МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Костромской государственный университет» (КГУ)

Институт автоматизированных систем и технологий

Кафедра информационных систем и технологий

Направление подготовки/Специальность\* 09.03.02

Информационные системы и технологии

Профиль Разработка программного обеспечения информационных систем

Дисциплина Технологии компьютерного зрения

ОТЧЁТ

Лабораторная работа №10: Нейронные сети

.

Выполнили студенты Смирнов Кирилл Андреевич

Шкунов Владимир Викторович

Группа 22-ИСбо-2а

Проверил Орлов Александр Валерьевич

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_

Кострома 2024

**Задание**

**Задание 1**: Создайте программу, обучащую простую нейронную сеть с одним скрытым слоем распознавать цифры на базе изображений MNIST.

* Для загрузки данных используйте модуль **tensorflow.keras.datasets.mnist**. Убедитесь, что загруженная база сохраняется в каталоге скрипта (см. документацию к функции **load\_data()**).
* Модель сети должна иметь тип Sequential и состоять из трёх слоев: входного слоя типа Flatten (так как входные данные - изображения), скрытого слоя типа Dense(размер слоя 32 и функция активации relu) и выходного слоя типа Dense(10 нейронов).
* При компиляции модели укажите тип оптимизатора 'adam', функцию потерь SparseCategoricalCrossentropy и метрику точности ('accuracy').
* Обучите сеть методом .fit() в течении 10 эпох, затем оцените точность работы сети на обучающей и контрольной методом **.evaluate()**.
* Сохраните обученную сеть методом **.save()** в файл с расширением **.h5**.

**Задание 2**: Модифицируйте созданную в задании 1 программу следующим образом.

* Используйте пакет tensorflow-datasets для загрузки набора данных. Сформируйте pipeline для обучающих и тестовых данных.
* Проведите нормализацию данных перед обучением сети. Зафиксируйте начальную (эпоха 1) и конечную (эпоха 10) достигнутые точности без нормализации и с нормализацией.
* Измените архитектуру сети следующим образом: используйте один слой на 64 нейрона, либо два слоя на 32 нейрона каждый. Сравните достигнутую точность и время обучения в обоих случаях.
* Задайте для оптимизатора параметр learning\_rate, равный 10-2, 10-3 или 10-4. Постройте графики роста точности обучения от эпохи к эпохе для всех трёх случаев. Для получения данных используйте объект History, возвращаемый методом **.fit()**.

**Задание 3:** Составьте программу, загружающую модель с наилучшими показателями, обученную в задании 2 (или в дополнительном), и использующую её для анализа пользовательского ввода.

* Используйте функцию **tf.keras.models.load\_model()** для загрузки обученной сети.
* Определите форму требуемого входа для сети.
* Постройте новую Sequential модель, содержащую загруженную сеть и дополнительный слой типа **Softmax**. Это требуется, чтобы преобразовать отклики сети в вероятности принадлежности входа к тому или иному классу.
* Программа должна показать квадратное окно размером 256х256 и позволить пользователю мышью нарисовать в нём цифру, белым цветом на черном фоне. По нажатию клавиши "пробел", программа должна уменьшить изображение до размера входа сети, нормализовать его, а затем передать его методу **.predict()** модели.
* Метод **.predict()** вернёт массив из 10 элементов. Найдите индекс наибольшего элемента в этом массиве и выведите его в консоль.

**Дополнительное задание**: Создайте свёрточную нейронную сеть следующей архитектуры. Свёрточный блок выделения особенностей:

1. Входной слой **Input** с формой входных данных (28, 28, 1).
2. Свёрточный слой **Conv2D** из 32 нейронов с окном 3х3 и функцией активации ReLU.
3. Слой прореживания **Pooling2D** с окном 2х2.
4. Свёрточный слой **Conv2D** из 32 нейронов с окном 3х3 и функцией активации ReLU.
5. Слой прореживания **Pooling2D** с окном 2х2.
6. Свёрточный слой **Conv2D** из 32 нейронов с окном 3х3 и функцией активации ReLU.

Непосредственно за свёрточным блоком должен идти полносвязный блок классификации:

1. Слой типа **Flatten** для преобразования изображения в вектор значений
2. Полносвязный слой **Dense** размером 64 нейрона и функцией активацией ReLU.
3. Выходной полносвязный слой **Dense** размером 10 нейронов.

При обучении сети используйте механизм ранней остановки. Создайте экземпляр класса **tf.keras.callbacks.EarlyStopping** отслеживающий точность (accuracy), с порогом останова 10-3 по величине и 3 эпохи по длительности. Передайте список из одного элемента, содержащий этот экземпляр, в параметре **callbacks** метода **model.fit()**.

Обучайте сеть в течении 20 эпох на наборе данных MNIST, и проверьте её работу с помощью программы, созданной в рамках задания 3. При этом в программе из задания 3 измените форму полученного массива на (1, 28, 28, 1).

**Вопросы**

1. Что такое параметры сети? Что такое гиперпараметры сети?
2. Зачем производится разделение выборки на обучающую и контрольную?
3. Что такое нормализация данных?
4. Для чего нужна функция активации нейрона?

