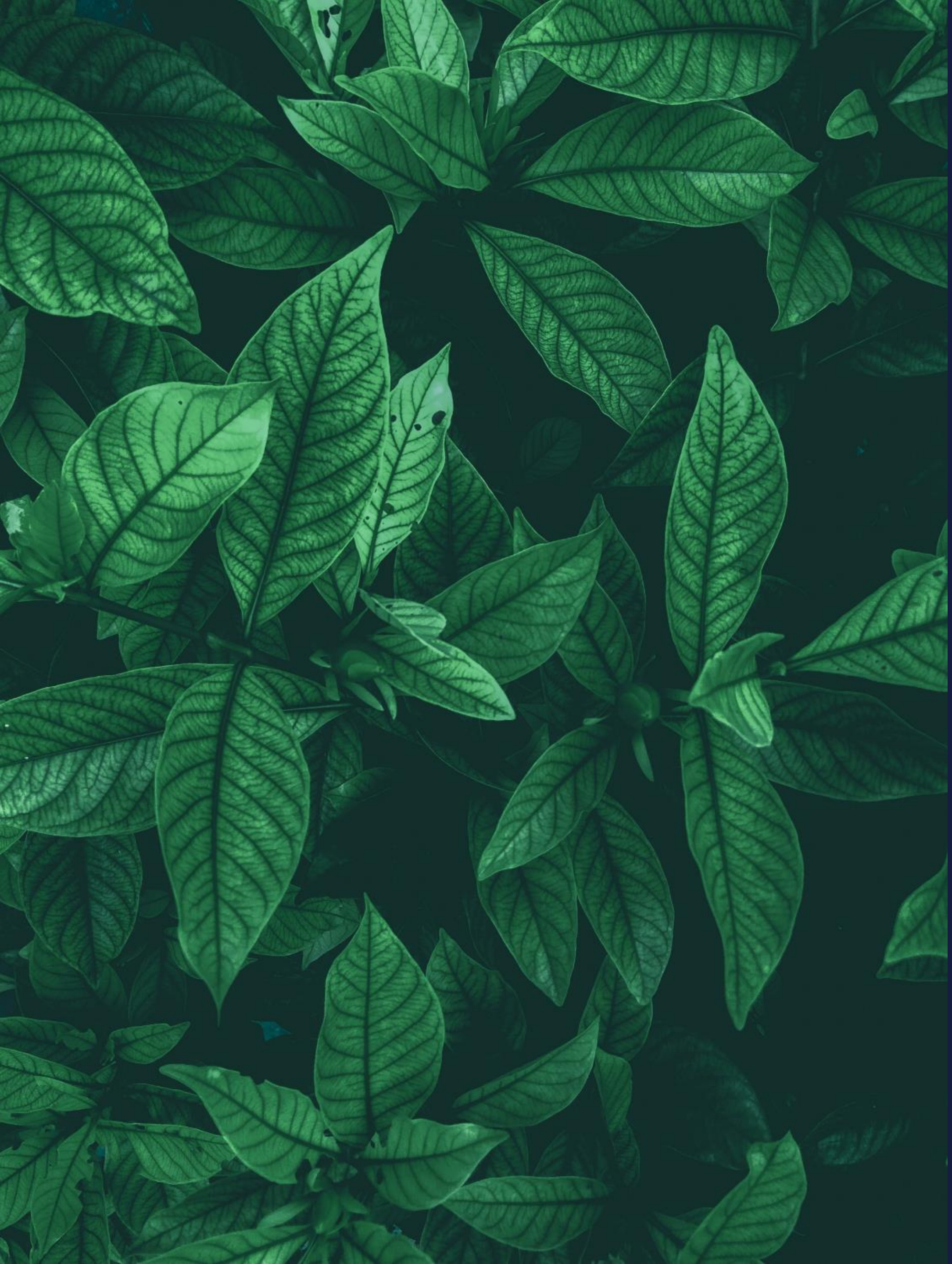


LEAFGUARDS

MUHAMMET HAKAN TAŞTAN

ERAY KURU

YASEMİN KÖSE

Inception Meme Generator on imgflip

To go deeper or not to go deeper

- Inception network is a network that has made significant contributions to field.
- Inception Module
- Complex yet efficient

