ФЕДЕРАЛЬНОЕ АГЕНСТВО СВЯЗИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

Московский технический университет связи и информатики

(МТУСИ)

Кафедра «Математическая кибернетика и информационные технологии» Дисциплина «Интеллектуальные системы»

Лабораторная работа № 4

Распознавание рукописных символов

Выполнил: \_\_\_\_\_\_\_ Д.С. Калининский

Проверил: \_\_\_\_\_\_\_ А.В. Шереметьев

Москва 2022

**Оглавление**

[1 Цель работы 3](#_Toc119580633)

[2 Задачи 3](#_Toc119580634)

[3 Выполнение работы 4](#_Toc119580635)

[4 Результаты 5](#_Toc119580636)

[5 Вывод 8](#_Toc119580637)

# **1 Цель работы**

Реализовать классификацию черно-белых изображений рукописных цифр (28x28) по 10 категориям (от 0 до 9).

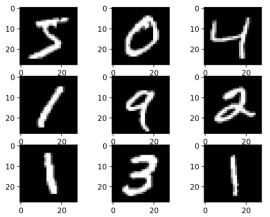


Рисунок 1 – Пример данных

Набор данных содержит 60,000 изображений для обучения и 10,000 изображений для тестирования (рисунок 1).

# **2 Задачи**

1) Ознакомиться с представлением графических данных;

2) Ознакомиться с простейшим способом передачи графических данных нейронной сети;

3) Создать модель;

4) Настроить параметры обучения;

5) Написать функцию, позволяющая загружать изображение пользователи и классифицировать его.

# **3 Выполнение работы**

Набор данных MNIST уже входит в состав Keras в форме набора из четырех массивов Numpy. На рисунке 2 представлено подключение необходимых модулей.

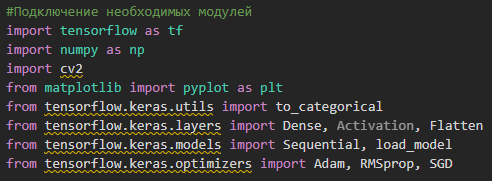


Рисунок 2 – Импорт модулей

Здесь train\_images и train\_labels — это тренировочный набор, то есть данные, необходимые для обучения. После обучения модель будет проверяться тестовым (или контрольным) набором, test\_images и test\_labels. Изображения хранятся в массивах Numpy, а метки — в массиве цифр от 0 до 9. Изображения и метки находятся в прямом соответствии, один к одному.

Исходные изображения представлены в виде массивов чисел в интервале [0, 255]. Перед обучением их необходимо преобразовать так, чтобы все значения оказались в интервале [0, 1].

Также необходимо закодировать метки категорий. В данном случае прямое кодирование меток заключается в конструировании вектора с нулевыми элементами со значением 1 в элементе, индекс которого соответствует индексу метки (рисунок 3).

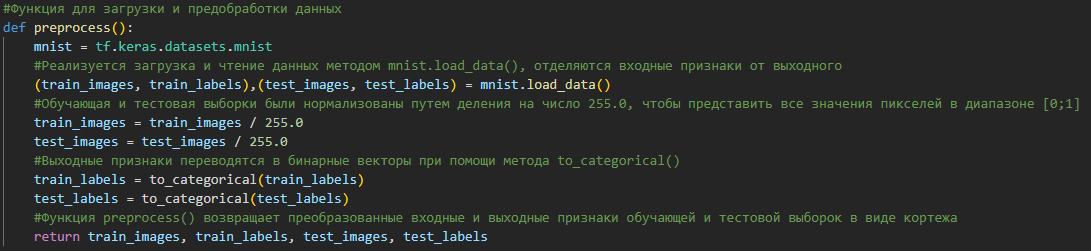


Рисунок 3 – Функция для загрузки и предобработки данных

Теперь можно задать базовую архитектуру сети.

Чтобы подготовить сеть к обучению, нужно настроить еще три параметра для этапа компиляции:

1) функцию потерь, которая определяет, как сеть должна оценивать качество своей работы на обучающих данных и, соответственно, как корректировать ее в правильном направлении;

2) оптимизатор — механизм, с помощью которого сеть будет обновлять себя, опираясь на наблюдаемые данные и функцию потерь;

3) метрики для мониторинга на этапах обучения и тестирования — здесь нас будет интересовать только точность (доля правильно классифицированных изображений).

