A screen shot of a cell phone

AI-generated content may be incorrect.

3223600019

Muhammad Bimo Fachrizky

Membuat Aplikasi Deep Learning tensorflow

Senin, 28 April 2025

**Praktikum Sistem Cerdas**

**Praktikum 9**

Membuat Aplikasi Deep Learning tensorflow

1. Tujuan Pembelajaran

* Mahasiswa dapat memahami dan menjelaskan konsep Deep Learning
* Mahasiswa dapat menjelaskan model Deep Learning
* Mahasiswa dapat membuat aplikasi Deep Learing tensorflow

Software yang di perlukan

* Microsoft Visual C++
* PyCharm

1. Langkah percobaan
2. Simple\_Regression

import tensorflow as tf

import numpy as np

import matplotlib.pyplot as plt

from tensorflow import keras

def plot\_data(x\_data, y\_data):

fig, ax = plt.subplots()

ax.plot(x\_data, y\_data, 'ro')

plt.pause(1)

return None

if \_\_name\_\_ == "\_\_main\_\_":

# Define and compile the neural networks

model = tf.keras.Sequential([

keras.layers.Dense(units=1, input\_shape=[1])

])

model.compile(optimizer="sgd", loss="mean\_squared\_error")

# Providing the data

xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0])

ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0]) # y = 2x - 1

# Plotting data will help us to understand faster

# visually about the data

plot\_data(xs, ys)

# Train the neural network

model.fit(xs, ys, epochs=500)

# Predict using the model

print(model.predict(np.array([10.0])))

1. Fashion\_Classifier

import numpy as np

import matplotlib.pyplot as plt

import matplotlib

import tensorflow as tf

if \_\_name\_\_ == "\_\_main\_\_":

# Load the dataset

mnist = tf.keras.datasets.fashion\_mnist

(training\_images, training\_labels), (test\_images, test\_labels) = mnist.load\_data()

# Plot an image from the dataset

plt.imshow(training\_images[0], cmap="Greys")

plt.pause(1) # Increase the pause time to give time for plotting

print(training\_labels[0])

print(training\_images[0])

# Normalizing image intensities to [0, 1]

training\_images = training\_images / 255.0

test\_images = test\_images / 255.0

# Create a model

model = tf.keras.models.Sequential([

tf.keras.layers.Flatten(),

tf.keras.layers.Dense(128, activation=tf.nn.relu),

tf.keras.layers.Dense(10, activation=tf.nn.softmax)

])

# Compile the model and train it to the dataset

model.compile(

optimizer=tf.optimizers.Adam(),

loss="sparse\_categorical\_crossentropy",

metrics=["accuracy"]

)

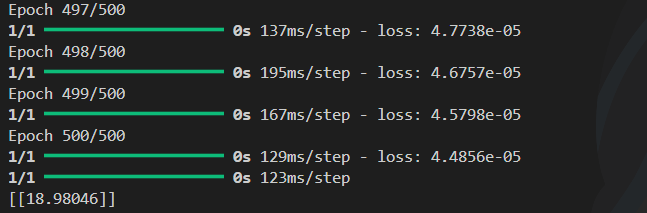
model.fit(training\_images, training\_labels, epochs=5)

model.evaluate(test\_images, test\_labels)

1. Hasil Percobaan
2. Simple\_Regression

A screenshot of a computer screen

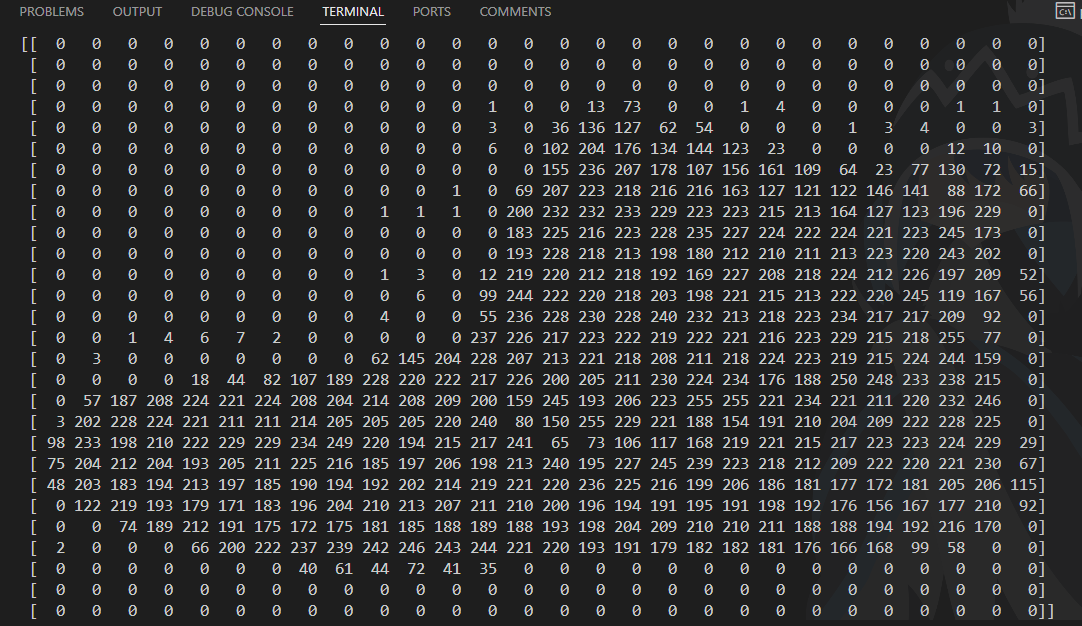
AI-generated content may be incorrect.



