|  |  |
| --- | --- |
| Федеральное агентство железнодорожного транспорта  Омский государственный университет путей сообщения  Кафедра «Автоматика и системы управления»  ЗНАКОМСТВО С PYTHON  Лабораторная работа № 8  по дисциплине «Информационные технологии» | |
|  | Студент гр. 23М                            Е.М Кривощеков.                            Д.А Григорьев  «     »                         2024 г.  Руководитель –  старший преподаватель кафедры «АиСУ»                            Т.В. Васеева  «    »                  2024 г. |
| Омск 2024 | |

# Цель работы

Целью данной лабораторной работы является знакомство языком программирования Python, а также с сортировкой фото с помощью нейронной сети.

# Выполнение работы

## Для начала среду разработки Anaconda (рисунок 1)

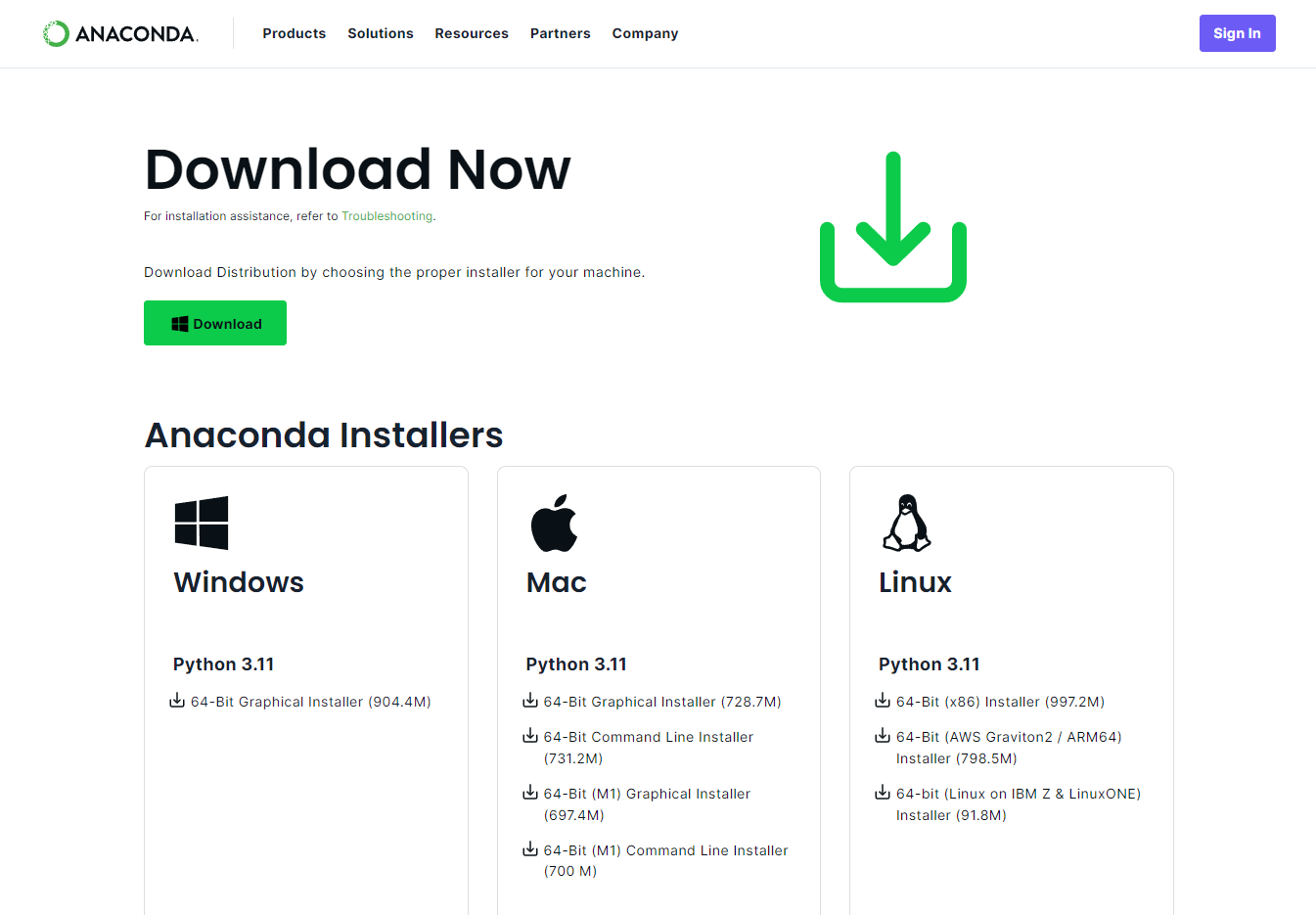


Рисунок 1 – Установка Anaconda

## Установим Tensorflow (рисунок 2)

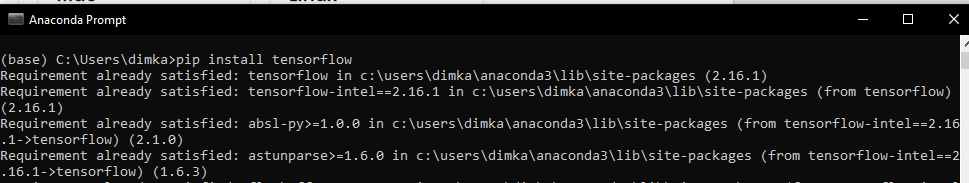


Рисунок 2 – Установка tensorflow

## Подгрузим нужные нам библиотеки (рисунок 3)

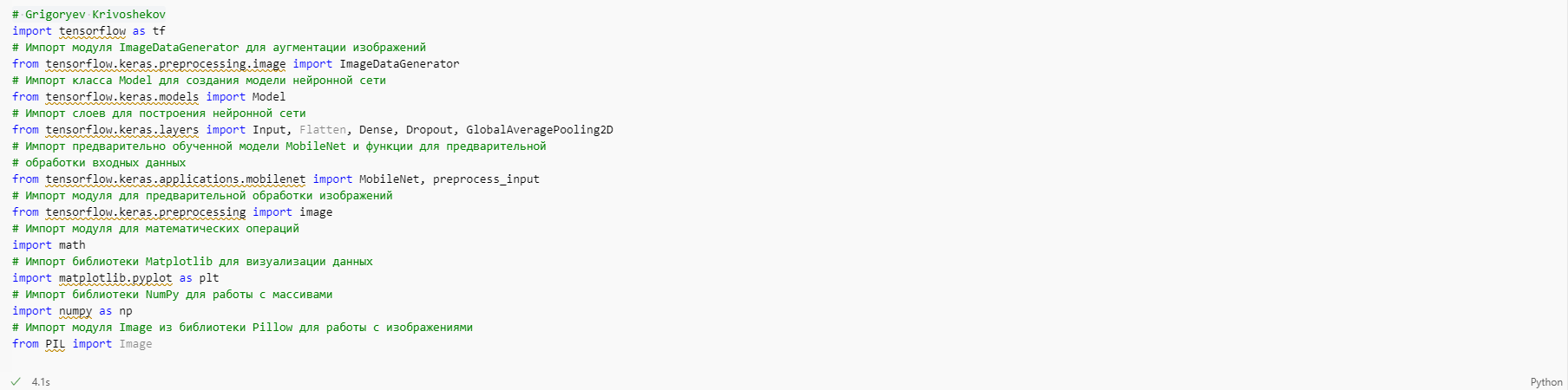


Рисунок 3 – Библиотеки

## Скачаем файлы для обучения нашей нейросети (рисунок 4)

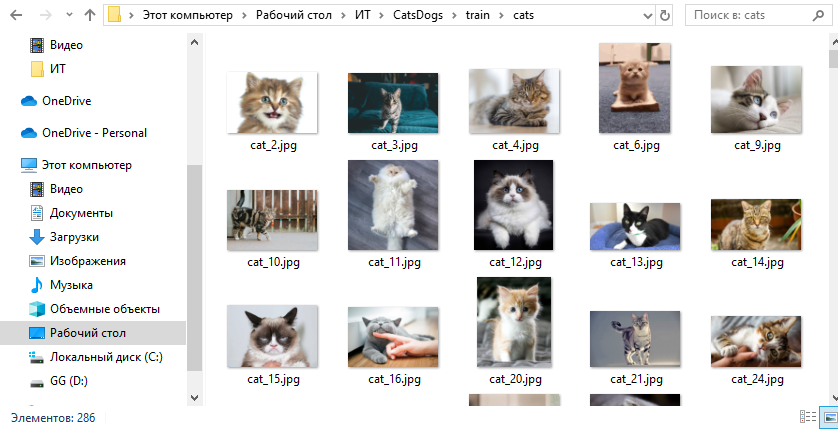


Рисунок 4 – Файлы для обучения

## Подключаемся к серверу Yupiter и подключаем python3

В приложении Anaconda-Navigator нажимаем Jupiter notebook после чего появляется терминал, из него мы копируем адрес сервера  
(рисунок 5). В VSCode нажимаем выбрать ядро. Выбираем python3 и вставляем скопированный токен (рисунок 6).