Теперь можно начинать обучение сети, для чего в случае использования библиотеки Keras достаточно вызвать метод fit сети — он пытается адаптировать (fit) модель под обучающие данные.

В процессе обучения отображаются две величины: потери сети на обучающих данных и точность сети на обучающих данных.

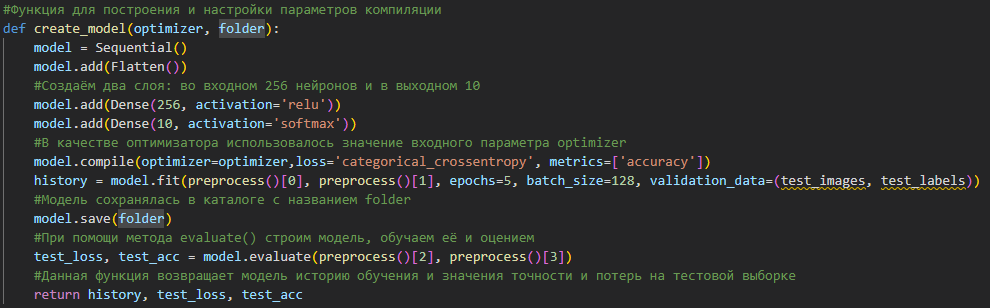


Рисунок 4 – Функция для построения и настройки параметров компиляции

Теперь проверим, как модель распознает контрольный набор (рисунок 5).

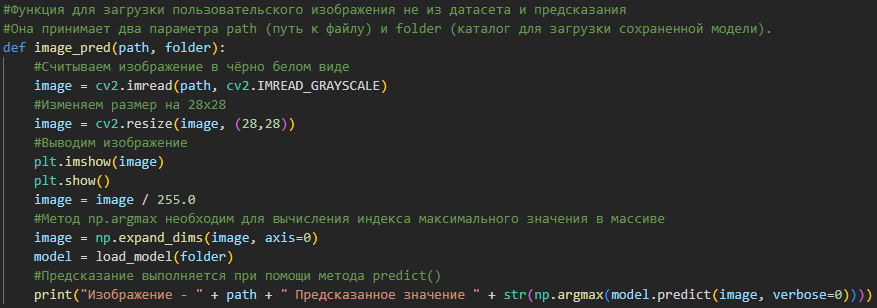


Рисунок 5 – Функция для загрузки изображения

# **4 Результаты**

Результаты тестирования моделей представлены в таблице 1 и на рисунках 6-9.

Таблица 1 – Результаты

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Оптимизатор | Скорость обучения | test\_accuracy | test\_loss |
| 1 | SGD | 0.01 | 0.909399 | 0.325536 |
| 2 | RMSprop | 0.01 | 0.972299 | 0.172193 |
| 3 | Adadelta | 0.01 | 0.885501 | 0.509438 |
| 4 | Adam | 0.01 | 0.972801 | 0.111144 |
| 5 | Adagrad | 0.01 | 0.934001 | 0.237191 |
| 6 | Adamax | 0.01 | 0.980199 | 0.065466 |
| 7 | Nadam | 0.01 | 0.969501 | 0.134356 |
| 8 | Ftrl | 0.01 | 0.900901 | 0.343943 |

