МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Костромской государственный университет»

(КГУ)

ИАСТ

Кафедра автоматизированных систем и технологий

09.03.02

Направление подготовки/Специальность Информационные системы и технологии

Дисциплина Технологии компьютерного зрения

# Лабораторная №10.

# Нейронные сети.

Выполнили студенты Копосов Лев Владимирович

Копосов Владимир Владимирович

Группа 22-ИСбо-1б

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

Кострома

Вопросы:

1. Что такое параметры сети? Что такое гиперпараметры сети?

Параметры сети - это веса и смещения сети. Это параметры, которые задаются в процессе обучения сети.

Гиперпараметры сети - это структура сети, то есть число и размер слоев, функция активации, скорость обучения и другие. Подбор значений гиперпараметров - один из важнейших этапов в создании нейронной сети. Это параметры, которые задаются до обучения сети.

1. Зачем производится разделение выборки на обучающую и контрольную?

Разделение выборки на обучающую и контрольную производится для предотвращения переобучения модели. Обучающая выборка используется для обучения модели. Контрольная выборка используется для проверки обучающей модели (на наличие переобучения или неточностей).

1. Что такое нормализация данных?

Нормализация данных - это процесс преобразования входных данных, в результате которого значения признаков будут приведены к заданному диапазону. Нормализация используется для повышения скорости и качества обучения модели.

1. Для чего нужна функция активации нейрона?

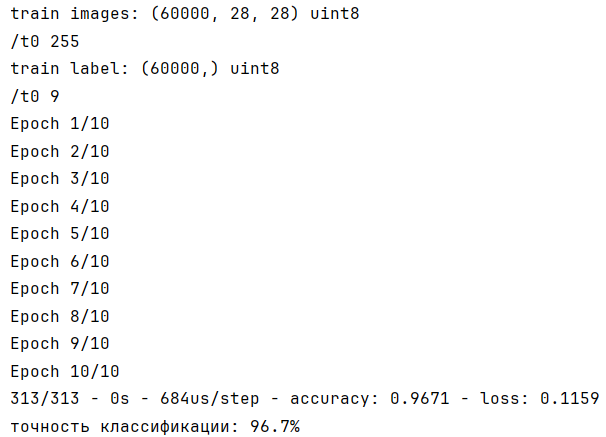
Взвешенная сумма входных значений, то есть ее линейная комбинация входных значений может описывать только прямую, и линейная комбинация прямых (для более высоких размерностей - гиперплоскостей) тоже дает только прямую. В итоге мы сильно ограничены в том, какие решающие особенности сеть может описывать, а значит, какие виды данных она может разделять. По сути нейросеть в описанном виде может обрабатывать только линейно разделимые данные. Выход нейрона может иметь любое значение, что не удобно на практике. Чтобы избавиться от этих недостатков, результат суммирования пропускается через функцию активации. Функция активации должна быть нелинейной, легко дифференцируемой и неубывающей, то есть чем больше значений на входе, тем больше должно быть значение на выходе.

Вывод в консоль.

##### **Задание 1**

Создайте программу, обучающую простую нейронную сеть с одним скрытым слоем распознавать цифры на базе изображений MNIST.

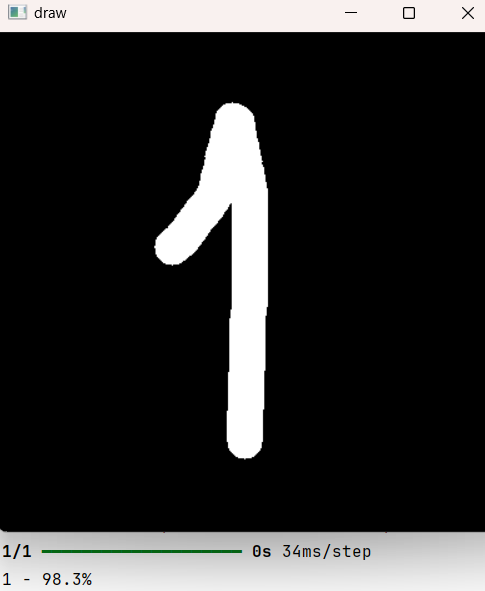
* Для загрузки данных используйте модуль **tensorflow.keras.datasets.mnist**. Убедитесь, что загруженная база сохраняется в каталоге скрипта (см. документацию к функции **load\_data()**).
* Модель сети должна иметь тип Sequential и состоять из трёх слоев: входного слоя типа Flatten (так как входные данные - 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 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 свёрточный слой **Conv2D** из 32 нейронов с окном 3х3 и функцией активации ReLU. Укажите форму входных данных (28,28,1).
2. Слой прореживания **Pooling2D** с окном 2х2.
3. Свёрточный слой **Conv2D** из 32 нейронов с окном 3х3 и функцией активации ReLU.
4. Слой прореживания **Pooling2D** с окном 2х2.
5. 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.

