Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

ЛАБОРОТОРНАЯ РАБОТА №2

по дисциплине «Искусственный интеллект и анализ данных»

Выполнил
студент группы КТбо3-7 _____ О. С. Халепо

Принял
Преподаватель МОП ЭВМ _____ К. С. Чичерина

Таганрог 2023

1. Постановка задачи

Необходимо решить задачу распознавания рукописных цифр (датасет MNIST). Задача решается в рамках платформы онлайн-конкурсов по машинному обучению Kaggle. Ссылка на задание: https://www.kaggle.com/competitions/digit-recognizer.

Провести предподготовку данных: В задаче каждая фотография рукописной цифры задана в виде строки из 784 (28x28) значений градации серого для каждого пикселя. Градация от 0 до 255 (0 - белый, 255 - черный). Необходимо отнормировать значения для каждой картинки и получить train и validation датасеты.

Создать модель многослойной нейронной сети. Используя библиотеку для глубокого обучения* (см. вариант группы) необходимо создать нейронную сеть из нескольких слоев. Изучить возможности библиотеки, уметь отвечать на вопросы про Dense слои, методы активации, размерности входных/выходных матриц, метод compile и fit.

2. Ход работы

Работа будет производиться на платформе Google Colabolatory. После создания файла приступим к работе. Первым делом нужно импортировать все библиотеки для работы (см. рисунок 1).

```
import numpy as np # линейная алгебра
import pandas as pd # обработка данных, ввод-вывод CSV-файла (например, pd.read_csv)
import matplotlib.pyplot as plt

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers,models
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.layers import Input, GlobalAveragePooling2D
from tensorflow.keras.utils import to_categorical
from keras.callbacks import ReduceLROnPlateau
```

Рисунок 1 – Импортированные библиотеки

После нужно скачать датасеты с сайта Kaggle, загрузить их в Colab, теперь их можно использовать в проекте. Прочитаем два файла, первый – тренировочные наборы, второй – тестовый набор. Файл состоит из рисунков

28x28, которые представлены значениями от 0 до 255. Нам нужно разбить файл так, чтобы в одном массиве остались рисунки, а в другом массиве были ответы на рисунки (см. рисунок 2).

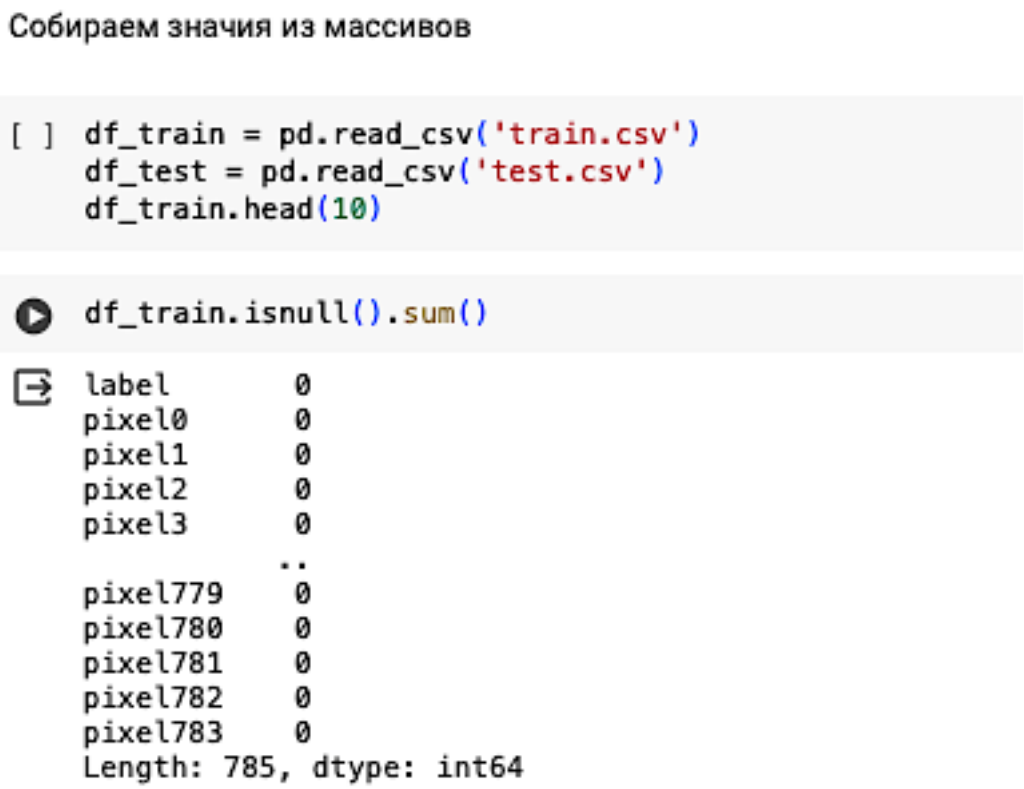

Рисунок 2 – Чтение файла

Теперь нам нужно разбить тренировочный файл на обучающую и тестовую часть. Сделаем это с помощью библиотеки sklearn (см. рисунок 3).