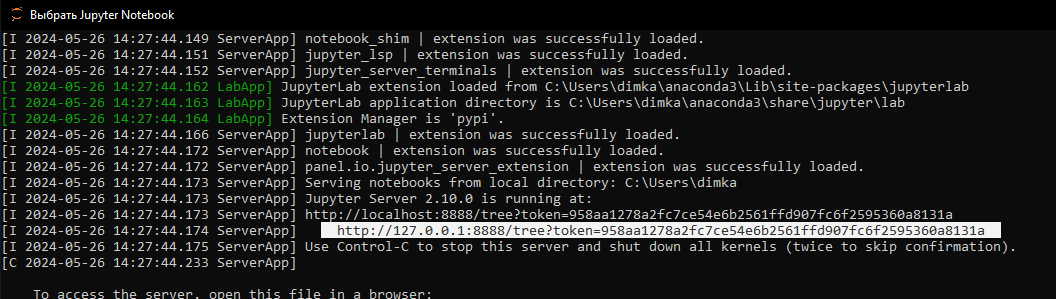


Рисунок 5 – Адрес сервера

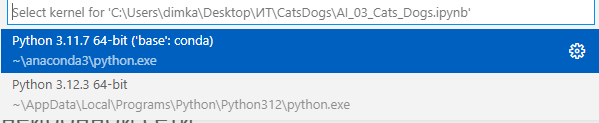


Рисунок 6 – Pychon3

## Эти константы (рисунок 7) служат для установления параметров и структуры обучения нейронной сети в процессе машинного обучения, особенно при работе с изображениями

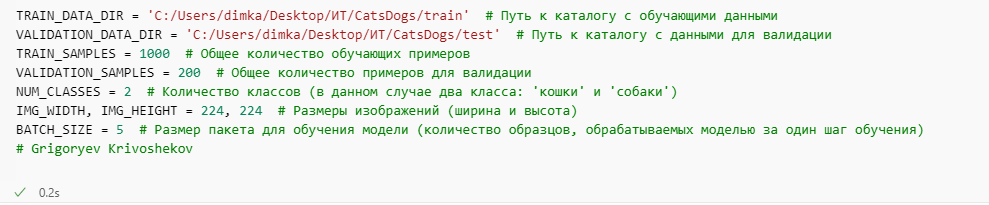


Рисунок 7 – Константы

## Создаются два генератора данных. Для обучения нейросети недостаточно пары сотен фотографий.

Поэтому будет использоваться такой прием как аргументация – это техника, используемая в машинном обучении, особенно при работе с изображениями, для искусственного увеличения объема обучающих данных путем создания модифицированных версий уже существующих данных. Это особенно важно в ситуациях, когда количество доступных данных ограничено, чтобы обеспечить достаточное разнообразие и предотвратить переобучение модели (рисунок 8).