Challenges

Photo by billow926 on Unsplash

Photo by
Elena Loshina
on Unsplash

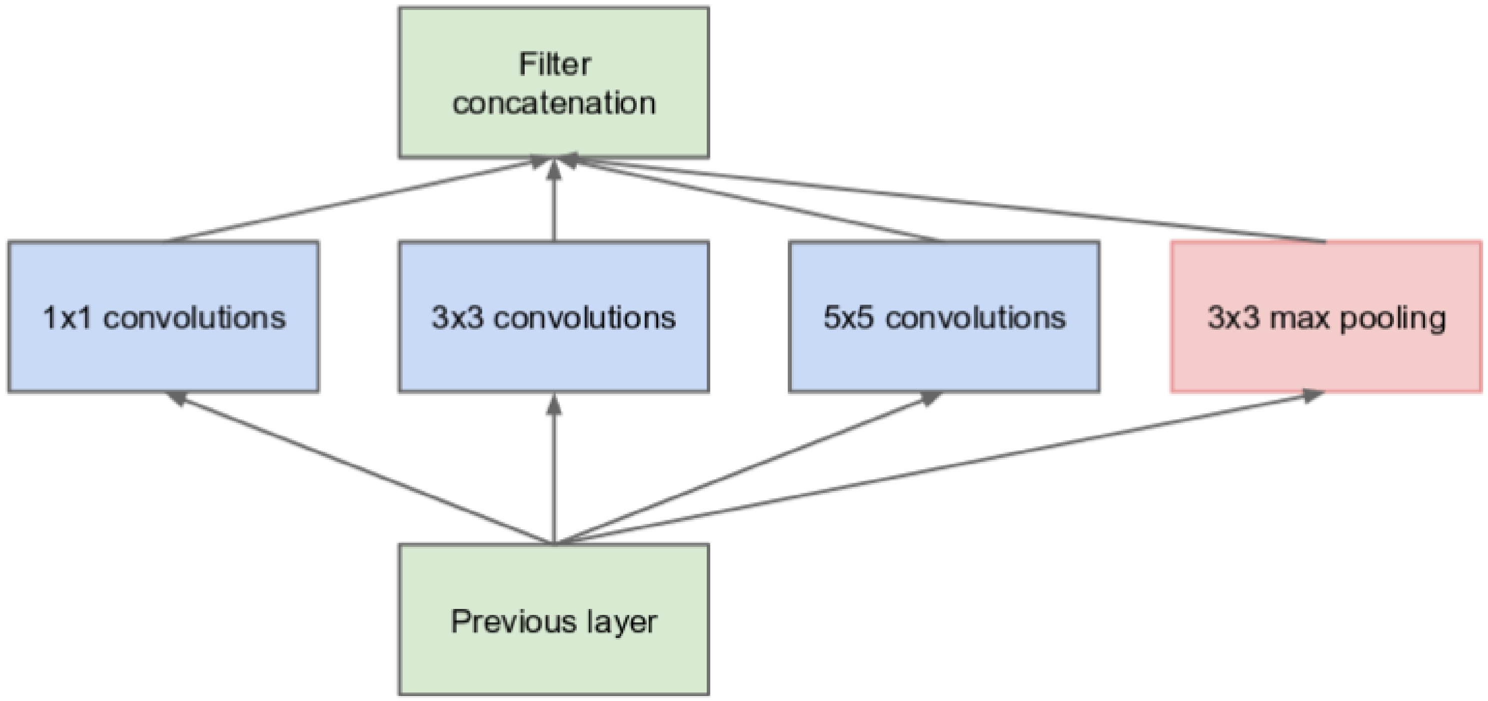
Photo by Debbie Molle on Unsplash

Solution

Buzz And Woody Meme on imgflip

INCEPTION MODULE

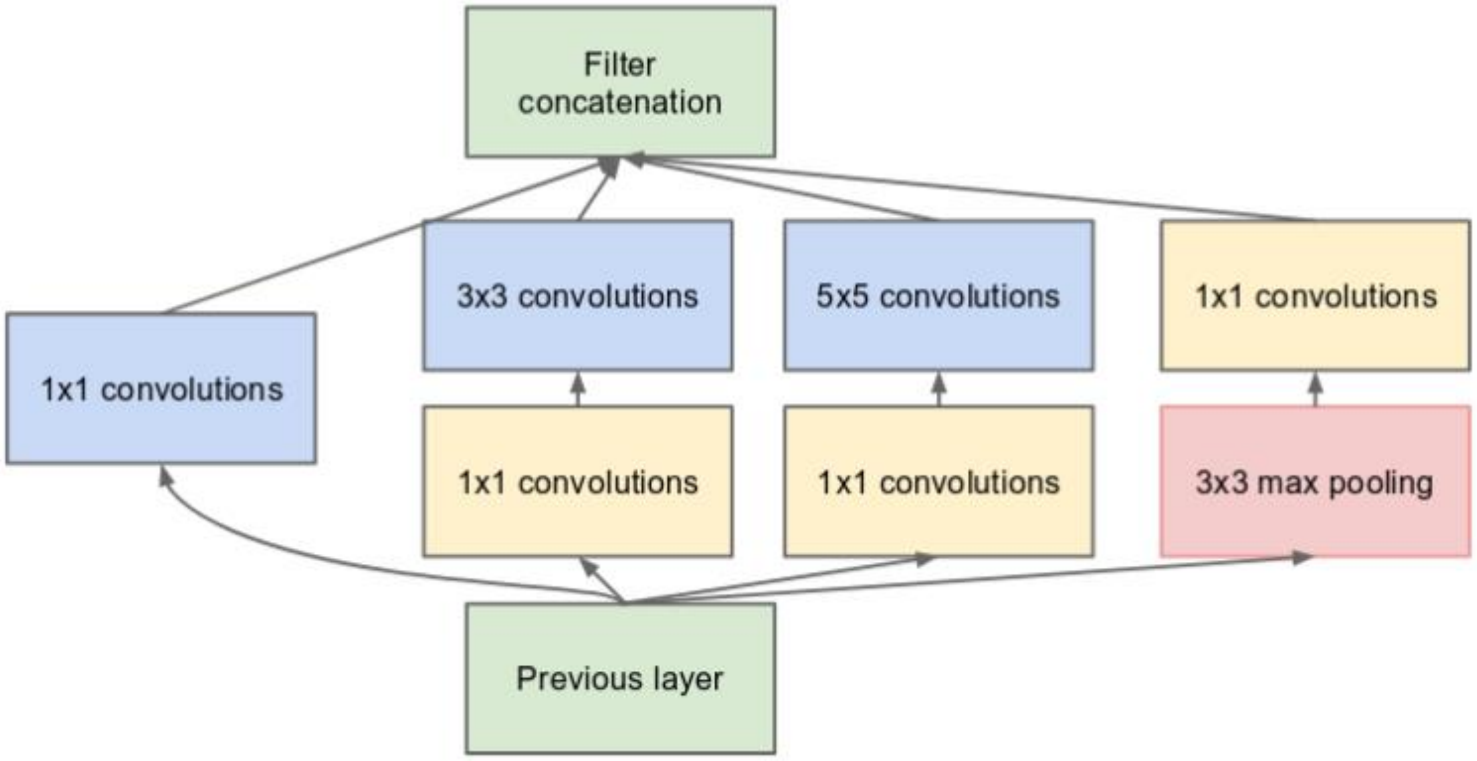
- the inception module enables the network to extract a richer set of features



(a) Inception module, naïve version

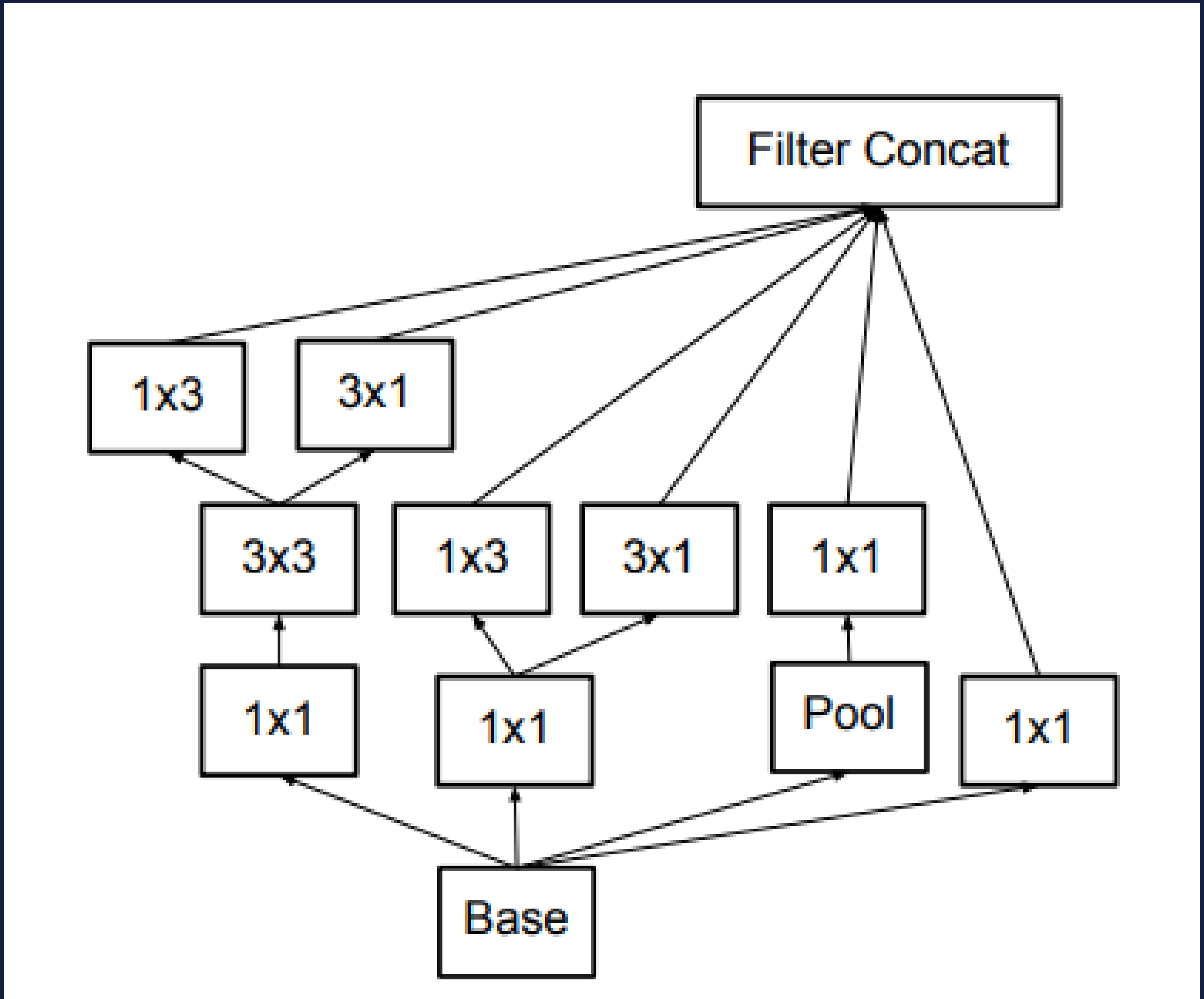
C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

The addition of 1x1 convolutional layer within the inception module helps reduce the number of parameters and computational cost.