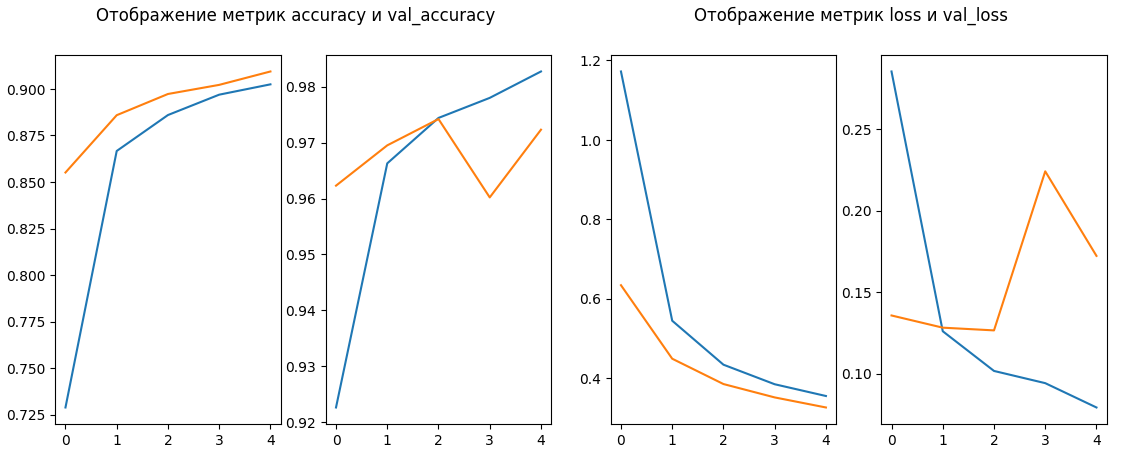


Рисунок 6 – Результаты тестирования модели 1 и модели 2

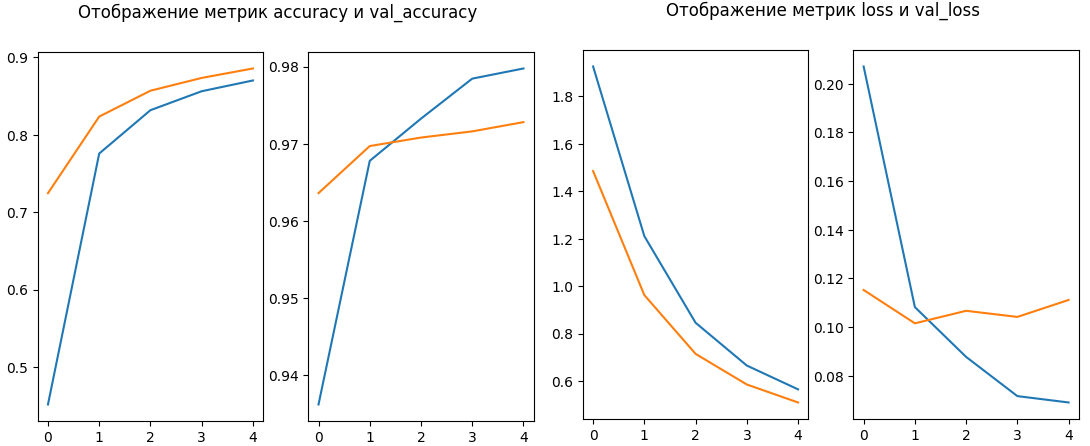


Рисунок 7 – Результаты тестирования модели 3 и модели 4

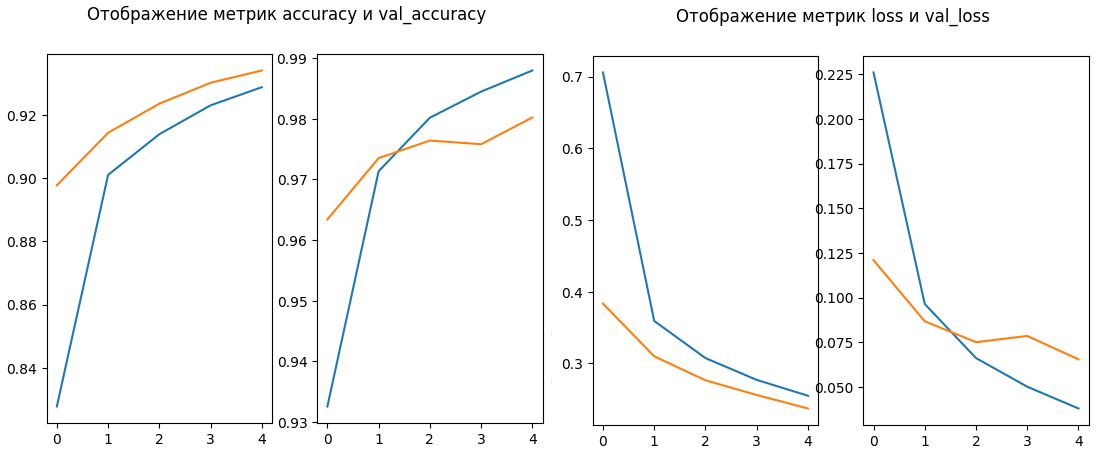


Рисунок 8 – Результаты тестирования модели 5 и модели 6

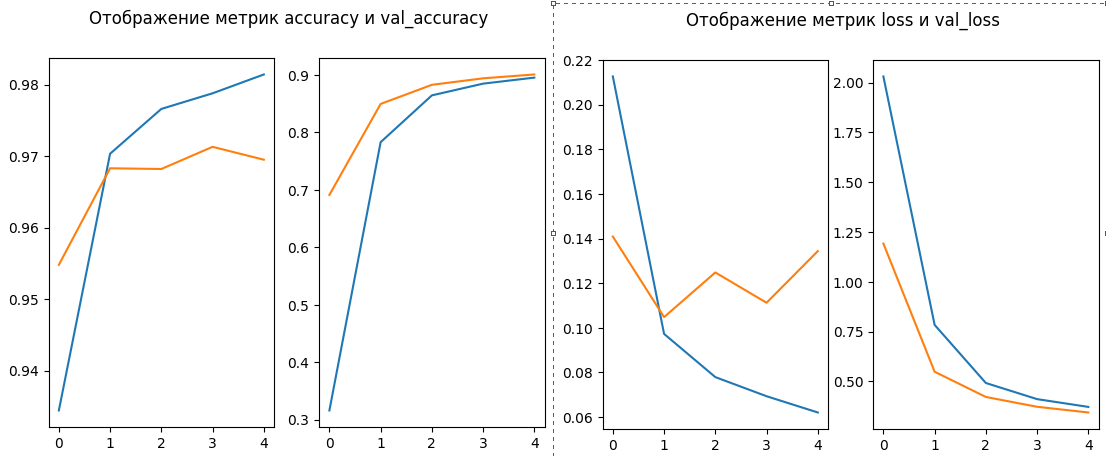


Рисунок 9 – Результаты тестирования модели 7 и модели 8

Для тестирования пользовательских изображений была выбрана модель 6, так как она обладает наибольшей точностью и наименьшими потерями (рисунок 10).