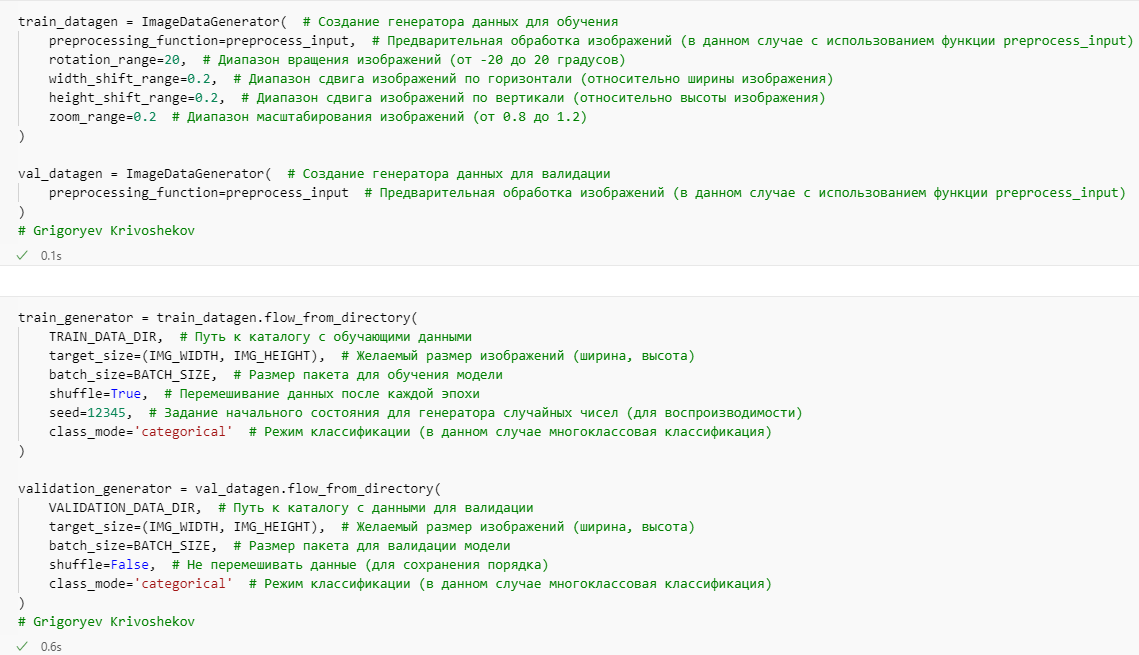


Рисунок 8 – Аргументация

## Создание модели на основе нейронной сети

В базовой модели нужно отключить обучение (для всех слоев устанавливается trainable = False). Входное изображение поступает на вход базовой модели. После базовой модели добавляется несколько своих слоев:слой пулинга GlobalAveragePooling2D; полносвязный слой из 16 нейронов Dense; используется механизм Дропаут (случайное