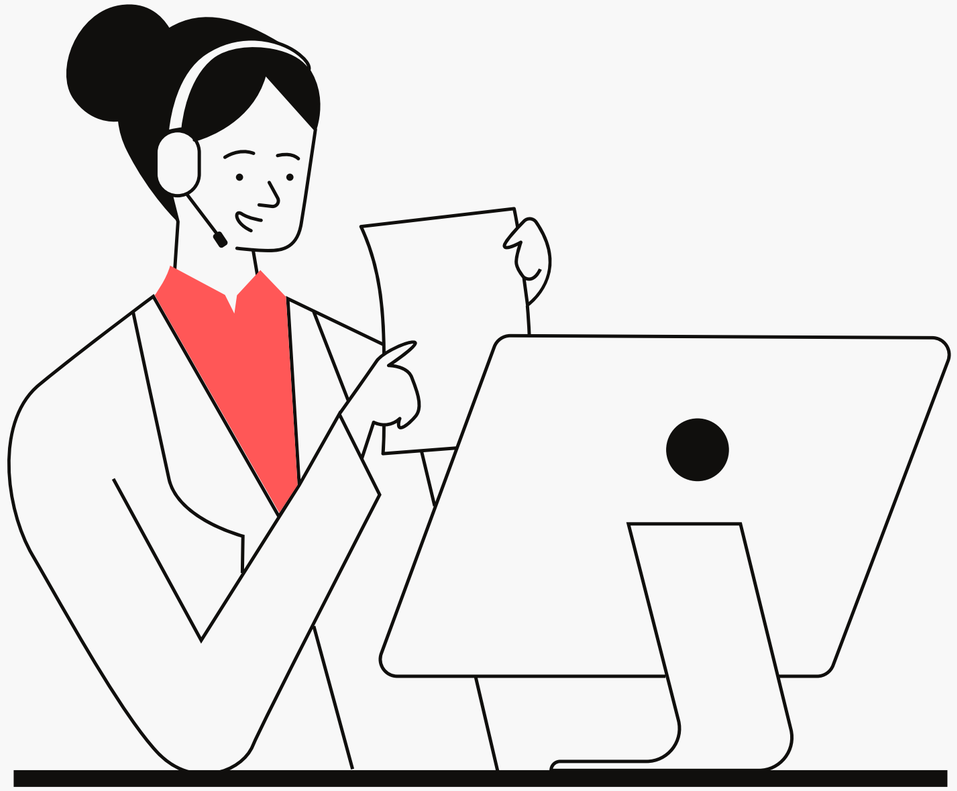
Laporan Presentasi UAS

Analisis Model CNN untuk klasifikasi MNIST Angka

Dheandra Alfarrelwijaya
1103213048

01

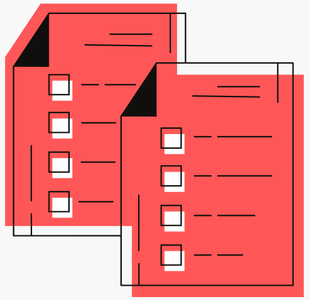


Database MNIST

MNIST (Modified National Institute of Standards and Technology) adalah dataset yang umum digunakan dalam pelatihan dan pengujian model pengenalan pola, terutama dalam pengenalan tulisan tangan digit (0-9). Dataset MNIST pun seringkali digunakan untuk menguji salahsatu kemampuan algoritma Machine Learning dan deep learning.

Tujuan

Tujuan dari laporan ini adalah untuk mempresentasikan hasil penerapan Neural Network dalam klasifikasi digit menggunakan database MNIST. dengan mengharapkan dengan menghasilkan mencapai akurasi diatas 90, dan menampilkan confusion matrix untuk melihat detail kinerja pada tiap class



Apa itu MNIST?

