# Описание работы (теория)

## Принцип работы бота:

Доступ к боту осуществляется по уникальному токену, сгенерированному при его создании.

В начале бот выводит вводное сообщение, в котором описано, как начать работу. Пользователю будет предложено воспользоваться каомандой /begin для инициализации процесса обработки фотографии.

Далее процесс идёт по следующему алгоритму: пользователь загружает первое изображение (важно чтобы изображение было в формате .png или .jpg и выбрана опция «сжать изображение» в Telegram). Сохранение информации о фотографиях проиходит в словаре. Ключём является id чата с пользователем, значением массив фотографий, с подобранными уникальными именами. Разделение фотографий из сообщений происходит по принципу, наличия уникального ключа: при обработке первого сообщения ключ создаётся, а в массив изображений загружается первое изображение, при обработке второго сообщения проверяется наличие ключа, и в случае положительного результата в массив значений записывается информация о второй и выходной фотографиях. Это необходимо для того, чтобы не происходило путаници при работе с ботом нескольких пользователей.

После этого пользователь может выбрать: отменить процесс с помощью кнопки «Stop», которая переведёт бота в началное состояние до /begin, или продолжить с помощью «Continue». Во втором случае предлагается загрузить изображение со стилем.

Затем пользователю дадут выбрать количество итераций с помощью соответствующих кнопок, либо завершить процесс той же кнопкой «Stop». Так, нейронная сеть начнёт работу по наложению стиля на исходное изображение. Массив значений каждого пользователя подаётся на вход нейронной сети с указанием количества итераций. В процессе работы нейронной сети создаётся и сохраняется итоговый файл, который в последствии передаётся ботом пользователю. После программа очищает информацию о текущем пользователе, удаляя загруженнеы и созданные, его ключ и значение по ключу из словаря, и бот переходит в начальное состояние.

Также в любой момент времени вы можете вызвать команду «/info» которая выведет подсказку на любом этапе процесса.

## Принцип работы нейронной сети:

Для работы с изображениями будет использоваться известная свёрточная нейронная сеть VGG19, которая предварительно настроена на задачу классификации изображений. Принцип передачи стиля заключается в определении двух функций, одна из которых показывает отличие двух изображений, а другая описывает разницу между двумя стилями изображений. Сеть пытается преобразовать входное изображение так, чтобы минимизировать расстояние от функции различия изображения и функции различия стиля.

Функция начала работы нейронной сети получает на вход два изображения, которые мы представляем в виде массивов NumPy и подготавливаем к обработке сетью VGG19 с помощью функции библиотеки Keras.

После этого необходимо настроить модель VGG19, в которую мы загружаем промежуточное слои - это карты признаков, которые по мере углубления становятся более упорядоченными, они позволят сети лучше понимать изображения и выявлять особенности. Функция построения модели возвращает данные с промежуточных слоёв VGG19.

Теперь нужно определить функции потерь, о которых говорилось выше. Для различия изображений, в кратце, она описывает расстояние содержимого Lcontent между входным изображением X и изображением контента P по формуле:



Здесь Fˡᵢⱼ(x) и Pˡᵢⱼ(x) описывают соответствующие промежуточные представления о X и P.

Для различия стилей этот процесс заключается в описании функции потерь для стиля главного изображения X, изображения стиля A и расстояния между представлениями матрицей Грама стиля этих двух картинок.

Суммарное влияние каждого слоя на функцию потерь:



Здесь Gˡᵢⱼ — это внутреннее произведение между векторизованной картой признаков i и j в слое L.

Gˡᵢⱼ и Aˡᵢⱼ — это соответствующие представления на слое L входного изображения X и изображения стиля A.

Nl описывает количество карт объектов, каждая из которых имеет размер Ml = (высота \* ширина).

Функция потерь всех слоёв:



Тут взвешивается влияние потери каждого слоя от какого-либо фактора wl.



После работы сети получается массив NumPy выходного изображения, который мы дообрабатываем от VGG19 и выводим.