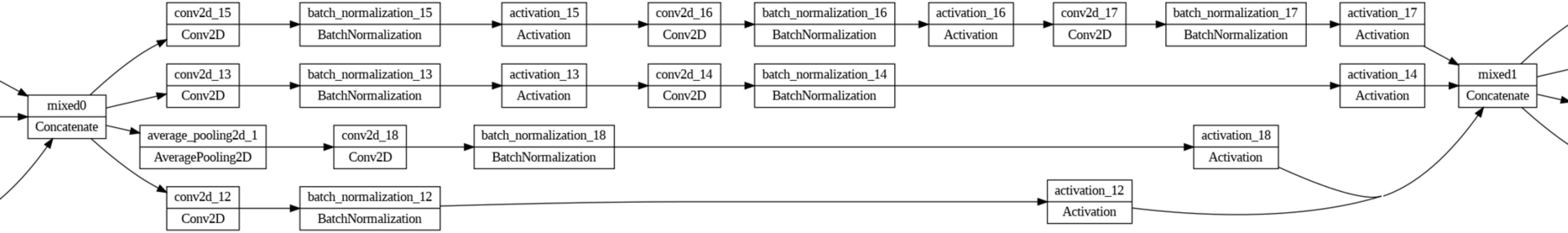


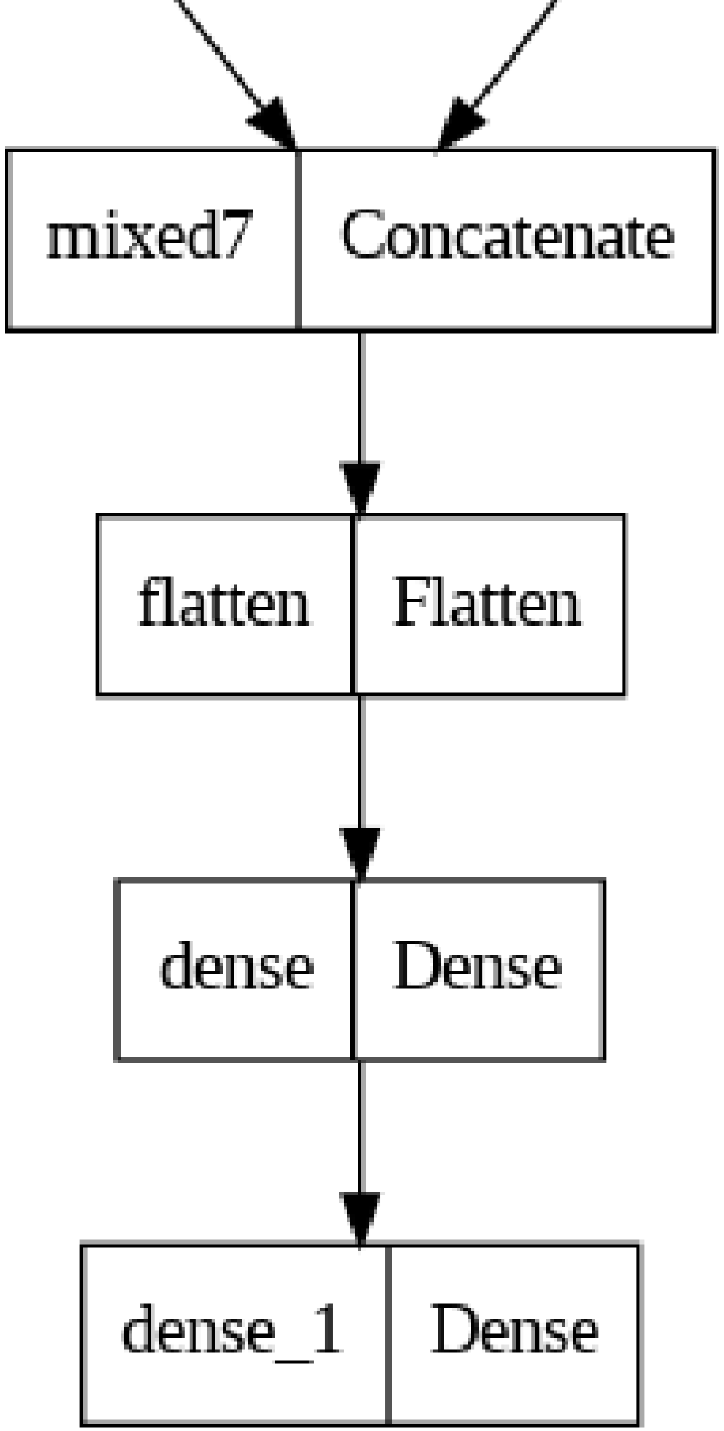
(b) Inception module with dimension reductions

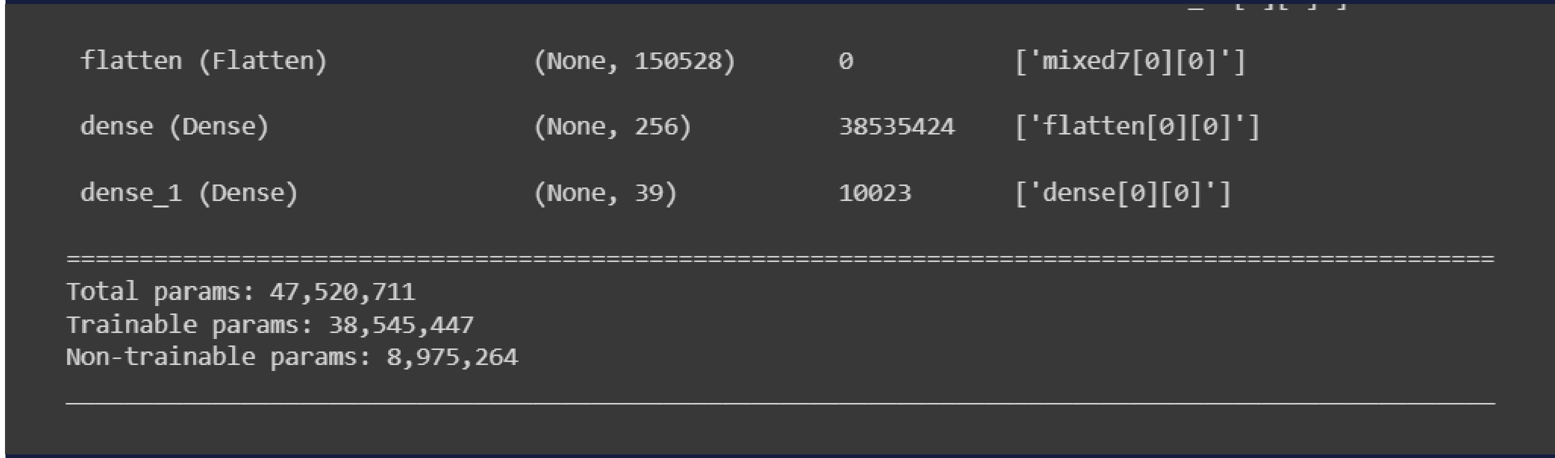
C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.