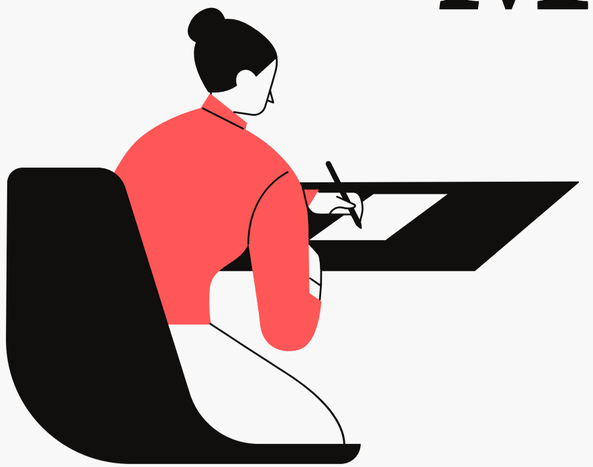
MNIST adalah dataset yang terdiri dari 70,000 gambar digit tulisan tangan. Dataset ini terbagi menjadi 60,000 gambar untuk pelatihan dan 10,000 gambar untuk pengujian. Setiap gambar berukuran 28x28 piksel dalam format grayscale. Label dari gambar ini merupakan angka dari 0 sampai 9,

Apa itu CNN?

Convolutional Neural Network (CNN) adalah salah satu jenis neural network yang biasa digunakan pada data image. CNN bisa digunakan untuk mendeteksi dan mengenali object pada sebuah image.

Neural Network (Jaringan Saraf Tiruan) adalah salah satu teknik dalam bidang kecerdasan buatan yang terinspirasi oleh cara kerja otak manusia. Teknik ini meniru jaringan neuron di otak, dengan harapan dapat menangkap pola kompleks dan melakukan tugas-tugas yang sulit diprogram secara eksplisit.

Apa Itu Confusion Matrix?



Confusion matrix adalah alat evaluasi kinerja yang penting dalam klasifikasi data mining, memberikan gambaran menyeluruh tentang hasil prediksi model.

Mengimport Library yang akan digunakan

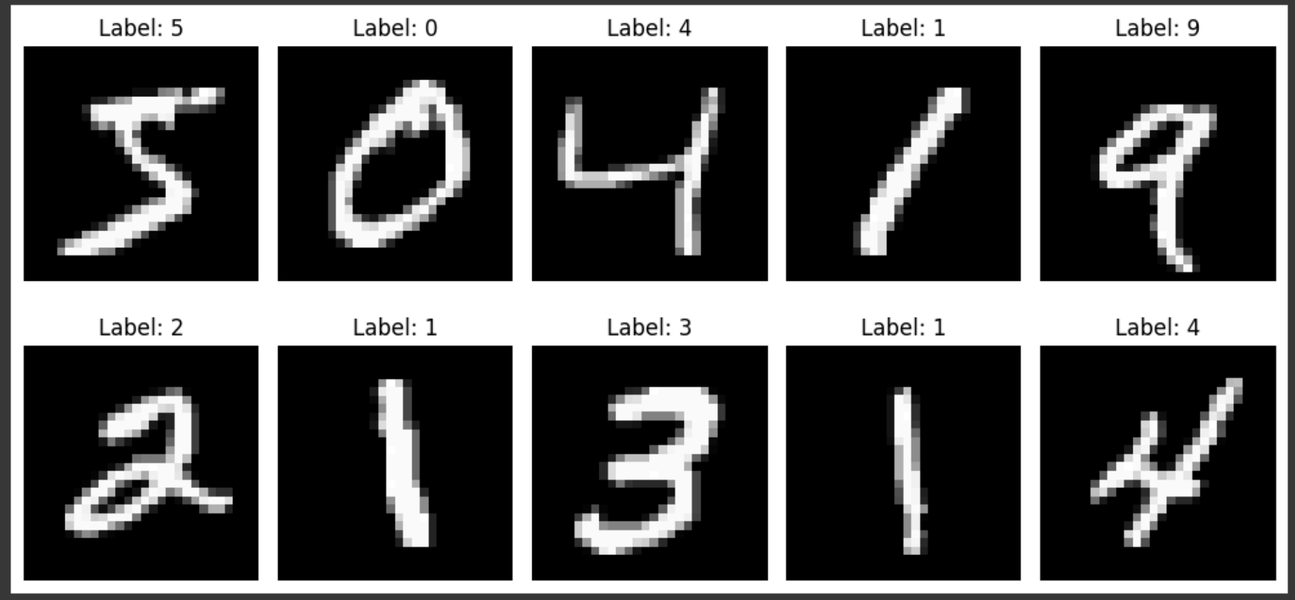
```
[11] import tensorflow as tf
     from tensorflow.keras import layers, models, datasets
     import matplotlib.pyplot as plt
     from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
     import numpy as np
```

