

**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 нейромережових систем»

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

**Виконав(ла)**

*ІП-01 Галько М.В.*

(шифр, прізвище, ім'я, по батькові)

**Перевірів(ла)**

*Шимкович В. М.*

(прізвище, ім'я, по батькові)

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()
```

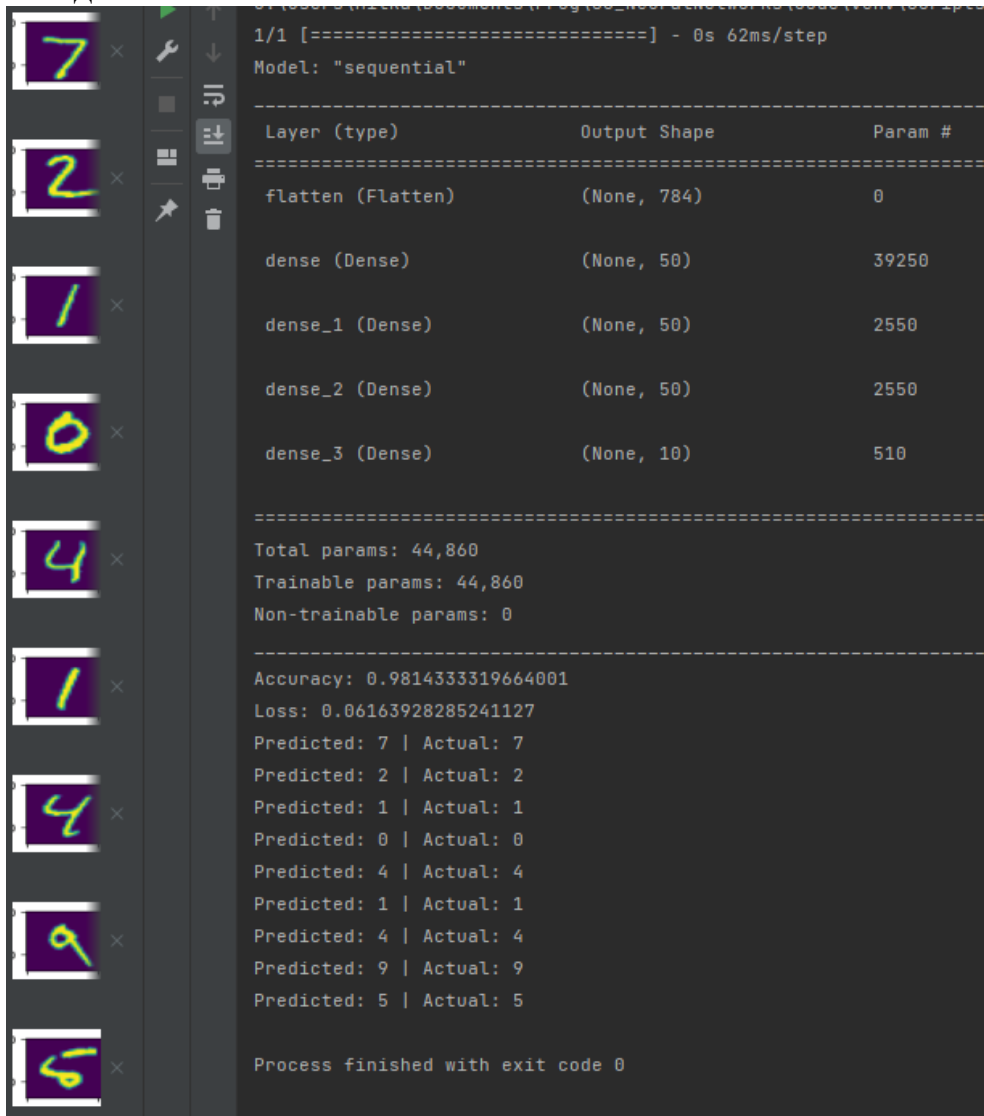
### model\_init.py

```
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:



```
1/1 [=====] - 0s 62ms/step
Model: "sequential"

-----
Layer (type)              Output Shape              Param #
-----
flatten (Flatten)         (None, 784)               0
dense (Dense)              (None, 50)               39250
dense_1 (Dense)            (None, 50)               2550
dense_2 (Dense)            (None, 50)               2550
dense_3 (Dense)            (None, 10)               510
-----

Total params: 44,860
Trainable params: 44,860
Non-trainable params: 0

-----
Accuracy: 0.9814333319664001
Loss: 0.06163928285241127
Predicted: 7 | Actual: 7
Predicted: 2 | Actual: 2
Predicted: 1 | Actual: 1
Predicted: 0 | Actual: 0
Predicted: 4 | Actual: 4
Predicted: 1 | Actual: 1
Predicted: 4 | Actual: 4
Predicted: 9 | Actual: 9
Predicted: 5 | Actual: 5

Process finished with exit code 0
```

## 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 картини 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.