**Міністерство освіти і науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

Комп‘ютерного практикуму № 3 з дисципліни

«Програмні засоби проектування та реалізації нейромережевих систем»

**«Нейронної мережі прямого розповсюдження для розпізнавання зображення»**

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-01 Галько М.В.*

**Перевірив(ла)**

(прізвище, ім'я, по батькові)

*Шимкович В. М.*

Київ 2022

**Завдання:** Написати програму що реалізує нейронну мережу прямого розповсюдження для розпізнавання рукописних цифр.

**Виконання:**

Дані:

hidden\_layers = [50, 50, 50]  
num\_epochs, numbers = 10, 9  
batch\_size = 32  
(x\_train, y\_train), (x\_test, y\_test) = tf.keras.datasets.mnist.load\_data()  
learning\_rate = tf.keras.optimizers.schedules.ExponentialDecay(  
 initial\_learning\_rate=0.001,  
 decay\_steps=int(len(x\_train) / batch\_size),  
 decay\_rate=(1 / 10) \*\* (1 / num\_epochs)  
)

main.py

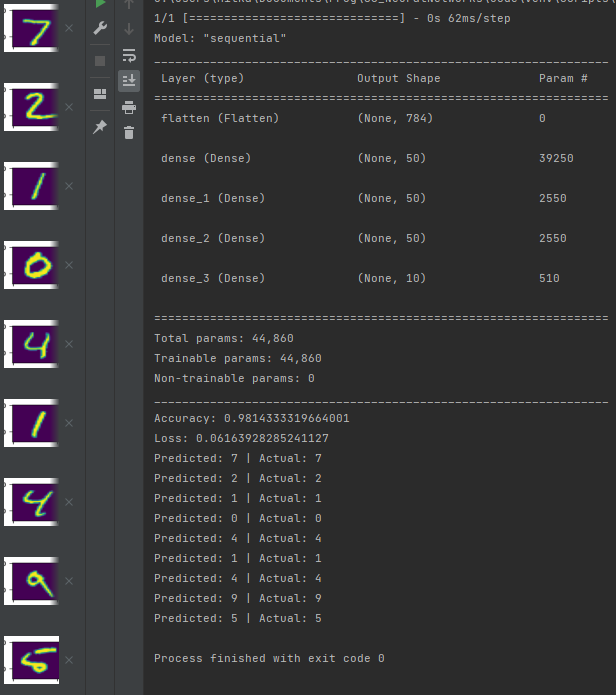
import matplotlib.pyplot as plt  
import tensorflow as tf  
from model\_init import get\_model, compile\_model

if \_\_name\_\_ == '\_\_main\_\_':  
 model = get\_model(hidden\_layers)  
 compile\_model(model, learning\_rate)  
 history = model.fit(x\_train, y\_train, epochs=num\_epochs, validation\_data=(x\_test, y\_test), verbose=0)  
 trained\_results, right\_results, = [y.argmax() for y in model.predict(x\_test[:numbers])], y\_test[:numbers]  
  
 model.summary()  
 print(f'Accuracy: {history.history["accuracy"][-1]}')  
 print(f'Loss: {history.history["loss"][-1]}')  
 for i in range(numbers):  
 print(f'Predicted: {trained\_results[i]} | Actual: {right\_results[i]}')  
 plt.figure(figsize=(1, 1))  
 plt.imshow(x\_test[i])  
 plt.show()

model\_init.py

import tensorflow as tf  
  
  
def get\_model(hidden\_layers):  
 layers = [tf.keras.layers.Flatten()]  
 for layer in hidden\_layers:  
 layers.append(tf.keras.layers.Dense(layer, activation='relu'))  
 layers.append(tf.keras.layers.Dense(10, activation='softmax'))  
 return tf.keras.Sequential(layers)  
def compile\_model(model, learning\_rate=0.001):  
 model.compile(  
 loss=tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=False),  
 optimizer=tf.keras.optimizers.Adam(learning\_rate=learning\_rate),  
 metrics=['accuracy'],  
 )

**Вивід:**

****

**Висновок:**

В цілому, модель нейронної мережі прямого розповсюдження успішно пройшла тренування та оцінку на наборі даних MNIST. За результатами виводу можна зробити такі висновки:

* Модель досягла точності приблизно 98.14% та втрати 0.0616 на тестових даних.
* Передбачені значення моделі для підмножини тестових даних відповідають фактичним цифрам, що свідчить про успішність тренування та оцінки.
* Графічне відображення кожної цифрової картинки зображене з використанням plt.imshow(x\_test[i]), що показує відповідну цифрову картинку з тестового набору.

Отже, можна стверджувати, що модель успішно виконує класифікацію на наборі даних MNIST з високою точністю.