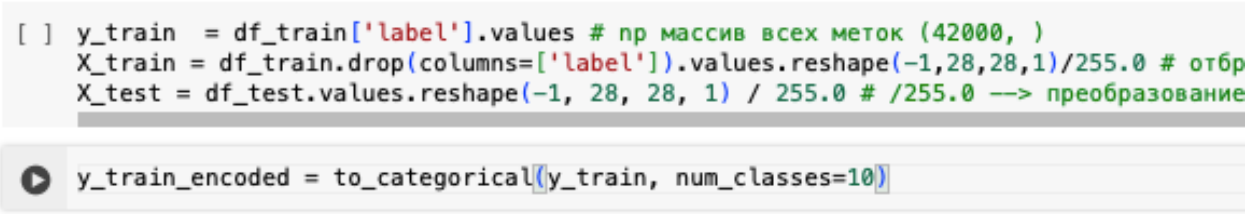

Рисунок 3 – Разбиение тренировочного файла

Теперь нужно преобразовать данные для обучения нейросети, после вывода информации по каждому массиву, видим, что формат не подходит, преобразуем формат и применим метод to_numpy() для преобразования из DataFrame в массив.

Теперь нам нужно получить в массиве с цветами данные в диапазоне от 0 до 1, а в массиве со значениями сделать вектор типа {0, 0, 0, 1, 0, 0, 0, 0, 0 , 0}, где 1 показывает число, такая функция присутствует в keras (см. рисунок 4).

```
X_test = df_test.values.reshape(-1, 28, 28, 1)
/ 255.0 # /255.0 --> преобразование этих значений пикселей в диапазон [0, 1]
```

Рисунок 4 – Нормализация данных

Теперь создадим модель обучения, которая будет состоять из входного слоя, скрытого слоя и выходного слоя. Первый слой Flatten получает изображение 28x28 и преобразует в 784 входных нейрона. Следующий слой Dense, состоящий из 128 нейронов, которые полносвязны с входными и выходными нейронами, метод активации – relu. Выходной слой Dense с методом активации softmax (см. рисунок 5).

```
model = models.Sequential([

    layers.Conv2D(filters=64, kernel_size=3, padding='same', activation='relu', input_shape=(28,28,1)),
    layers.MaxPool2D(pool_size=2, padding='same'),

    layers.Flatten(),

    layers.Dense(units=256, activation='relu'),
    layers.Dense(units=10, activation='softmax'),
])
```

Рисунок 5 – создание модели

Далее нужно скомпилировать созданную модель, для этого применяется функция compile, в ней есть 3 параметра(optimizer - алгоритм обучения НС, loss - функция ошибки, которую оптимизирует сеть, metrics - метрика качества обучения, измеряется после каждой эпохи обучения) (см. рисунок 6).

```
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

reduce_lr = ReduceLROnPlateau(monitor='loss', factor=0.3, verbose=1,
                              patience=2, min_lr=0.00000001)

history = model.fit(
    X_train, y_train_encoded,
    epochs=5,
    validation_split=0.1,
    callbacks=[reduce_lr],
)
```

Рисунок 6 – Компиляция модели

Теперь обучим нашу нейронную сеть, для этого применим функцию fit (см. рисунок 7).

```
history = model.fit(
    X_train, y_train_encoded,
    epochs=5,
    validation_split=0.1,
    callbacks=[reduce_lr],
)
```

Рисунок 7 – Обучение нейронной сети

После обучения можем проверять нашу нейронную сеть на тестовых данных, для этого нормализуем тестовые данные, воспользуемся функцией predict для прогона значений через нейронную сеть, после мы получим массив, который преобразуем в таблицу и выведем на экран (см. рисунок 8).

```
[ ] # Преобразуем выходные данные вероятности модели в предсказания цифр
    predicted_labels = np.argmax(predictions, axis=1)

    # Создаем DataFrame для отправки
    submission = pd.DataFrame({
        "ImageId": range(1, len(predicted_labels) + 1),  # ImageId starts from 1
        "Label": predicted_labels
    })

    # Сохраняем DataFrame в файл CSV
    submission.to_csv('submission.csv', index=False)
    print(submission)
```

Рисунок 8 – Результат тестовых данных

На заключительном этапе сформируем excel файл (см. рисунок 9).

```
# Сохраняем DataFrame в файл CSV
submission.to_csv('submission.csv', index=False)
print(submission)
```

Рисунок 9 – Формирование выходного файла

3. Вывод

В результате выполнения лабораторной работы была освоена библиотека keras, были изучены методы написания нейронной сети для распознавания рукописного текста.