Ушаков Владимир Александрович, группа МПиКИ

Лабораторная работа № 1

Знакомство со средой Google Collab и библиотеками Tensorflow и Keras

**Цель работы**

Ознакомиться со средой Google Collab и библиотеками Tensorflow и Keras, а также изучить поведение нейронной сети при изменении различных гиперпараметров

**Задание**

1. Провести исследование «activation»

network.add(layers.Dense(512, activation='relu', input\_shape=(28 \* 28,)))

network.add(layers.Dense(10, activation='softmax'))

2. Провести исследование «optimizer», «loss», «metrics»

network.compile(optimizer='rmsprop',

 loss='categorical\_crossentropy',

 metrics=['accuracy'])

3. Провести исследование «epochs»,«batch\_size»

network.fit(train\_images, train\_labels, epochs=5, batch\_size=128)

4. Разработать ПО, для реализации задачи распознавания рукописный цифр (с использованием набора mnist).

5. Создать примеры, показывающие работу сети (сеть распознает верно).

6. Создать примеры, показывающие работу сети (сеть распознает не верно).

**Код программы (внесённые изменения в шаблон кода выделены)**

from keras.datasets import mnist  
(train\_images, train\_labels), (test\_images, test\_labels) = mnist.load\_data()  
  
train\_images.shape  
  
 len(train\_labels)  
#%%  
train\_labels  
#%%  
test\_images.shape  
#%%  
len(test\_labels)  
#%%  
test\_labels  
#%%  
from keras import models  
from keras import layers  
network = models.Sequential()  
network.add(layers.Dense(512, activation='relu', input\_shape=(28 \* 28,)))  
network.add(layers.Dense(10, activation='softmax'))  
#%%  
network.compile(optimizer='rmsprop',  
 loss='categorical\_crossentropy',  
 metrics=['accuracy'])  
#%%  
train\_images = train\_images.reshape((60000, 28 \* 28))  
train\_images = train\_images.astype('float32') / 255  
test\_images = test\_images.reshape((10000, 28 \* 28))  
test\_images = test\_images.astype('float32') / 255   
#%%  
from tensorflow.keras.utils import to\_categorical  
train\_labels = to\_categorical(train\_labels)  
test\_labels = to\_categorical(test\_labels)  
  
#%%  
network.fit(train\_images, train\_labels, epochs=5, batch\_size=128)  
#%%  
test\_loss, test\_acc = network.evaluate(test\_images, test\_labels)  
print('test\_acc:', test\_acc)  
#%%  
from PIL import Image  
import numpy as np  
import matplotlib.pyplot as plt  
  
im = Image.open("img1.png").convert('L').resize((28, 28))  
arr = np.array(im)  
  
digit = arr.reshape(1, 28 \* 28)  
digit = 1 - digit.astype('float32') / 255  
plt.imshow(digit[0].reshape((28, 28)), cmap=plt.cm.binary)  
plt.show()  
#%%  
digit  
#%%  
cl = network.predict(digit)[0]  
np.argmax(cl)  
#%%  
im = Image.open("img2.png").convert('L').resize((28, 28))  
arr = np.array(im)  
  
digit = arr.reshape(1, 28 \* 28)  
digit = 1 - digit.astype('float32') / 255  
plt.imshow(digit[0].reshape((28, 28)), cmap=plt.cm.binary)  
plt.show()  
#%%  
cl = network.predict(digit)[0]  
np.argmax(cl)  
#%%

