Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**Департамент анализа данных и машинного обучения**

Пояснительная записка к курсовой работе по дисциплине

«Технологии анализа данных и машинное обучение»

на тему:

«Машинное обучение в задачах распознавания объектов на фотографии»

Выполнил:

студент группы ПИ19-2 факультета

информационных технологий

и анализа больших данных

Бедак И.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Научный руководитель:

доцент, к. ф.-м. н.

Романова Е.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2022 г

Оглавление

[Введение 2](#_Toc102614910)

[Теоретическая часть 3](#_Toc102614911)

[Что такое машинное обучение? 3](#_Toc102614912)

[Обучение с учителем 3](#_Toc102614913)

[Нейронные сети 4](#_Toc102614914)

[История создания 4](#_Toc102614915)

[Нейрон и нейросеть 4](#_Toc102614916)

[Функции активации 6](#_Toc102614917)

[Архитектуры нейронных сетей 9](#_Toc102614918)

[Преимущества и недостатки 10](#_Toc102614919)

[Практическая часть 12](#_Toc102614920)

[Выбор данных 12](#_Toc102614921)

[Аугментация обучающей выборки 13](#_Toc102614922)

[Создание нейросети 16](#_Toc102614923)

[Тестирование модели на реальных изображениях 24](#_Toc102614924)

[Заключение 26](#_Toc102614925)

[Источники 27](#_Toc102614926)

# Введение

С каждым годом количество информации в мире становится всё больше и больше. Часто бывает так, что человек что-то забывает, что-то изобретает.

Наш мозг является самым сложным и самым таинственным «инструментом» в нашем организме. Многие учёные пытались и пытаются до сих пор обуздать его природу.

Стоит только представить, что происходит внутри этого «черного ящика», в результате работы которого мы сейчас можем читать эту работу, смотреть на окружающих нас людей и понимать кто из них есть кто.

В результате ряда экспериментов было показано, что для того, чтобы человек распознал образ ему требуется 13 миллисекунд, что является очень быстрым результатом. И только представьте, как далеко шагнёт наука, если у человека получится воссоздать такой результат искусственно.

Цель данной работы исследовать возможные алгоритмы машинного обучения, а в практической части продемонстрировать процесс распознавания объектов на изображении машиной.

В дальнейшем весь приведённый код был написан с использованием языка программирования Python на платформе JupyterNotebook.

# Теоретическая часть

## Что такое машинное обучение?

Машинное обучение – это методы искусственного интеллекта, особенностью которых является не решение задач напрямую, а обучение модели используя набор схожих по смыслу задач.

В текущее время появляется всё больше и больше различных методов машинного обучения, однако можно с уверенностью выделить основные из них:

* Обучение с учителем
* Обучение без учителя
* Обучение с подкреплением

Отличаются они исключительно подходом к обучению модели и, в следствие, применяемыми алгоритмами. Так в обучении с учителем для каждого «прецендента» существует пара (входные данные – требуемый ответ). Таким методом обычно пользуются в задачах классификации. Обучение без учителя же уже не требует необходимый ответ, тем самым модель должна сгруппировать все существующие «преценденты» в кластеры исходя из их сходства между собой. Для обучения с подкреплением уже, в качестве исходных данных, приводятся пары (ситуация – принятое решение) а сама модель обучается, взаимодействуя с некой средой. Классический пример для такого типа обучения – генетические алгоритмы.

## Обучение с учителем

Так как задача данной работы – классификация, для дальнейшей работы будет выбрано обучение с учителем. Рассмотрим какие методы решения входят в данный тип машинного обучения.

* Байесовский классификатор (включает в себя такие алгоритмы как метод ближайших соседей)
* Нейронная сеть (персептрон, многослойный персептрон)
* Метод опорных векторов

Для каждого метода свойственны подходящие области применения и типы входных данных. Т.к. в своей работе я рассматриваю работу с изображениями, для дальнейшей работы будет выбраны нейронные сети, так как искусственные нейросети справляются с этой задачей намного лучше.