Load Dataset Dan Preproccess

```
[22] # Melakukan Preprosses dan melakukan load pada dataset mnist
     (x_train, y_train), (x_test, y_test)  = datasets.mnist.load_data()
     x_train, x_test = x_train / 255.0, x_test / 255.0
```

Menampilkan isi gambar dataset

```python
# Menampilkan gambar yang berada dalam dataset MNIST
plt.figure(figsize=(10, 5))
for i in range(10):  # Menampilkan 10 gambar (bisa dirubah)
    plt.subplot(2, 5, i + 1)
    plt.imshow(train_images[i], cmap='gray')
    plt.title(f'Label: {train_labels[i]}')
    plt.axis('off')
plt.tight_layout()
plt.show()
```



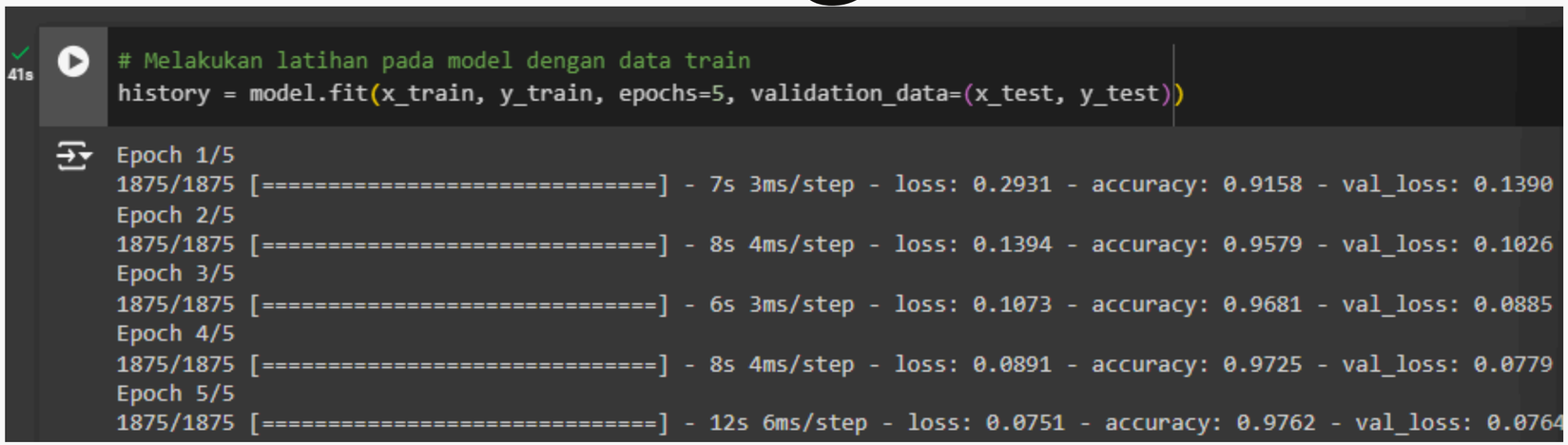
Menginput Model CNN

```
[24] # Membangun model neural network
     model = models.Sequential([
         layers.Flatten(input_shape=(28, 28)),  # Mengubah gambar 2D menjadi 1D
         layers.Dense(128, activation='relu'),   # Lapisan tersembunyi dengan 128 neuron
         layers.Dropout(0.2),  # Lapisan tersembunyi dengan 64 neuron
         layers.Dense(10, activation='softmax')  # Lapisan output dengan 10 neuron (klasifikasi 10 kelas)
     ])
```

Melakukan Kompilasi

```
[25] # Melakukan kompilasi pada data
     model.compile(optimizer='adam',
                   loss='sparse_categorical_crossentropy',
                   metrics=['accuracy'])
```

Training data



Mengevaluasi Dataset

```
[27] # Mengevaluasi Model dengan dataset MNIST
     test_loss, test_acc = model.evaluate(x_test, y_test, verbose=2)
     print('\nTest accuracy:', test_acc)
```

```
313/313 - 0s - loss: 0.0764 - accuracy: 0.9785 - 430ms/epoch - 1ms/step

Test accuracy: 0.9785000085830688
```

