Міністерство освіти і науки

України Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЗВІТ

лабораторної роботи №3

з курсу «Програмні засоби проєктування і реалізації неромережевих систем»

Тема: «Нейронні мережі прямого розповсюдження для розпізнавання зображення»

Перевірив:

Шимкович В. М.

Виконав:

Студент Гр. ІП-01 Шпилька В.С.

Київ 2023

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

Модель:

```python
import tensorflow as tf

def Model(hidden_neurons):
    model = tf.keras.Sequential()
    model.add(tf.keras.Input(shape=(28,28)))
    model.add(tf.keras.layers.Flatten())
    for neurons in hidden_neurons:
        model.add(tf.keras.layers.Dense(neurons, activation='relu'))
    model.add(tf.keras.layers.Dense(10, activation='softmax'))

    return model
```

Архітектура:

```
model = tf.keras.models.load_model("Artifacts/Models/v1/Model/tf")
model.summary()
```

[14] ✓ 0.5s

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 70)                54950

 dense_1 (Dense)             (None, 70)                4970

 dense_2 (Dense)             (None, 70)                4970

 dense_3 (Dense)             (None, 10)                710

=================================================================
Total params: 65,600
Trainable params: 65,600
Non-trainable params: 0
_________________________________________________________________
```

Функція тренування:

```
def train(version,
          hidden_neurons = settings.HIDDEN_NEURONS,
          save_folder = settings.SAVE_FOLDER,
          epochs = settings.EPOCHS,
          batch_size = settings.BATCH_SIZE,
          lr = settings.DEFAULT_LR):
    (x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()
    model = Model(hidden_neurons=hidden_neurons)

    #values for schedules
    initial_learning_rate = 10**(-3)
    final_learning_rate = 10**(-5)
    learning_rate_decay_factor = (final_learning_rate / initial_learning_rate)**(1/epochs)
    steps_per_epoch = int(len(x_train)/batch_size)

    learning_rate = lr
    if(lr == -1):
        learning_rate = tf.keras.optimizers.schedules.ExponentialDecay(
                        initial_learning_rate=initial_learning_rate,
                        decay_steps=steps_per_epoch,
                        decay_rate=learning_rate_decay_factor
                    )

    model.compile(loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'],
                  optimizer=tf.keras.optimizers.Adam(learning_rate=learning_rate))

    path_to_save = save_folder + '/' + version + '/'
```

```
    checkpoint_dir = path_to_save + "Checkpoints/"
    checkpoint_path = checkpoint_dir + "cp-{epoch:04d}.ckpt"
    checkpoint = ModelCheckpoint(filepath=checkpoint_path,
                                 monitor='val_loss',
                                 verbose=1,
                                 save_weights_only = True,
                                 mode='auto')

    tf_path = path_to_save + "Model/tf"
    fullModelSave = ModelCheckpoint(filepath=tf_path,
                                 monitor='val_loss',
                                 verbose=1,
                                 save_best_only=True,
                                 mode='auto')


    log_dir = path_to_save + "Logs/"
    tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1)

    callbacks_list = [checkpoint, tensorboard_callback, fullModelSave]

