kuad2p5nZNSU9g3FAfBj6dggBN?usp=sharing#scrollTo=t2...

LabDL\_09\_GAN\_Praktikan.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

Table of contents

- Lab DL 9: GAN untuk Image Generation
- 1. Pendahuluan
- 2. Mounting Google Drive
- 3. Download Dataset
- 4. Fungsi untuk Mengolah Dataset
- 5. Membangun Model GAN
- [Part 1] GAN Training
  - 1. Define Models
    - Generator Model
    - Discriminator Model
  - 5.1. Generator Model
  - 5.2. Discriminator Model
  - 5.3. Gabungkan Kedua Model
- 6. Pelatihan Model GAN
- 2. Training Functions
  - 6.1. Fungsi Pelatihan
  - 6.2. Pelatihan GAN
- 7. Evaluasi Model
- Section

```
def discriminator_model():
    model = Sequential([
        Conv2D(32, (5, 5), input_shape=(28, 28, 1), padding='same'),
        LeakyReLU(alpha=0.01),
        MaxPooling2D(pool_size=(2, 2)),
        Conv2D(64, (5, 5), padding='same'),
        LeakyReLU(alpha=0.01),
        MaxPooling2D(pool_size=(2, 2)),
        Flatten(),
        Dense(4 * 4 * 64, activation='tanh'),
        Dense(1, activation='sigmoid')
    ])
    return model

discriminator_model().summary()
```

Model: "sequential\_1"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 32)	832
leaky_re_lu (LeakyReLU)	(None, 28, 28, 32)	0
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0
conv2d_1 (Conv2D)	(None, 10, 10, 64)	51264
leaky_re_lu_1 (LeakyReLU)	(None, 10, 10, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 5, 5, 64)	0

19m 1s completed at 07:37

The screenshot shows a Google Colab notebook titled 'LabDL\_09\_GAN\_Praktikan.ipynb'. The left sidebar contains a table of contents with sections: 1. Pendahuluan, 2. Mounting Google Drive, 3. Download Dataset, 4. Fungsi untuk Mengolah Dataset, 5. Membangun Model GAN, [Part 1] GAN Training, 5.1. Generator Model, 5.2. Discriminator Model, 5.3. Gabungkan Kedua Model, 6. Pelatihan Model GAN, 2 - Training Functions, 6.1. Fungsi Pelatihan, 6.2. Pelatihan GAN, 7. Evaluasi Model, and Section. The main code area shows the output of a model summary:

```
max_pooling2d_1 (MaxPooling (None, 5, 5, 64) 0)
flatten (Flatten) (None, 1600) 0
dense_2 (Dense) (None, 1024) 1639424
dense_3 (Dense) (None, 1) 1025
```

Below the summary, it states: Total params: 1,692,545, Trainable params: 1,692,545, Non-trainable params: 0. The section header '5.3. Gabungkan Kedua Model' is visible, followed by a code cell for 'def combine\_model(g, d):'.

4. Jalankan tahap **6. Training Model GAN**. Tahapan ini akan melatih model GAN yang dibuat. Jalankan kode di **6.1. Fungsi Pelatihan** untuk membuat fungsi-fungsi yang diperlukan saat pelatihan dan mengatur konfigurasi dari hyperparameter dan jalankan kode di **6.2. Pelatihan GAN** untuk memulai pelatihan.