**Вопрос №1**

Параметры сети - это внутренние переменные, которые оптимизируются во время обучения нейронной сети. Это, например, веса и смещения нейронов. Они изменяются во время обучения сети

Гиперпараметры сети - это параметры, которые задаются до обучения сети и не оптимизируются во время обучения. Это, например, количество слоев в сети, количество нейронов в каждом слое, размеры ядер свертки, величина, скорость обучения и другие параметры.

**Вопрос №2**

Разделение выборки на обучающую и контрольную производится для того, чтобы оценить качество обученной модели на данных, которые не использовались при обучении. Это необходимо для того, чтобы избежать переобучения модели.

**Вопрос №3**

Нормализация данных - это процесс преобразования данных в определенный диапазон или масштаб. Это необходимо для того, чтобы все признаки имели одинаковый вес при обучении модели и не было смещения в сторону признаков с большими значениями.

**Вопрос №4**

Функция активации нейрона необходима для введения нелинейности в модель, позволяя нейрону "активироваться" только при определенных входных сигналах и "забывать" незначительные сигналы, что повышает устойчивость модели к шуму в данных. Без функции активации нейронная сеть была бы линейной моделью, не способной решать сложные задачи.

**Скриншоты работы**

**Задание 1**

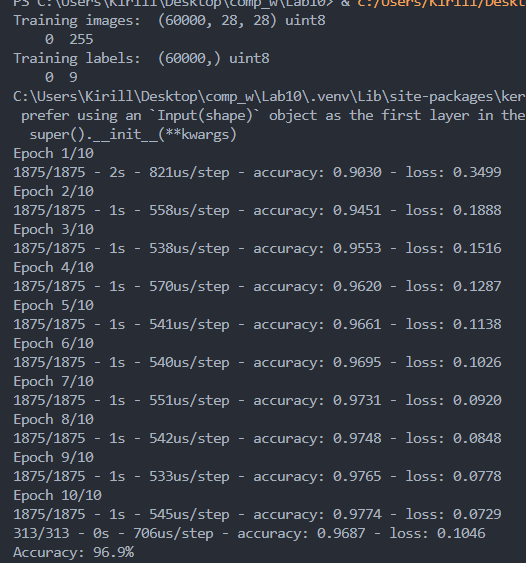
****

Рис. 1. Вывод первого задания

**Задание 2**

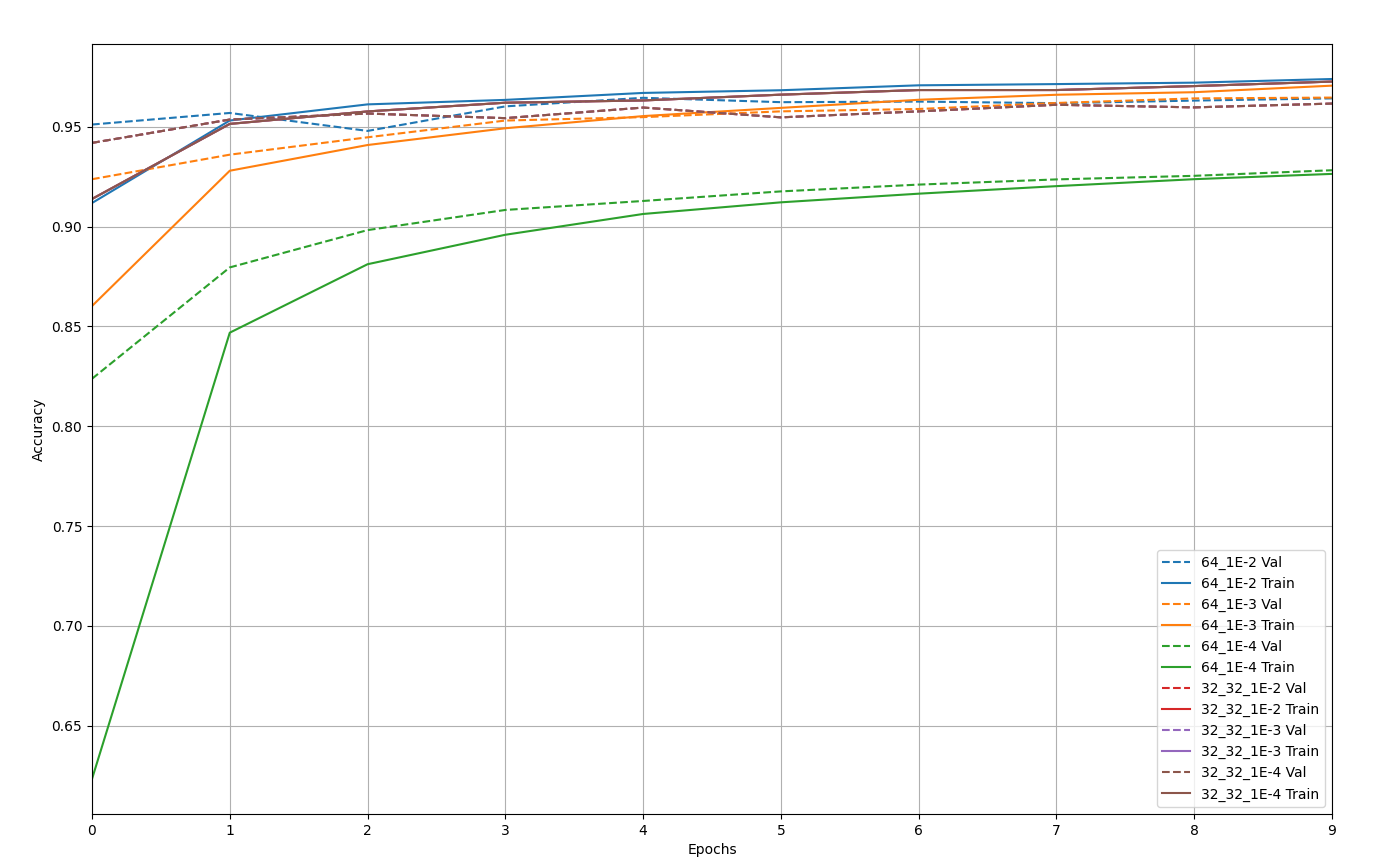
****

Рис. 2. Вывод второго задания

**Задание 3**

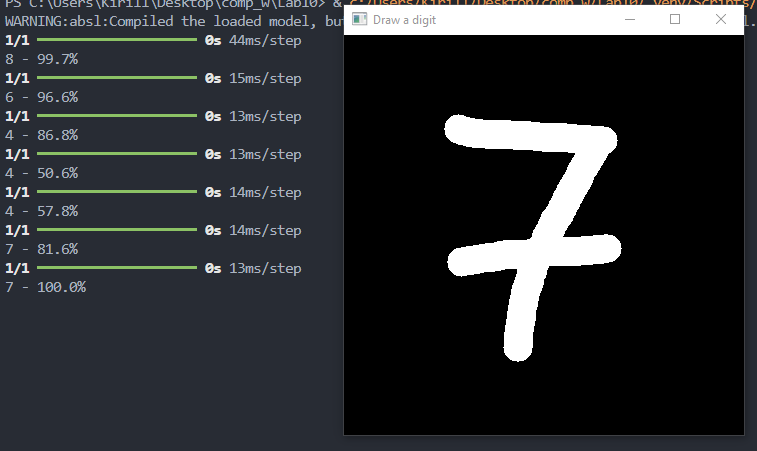
****

Рис. 3. Вывод третьего задания

**Дополнительное задание**

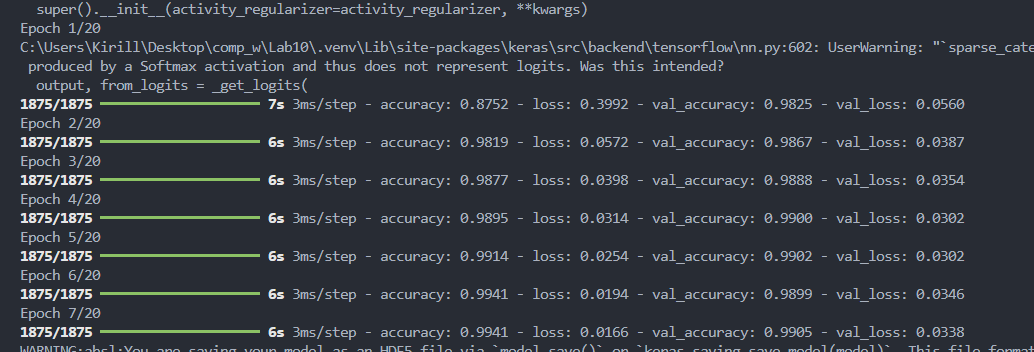
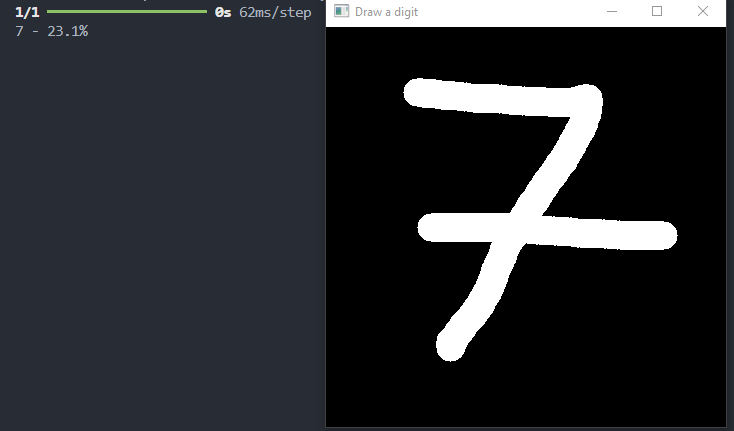
****

Рис. 4. Обучение

  
Рис. 5. Вывод третьего задания со сверточной нейронной сетью из дополнительного залания