отключение нейронных связей во время обучения); последний слой должен выдавать два значения (вероятности того, что на фото кошка или собака – два класса). Таким образом мы решаем задачу двухклассовой классификации (рисунок 9).

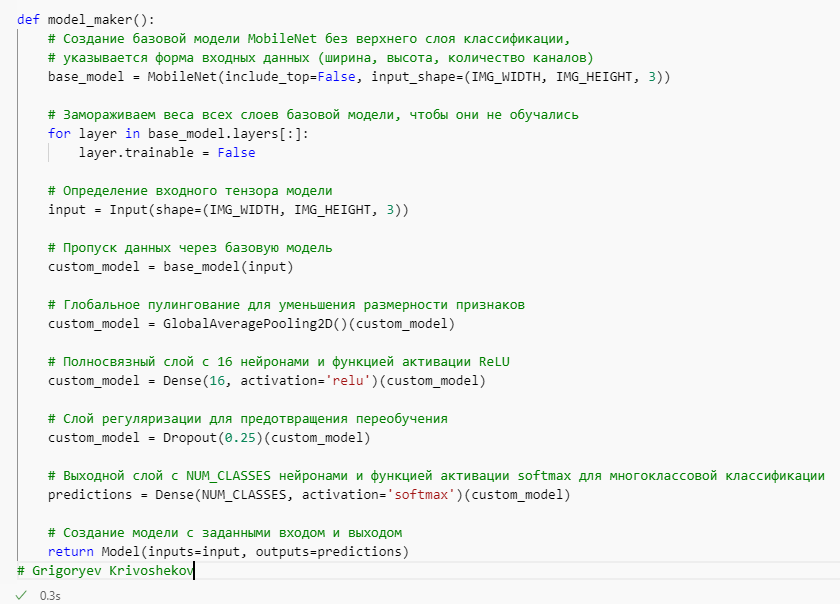


Рисунок 9 – Создание модели

## Компилируем и обучаем модель (рисунок 10)

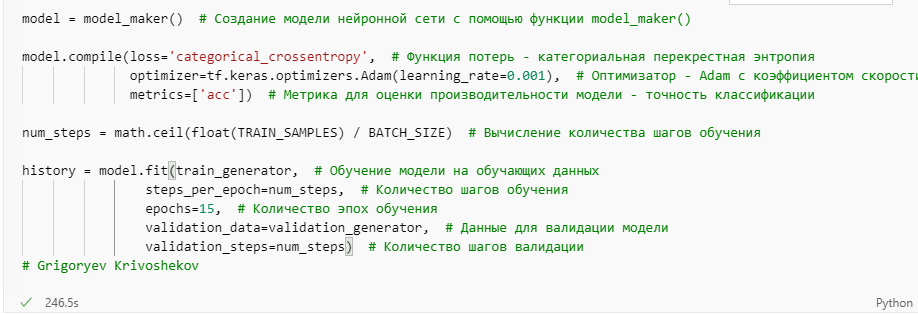
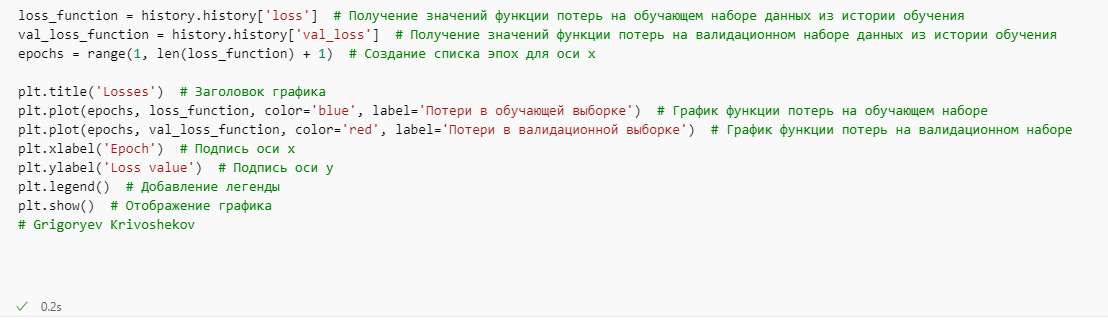


