Workshop

Machine Learning / Deep Learning menggunakan Google colab

Sesi II

10 April 2021

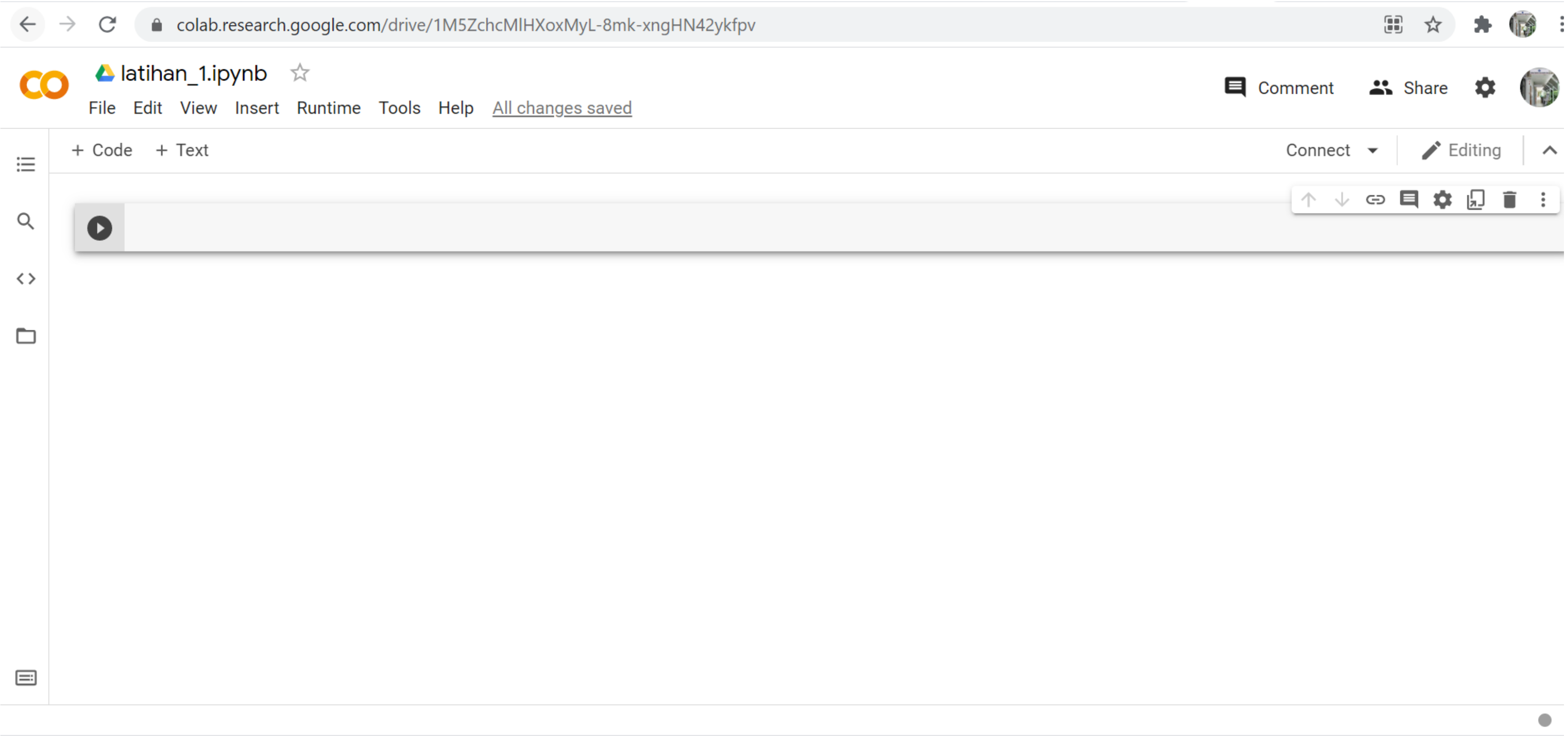
Agenda

- Tools
- Mengenal Google Colab
- Hands On

Tools

- Python
- Library
  - Keras
  - Tensorflow

https://colab.research.google.com/



Tutorial

- https://colab.research.google.com/github/lexfridman/mit-deep-learning/blob/master/tutorial_deep_learning_basics/deep_learning_basics.ipynb#scrollTo=lLek82SqrGA_

MNIST Dataset

- Merupakan dataset tulisan tangan angka
- Sebagai dataset untuk benchmarking metode machine learning / deep learning
- http://yann.lecun.com/exdb/mnist/