## Нейронные сети

### История создания

Началом эпохи искусственных нейронных сетей считается 1943 год. В этот год была выпущена одна из самых первых работ по этой теме. Авторами выступили Уолтер Питтс и Уоррен Мак-Каллок.

Через 6 лет была выпущена книга «Организация поведения» Дональд Хебб, в которой описали процесс обучения нейронной сети.

И наконец в 1957 был представлен первый математический прототип нейрона. Американский учёный Фрэнк Розенблатт показал миру перцептрон, который начали использовать сразу же для простейших задач.

Завершающим событием, разогнавшим развитие данной отрасли, можно считать работу советского учёного Галушкина А.И. В 1974 году он описал алгоритм метода обратного распространения ошибки, что позволяло обучать искусственные нейронные сети эффективнее.

### Нейрон и нейросеть

Еще с давних пор человечество интересовало устройство человеческого мозга. И в правду, человеческий мозг является уникальным органом, природа которого хоть и изучена, но не описана на 100% до сих пор.

Вся нервная система человека и не только представляется как внушающая по размерам биологическая нейронная сеть. Она обрабатывает сигналы от различных органов чувств (таких как глаза, кожа и т.д.) и отправляет эти сигналы прямиком в мозг. Давайте же рассмотрим структуру биологического нейрона и сравним его с искусственным.

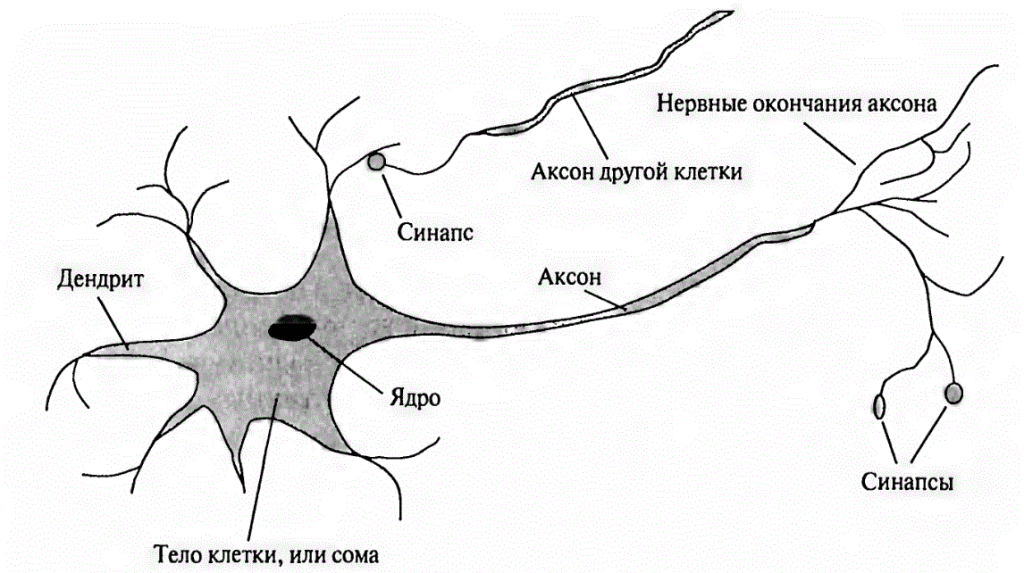


Рис 1. Строение нейрона

Вся нейронная система человека состоит из одинаковых компонентов, называемых нейроном. Главная часть нейрона – его ядро. Оно накапливает электрический заряд для передачи в другие нейроны. Так же у нейрона есть отростки двух типов. Первый тип – это маленькие отростки, называемые дендритами. Эти отростки служат «транспортным узлом» электрического разряда от других нейронов к нашему нейрону. Второй тип отростков – это большой отросток аксон. По этому отростку уже наш нейрон передаёт сигнал другим нейронам. Помимо различных отростков в транспортировке сигнала участвуют синапсы. Синапс – это место соприкосновения дендрита и аксона. Синапс может быть сильным, а может быть слабым. Его пропускная способность напрямую зависит от его силы. Синапс может изменяться со временем. Обучение таких нейронов – это и есть настройка силы синапсов для достижения необходимого результата.

