МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Южно-Уральский государственный университет (национальный исследовательский университет)»

Институт естественных и точных наук

Кафедра прикладной математики и программирования

Отчет по лабораторной работе № 2

Разработка полностью связанной нейронной сети

по дисциплине

«Современные нейросетевые технологии»

ЮУрГУ - 01.04.02. 2025. 306/010. Р

Руководитель, преподаватель

_____/ Д.М. Кичеев /

«_____» мая 2025 г.

Автор

студент группы ИЕТН - ЕТ-122

_____/ О.В. Ростова /

«_____» мая 2025 г.

Челябинск 2025

Цель настоящей работы состоит в том, чтобы получить базовые навыки работы с одной из библиотек глубокого обучения (Caffe, Torch, TensorFlow или MXNet на выбор) на примере полностью связанных нейронных сетей.

Выполнение практической работы предполагает решение следующих **задач**:

1. Выбор библиотеки для выполнения практических работ курса.

2. Установка выбранной библиотеки на кластере.

3. Проверка корректности установки библиотеки. Разработка и запуск тестового примера сети, соответствующей логистической регрессии, для решения задачи классификации рукописных цифр набора данных MNIST (пример разобран в лекционных материалах).

4. Выбор практической задачи компьютерного зрения для выполнения практических работ.

5. Разработка программ/скриптов для подготовки тренировочных и тестовых данных в формате, который обрабатывается выбранной библиотекой.

6. Разработка нескольких архитектур полностью связанных нейронных сетей (варьируются количество слоев и виды функций активации на каждом слое) в формате, который принимается выбранной библиотекой.

7. Обучение разработанных глубоких моделей.

8. Тестирование обученных глубоких моделей, формирование вывода относительно разработанных архитектур.

9. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

Выполнение лабораторной работы осуществлено мною в Google Colab на языке программирования Python. Результаты и отчет размещены в личном профиле на GitHub.

В процессе выполнения лабораторной работы, мною были выполнены следующие шаги:

1. Выбрана библиотека TensorFlow с модулем Keras из-за простоты использования и встроенной поддержки в Google Colab.

2. TensorFlow версии 2.18.0 предустановлен в Google Colab, установка не потребовалась.

3. Для проверки корректности установки библиотеки, разработан и запущен тестовый пример сети, соответствующей логистической регрессии, для решения задачи классификации рукописных цифр набора данных MNIST с использованием TensorFlow.

Код загружает данные MNIST, нормализует их, создает модель с одним полносвязным слоем и обучает её.

Запуск тестового примера показал, что библиотека установлена корректно, а базовая модель обучается на простом наборе данных.

Доля правильно классифицированных изображений на тестовой выборке 92,24%, это говорит о хорошей способности модели распознавать цифры MNIST.

Значение функции потерь (categorical_crossentropy) на тех же данных. Кросс-энтропия показывает насколько прогнозируемые вероятности отличаются от идеального one-hot распределения. Test loss = 0,2814 - это типичное значение для простой однослойной сети, указывающее на умеренную неуверенность в некоторых предсказаниях.

4. Для выполнения практических работ я выбрала практическую задачу компьютерного зрения - задачу классификации изображений из набора данных CIFAR-10.

Задача: Многоклассовая классификация изображений CIFAR-10

Набор данных CIFAR-10 (Canadian Institute For Advanced Research) - это коллекция изображений, широко используемых в алгоритмах машинного обучения и компьютерного зрения. Он был разработан исследователями из института CIFAR и состоит из 60 000 цветных изображений размером 32x32 пикселя, разделенных на 10 классов.

Ссылка на набор данных: https://www.kaggle.com/c/cifar-10. Также этот набор данных доступен через TensorFlow.

На входе: вектор из 3072 чисел (нормализованные пиксели цветного изображения (RGB) размером 32x32x3).

На выходе: вероятность принадлежности к одному из 10 классов (метка одного из 10 классов: самолет, автомобиль, птица, кошка, олень, собака, лягушка, лошадь, корабль, грузовик).

Математическая постановка задачи:

Дан набор данных:

$$D = \{(x_i, y_i)\}_{i=1}^{N}$$

где $x_i \in R^{32\times32\times3}$ - изображение,

$y_i \in \{0,1,\ldots,9\}$ - метка класса.

Цель - обучить модель f (x; θ), минимизирующую функцию потерь $L(f(x_i; \theta), y_i)$, например, кросс-энтропию:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{10} y_{i,k} * \log(f_k(x_i; \theta)),$$

где $y_{i,k}$ - one-hot кодировка метки,

f_k - вероятность класса k.