THE MNIST DATABASE

of handwritten digits

Yann LeCun, Courant Institute, NYU
Corinna Cortes, Google Labs, New York
Christopher J.C. Burges, Microsoft Research, Redmond

Please refrain from accessing these files from automated scripts with high frequency. Make copies!`

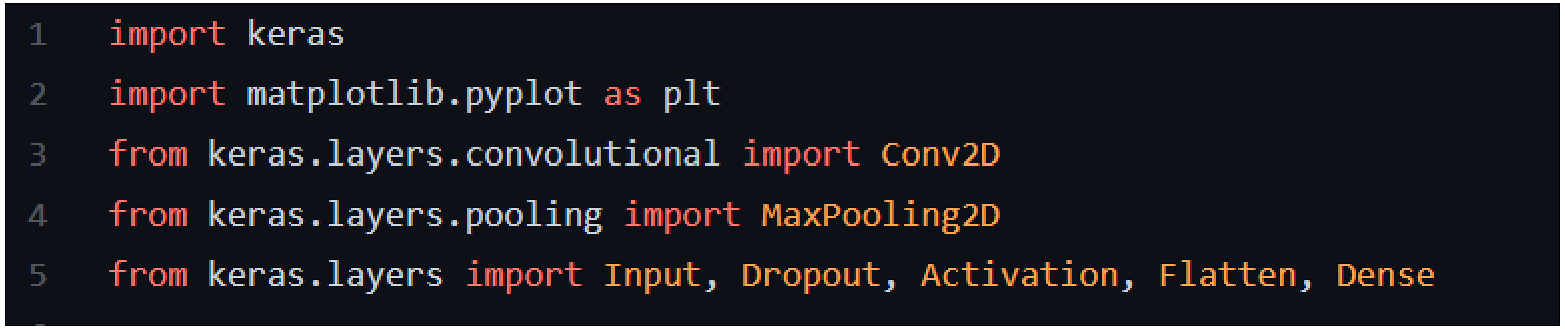
The MNIST database of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

It is a good database for people ng.

Repository

- https://github.com/totoharyantoui/polindra-workshop

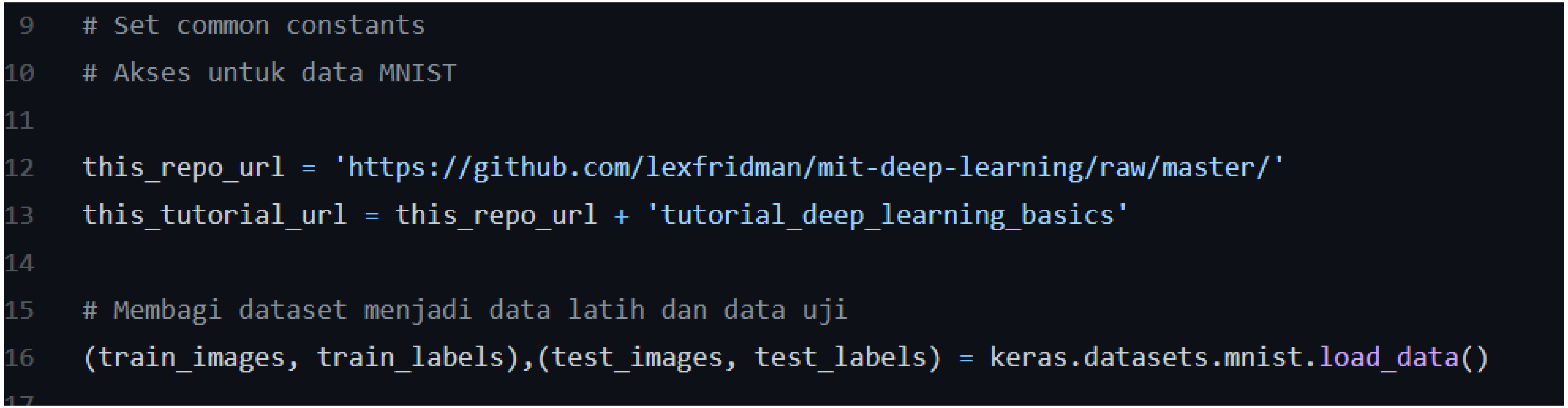
Import Library



```python
import keras
import matplotlib.pyplot as plt
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
from keras.layers import Input, Dropout, Activation, Flatten, Dense
```