К счастью, для описания искусственного нейрона нам не понадобятся различные синапсы, дендриты и т.д.

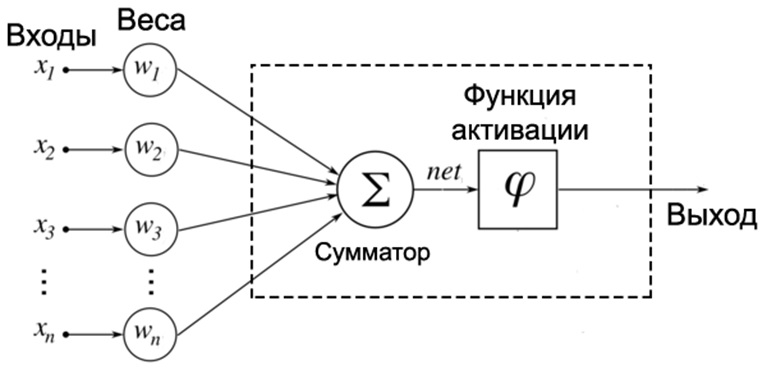


Рис 2. Искусственный нейрон

В математической модели нейрона есть входы, веса, сумматор, функция активации и выход. Работа такого нейрона заключается в получении сигналов с входов, учёт весов для каждого входа, суммирование всех полученных сигналов с входов и получение выходного сигнала с помощью функции активации. Математически работу такого нейрона можно представить в виде формулы

где – функция активации, x – входы и w – соответствующие им веса.

При объединении нескольких таких нейронов мы и получаем необходимую нам искусственную нейронную сеть.

### Функции активации

Из формулы выше очевидно, что функция активации напрямую влияет на выходной сигнал из нейрона. Рассмотрим, какие существуют функции активации:

* Ступенчатая функция активации

Изображение выглядит как седзи, кроссворд, с плиткой, общедоступный

Автоматически созданное описание

Рис 3. Ступенчатая функция активации

Задаётся формулой является самой простой функцией активации. Однако она не является непрерывно дифференцируемой монотонной функцией, поэтому она не может осуществлять такие алгоритмы, как обратное распространение ошибки.

* Линейная функция активации

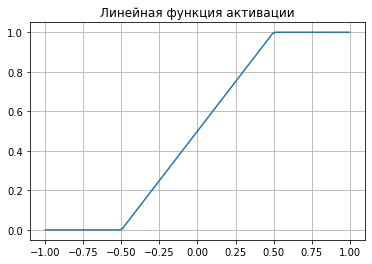


Рис 4. Линейная функция активации

* Сигмоида

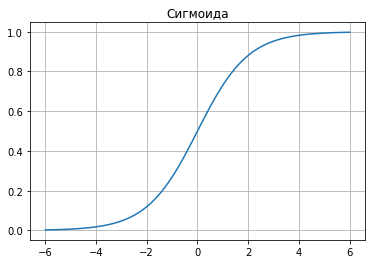


Рис 5. Сигмоидальная функция активации

Задаётся формулой . Данная функция сама по себе нелинейна, что даёт хороший результат, т.к. при значениях x близких к 0 функция реагирует сильно, когда же значения х далеки от 0, то функция реагирует слабо.

* ReLu

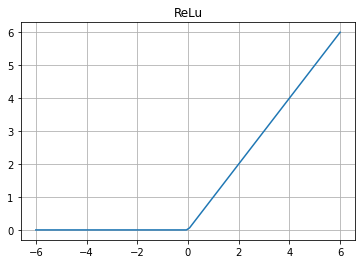


Рис 6. Функция активации ReLu 1

С одной стороны, эта функция выглядит очень похоже на линейную, однако по своей природе она нелинейна и комбинация таких функций тоже нелинейна. Однако для х < 0 функция ReLu является горизонтальной линией, это означает, что градиенты в этих значениях равны нулю. Из-за этого в последнее время вместо горизонтальной линии y = 0 часто берут y = 0.001\*x. Основное же преимущество данной функции является минимальные требования к вычислениям, что позволяет её применять в глубоких нейронных сетях