Рисунок 10 – Компиляция и обучение модели

## Построим график для визуализации обучения (рисунок 11)

  
Рисунок 11 – График

## Визуализированная проверка изображений (рисунок 12)

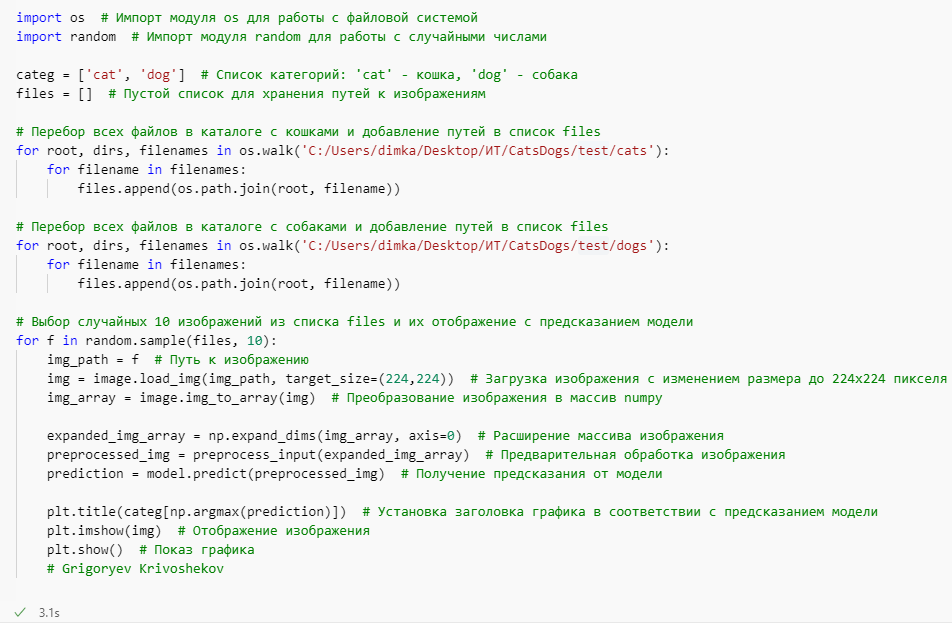


Рисунок 12 – Проверка изображений

## Теперь модель можно использовать и проверить, как она работает, Наша модель обучилась за 4 минуты. (рисунки 13-15)

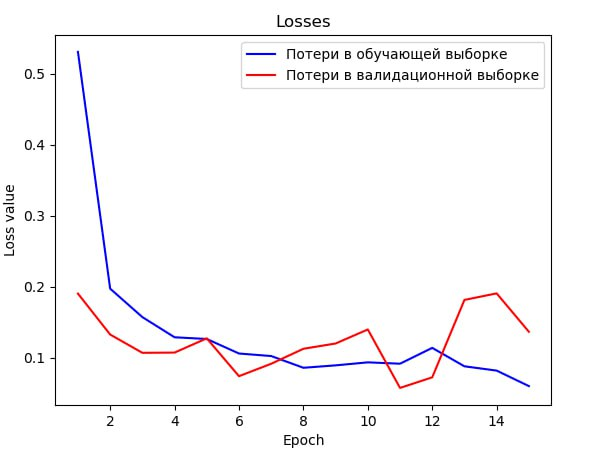


Рисунок 13 – График

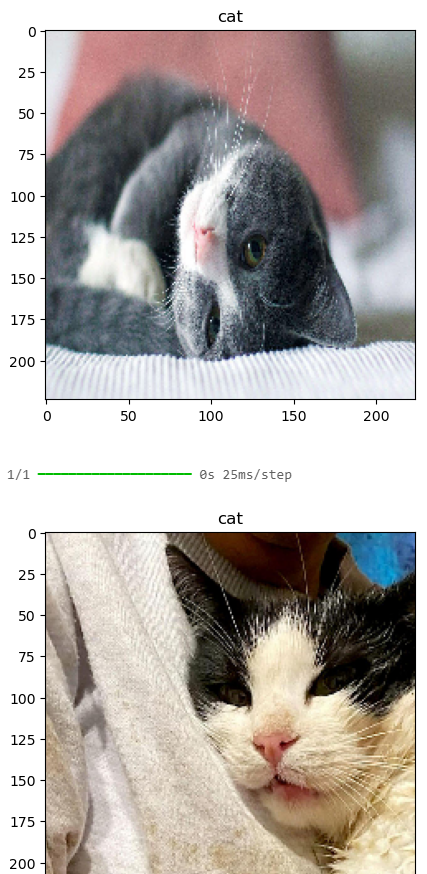


Рисунок 14 – Соответствие изображений

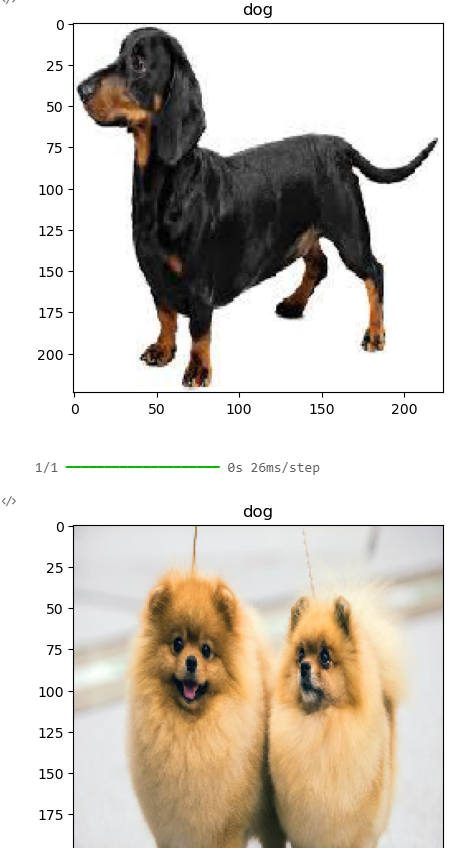


Рисунок 15 – Соответствие изображений

## Попробуем поменять количество эпох обучения (рисунок 16) и полносвязный слой с 16 нейронами и функцией активации ReLU (рисунок 17)

После посмотрим, как изменился наш график (рисунок 18). Наша модель обучилась за 14,5 минут.