Метрика: Точность (accuracy):

$$\text{Accuracy} = \frac{\text{Количество правильно классифицированных примеров}}{\text{Общее количество примеров}}$$

Точность измеряет долю правильных предсказаний на тестовом наборе.

5. Произведена разработка программ/скриптов для подготовки тренировочных и тестовых данных в формате, который обрабатывается выбранной библиотекой.

Для подготовки тренировочных и тестовых данных набора CIFAR-10 разработан скрипт на языке Python с использованием библиотеки TensorFlow (модуль Keras). Скрипт выполняет загрузку данных, их предобработку и преобразование в формат, подходящий для обучения полностью связанных нейронных сетей.

Данные CIFAR-10 хранятся в формате NumPy массивов и загружаются с помощью функции tf.keras.datasets.cifar10.load_data(). Набор данных содержит 60 000 цветных изображений размером 32×32 пикселя, разделенных на 50 000 тренировочных и 10 000 тестовых примеров. Каждый пример имеет следующие характеристики:

- **изображения**: тип данных uint8 (значения пикселей от 0 до 255), форма (количество примеров, 32, 32, 3);

- **метки**: тип данных int32, форма (количество примеров, 1), где значения от 0 до 9 соответствуют 10 классам: самолёт (0), автомобиль (1), птица (2), кошка (3), олень (4), собака (5), лягушка (6), лошадь (7), корабль (8), грузовик (9).

Скрипт выполняет следующие действия:

- Загружает данные CIFAR-10 с помощью tf.keras.datasets.cifar10.load_data().

- Нормализует изображения делением на 255.

- Преобразует изображения в векторы для полностью связанной сети с помощью метода reshape.

- Преобразует метки в категориальный формат one-hot encoding с помощью to_categorical.

- Выводит размеры тренировочных и тестовых выборок, а также примеры меток для проверки корректности подготовки данных.

Скрипт успешно подготавливает данные CIFAR-10 для обучения полностью связанных нейронных сетей. Данные нормализованы, преобразованы в нужный формат, и их корректность подтверждена проверкой размеров и выводом примеров меток.

Изображения: float32, нормализованы к [0, 1], преобразованы в векторы, форма (количество примеров, 3072). Метки: float32, one-hot encoding, форма (количество примеров, 10).

6. Разработано 3 архитектуры полностью связанных нейронных сетей с разным количеством слоёв и функциями активации. Это позволит сравнить их производительность и понять, как сложность модели влияет на качество классификации.

Архитектуры:

	Скрытые слои	Выходной слой	
Модель 1	1 скрытый слой (512 нейронов, активация ReLU)	10 нейронов, softmax	простая модель, подходит для быстрого обучения; ReLU добавляет нелинейность
Модель 2	2 скрытых слоя (1024 и 512 нейронов, ReLU)	10 нейронов, softmax	более сложная, с большей вычислительной мощностью; 2 скрытых слоя для большей выразительности
Модель 3	3 скрытых слоя (2048, 1024, 512 нейронов, ReLU)	10 нейронов, softmax	самая глубокая, с комбинацией активаций для разнообразия; 3

			скрытых слоя для максимальной сложности; Dropout (0,3) после каждого слоя для повышения точности.

7. Произведено обучение разработанных глубоких моделей.

Обучение выполняется в функции **train_model** для каждой модели:

Компиляция: используется оптимизатор Adam (адаптивный, хорошо подходит для CIFAR-10), функция потерь sparse_categorical_crossentropy (для целочисленных меток), метрика - точность.

Обучение: 20 эпох, размер пакета 64, 20% обучающих данных (10 000 примеров) используются для валидации.

Время: замеряется с помощью time.time() для сравнения скорости обучения.

Визуализация: график точности по эпохам для обучения и валидации.

Возврат: история обучения (history), тестовая, валидационная и обучающая точность, время обучения.

8. Проведено также тестирование обученных глубоких моделей. Рассмотрим результаты тестирования и сформируем выводы относительно разработанных архитектур.

Тестирование также включено в функцию train_model. Точность на тестовых данных выводится для каждой модели.

Рассмотрим графики точности обучения каждой модели. Все 3 модели прошли обучение – 20 эпох (на оси Y). Точность (accuracy) или доля правильно классифицированных примеров (от 0 до 1) отражены на оси X.

Эти графики показывают, как изменяется точность (accuracy) на обучающей и валидационной выборках по эпохам.