The screenshot shows the same Google Colab notebook, but now displaying the code for training a GAN. The code defines a function 'train\_gan' and a sampling function 'sampling\_fn'.

```
def train_gan(X_train, Y_train,
              batch_size, epochs,
              models, samplings='normal',
              save_every=500, print_every=100):

    def sampling_fn(shape):
        if sampling == 'normal':
            return tf.random.normal(shape)
        else:
            return tf.random.uniform(shape, minval=-1, maxval=1)

    g, d, d_on_g = models

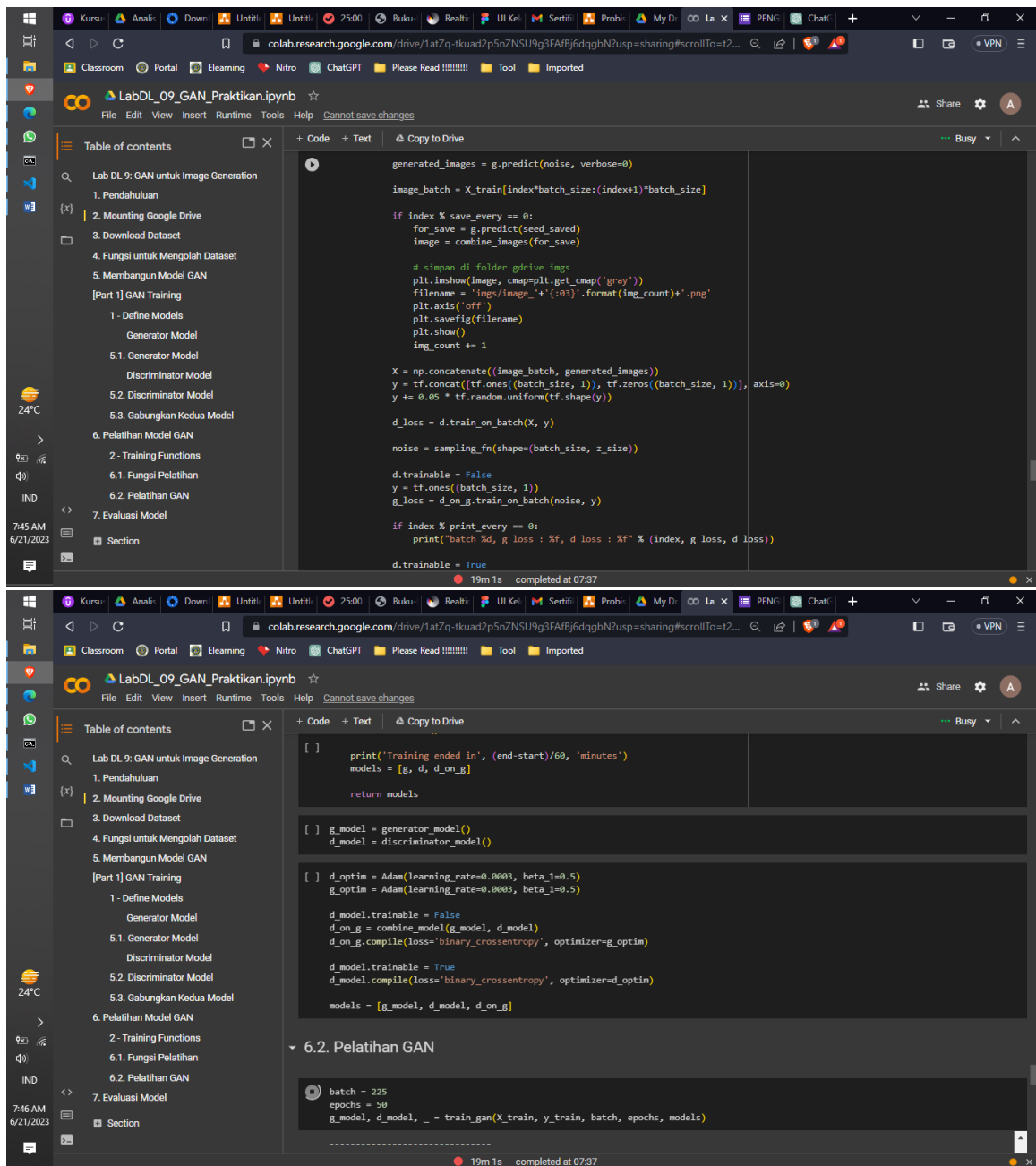
    print('Start training using random', sampling, 'distribution')
    print('Number of batches', int(X_train.shape[0]/batch_size))

    z_size = g.layers[0].input_shape[1]
    seed_saved = sampling_fn(shape=(16, z_size))
    img_count = 0

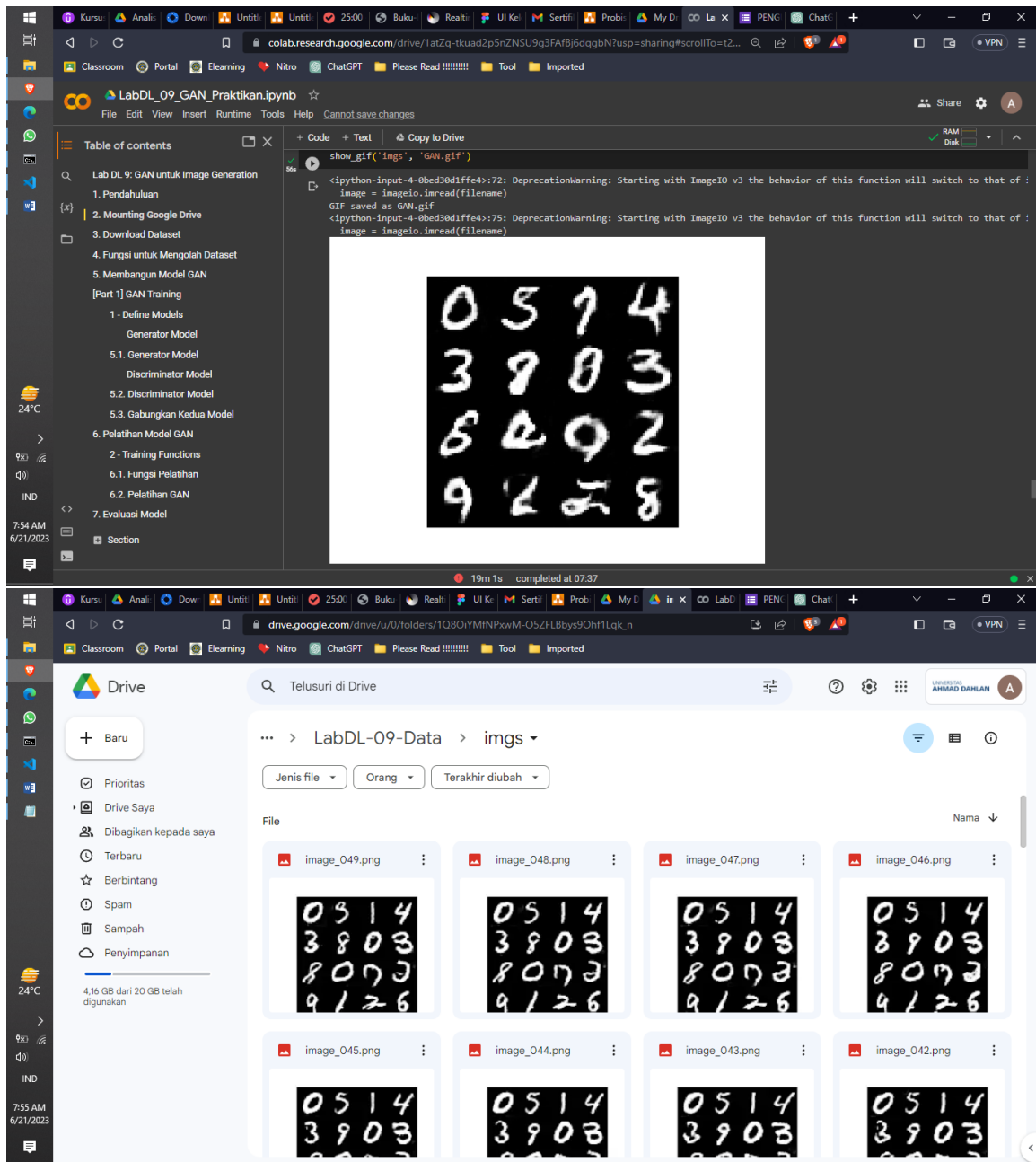
    start = time.time()

    for epoch in range(epochs):
        print("\n-----\nEpoch is", epoch)

        for index in range(int(X_train.shape[0]/batch_size)):
            noise = sampling_fn(shape=(batch_size, z_size))
```