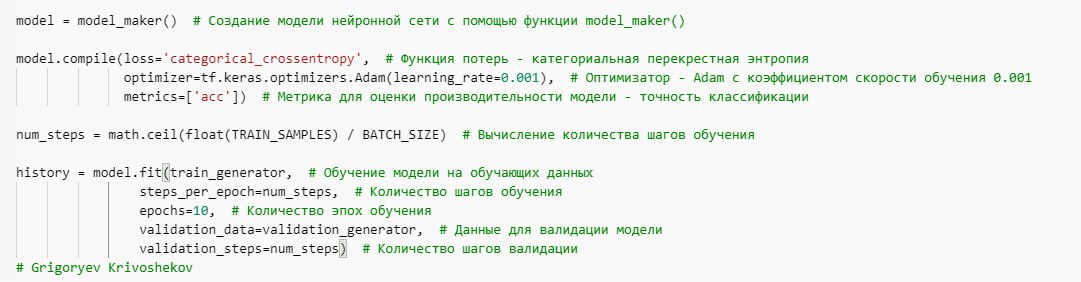


Рисунок 16 – Изменили количество эпох



Рисунок 17 – Изменили dense

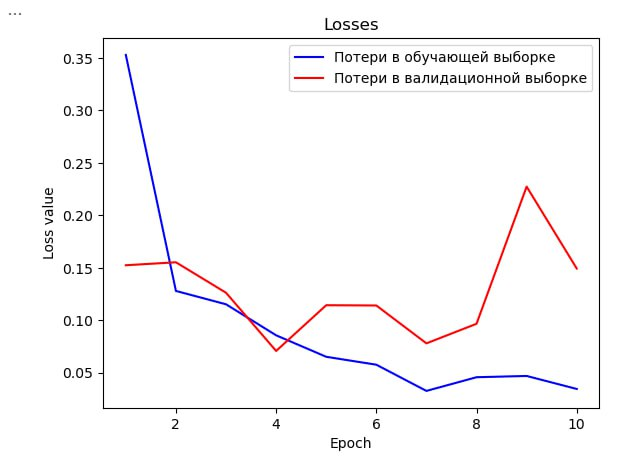


Рисунок 18 – Измененный график

# Задания для защиты

## Какие есть еще готовые нейронные сети, которые умеют выделять осмысленно информацию, кроме MobileNet?

YOLO (You Only Look Once) – архитектура нейронных сетей, предназначенная для детекции объектов на изображении. Один из способов решения задачи детекции заключается в разбиении изображения на квадратные области, затем классификация этих областей на наличие объекта и классификация самого объекта.

YОLOv8 и МоbіlеNеt V2 – это модели, которые часто используются в проектах компьютерного зрения.

YОLOv8 – это передовая модель для обнаружения объектов и сегментации изображений, созданная Ultrаlуtісs, разработчиками YOLOv5. Он использует архитектуру YOLOv5 и СNN. Эта модель обучается с использованием РуТоrсh

MоbіlеNеt V2 – это модель GооglеАІ, которая хорошо подходит для классификации в реальном времени на устройстве. Отличается от МоbіlеNеtSSD, Sіnglе Shоt Dеtесtоr. Эта модель использует технику переноса обучения с ІmаgеNеt на ваш набор данных.

Обе модели имеют свои преимущества и могут быть выбраны в зависимости от конкретных требований проекта. Например, YОLОv8 может быть предпочтительнее для задач, требующих точного обнаружения объектов и сегментации, в то время как МоbіlеNеt V2 может быть более подходящим для задач, требующие быстрой классификации на устройстве.

SSD (Single Shot MultiBox Detector) – используются наиболее удачные «хаки» архитектуры YOLO (например, non-maximum suppression) и добавляются новые, чтобы нейросеть быстрее и точнее работала. Отличительная особенность: различение объектов за один прогон с помощью заданной сетки окон (default box) на пирамиде изображений.

SqueezeNet – очень маленькая, но точная нейросеть. Сама по себе не решает задачу обнаружения объектов. Однако может применяться при комбинации различных архитектур. И использоваться в мобильных устройствах. Отличительной особенностью является то, что сначала данные сжимаются до четырех 1×1 сверточных фильтров, а затем расширяются до четырех 1×1 и четырех 3×3 сверточных фильтров. Одна такая итерация сжатия-расширения данных называется «Fire Module».

DeepLab (Semantic Image Segmentation with Deep Convolutional Nets) – сегментация объектов на изображении. Отличительной особенностью архитектуры является разряженная (dilated convolution) свертка, которая сохраняет пространственное разрешение. Затем следует стадия постобработки результатов с использованием графической вероятностной модели (conditional random field), что позволяет убрать небольшие шумы в сегментации и улучшить качество отсегментированного изображения.

## Как влияет количество эпох обучения на результат нейронной сети MobileNet?

Количество эпох обучения влияет на качество модели. С увеличением числа эпох модель получает больше возможностей для изучения разнообразных примеров из обучающего набора данных, что может улучшить ее способность обобщать информацию на новых данных. Однако слишком большое количество эпох может привести к переобучению, когда модель слишком точно подстраивается под обучающий набор и плохо работает на новых данных.

## Отсортируем собственные фото с помощью нейронной сети

Для этого нам потребуется найти примерно по 100 фотографий разных машин, например human и monkey. Поставим нашей модели обучение на 50 эпох. По итогу модель обучилась за 7,5 минут. Посмотрим на результат обучения (рисунок 19-2).