**Результаты выполнения задания**

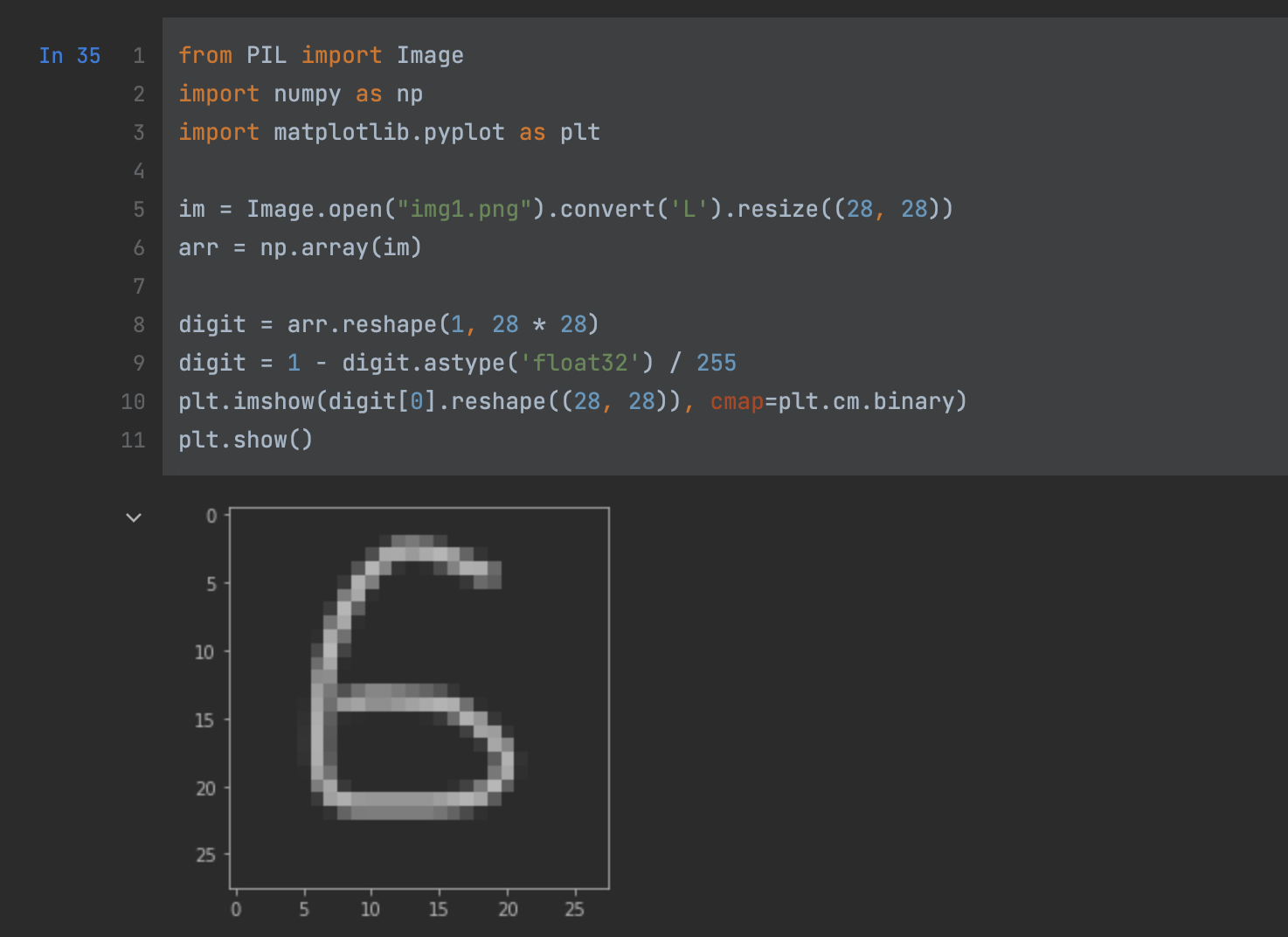
Для проверки работоспособности сети были выбраны следующие изображения с числами 6 и 7