Точность (обучение) – здесь отражена точность на обучающей выборке (40 000 примеров, так как 20% из 50 000 используются для валидации).

Точность (валидация) – здесь отражена точность на валидационной выборке (10 000 примеров).

На графиках мы можем увидеть, как модель обучается, насколько хорошо она обобщает данные и есть ли признаки переобучения или недообучения.

Поскольку CIFAR-10 - сложный набор данных для полностью связанных нейронных сетей, точность на тестовых данных составляет около 40-50%%.

Модели показали следующие результаты:

name	training_time	train_accuracy	val_accuracy	test_accuracy
Модель 1	477,680711	0,524050	0,4760	0,4842
Модель 2	1280,968456	0,590525	0,4972	0,5002
Модель 3	2672,024653	0,379700	0,4028	0,4042

Train Accuracy – тренировочные данные (состоит из 50 000 для CIFAR-10), отражает способность модели подстраиваться под тренировочные данные.

Val Accuracy – валидационные данные (10 000, выделенные из тренировочной выборки), используется для оценки обобщения на части тренировочного набора во время обучения.

Test Accuracy – тестовые данные (10 000 для CIFAR-10), показывает, насколько хорошо модель работает на новых данных.

На рисунке 1 представлена точность модели 1:

train_accuracy = 52,41%, val_accuracy = 47,60%, test_accuracy = 48,42%.

Тренировочная точность выше валидационной, что указывает на небольшое переобучение. Тестовая точность близка к валидационной, что подтверждает слабую обобщающую способность модели. Низкие значения всех метрик (около 50%) говорят о том, что модель слишком простая для сложных данных CIFAR-10.