5. Tunggu pelatihan sampai selesai karena akan memakan waktu lama. Pada saat pelatihan anda akan melihat proses pembentukan citra digit per epochnya. Proses tersebut akan disimpan di folder imgs pada google drive anda.



6. Jalankan tahap **7. Evaluasi Model**. Pada tahapan ini model akan membuat citra digit yang sudah dipelajari saat pelatihan. Pengujian GAN biasanya menggunakan analisis visual pada citra yang dihasilkan oleh manusia.



colab.research.google.com/drive/1atZq-tkuad2p5nZNSU9g3FAF8j6dqqbN?usp=sharing#scrollTo=t2...

LabDL\_09\_GAN\_Praktikan.ipynb


Table of contents

- Lab DL 9: GAN untuk Image Generation
- 1. Pendahuluan
- 2. Mounting Google Drive
- 3. Download Dataset
- 4. Fungsi untuk Mengolah Dataset
- 5. Membangun Model GAN
- [Part 1] GAN Training
  - 1 - Define Models
    - Generator Model
    - 5.1. Generator Model
    - Discriminator Model
    - 5.2. Discriminator Model
    - 5.3. Gabungkan Kedua Model
  - 6. Pelatihan Model GAN
    - 2 - Training Functions
    - 6.1. Fungsi Pelatihan
    - 6.2. Pelatihan GAN
  - 7. Evaluasi Model
- Section

```
seed = tf.random.normal(shape=(4, 128))
images = g_model.predict(seed)

for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(np.reshape(images[i], (28,28)), cmap=plt.get_cmap('gray'))
    plt.axis('off')
plt.show()
```

1/1 [=====] - 0s 114ms/step



19m 1s completed at 07:37

colab.research.google.com/drive/1atZq-tkuad2p5nZNSU9g3FAF8j6dqqbN?usp=sharing#scrollTo=t2...

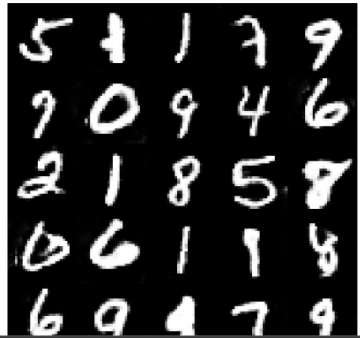
LabDL\_09\_GAN\_Praktikan.ipynb

Table of contents

- Lab DL 9: GAN untuk Image Generation
- 1. Pendahuluan
- 2. Mounting Google Drive
- 3. Download Dataset
- 4. Fungsi untuk Mengolah Dataset
- 5. Membangun Model GAN
- [Part 1] GAN Training
  - 1 - Define Models
    - Generator Model
    - 5.1. Generator Model
    - Discriminator Model
    - 5.2. Discriminator Model
    - 5.3. Gabungkan Kedua Model
  - 6. Pelatihan Model GAN
    - 2 - Training Functions
    - 6.1. Fungsi Pelatihan
    - 6.2. Pelatihan GAN
  - 7. Evaluasi Model
- Section

```
[14] images = generate_images(g_model, 25)
plt.figure(figsize = (6,6))
plt.imshow(images, cmap=plt.get_cmap('gray'))
plt.axis('off')
plt.show()
```

1/1 [=====] - 0s 62ms/step



19m 1s completed at 07:37



colab.research.google.com/drive/1atZq-tkuad2p5nZNSU9g3FAfBj6dqqbN7usp=sharing#scrollTo=t2...