## Архитектуры нейронных сетей

Теперь мы знаем, из каких компонентов состоят нейронные сети. Для наглядности приведу схему типичной нейронной сети

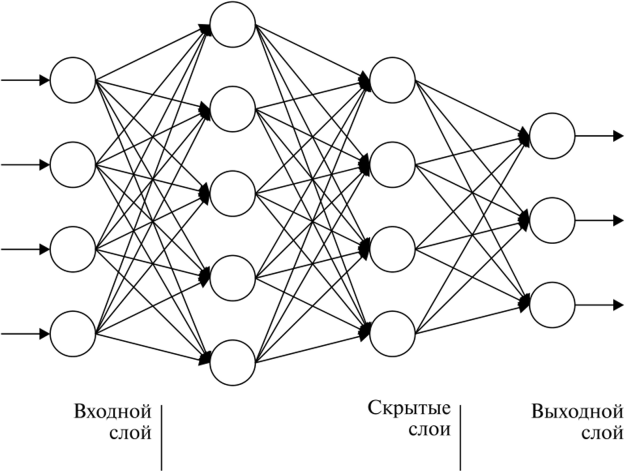


Рис 7. Нейронная сеть

Как мы можем заметить, нейросеть состоит из некоторого количество слоёв, состоящих из нейронов. Самый первый слой, в который поступают начальные данные называется входным слоем. Самый последний соответственно выходной. Между ними может быть какое-то количество внутренних или, как их еще называют, скрытых слоёв. Каждый нейрон связан со всеми нейронами из предыдущего и следующего слоёв.

Нейросети по своей структуре (архитектуре) можно разделить на следующие типы:

* Неглубокая нейросеть

Такие нейросети, как правило, состоят не более чем из двух скрытых слоёв. Иногда скрытые слои и вовсе отсутствуют. Используются же такие нейросети для простейших задач, в которых нет необходимости для сложных вычислений.

* Глубокая нейросеть

Такой тип нейросетей состоит из множества скрытых слоёв. Это лучший выбор для работы со сложными функциями

* Свёрточные нейросети

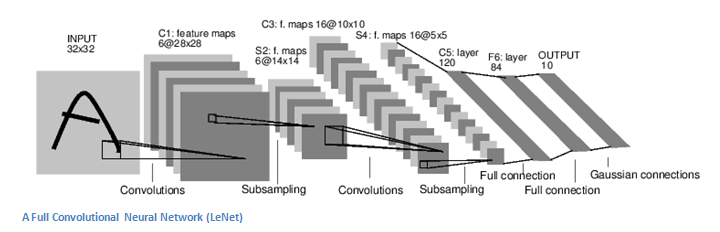


Рис 9. Схема свёрточной нейросети

Такие нейронные сети очень хорошо справляются с задачами классификации изображений. Когда изображение подаётся на вход в такую нейросеть, оно проходить ряд сверточных слоёв и слоёв объединения, после чего переходит в серию полносвязных слоёв (глубокая нейросеть) и генерируется результат.

По своей сути свёрточные слои подобны фильтрам, которые отбирают необходимые паттерны, схожие с какими-либо выходными классами

Прочитав достаточное количество информации по различным архитектурам, для дальнейшей работы будет выбрана архитектура свёрточных сетей из-за своей высокой эффективности в работе с изображениями.

## Преимущества и недостатки

Какими бы хорошими алгоритмами ни обладала нейронная сеть, у любого алгоритма есть свои преимущества и недостатки.

Преимущества:

* Свёрточные нейросети требуют меньше времени и вычислительных ресурсов, чем классические полносвязные сети при сопоставимом размере из-за свёрточных слоёв
* Так же свёрточные слои уберегают модель от банального запоминания пикселей и дают возможность запоминать области и образы

Недостатки:

* Необходимость в большом количестве примеров при обучении с учителем
* В большинстве имеет высокую эффективность только при работе с изображениями (встречаются применения в NLP, такие как «word2vec»)
* Высока вероятность переобучения
* Большое количество изменяемых параметров внутри свёрточных слоёв
* Ответ, как и у всех нейросетей, всегда приблизительный

Несмотря на представленные недостатки, свёрточные нейросети полностью подходят для исследуемой темы.