1. Fashion\_Classifier

A computer screen shot of a shoe

AI-generated content may be incorrect.



1. Analisa

Praktikum ini menggunakan program yang pertama yaitu Simple Regression dengan tensorflow. Program pertama merupakan implementasi dasar dari algoritma regresi linear menggunakan TensorFlow dan Keras. Tujuan utamanya adalah membuat sebuah model yang bisa memetakan hubungan antara input x dan output y, di mana pola yang digunakan adalah fungsi linear sederhana: y = 2x - 1. Pada awal program, dilakukan proses impor modul yang diperlukan seperti tensorflow, numpy, dan matplotlib.pyplot, diikuti oleh definisi fungsi plot\_data() untuk menampilkan grafik scatter dari data input dan output dengan warna merah ('ro'). Pada bagian utama program (if \_\_name\_\_ == "\_\_main\_\_":), model neural network dibangun dengan menggunakan Sequential, yang terdiri dari satu lapisan Dense dengan units=1 dan input\_shape=[1]. Ini berarti model hanya memiliki satu neuron yang menerima satu fitur input—sesuai dengan struktur regresi linier. Model dikompilasi menggunakan optimizer sgd (Stochastic Gradient Descent) yang cocok untuk permasalahan linier sederhana, dan fungsi loss mean\_squared\_error yang umum digunakan pada regresi. Setelah menampilkan plot dari data, model dilatih (fit) sebanyak 500 epoch, yang memungkinkan model mempelajari pola dengan akurasi tinggi. Setelah pelatihan selesai, model digunakan untuk memprediksi hasil output dari input 10.0.

Program kedua merupakan implementasi dari klasifikasi gambar menggunakan dataset Fashion MNIST, yang terdiri dari 70.000 gambar grayscale berukuran 28x28 piksel, masing-masing merepresentasikan 10 jenis pakaian yang berbeda. Program dimulai dengan mengimpor library penting: numpy, matplotlib.pyplot, dan tensorflow. Di dalam blok utama, data dibagi menjadi dua bagian: training\_images dan training\_labels untuk pelatihan, serta test\_images dan test\_labels untuk pengujian. Data diambil menggunakan fungsi load\_data() dari tf.keras.datasets.fashion\_mnist. Langkah penting berikutnya adalah normalisasi data, di mana seluruh nilai piksel dibagi dengan 255.0 sehingga semua nilainya berada dalam rentang [0, 1]. Ini adalah langkah penting untuk mempercepat konvergensi dan membuat pelatihan lebih stabil. Model neural network yang dibangun memiliki tiga lapisan: Flatten() untuk mengubah input 2D menjadi 1D, Dense(128, activation=tf.nn.relu) sebagai hidden layer untuk mendeteksi fitur kompleks, dan Dense(10, activation=tf.nn.softmax) sebagai output layer dengan 10 neuron yang mewakili masing-masing kelas pakaian. Model dilatih (fit) selama 5 epoch pada data pelatihan, dan setelah pelatihan selesai, model dievaluasi (evaluate) terhadap data pengujian untuk menghitung akurasi dan loss pada data yang belum pernah dilihat sebelumnya.

1. Kesimpulan

TensorFlow dan Keras merupakan tools powerful dalam pengembangan model pembelajaran mesin, baik untuk kasus regresi sederhana maupun klasifikasi gambar kompleks. Program pertama menunjukkan bahwa model neural network bahkan dengan arsitektur paling dasar sekalipun (satu neuron) mampu mempelajari hubungan linear antara input dan output, dengan tingkat akurasi yang tinggi setelah melalui proses pelatihan berulang (epoch). Ini membuktikan kemampuan jaringan saraf dalam menyelesaikan masalah regresi secara efisien. Sementara itu, program kedua memperlihatkan proses pembuatan sistem klasifikasi gambar dengan dataset Fashion MNIST, yang mencakup langkah-langkah penting seperti normalisasi data, pembuatan arsitektur jaringan dengan hidden layer, serta evaluasi performa model terhadap data uji. Dengan menerapkan prinsip-prinsip deep learning yang tepat, model mampu mencapai tingkat akurasi tinggi dalam mengenali objek visual dari gambar 2D. Kedua program juga mempertegas pentingnya pemahaman preprocessing data dan pemilihan fungsi aktivasi, loss, dan optimizer yang sesuai dengan jenis permasalahan.