LabDL\_09\_GAN\_Praktikan.ipynb

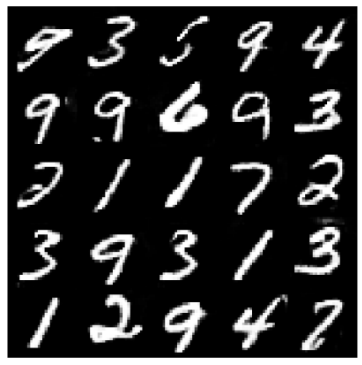
File Edit View Insert Runtime Tools Help Cannot save changes

Table of contents

- Lab DL 9: GAN untuk Image Generation
  - 1. Pendahuluan
  - 2. Mounting Google Drive
  - 3. Download Dataset
  - 4. Fungsi untuk Mengolah Dataset
  - 5. Membangun Model GAN
    - [Part 1] GAN Training
      - 1. Define Models
        - Generator Model
        - 5.1. Generator Model
        - Discriminator Model
        - 5.2. Discriminator Model
        - 5.3. Gabungkan Kedua Model
      - 6. Pelatihan Model GAN
        - 2. Training Functions
        - 6.1. Fungsi Pelatihan
        - 6.2. Pelatihan GAN
      - 7. Evaluasi Model

1x

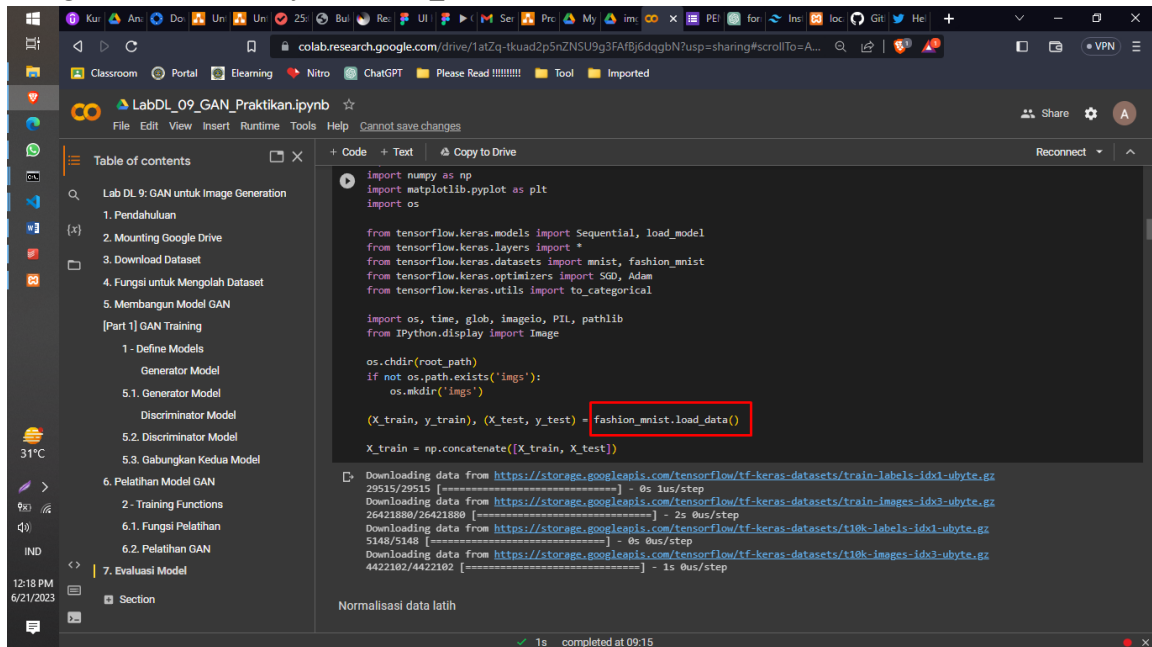
```
images = generate_best_images(g_model, d_model, 25)
plt.figure(figsize = (6,6))
plt.imshow(images, cmap=plt.get_cmap('gray'))
plt.axis('off')
plt.show()
```