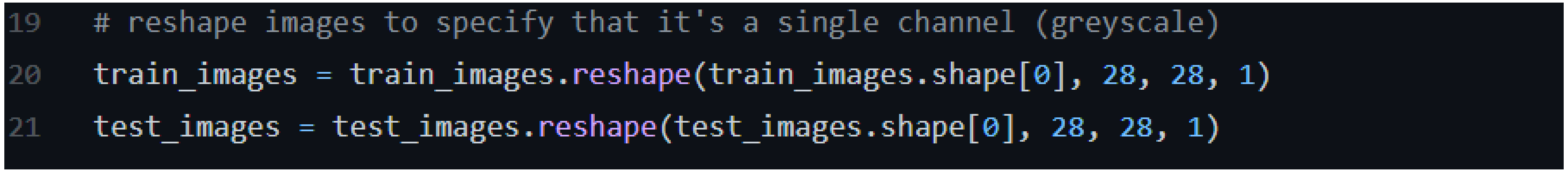
Library di atas umumnya digunakan untuk melakukan training pada arsitektur deep learning

Akses data MNIST



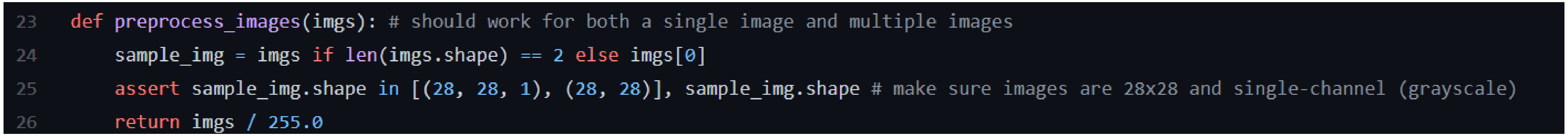
```
# Set common constants
# Akses untuk data MNIST

this_repo_url = 'https://github.com/lexfridman/mit-deep-learning/raw/master/'
this_tutorial_url = this_repo_url + 'tutorial_deep_learning_basics'

# Membagi dataset menjadi data latih dan data uji
(train_images, train_labels),(test_images, test_labels) = keras.datasets.mnist.load_data()
```

Augmentasi

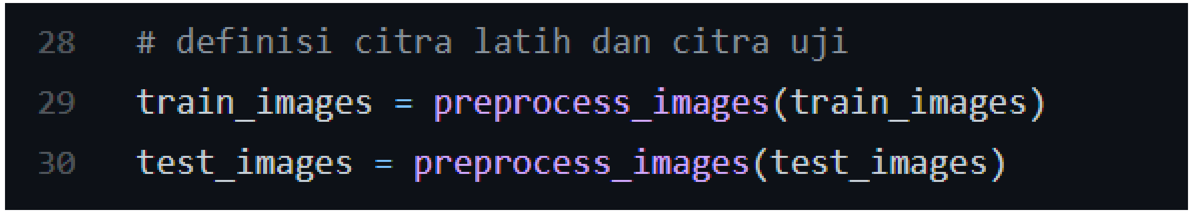


```python
# reshape images to specify that it's a single channel (greyscale)
train_images = train_images.reshape(train_images.shape[0], 28, 28, 1)
test_images = test_images.reshape(test_images.shape[0], 28, 28, 1)
```