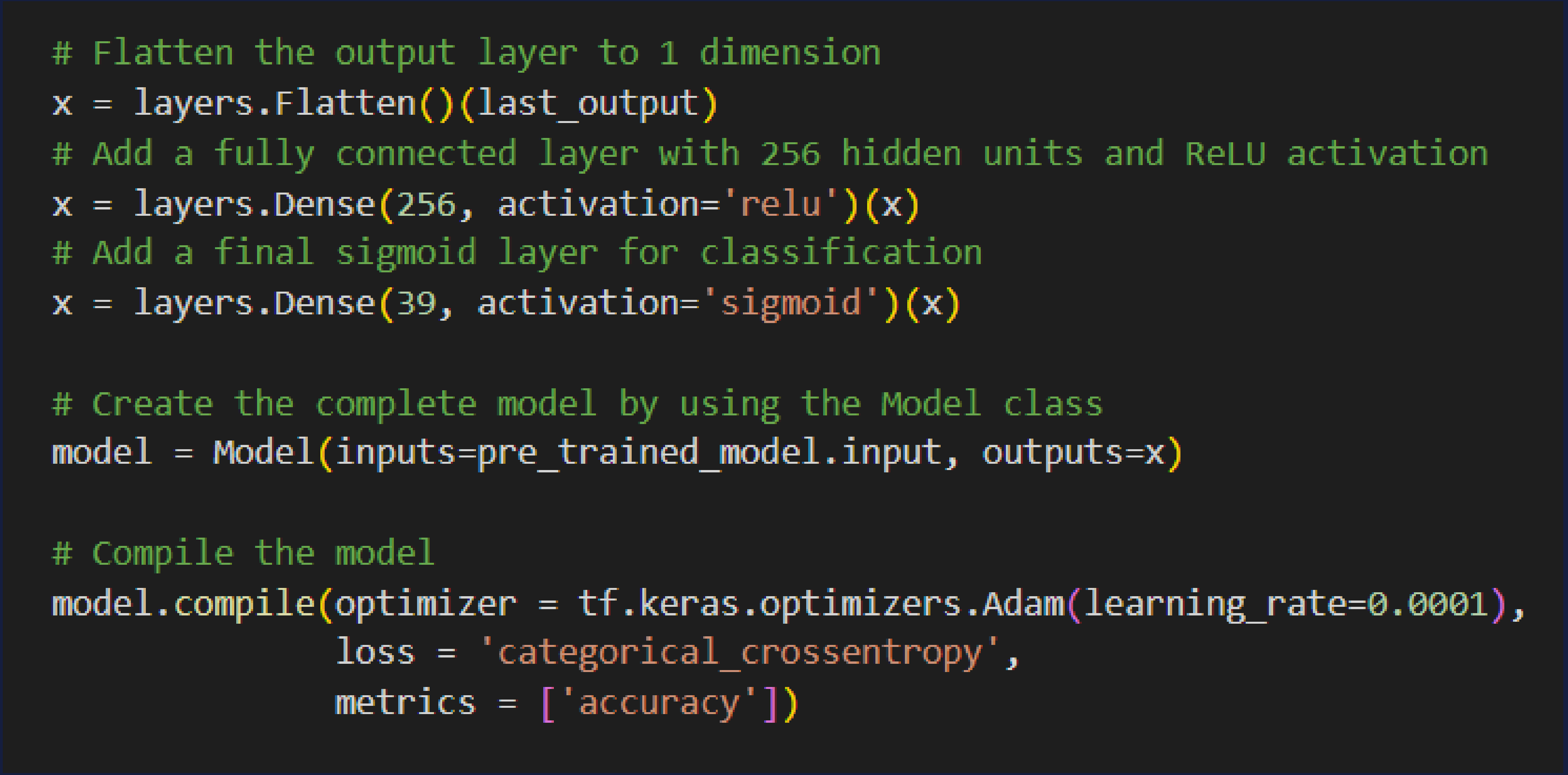
C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 2818-2826, doi: 10.1109/CVPR.2016.308.


mixed0
Concatenate
conv2d_15
Conv2D
batch_normalization_15
BatchNormalization
activation_15
Activation
conv2d_16
Conv2D
batch_normalization_16
BatchNormalization
activation_16
Activation
conv2d_17
Conv2D
batch_normalization_17
BatchNormalization
activation_17
Activation
conv2d_13
Conv2D
batch_normalization_13
BatchNormalization
activation_13
Activation
conv2d_14
Conv2D
batch_normalization_14
BatchNormalization
activation_14
Activation
average_pooling2d_1
AveragePooling2D
conv2d_18
Conv2D
batch_normalization_18
BatchNormalization
activation_18
Activation
conv2d_12
Conv2D
batch_normalization_12
BatchNormalization
activation_12
Activation
mixed1
Concatenate

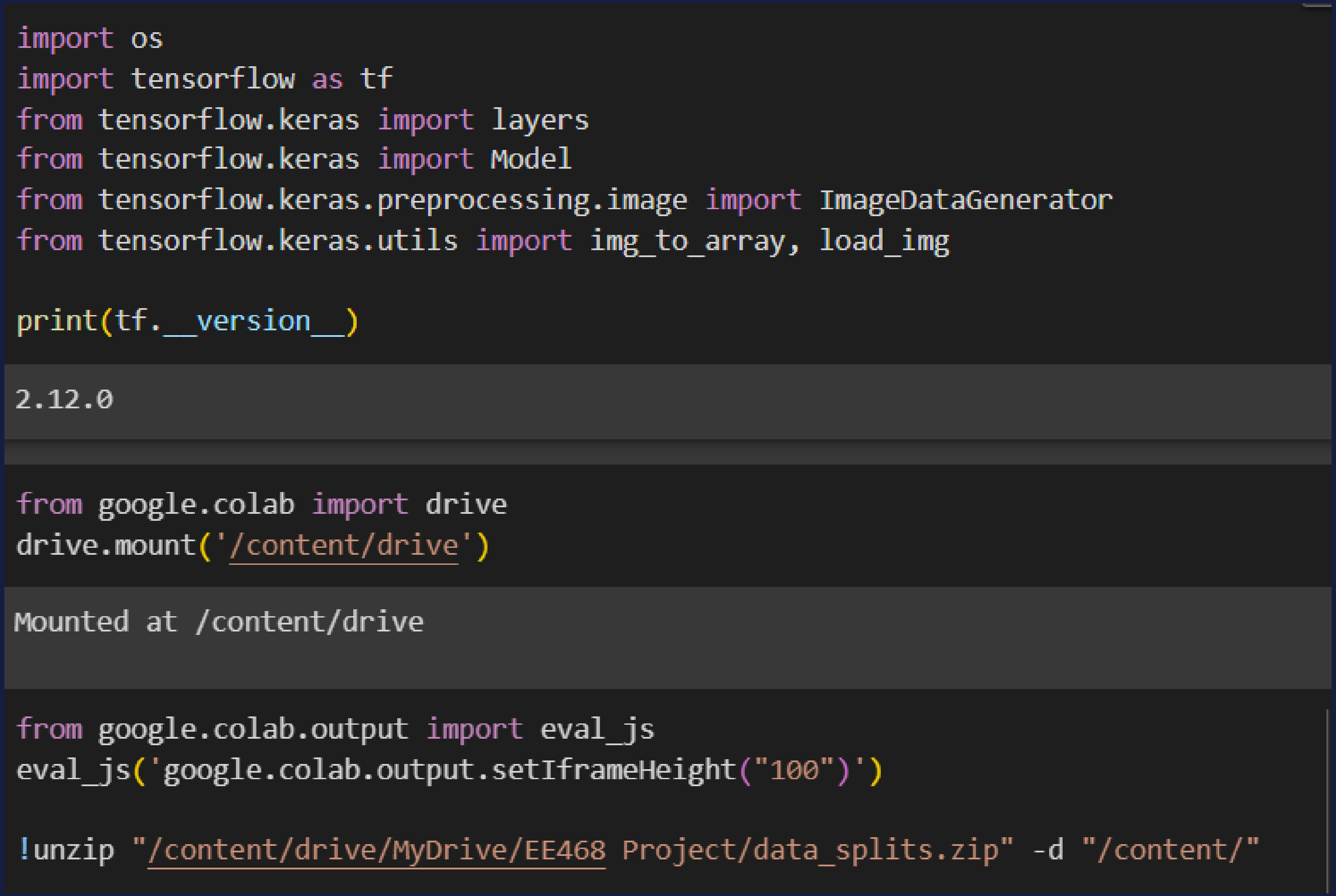




```
 flatten (Flatten)              (None, 150528)       0           ['mixed7[0][0]']

 dense (Dense)                  (None, 256)          38535424    ['flatten[0][0]']

 dense_1 (Dense)                (None, 39)           10023       ['dense[0][0]']

==================================================================================================
Total params: 47,520,711
Trainable params: 38,545,447
Non-trainable params: 8,975,264
__________________________________________________________________________________________________
```



```
# Flatten the output layer to 1 dimension
x = layers.Flatten()(last_output)
# Add a fully connected layer with 256 hidden units and ReLU activation
x = layers.Dense(256, activation='relu')(x)
# Add a final sigmoid layer for classification
x = layers.Dense(39, activation='sigmoid')(x)

# Create the complete model by using the Model class
model = Model(inputs=pre_trained_model.input, outputs=x)

# Compile the model
model.compile(optimizer = tf.keras.optimizers.Adam(learning_rate=0.0001),
              loss = 'categorical_crossentropy',
              metrics = ['accuracy'])
```