**Изображение выглядит как вешалка

Автоматически созданное описаниеИзображение выглядит как вешалка, лампа, другой

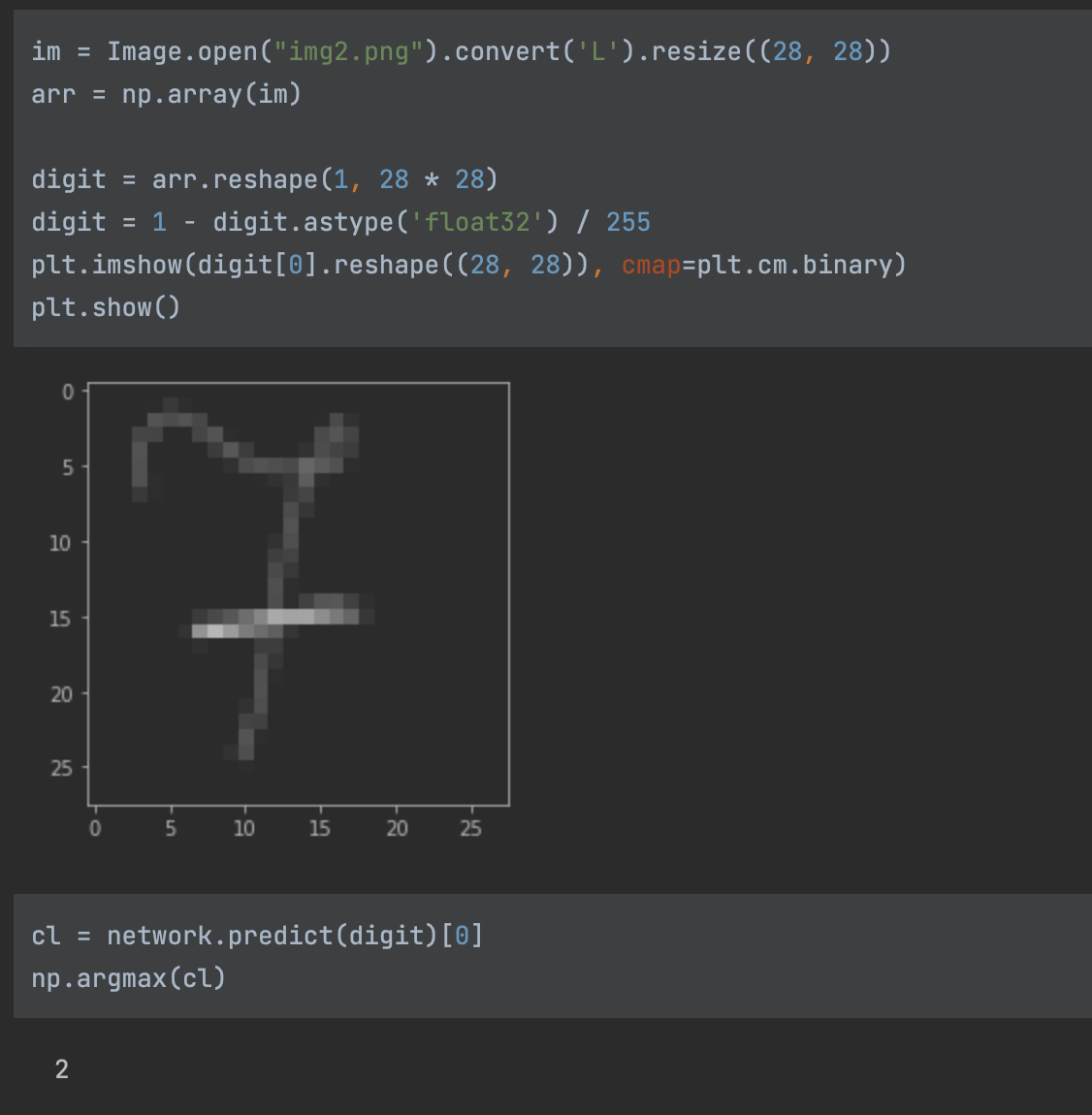
Автоматически созданное описание**

В результате первое число было успешно распознано, второе дало сбой (можно связать с тем, что линия была тонкой, использовалась черточка и число было написано с большим количеством «завитушек»)



Изображение выглядит как текст

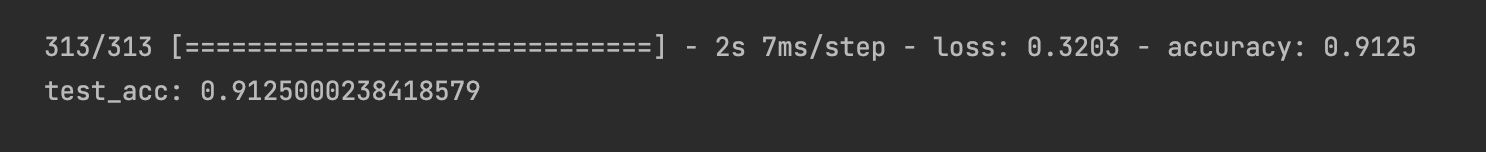
Автоматически созданное описание



Также в работе были изучены различные параметры сети. Ниже представлены их параметры и результаты точноcти:

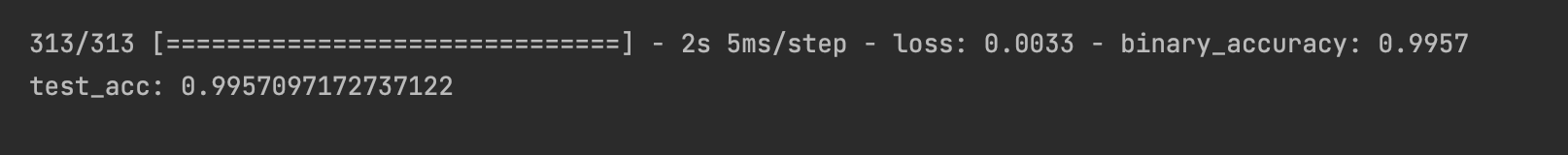
Изображение выглядит как текст

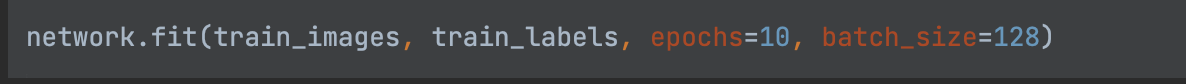
Автоматически созданное описание

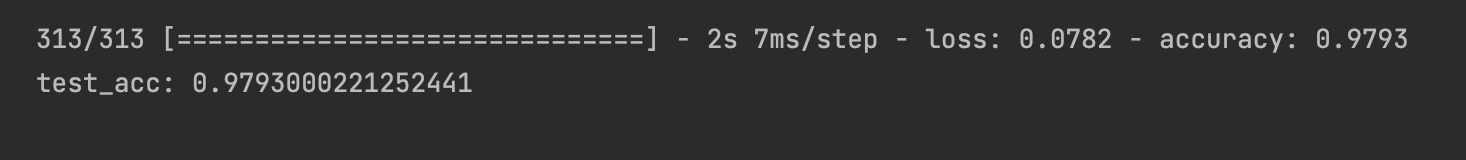


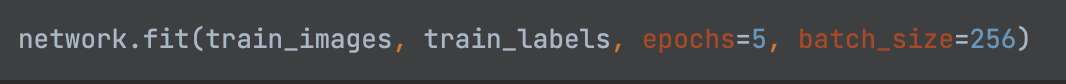
Изображение выглядит как текст

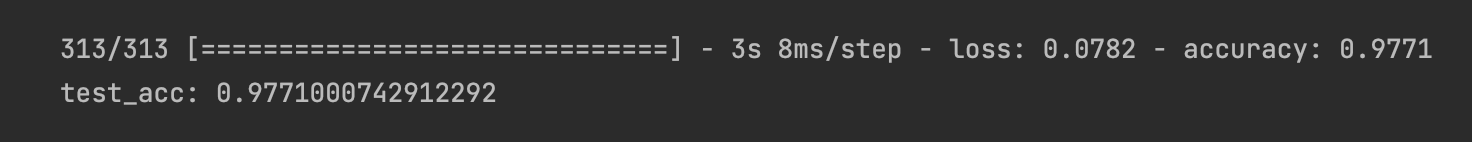
Автоматически созданное описание











Изображение выглядит как текст

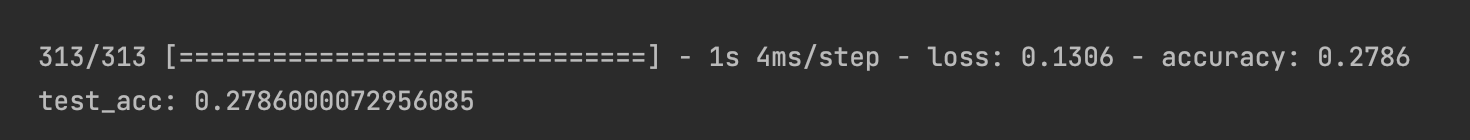
Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание



# Выводы

1. В результате проведения экспериментов с гиперпараметрами были найдены комбинации, позволяющие повысить точность классификации изображений
2. Успешно найдены примеры изображений, которые сеть правильно и неправильно классифицирует