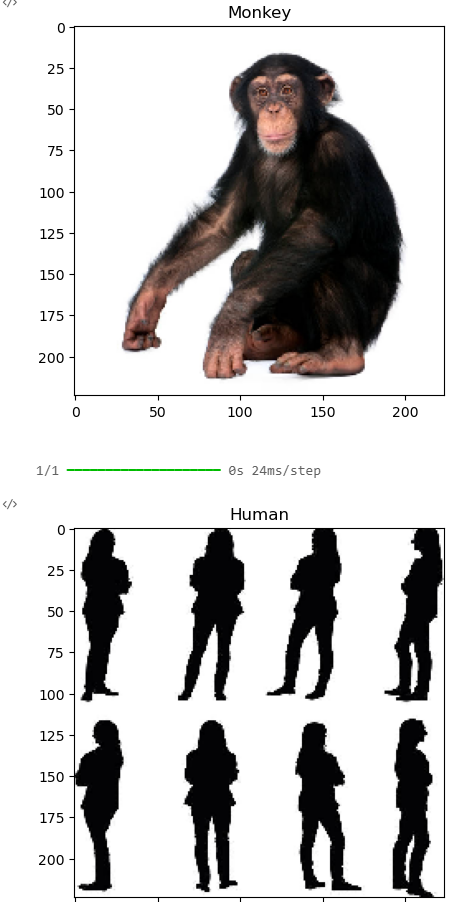


Рисунок 19 – Соответствие изображений

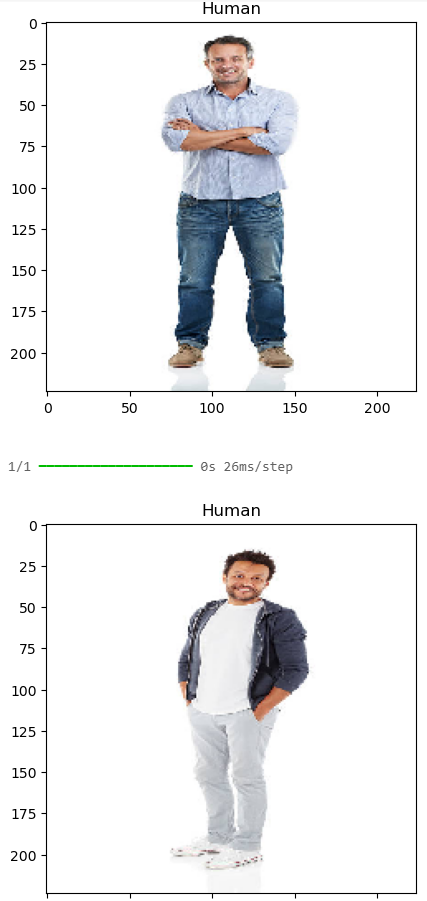


Рисунок 20 – Соответствие изображений

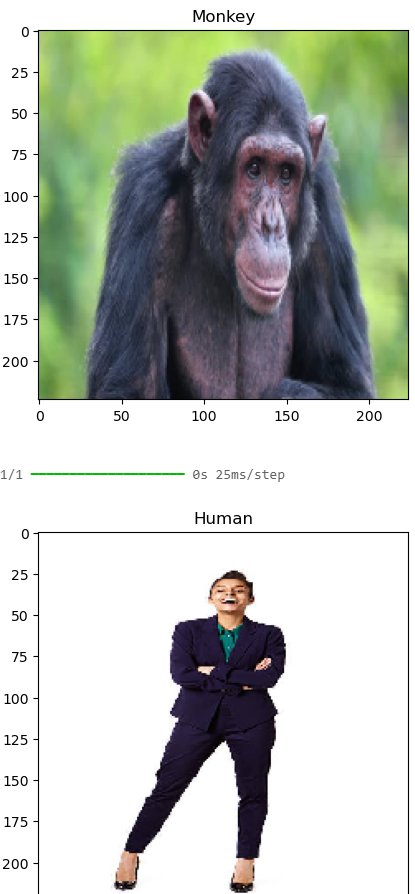


Рисунок 21 – Соответствие изображений

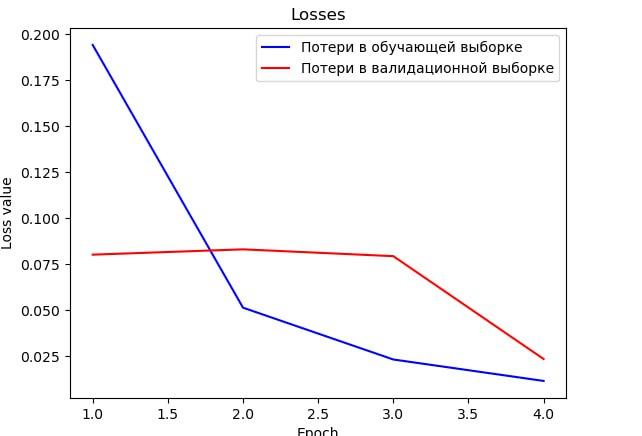


Рисунок 22 – График

Вывод

В ходе данной лабораторной работы мы познакомились с языком программирования Python. Мы научились использовать этот язык для обучения нейронной сети. Испытали эту сеть сортируя фото кошек и собак. А также, протестировали со своими картинками разных людей и обезьян. В ходе выполнения работы ошибок с определением картинок не возникло. Данная лабораторная работа нам очень понравилась, она была достаточно легкой и интересной.