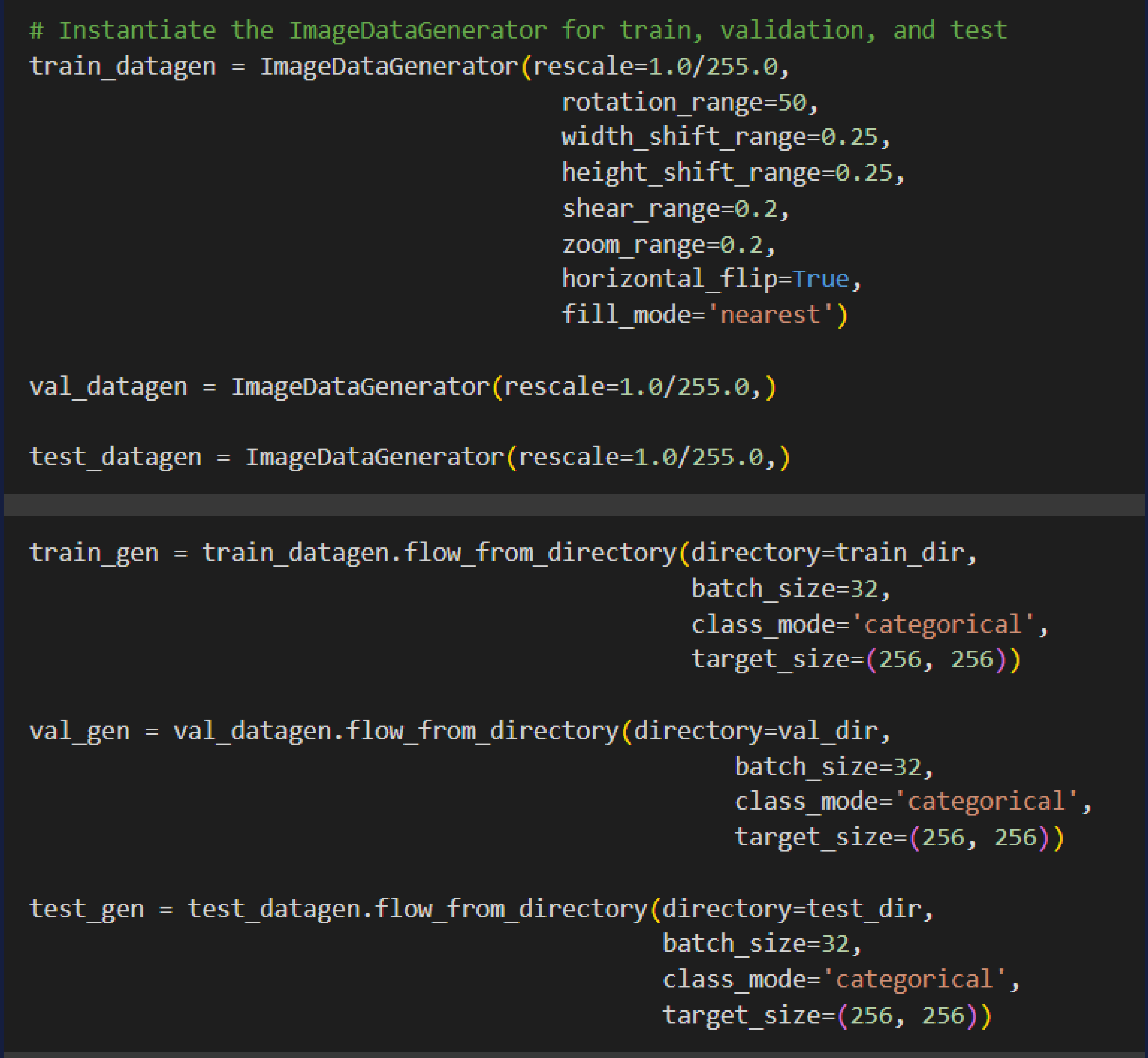
```
import os
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import Model
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import img_to_array, load_img

print(tf.__version__)
```

```
2.12.0
```

```
from google.colab import drive
drive.mount('/content/drive')
```