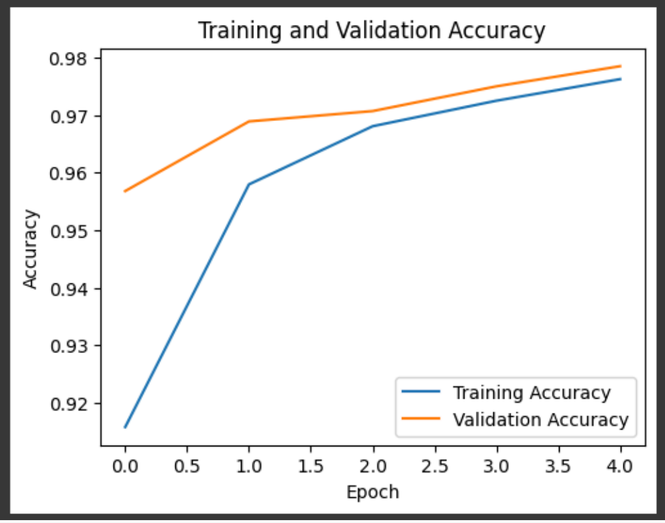
Memprediksi data uji

```
[28] # Prediksi pada data uji
     y_pred = np.argmax(model.predict(x_test), axis=1)
```

```
313/313 [==============================] - 1s 2ms/step
```

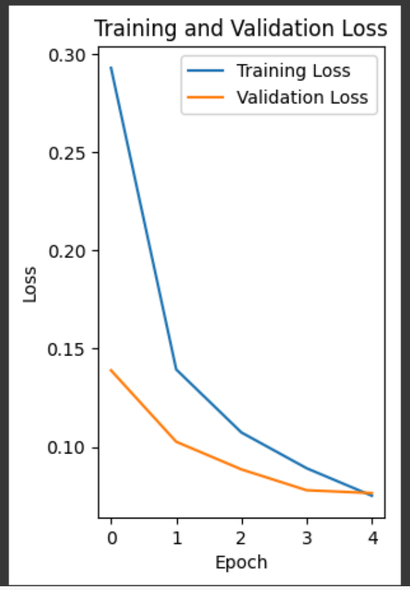
Grafik Akurasi

```python
# Menampilkan grafik akurasi
plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')
plt.show()
```



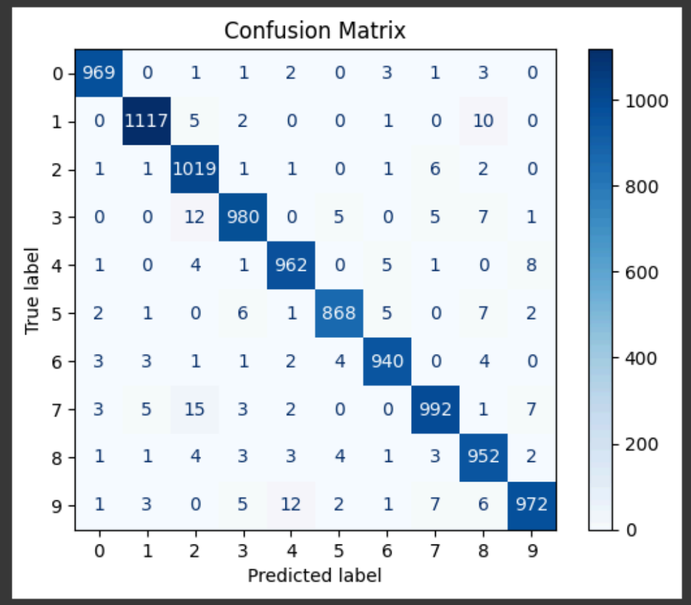
Grafik Loss

```python
# Menampilkan grafik akurasi
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')

plt.show()
```



Confusion Matrix

```python
# Membuat confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Menampilkan confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=np.arange(10))
disp.plot(cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.show()
```



Hasil Dan Kesimpulan

Model dilatih oleh augmentasi data 5 epoch

Akurasi data uji menampilkan mencapai 98% yang artinya hampir sempurna

Teknik ini membantu untuk mencegah overfitting dan membantu meningkankan generalisasi model

Model neural network yang dibangun dan dilatih pada dataset MNIST menunjukkan kinerja yang baik dalam mengenali angka tulisan tangan. Dengan akurasi yang tinggi pada data uji, model ini mampu mengklasifikasikan sebagian besar angka dengan benar

Thank You