Fungsi untuk pre-proses

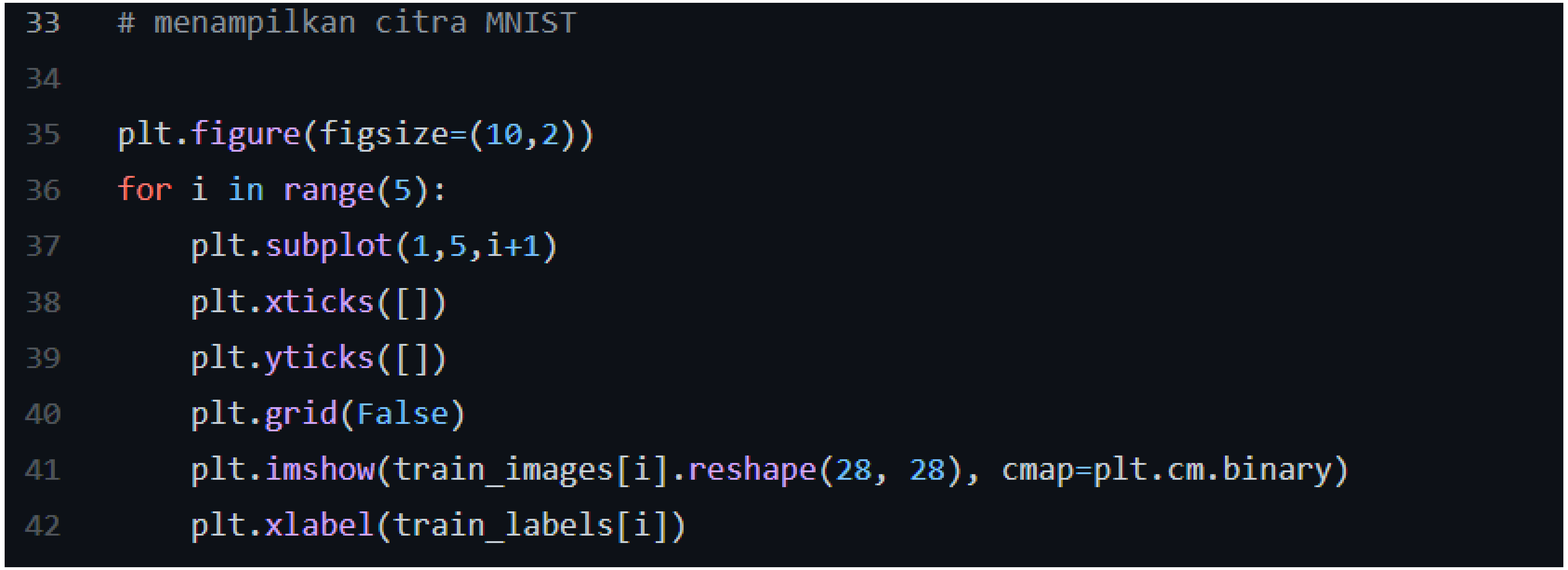


```python
def preprocess_images(imgs): # should work for both a single image and multiple images
    sample_img = imgs if len(imgs.shape) == 2 else imgs[0]
    assert sample_img.shape in [(28, 28, 1), (28, 28)], sample_img.shape # make sure images are 28x28 and single-channel (grayscale)
    return imgs / 255.0
```



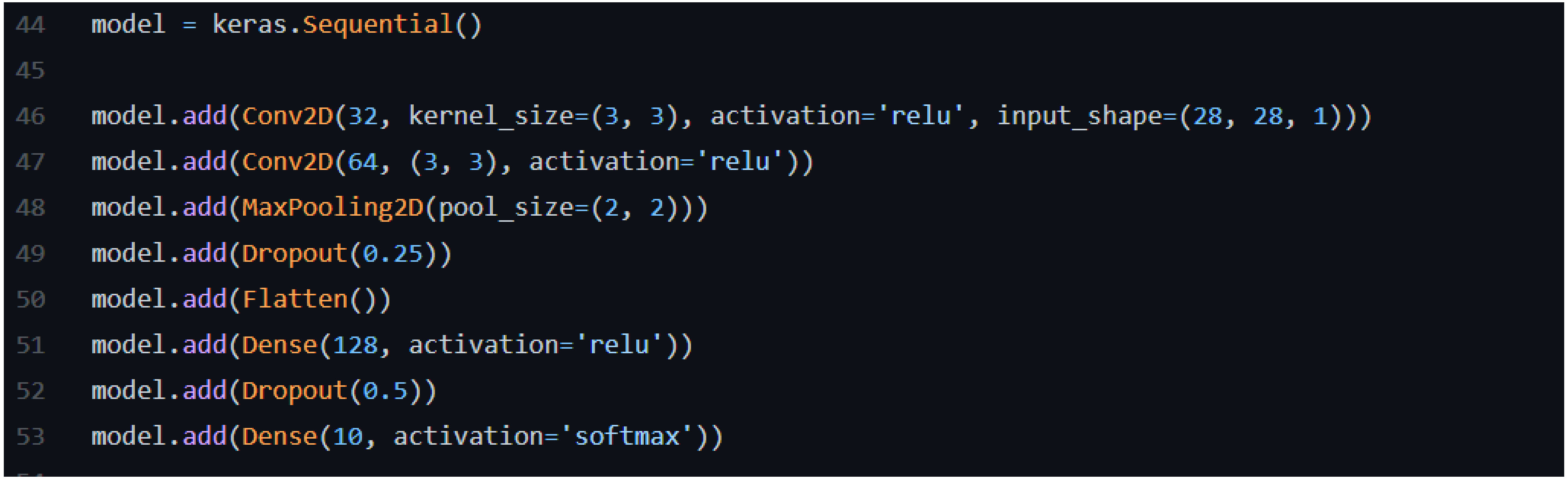
```python
# definisi citra latih dan citra uji
train_images = preprocess_images(train_images)
test_images = preprocess_images(test_images)
```