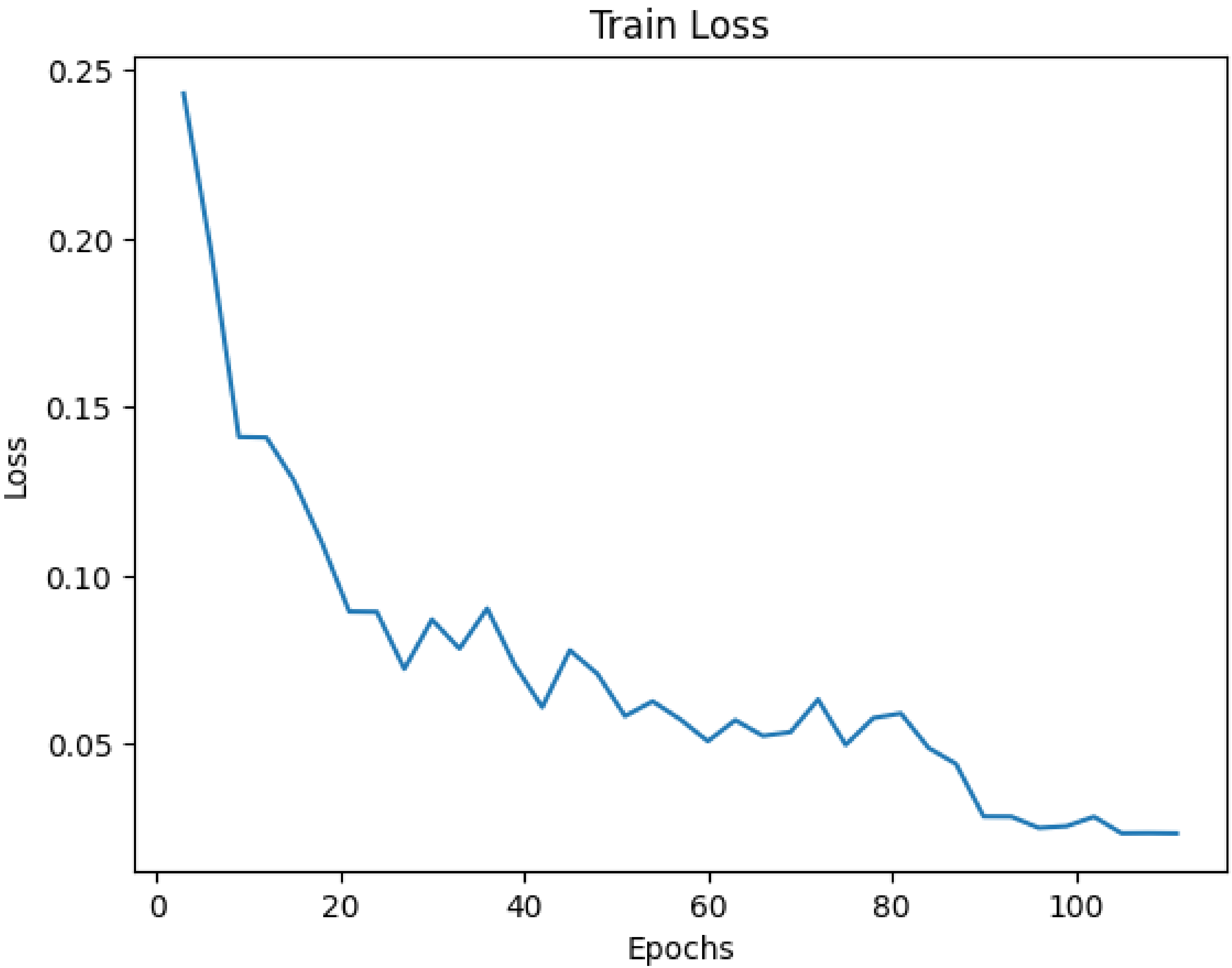
```
Mounted at /content/drive
```

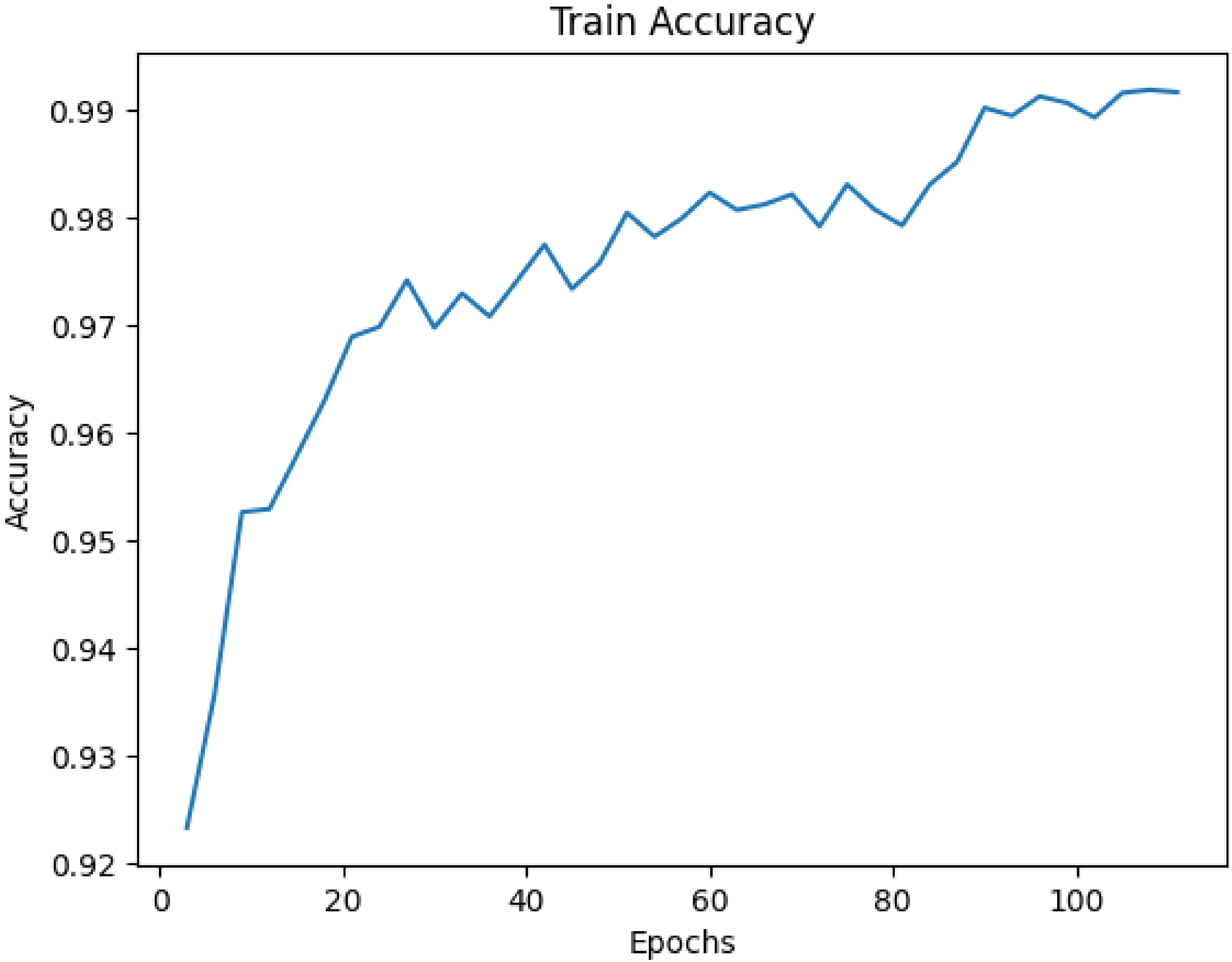
```
from google.colab.output import eval_js
eval_js('google.colab.output.setIframeHeight("100")')

!unzip "/content/drive/MyDrive/EE468 Project/data_splits.zip" -d "/content/"
```