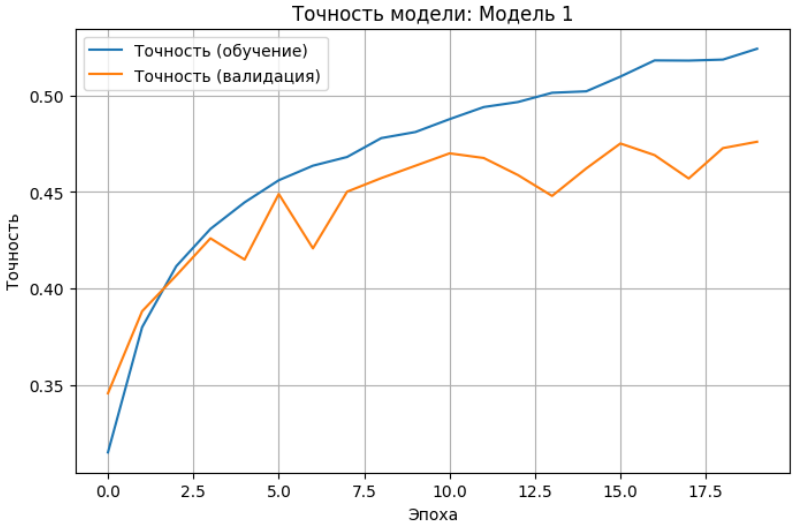

Рисунок 1 – Точность Модели 1

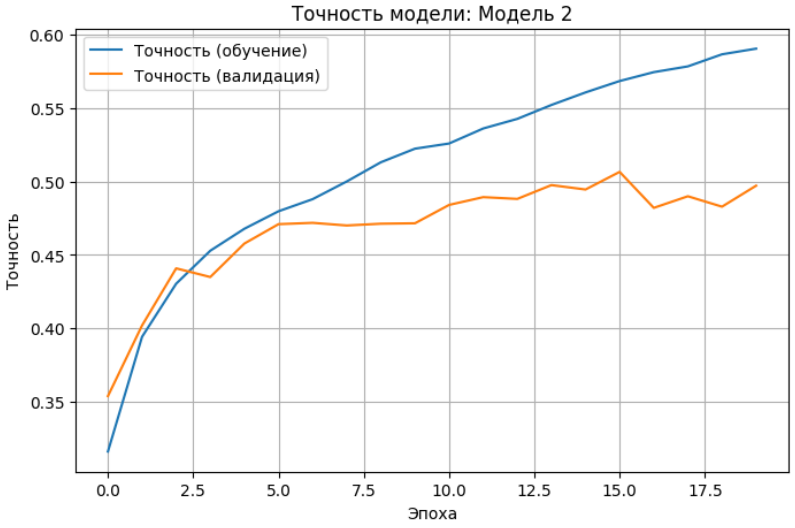

Рисунок 2 – Точность Модели 2

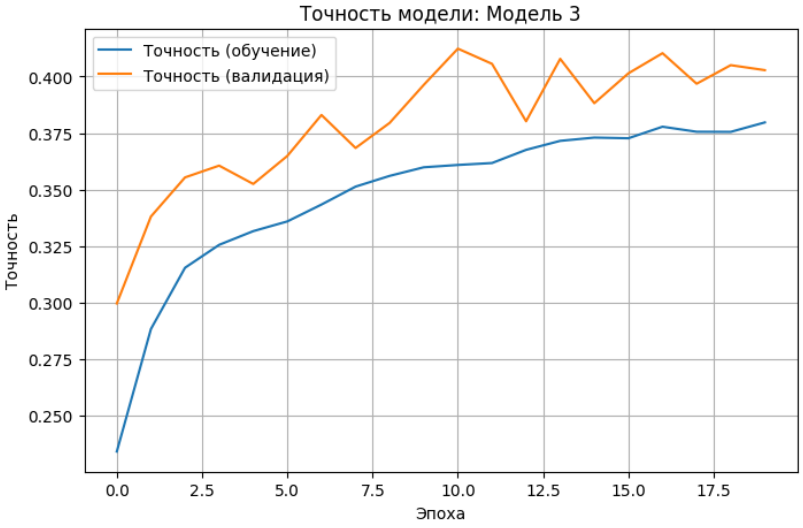

Рисунок 3 – Точность Модели 3

На рисунке 2 представлена точность модели 2:

train_accuracy = 59,05%, val_accuracy = 49,72%, test_accuracy = 50,02%.

Наивысшая тестовая точность среди всех моделей, но всё ещё низкая. Значительная разница между тренировочной и валидационной точностью (59,05% против 49,72%) указывает на переобучение, несмотря на использование Dropout. Эта модель лучше, чем остальные, вероятно, из-за большей глубины и активации ReLU.

На рисунке 3 представлена точность модели 3:

train_accuracy = 37,97%, val_accuracy = 40,28%, test_accuracy = 40,42%.

Самые низкие метрики, особенно тренировочная точность ниже валидационной, что необычно и может указывать на недообучение или проблемы с активациями (tanh и sigmoid склонны к затуханию градиентов). Тестовая точность подтверждает слабую производительность модели.

Для всех моделей тренировочная точность выше валидационной, что указывает на переобучение, особенно выраженное у Модели 2 (59,05% против 49,72%). Низкие значения всех метрик (37,97 - 59,05% для тренировочной, 40,28 - 49,72% для валидационной, 40,42 - 50,02% для тестовой) объясняются ограничениями полностью связанных сетей для задачи классификации изображений CIFAR-10, полностью связанные сети не оптимальны для обработки изображений, так как они не учитывают пространственную структуру данных (в отличие от сверточных нейронных сетей, CNN). Также это связано со сложностью CIFAR-10: набор данных содержит 10 классов с высокой вариативностью, что требует более сложных моделей. Также это может быть связано с недостаточное количество эпох, неподходящая скорость обучения или отсутствие аугментации данных.

Для каждой модели визуализированы предсказания на 5 случайно выбранных тестовых изображениях из набора CIFAR-10. Под каждым изображением указаны истинный класс, предсказанный класс и вероятность предсказания.

Таким образом, низкая точность моделей в основном обусловлена тем, что полностью связанные сети не оптимальны для CIFAR-10. И это нормальный результат для учебной задачи, так как ее цель - освоить базовые навыки.

9. Подготовлен отчет, содержащий минимальный объем информации по каждому этапу выполнения работы.

Таким образом, в процессе выполнения лабораторной работы мною изучены произведен выбор библиотеки для выполнения практических работ курса, проверка корректности установки библиотеки. Были разработан и запущен тестовый пример сети, соответствующей логистической регрессии, для решения задачи классификации рукописных цифр набора данных MNIST. При решении практической задачи компьютерного зрения для выполнения практических работ я выбрала задачу классификации изображений из набора данных CIFAR-10. В процессе разработки программ/скриптов для подготовки тренировочных и тестовых данных в формате, который обрабатывается выбранной библиотекой, мною было разработано 3 архитектуры полностью связанных нейронных сетей, проведено обучение и тестирование разработанных глубоких моделей.

Сформулированы выводы относительно разработанных архитектур, составлен отчет по каждому этапу выполнения работы.