Menampilkan Citra



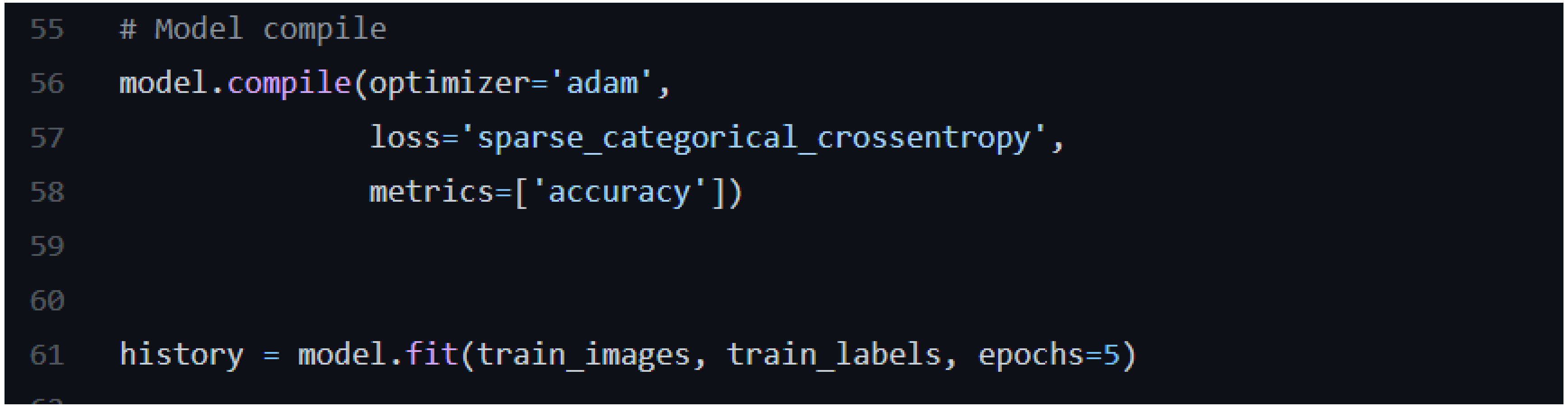
```
# menampilkan citra MNIST

plt.figure(figsize=(10,2))
for i in range(5):
    plt.subplot(1,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(train_images[i].reshape(28, 28), cmap=plt.cm.binary)
    plt.xlabel(train_labels[i])
```

Perancangan Arsitektur Deep Learning



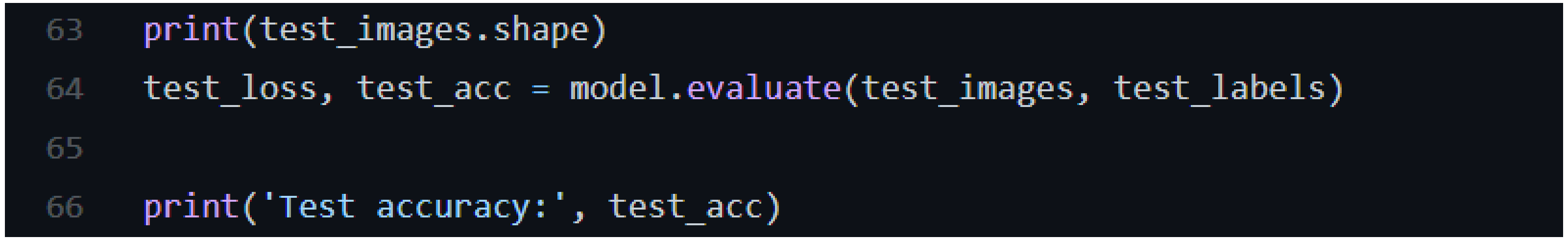
```
model = keras.Sequential()

model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
```

Kompilasi dan Proses Training



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


history = model.fit(train_images, train_labels, epochs=5)
```

Pengujian (Testing)



```python
print(test_images.shape)
test_loss, test_acc = model.evaluate(test_images, test_labels)

print('Test accuracy:', test_acc)
```

Mari Berlatih