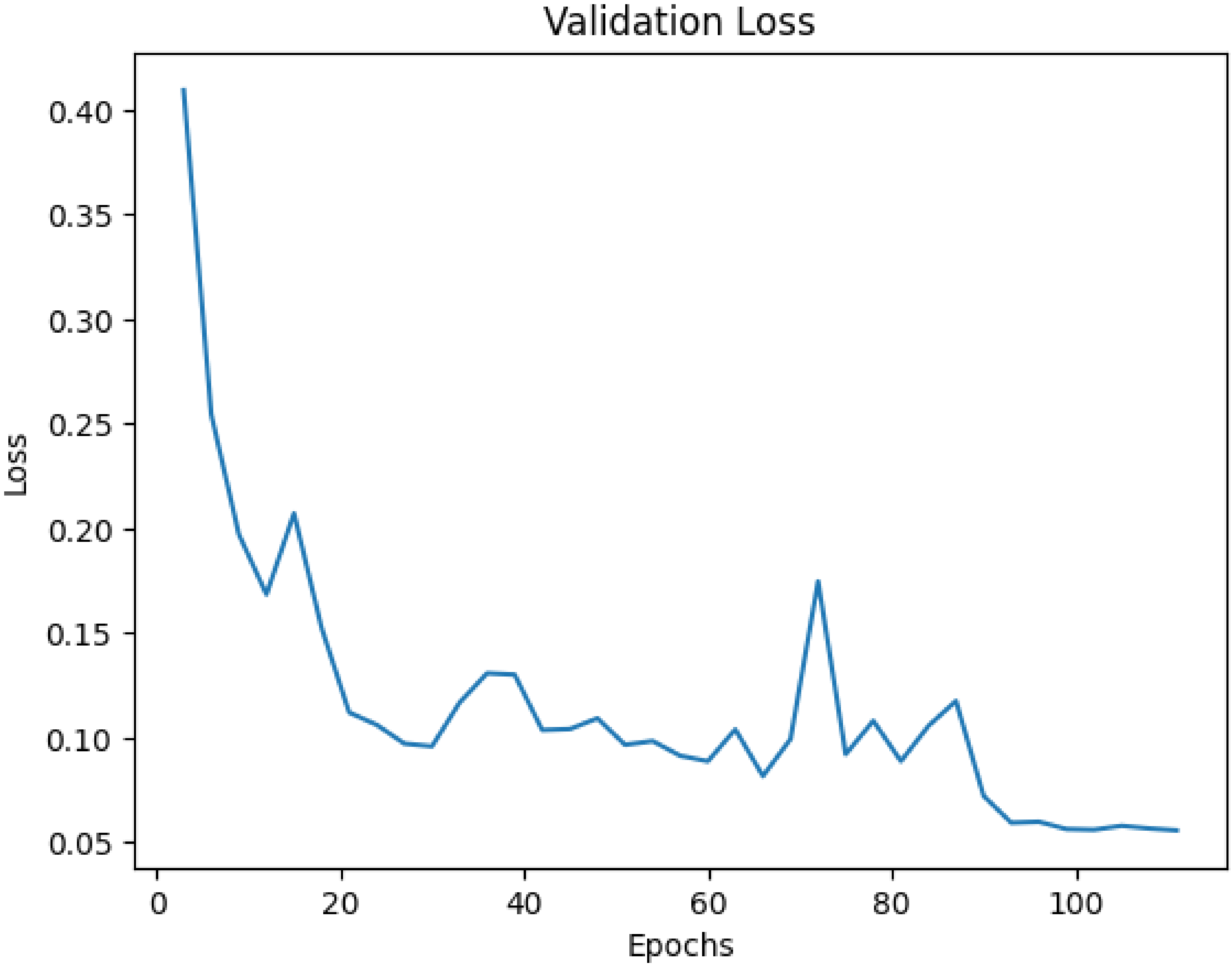


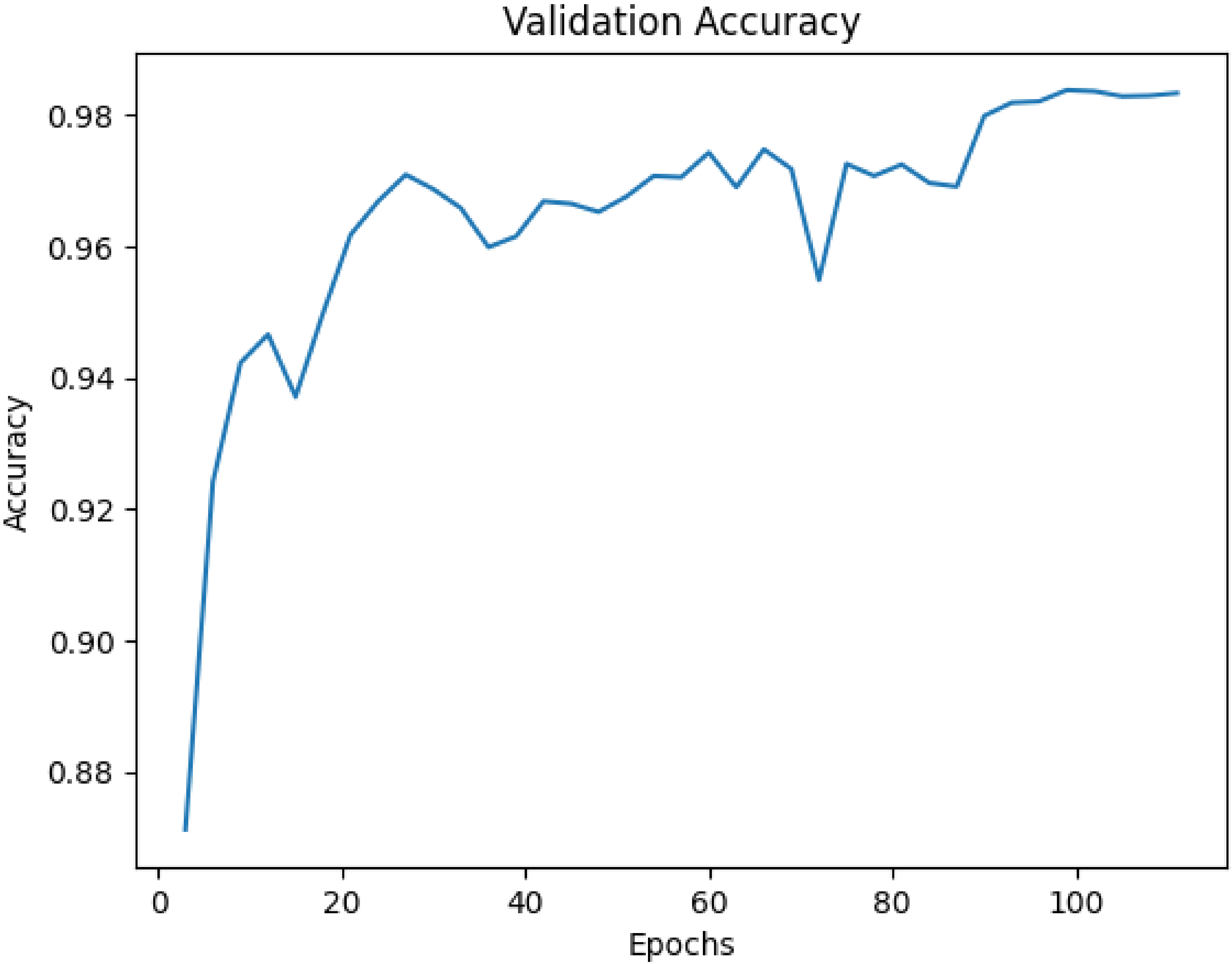
```
# Instantiate the ImageDataGenerator for train, validation, and test
train_datagen = ImageDataGenerator(rescale=1.0/255.0,
                                     rotation_range=50,
                                     width_shift_range=0.25,
                                     height_shift_range=0.25,
                                     shear_range=0.2,
                                     zoom_range=0.2,
                                     horizontal_flip=True,
                                     fill_mode='nearest')

val_datagen = ImageDataGenerator(rescale=1.0/255.0,)

test_datagen = ImageDataGenerator(rescale=1.0/255.0,)
```

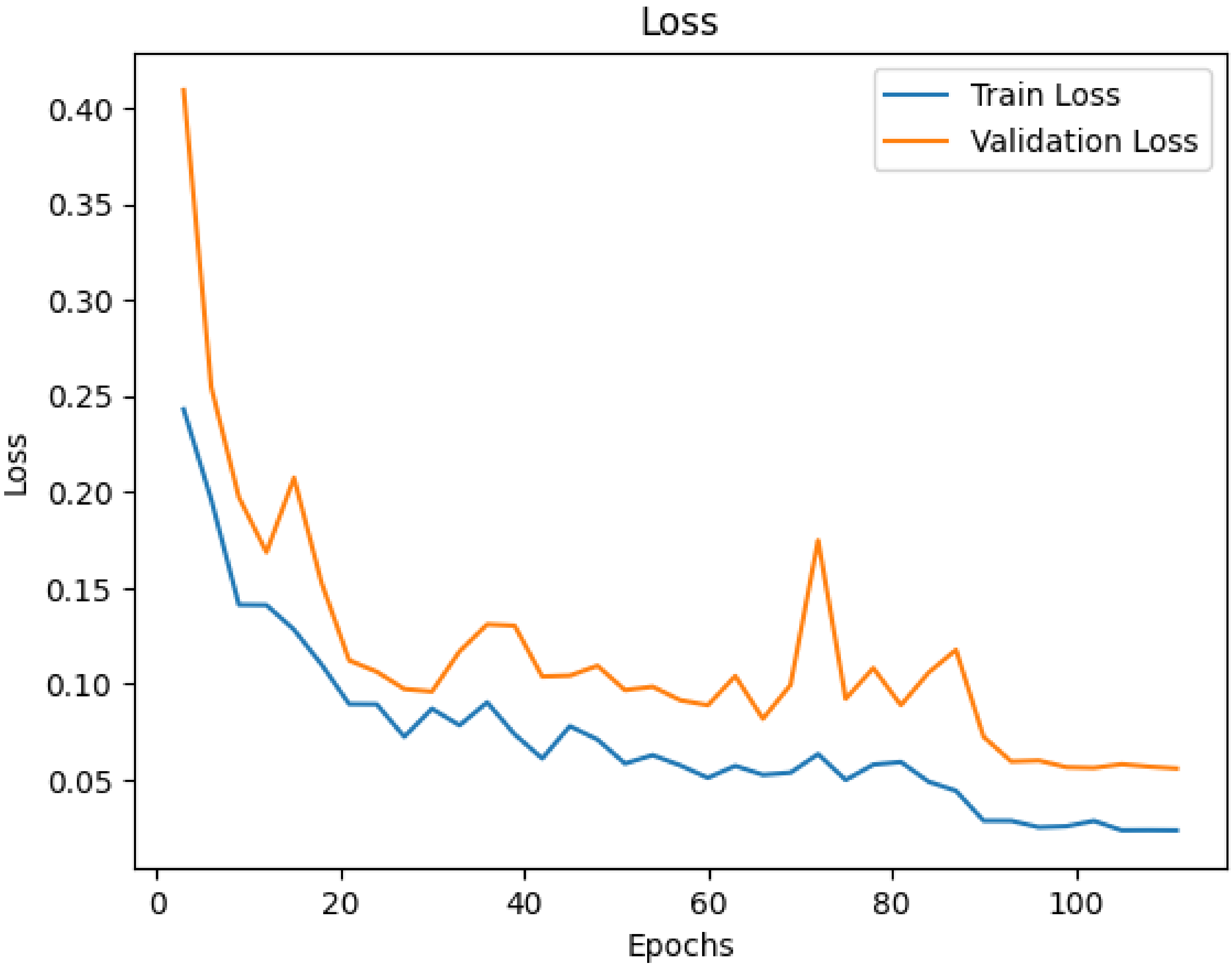
```
train_gen = train_datagen.flow_from_directory(directory=train_dir,
                                              batch_size=32,
                                              class_mode='categorical',
                                              target_size=(256, 256))

val_gen = val_datagen.flow_from_directory(directory=val_dir,
                                                 batch_size=32,
                                                 class_mode='categorical',
                                                 target_size=(256, 256))

test_gen = test_datagen.flow_from_directory(directory=test_dir,
                                            batch_size=32,
                                            class_mode='categorical',
                                            target_size=(256, 256))
```