# Практическая часть

## Выбор данных

Выбор данных для обучения нейросети – один из важнейших этапов создания модели. Нужно учитывать достоверность данных, а так же их количество.

Прочитав некоторое количество статей по данной тематике, было принято решение использовать один из самых популярных датасетов в сфере компьютерного зрения, а именно CIFAR-10. Загрузить этот датасет возможно с помощью библиотеки tourchvision, которая хранит в себе множество различных датасетов для задач компьютерного зрения, в том числе и CIFAR-10.

Изображение выглядит как текст

Автоматически созданное описание

Рис 10. Загрузка датасета

Этот датасет представляет из себя 60000 изображений разрешением 32х32 пикселей. Внутри этого содержатся изображения, принадлежащие к одному из следующих классов: 'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck'.

Изображение выглядит как текст, размытый

Автоматически созданное описание

Рис 11. Пример изображения

Конечно же, изображения, полученные на вход при уже тестировании реальными картинками, не будут иметь размерность 32х32, поэтому при применении модели после её обучения необходимо написать функцию предобработки входных данных.

Изображение выглядит как текст

Автоматически созданное описание

Рис 12. Функция получение предсказания

В данной функции происходит приведение входного изображения к необходимым размерностям. Так же данная функция предусматривает возможность работы модели на графическом процессоре.

## Аугментация обучающей выборки

В выбранной выборке присутсвует 60000 изображений, причём они сразу разделены на тренировочную и тестовую выборки.

Изображение выглядит как текст

Автоматически созданное описание

Рис 13. Размеры выборки

Однако, хоть данных и достаточно, для лучшего обучения, многие проводят предварительную аугментацию данных.

Для чего это необходимо? Зачастую бывает, что допустим в тренировочных и тестовых изображениях присутствуют изображения котов, не смещённых вбок или расположенных строго вертикально. Однако реальные изображения котов могут быть в любом их положении. Из-за этого модель может распознавать исключительно котов, повёрнутых прямо к камере, остальные же модель определит к любому случайному классу.

Для этого перед обучением исходные данные проходят процесс предобработки, использующих различные трансформации.

Рассмотрим эти трансформации на изображении выше (Рис. 11)

* Горизонтальный поворот



* Поворот по центру



* Горизонтальное и вертикальное смещение



* Кадрирование



Все эти изменения происходят рандомно в каком-то заданном интервале. Для того, чтобы воспользоваться ими необходимо указать их в параметре transform при загрузке датасета:

Изображение выглядит как текст

Автоматически созданное описание

Рис 14. Аугментация данных

## Создание нейросети

Самое первое, что необходимо сделать для создания нейросети – импортировать необходимые библиотеки.

Изображение выглядит как текст

Автоматически созданное описание

Рис 15. Импорт библиотек

В своей работе я буду использовать библиотеку torch.

Torch – это фреймворк машинного обучения для языка Python с открытым исходным кодом, созданный на базе Torch. Разрабатывается преимущественно группой искусственного интеллекта Facebook.

Далее обозначим ключевые константные значения, такие как количество эпох обучения, количество изображений в одной эпохе, скорость обучения, количество выходных классов и т.д.

Изображение выглядит как текст

Автоматически созданное описание

Рис 16. Ключевые константы

После этого необходимо загрузить сам датасет (либо удостовериться в его целостности).

Изображение выглядит как текст

Автоматически созданное описание

Рис 17. Загрузка датасета

Для достоверности выведем любое изображение.

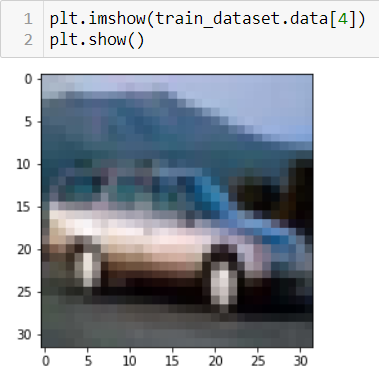


Рис 18. Проверка целостности изображений

Как можем заметить изображение выводится корректно и можем наблюдать на нём автомобиль.