    model.fit(
        x_train,
        y_train,
        batch_size,
        epochs = epochs,
        validation_data = (x_test, y_test),
        callbacks = callbacks_list,
        verbose = 1)
```

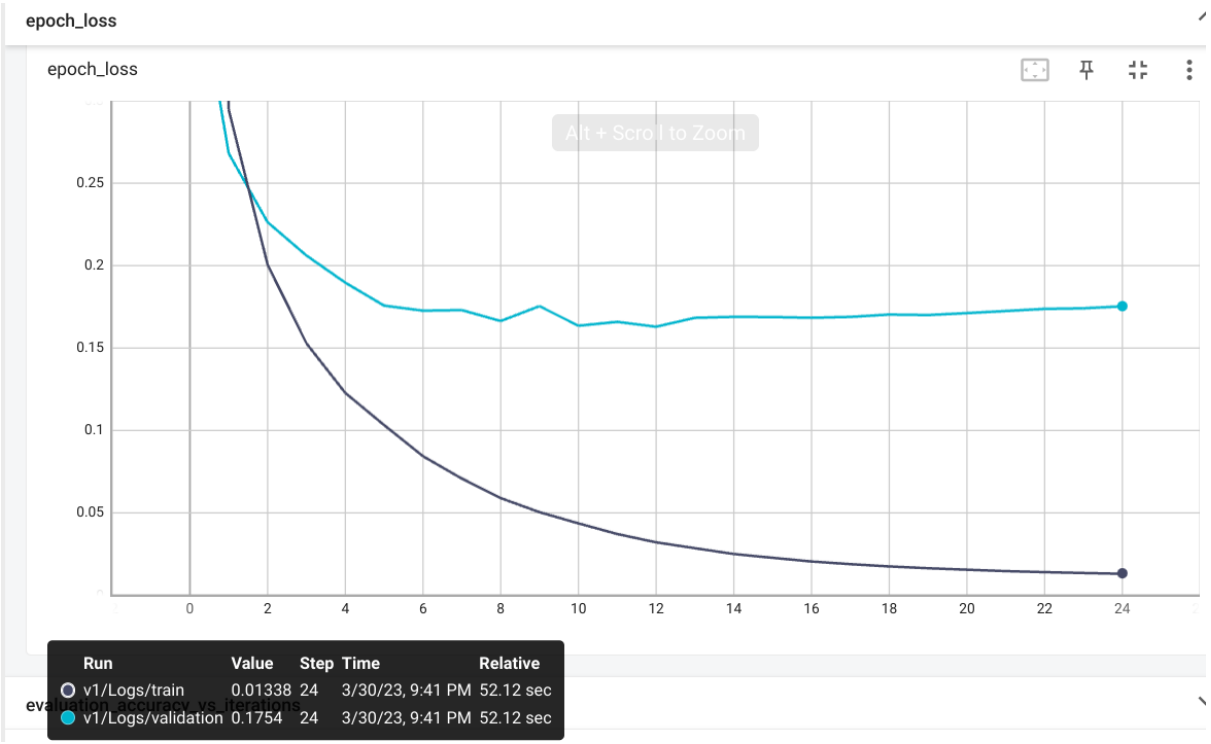
Гіперпараметри та константи:

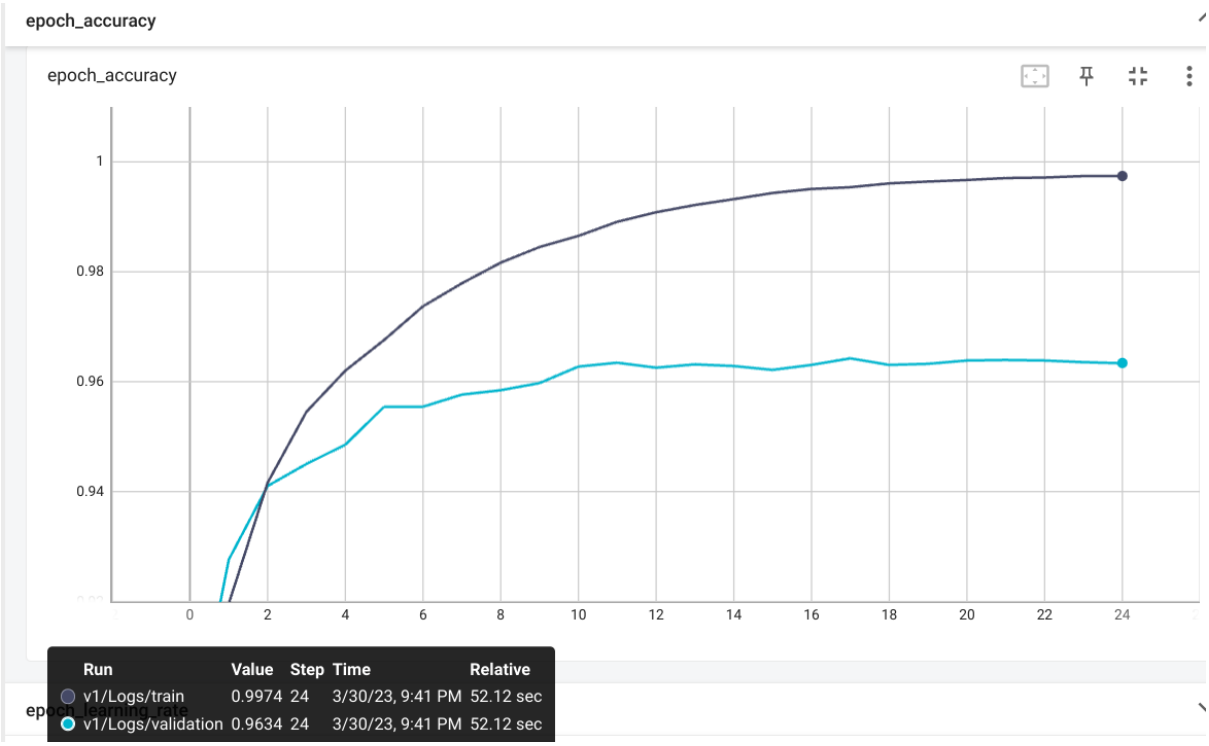
```
RANDOM_SEED = 42
BATCH_SIZE = 100
SAVE_FOLDER = './Artifacts/Models'
EPOCHS = 25
HIDDEN_NEURONS = [70, 70, 70]
```

В результаті тренування нейроної мережі було отримано точність в 96,26 відсотків:

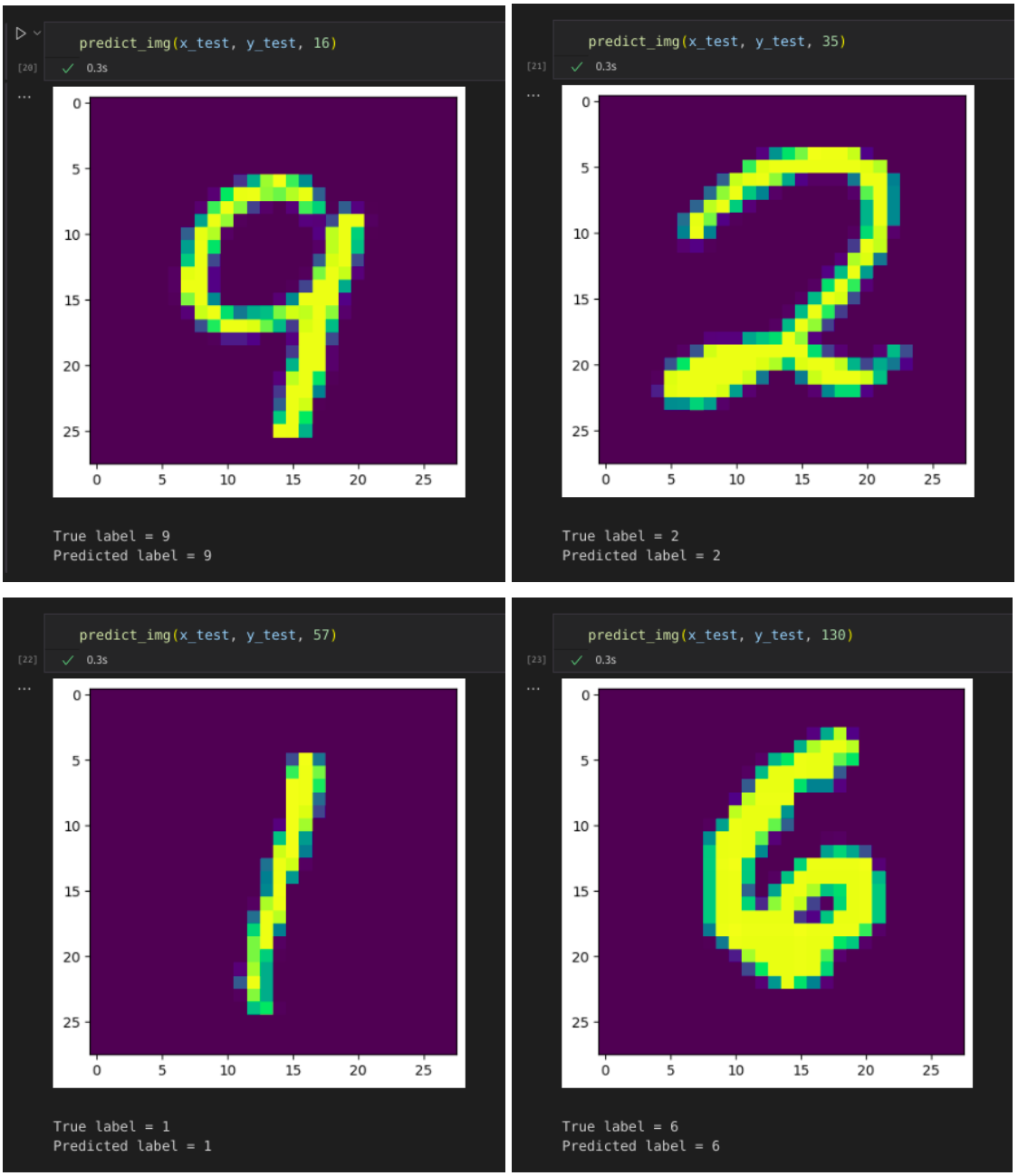
```
model.evaluate(x_test, y_test)
[7] ✓ 0.7s
... 313/313 [==============================] - 1s 2ms/step - loss: 0.1631 - accuracy: 0.9626

[0.16308008134365082, 0.9625999927520752]
```

Зміна loss і точності від епохи:





Приклади роботи:



Висновок:

В результаті виконання лабораторної роботи було побудовано нейрону мережу прямого розповсюдження для розпізнавання цифр з MNIST датасету. Мережа має 3 прихованих шари по 70 нейронів з функцією активації релу та останній шар з 10 нейронами та функцією активацією софтмакс. Всього 65600 параметрів. Для даного датасету дана нейрона мережа показала гарні результати, а саме точність в 96,25 відсотків. Це обумовлено маленьким розміром картинки(28 на 28 пікселів), картинка монохромна, відсутністю викидів та шумів. В більш складних ситуаціях, дана нейрона мережа навряд чи справиться также добре.