19m 1s completed at 07:37

# POSTEST

## 1. Mengubah dataset menjadi fashion\_mist



```
import numpy as np
import matplotlib.pyplot as plt
import os

from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import *
from tensorflow.keras.datasets import mnist, fashion_mnist
from tensorflow.keras.optimizers import SGD, Adam
from tensorflow.keras.utils import to_categorical

import os, time, glob, imageio, PIL, pathlib
from IPython.display import Image

os.chdir(root_path)
if not os.path.exists('imgs'):
    os.mkdir('imgs')

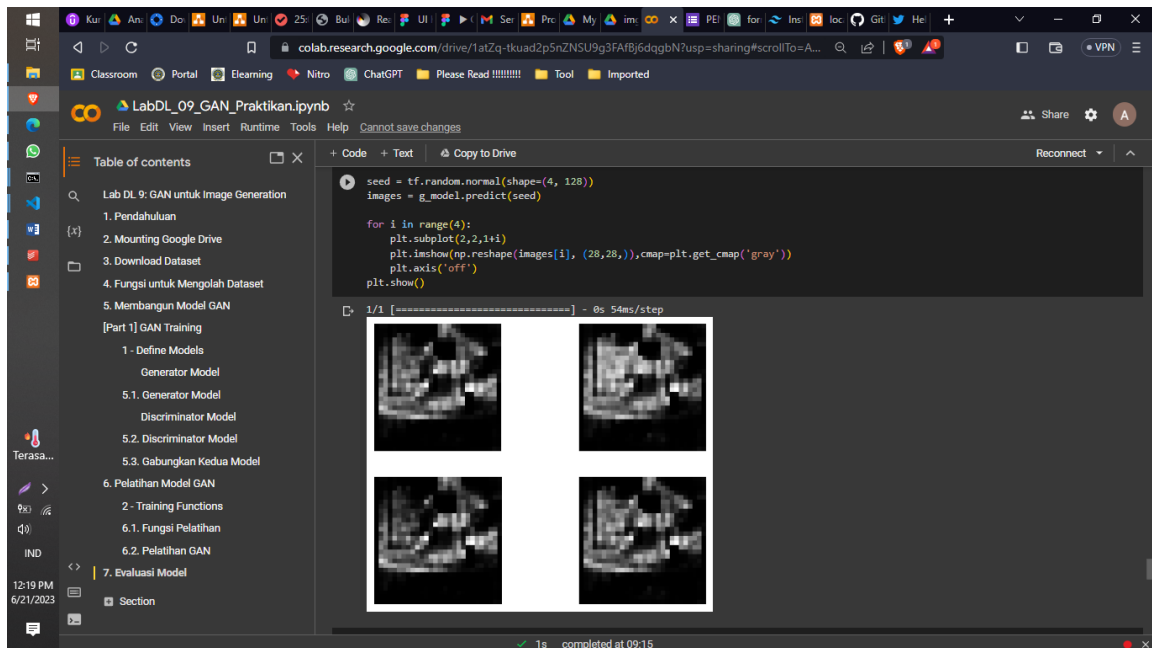
(X_train, y_train), (X_test, y_test) = fashion_mnist.load_data()

X_train = np.concatenate([X_train, X_test])
```

Download data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz> 29515/29515 [=====] - 0s 1us/step  
Download data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz> 26421880/26421880 [=====] - 2s 0us/step  
Download data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz> 5148/5148 [=====] - 0s 0us/step  
Download data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz> 4422102/4422102 [=====] - 1s 0us/step