Воспользуемся встроенным методом и проверим, те ли классы у нашей выборки.

Изображение выглядит как текст

Автоматически созданное описание

Рис 19. Классы датасета

Как можно заметить – всё хорошо. Далее нам необходимо создать некий итератор, который будет подавать картинки при обучении «мини-пакетами». Для этого воспользуемся классом torch.utils.data.DataLoader, который делает это довольно просто, а также использует модуль multiprocessing для ускорения поиска изображений.

Изображение выглядит как текст

Автоматически созданное описание

Рис 20. Итератор по датасету

Проверим его работу и выведем по 6 изображений для каждого класса в датасете.

Изображение выглядит как текст

Автоматически созданное описание

Рис 21. Тестирование итератора

Получим следующее:

Изображение выглядит как текст, другой

Автоматически созданное описание

Рис 22. Примеры картинок

DataLoader работает исправно. Также можем отметить, что предварительная аугментация данных тоже отработала исправно, некоторые из картинок перевёрнуты, повёрнуты и т.д.

Перед созданием нейросети необходимо добавить возможность обучать её на графическом процессоре (GPU), т.к. он быстрее обрабатывает изображения и, вследствие, ускорит процесс обучения нашей модели.



Рис 23. Переключение на GPU

Стоит заметить, что это возможно исключительно на видеокартах NVIDIA с технологиями GTX или RTX. Во всех остальных случаях обработчиком всех вычислений станет главный процессор компьютера.

Теперь можно перейти к созданию нейросети.

Изучив различные модели, я решил воспроизвести уже известную модель VGG16.

VGG16 – это свёрточная нейросеть, предложенная K. Simonyan и A. Zisserman в своей статье “Very Deep Convolutional Networks for Large-Scale Image Recognition”. Оригинальная модель достигает точности 92.7% при тестировании на датасете из 14 миллионов изображений, включающих в себя 1000 классов. Для создания такого прототипа найдём схему оригинальной нейронной сети.

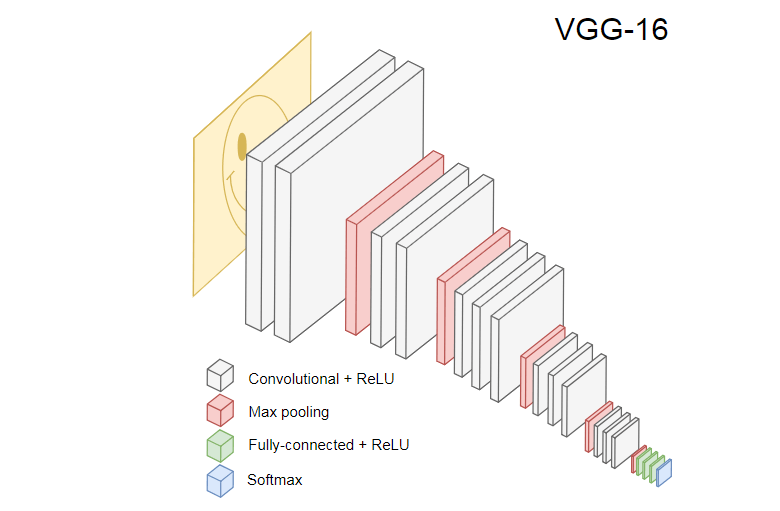


Рис 24. Схема VGG16

Модель содержит в себе 13 свёрточных слоёв, каждый из которых использует функцию активации ReLu и 3х слойную полносвязную нейросеть в конце, которая тоже использует функции активации ReLu.

Реализация данной модели находится в классе MyCNNModel в файле Бедак\_Иван\_ПИ19-2.ipynb

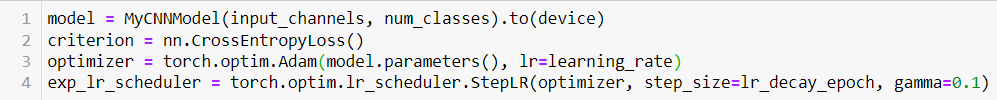


Рис 25. Создание модели

Создадим экземпляр модели и сразу же направим все вычисления на используемый процессор.