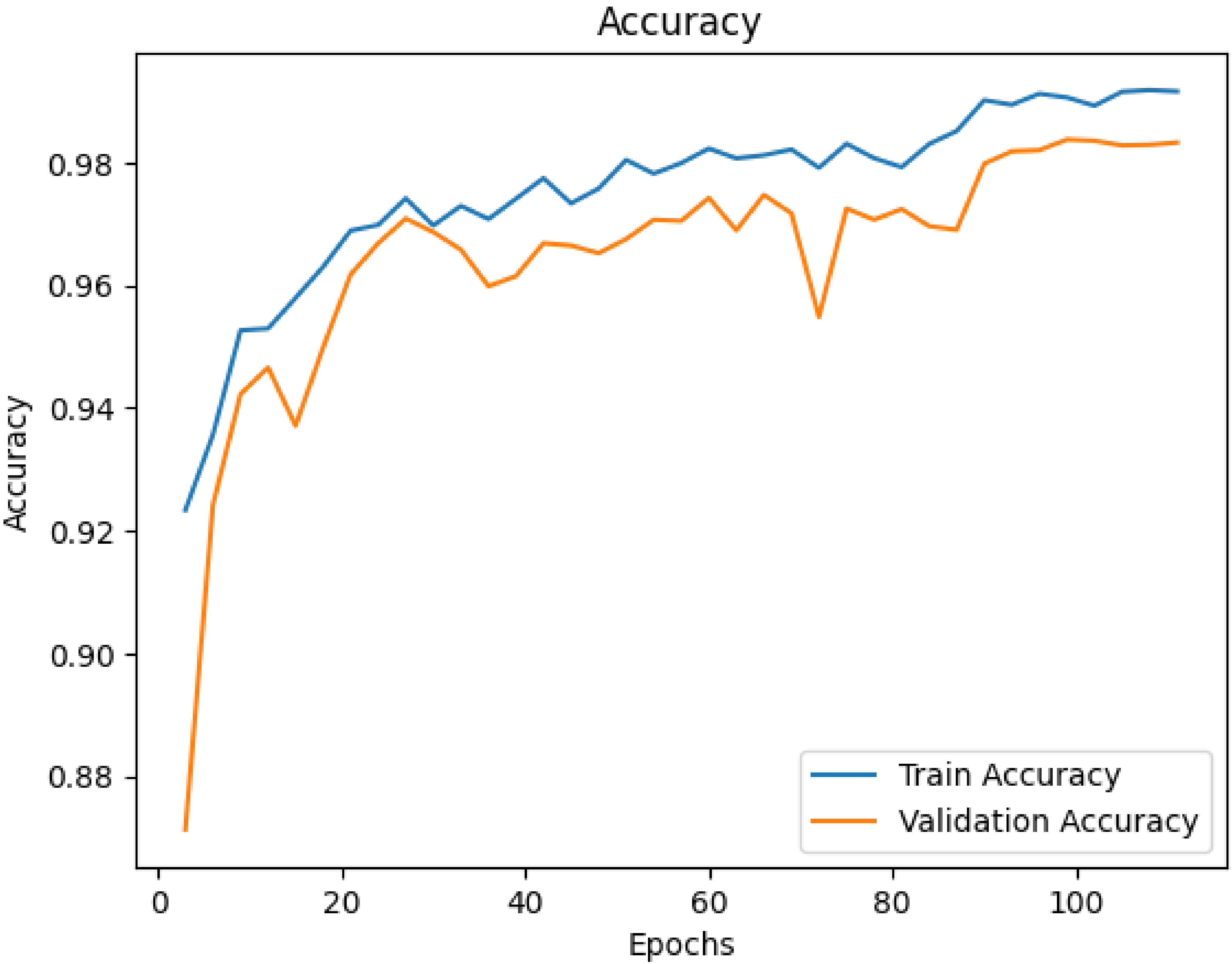

Train Loss
0.25
0.20
0.15
0.10
0.05
Loss
0
20
40
60
80
100
Epochs

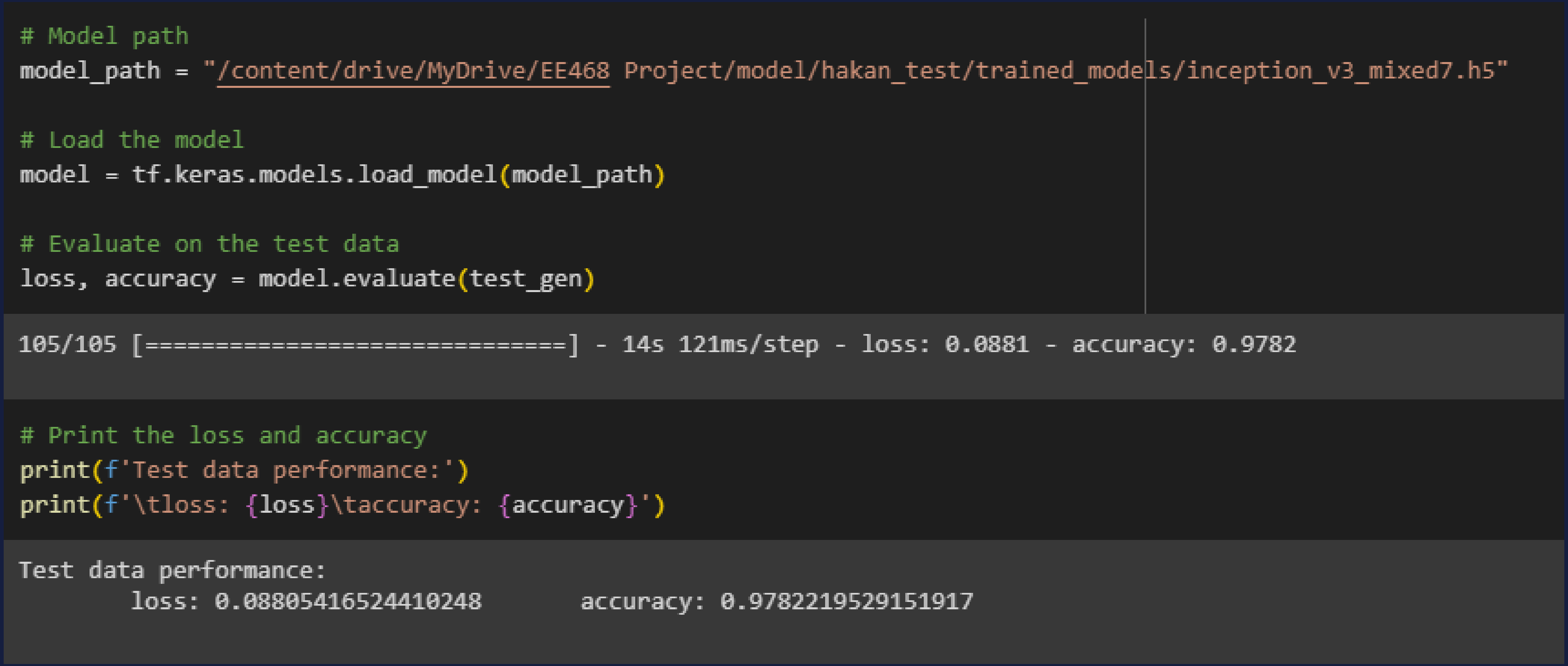

Train Accuracy
Accuracy
Epochs
0.92
0.93
0.94
0.95
0.96
0.97
0.98
0.99
0
20
40
60
80
100


Validation Loss
Loss
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
Epochs
0
20
40
60
80
100


Validation Accuracy
Accuracy
0.88
0.90
0.92
0.94
0.96
0.98
Epochs
0
20
40
60
80
100


Loss
Train Loss
Validation Loss
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
Loss
0
20
40
60
80
100
Epochs


Accuracy
Accuracy
0.88
0.90
0.92
0.94
0.96
0.98
0
20
40
60
80
100
Epochs
Train Accuracy
Validation Accuracy



```
# Model path
model_path = "/content/drive/MyDrive/EE468 Project/model/hakan_test/trained_models/inception_v3_mixed7.h5"

# Load the model
model = tf.keras.models.load_model(model_path)

# Evaluate on the test data
loss, accuracy = model.evaluate(test_gen)
```

```
105/105 [==============================] - 14s 121ms/step - loss: 0.0881 - accuracy: 0.9782
```

```
# Print the loss and accuracy
print(f'Test data performance:')
print(f'\tloss: {loss}\taccuracy: {accuracy}')
```

```
Test data performance:
	loss: 0.08805416524410248	accuracy: 0.9782219529151917
```

1- Photo by Riski Andriansyah on Unsplash

2- Image by Picsart AI Image generator on Picsart

3- C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

4- C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 2818-2826, doi: 10.1109/CVPR.2016.308.