Normalisasi data latih

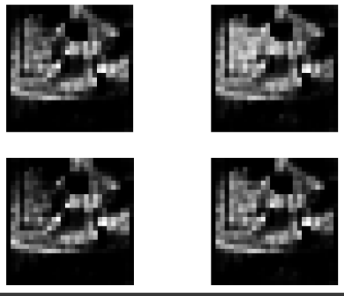
## 2. Hasil Evaluasi Model

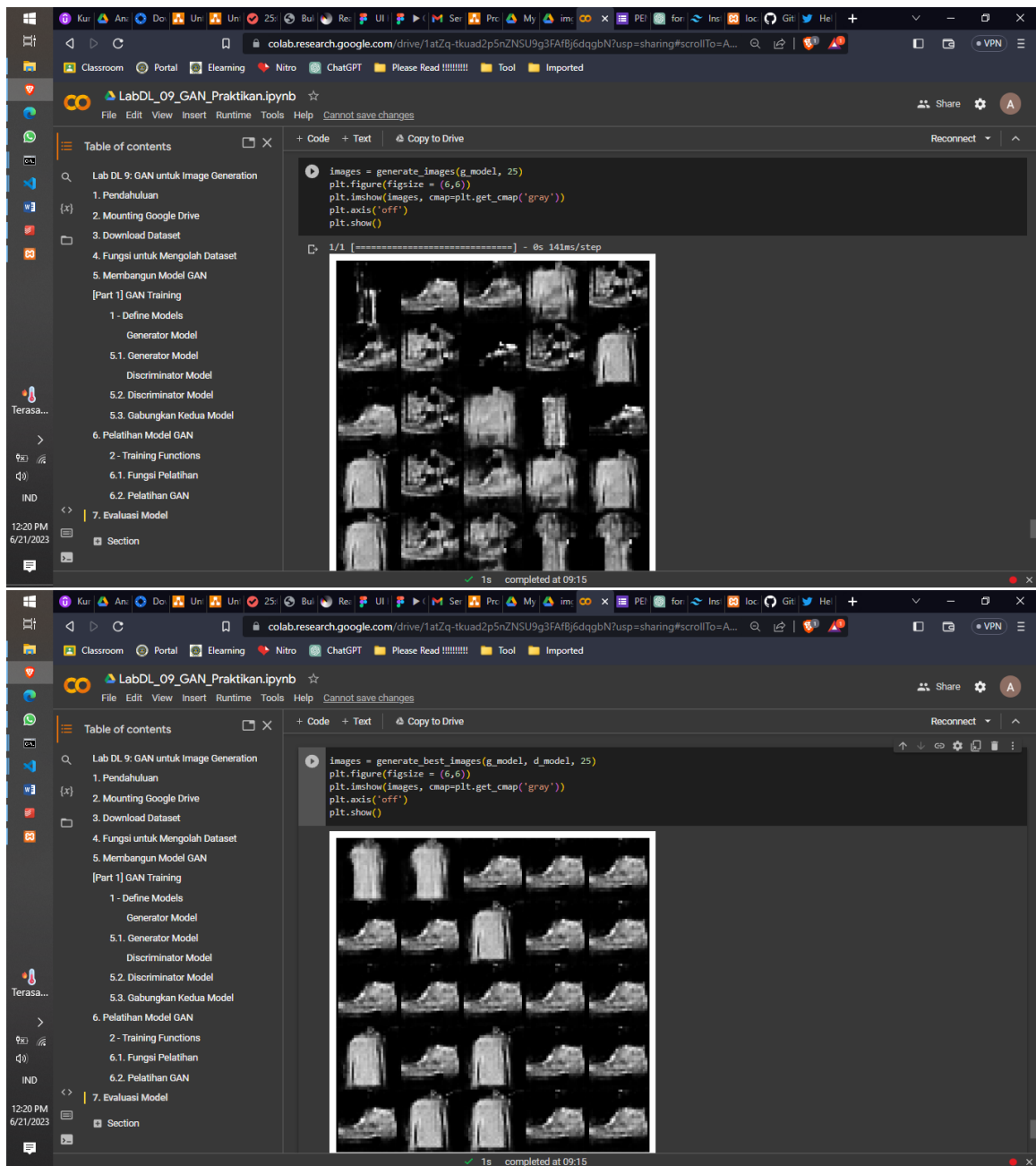


```
seed = tf.random.normal(shape=(4, 128))
images = g_model.predict(seed)

for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(np.reshape(images[i], (28,28)), cmap=plt.get_cmap('gray'))
    plt.axis('off')
plt.show()
```

1/1 [=====] - 0s 54ms/step





Dapat terlihat best image yang didapatkan pada posttest dengan praktikum sangatlah berbeda, alasannya karena gambar saat praktikum lebih sederhana dibandingkan dengan gambar dataset yang ada pada posttest, oleh karena itu best image yang dimiliki posttest hanyalah gambar baju dan sepatu, berbeda dengan praktikum.