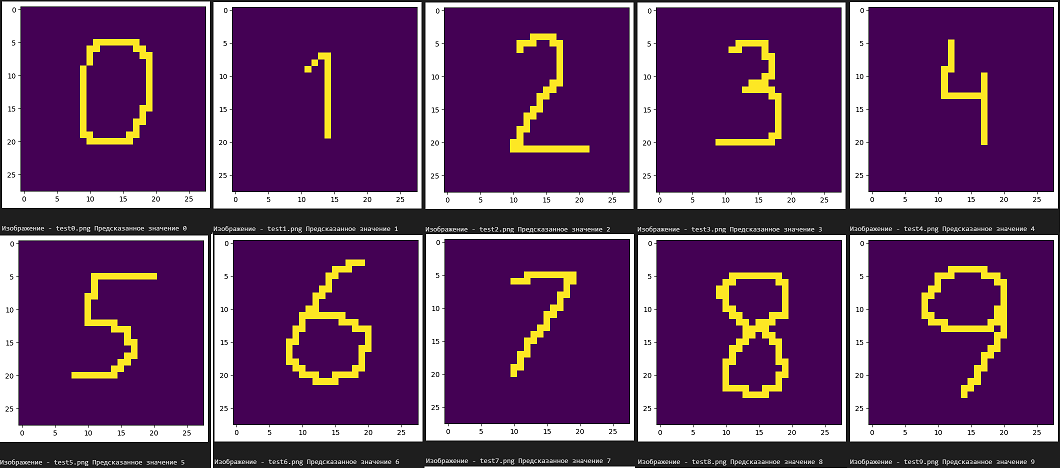


Рисунок 10 – Результат распознавания рукописных символов

# **5 Вывод**

В данной лабораторной работе, необходимо было реализовать классификацию черно-белых изображений рукописных цифр (28x28) по 10 категориям (от 0 до 9). Набор данных содержит 60,000 изображений для обучения и 10,000 изображений для тестирования.

В результате, была разработано 8 моделей и выбрана наилучшая, с который было протестировано 10 изображений с числами от 0 до 9. Все числа были распознаны правильно.