**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

Проект по теме:

«Разработка и обучение модели на данных MNIST с использованием TensorFlow и Keras»

по курсу

«Операционные системы и сети»

Выполнили:

студенты группы ИКПИ-14

Хохлов Т.В.

Принял:

доцент кафедры ПИиВТ

Дагаев А.В

Санкт-Петербург

2023 г.

**Введение:**

Целью данной работы было разработать и обучить модель машинного обучения на изображениях рукописных цифр из набора данных **MNIST**, используя библиотеки **TensorFlow** и **Keras**. Модель представляет собой сверточную нейронную сеть, способную классифицировать цифры от 0 до 9.

**Используемые библиотеки и инструменты:**

**TensorFlow**: открытая библиотека для машинного обучения и глубокого обучения.

**Keras**: высокоуровневый API для построения и обучения моделей глубокого обучения.

**Подготовка данных:**

Использован набор данных **MNIST**, содержащий 60 000 тренировочных изображений и 10 000 тестовых изображений рукописных цифр.

Изображения были предварительно обработаны: изменен размер, преобразованы в оттенки серого и нормализованы.

**Архитектура модели:**

Модель состоит из трех сверточных слоев (**Conv2D**) с функцией активации **ReLU** и двух слоев пулинга (**MaxPooling2D**).

Затем следуют слои **Flatten** и **Dense** для классификации.

Использована функция активации **Softmax** на последнем слое для получения вероятностных прогнозов.

# Создание модели

model = models.Sequential([

    layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(64, (3, 3), activation='relu'),

    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(64, (3, 3), activation='relu'),

    layers.Flatten(),

    layers.Dense(64, activation='relu'),

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

])

**Компиляция и обучение:**

Модель была скомпилирована с оптимизатором **'adam'**, функцией потерь **'categorical\_crossentropy**' и метрикой **'accuracy'**.

Обучение проведено на тренировочном наборе с разделением на валидационную подвыборку. Одна эпоха использована для примера.

# Компиляция модели

model.compile(optimizer='adam',

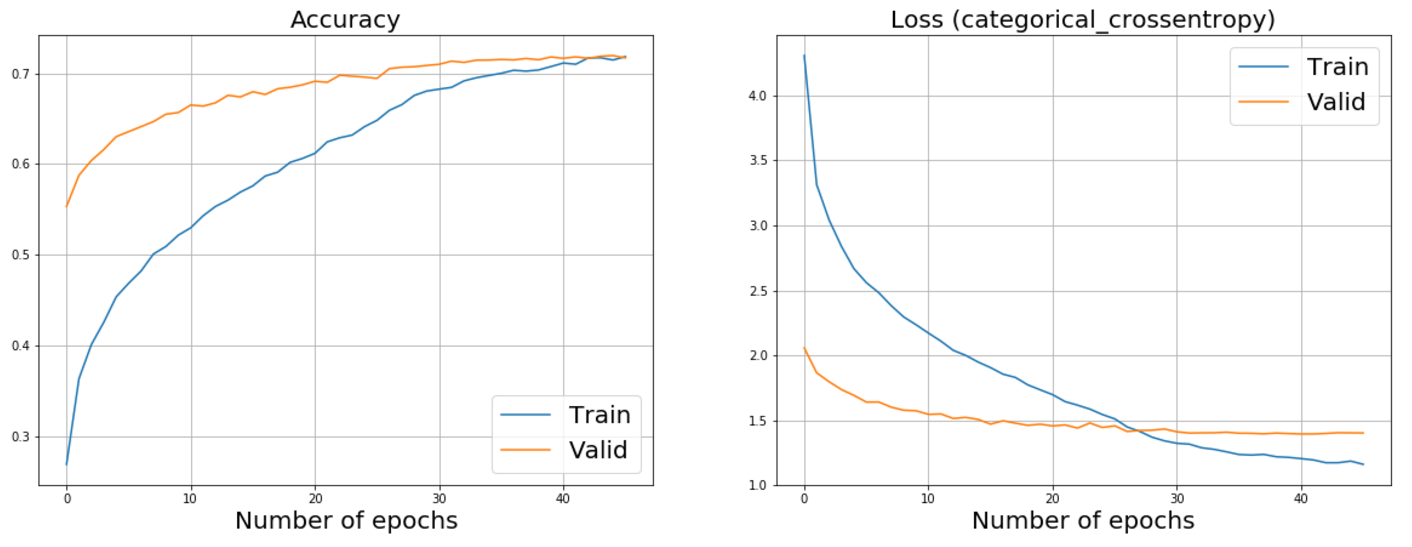
            loss='categorical\_crossentropy',

            metrics=['accuracy'])

# Обучение модели

model.fit(train\_images, train\_labels, epochs=1, batch\_size=64, validation\_split=0.2)

Модель была оценена на тестовом наборе данных для определения точности.



**Выводы**

В ходе проведённой лабораторной работы были получены следующие результаты:

1. Освоена библиотека **TensorFlow** языка **Python**.

2. Создана программа с использованием простенькой AI.

Исходный код программы

import tensorflow as tf

from tensorflow.keras import layers, models

from tensorflow.keras.datasets import mnist

import cv2

import numpy as np

def imgParse(image\_path, model):

    # Загрузка и предобработка изображения

    img = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)

    if img is None:

        print(f"Ошибка загрузки изображения {image\_path}")

        return

    # threshold, перевод в черно-белое

    img = cv2.threshold(img, 128, 255, cv2.THRESH\_BINARY)[1]

    img = cv2.resize(img, (28, 28))

    img = img.reshape((1, 28, 28, 1)).astype('float32') / 255

    # Предсказание

    predictions = model.predict(img)

    predicted\_label = np.argmax(predictions[0])

    # Вывод предсказания

    print(f"Модель предсказывает: {predicted\_label}")

# Загрузка данных MNIST

(train\_images, train\_labels), (test\_images, test\_labels) = mnist.load\_data()

# Предобработка данных

train\_images = train\_images.reshape((60000, 28, 28, 1)).astype('float32') / 255

test\_images = test\_images.reshape((10000, 28, 28, 1)).astype('float32') / 255

# Кодирование меток в формат one-hot

train\_labels = tf.keras.utils.to\_categorical(train\_labels)

test\_labels = tf.keras.utils.to\_categorical(test\_labels)

# Создание модели

model = models.Sequential([

    layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(64, (3, 3), activation='relu'),

    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(64, (3, 3), activation='relu'),

    layers.Flatten(),

    layers.Dense(64, activation='relu'),

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

])

# Компиляция модели

model.compile(optimizer='adam',

            loss='categorical\_crossentropy',

            metrics=['accuracy'])

# Обучение модели

model.fit(train\_images, train\_labels, epochs=1, batch\_size=64, validation\_split=0.2)

# Оценка модели на тестовых данных

test\_loss, test\_acc = model.evaluate(test\_images, test\_labels)

print(f"Точность на тестовом наборе: {test\_acc}")

imgParse('test.jpg', model)

imgParse('test2.jpg', model)

imgParse('test3.jpg', model)

imgParse('test4.jpg', model)

imgParse('test5.jpg', model)

imgParse('test6.jpg', model)