В качестве оптимизатора обучения модели был выбран алгоритм Адам.

Приступим к обучению модели. В течении всех эпох обучения будем получать данные из DataLoader, получать выходные значения, производить шаг градиента.

Изображение выглядит как текст

Автоматически созданное описание

Рис 26. Обучение модели

В конце каждой эпохи производим замер на тестовых значениях и выводим в консоль.

Изображение выглядит как текст, внутренний

Автоматически созданное описание

Рис 27. Вывод информации об обучении

Выведем график точности модели на тестовых данных для проверки на наличие переобучения.

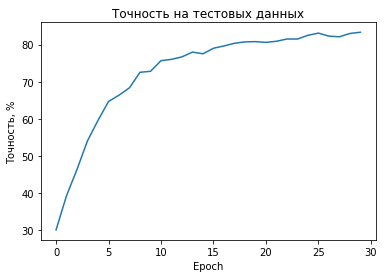


Рис 28. График точности

Из графика можно сделать вывод, что переобучения нет (т.к. замер производился на тестовых данных, а не на тренировочных)

Финальной точностью модели будем считать 83.9%, что я считаю довольно неплохим результатом. Сохраним нашу модель в файл, чтобы потом можно было ей воспользоваться.



Рис 29. Сохранение модели

Для лучшего анализа обученной модели визуализируем результаты тестирования модели на тестовых данных.

Изображение выглядит как текст

Автоматически созданное описание

Рис 30. Повторное тесторование модели

Построим тепловую карту для визуализации значений «ответ»-«результат работы модели».

Изображение выглядит как текст, электроника, клавиатура

Автоматически созданное описание

Рис 31. Тепловая карта тестирования

Сразу можно заметить, что самые большие числа сосредоточены на главное диагонали – ячеках, показывающих правильно угаданный класс.

Так же можно отметить, что, из ошибок модели, нейросеть чаще всего принимала кота за собаку и собаку за кота.

## Тестирование модели на реальных изображениях

Для тестирования была скачана картинка кота из интернета cat.jpg в папке img.



Рис 32. Изображение из интернета

Протестируем наше изображение на обученной модели. Для этого воспользуемся написанной функцией, показанной на рис. 12.

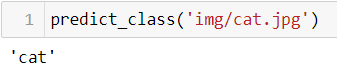


Рис 33. Результат тестирования

Как видим, обученная модель успешно справляется с поставленной целью.

# Заключение

В ходе данной работы:

* были рассмотрены различные подходы к машинному обучению, различные архитектуры нейросетей.
* был выбран самый эффективный алгоритм обучения нейросети для поставленной задачи
* в практической части была воссоздана одна из возможных реализаций модели распознавания объектов на фотографии
* была протестирована обученная модель и был получен приемлемый результат в 83.9% точности

После полученных результатов можно сделать вывод о высокой эффективности алгоритмов машинного обучения, а в частности глубокие и свёрточные нейросети, в задачах распознавания изображений.

Однако изображения не единственная сфера для создания моделей машинного обучения и именно поэтому я считаю, что будущее за искусственным интеллектом.

# Источники

1. Андреас Мюллер, Введение в машинное обучение с помощью Python [Руководство для специалистов по работе с данными] / Андреас Мюллер, Сара Гвидо, – Москва, 2016-2017. – 393 c.
2. Коротеев М.В. Об основных задачах дескриптивного анализа как подготовительного этапа машинного обучения - 2018.
3. Коротеев М.В. Учебное пособие по дисциплине «Анализ данных и машинное обучение» - Москва, 2018.
4. Karen Simonyan, Andrew Zisserman. VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION – University of Oxford – 2015.
5. Машинное обучение. Википедия. [Электронный ресурс] URL: <https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5> (дата обращения 03.04.2022)
6. PyTorch. Официальная документация. [Электронный ресурс] URL: <https://pytorch.org/> (дата обращения 20.04.2022)