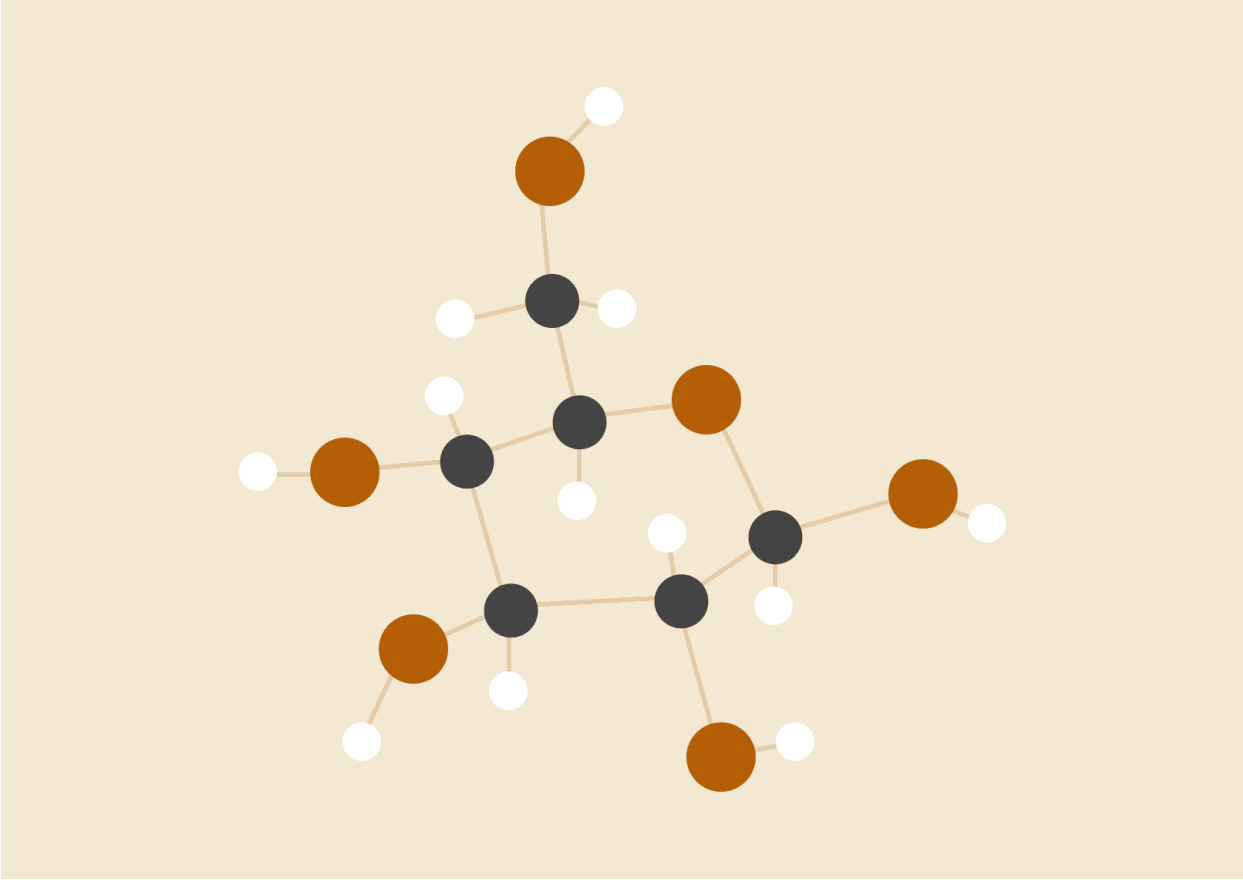
Отчет

по итоговому проекту: “Telegram Bot для стилизации изображений”



Александров Иван

10.02.2022

Deep Learning School

2021 осень, семестр 1, базовый поток

Stepic id: 99686862

GitHub code: https://github.com/Ivanich-spb/TGbot_NNStyleTransfer

Telegram Bot: https://t.me/NNTransferStyleBot

ВВЕДЕНИЕ

За основу взят baseline из лекции по StyleTransfer, реализующий алгоритм переноса стиля Леона А. Гатиса, Александра С. Эккера и Маттиаса Бетге.
(https://arxiv.org/abs/1508.06576)

Бот реализован на асинхронном фреймворке aiogram, размещён на хостинге FirstVDS (1 core cpu, 1 Gb RAM)

ТЮНИНГ МОДЕЛИ

- добавил нормализацию градиентов, чтобы бороться с взрывом лоссов
- слегка потюнил параметры оптимизатора и модели
- экспериментировал с разными оптимизаторами - наиболее “красивые” результаты получались с LBFGS
- в процессе деплоя на сервер оказалось, что при импорте pretrained VGG19 происходил скачек потребления памяти до 1.2Gb и соответственно процессу наступал kill. Обрезал неиспользуемые слои и сохранил в отдельном файле (model/my_new_model.pth). Получился файл 4Mb вместо 550. Потребление памяти сократилось почти вдвое - полезный опыт!

TELEGRAM BOT (фишки)

- реализован с помощью aiogram FSM(Finite State Machine) - следит за последовательностью действий пользователя, чтобы фото загружались одно за другим с возможностью отмены промежуточных действий, подгружает свои клавиатуры на каждый этап
- считывает токен бота из переменной окружения
- показывает пример работы бота
- проверка контента (проверяет, что загружены именно фотографии и не пускает на последующие шаги)
- предоставляет пользователю возможность выбрать из предустановленных стилей по кнопке
- после отправки результата пользователю - удаляет весь его контент с сервера

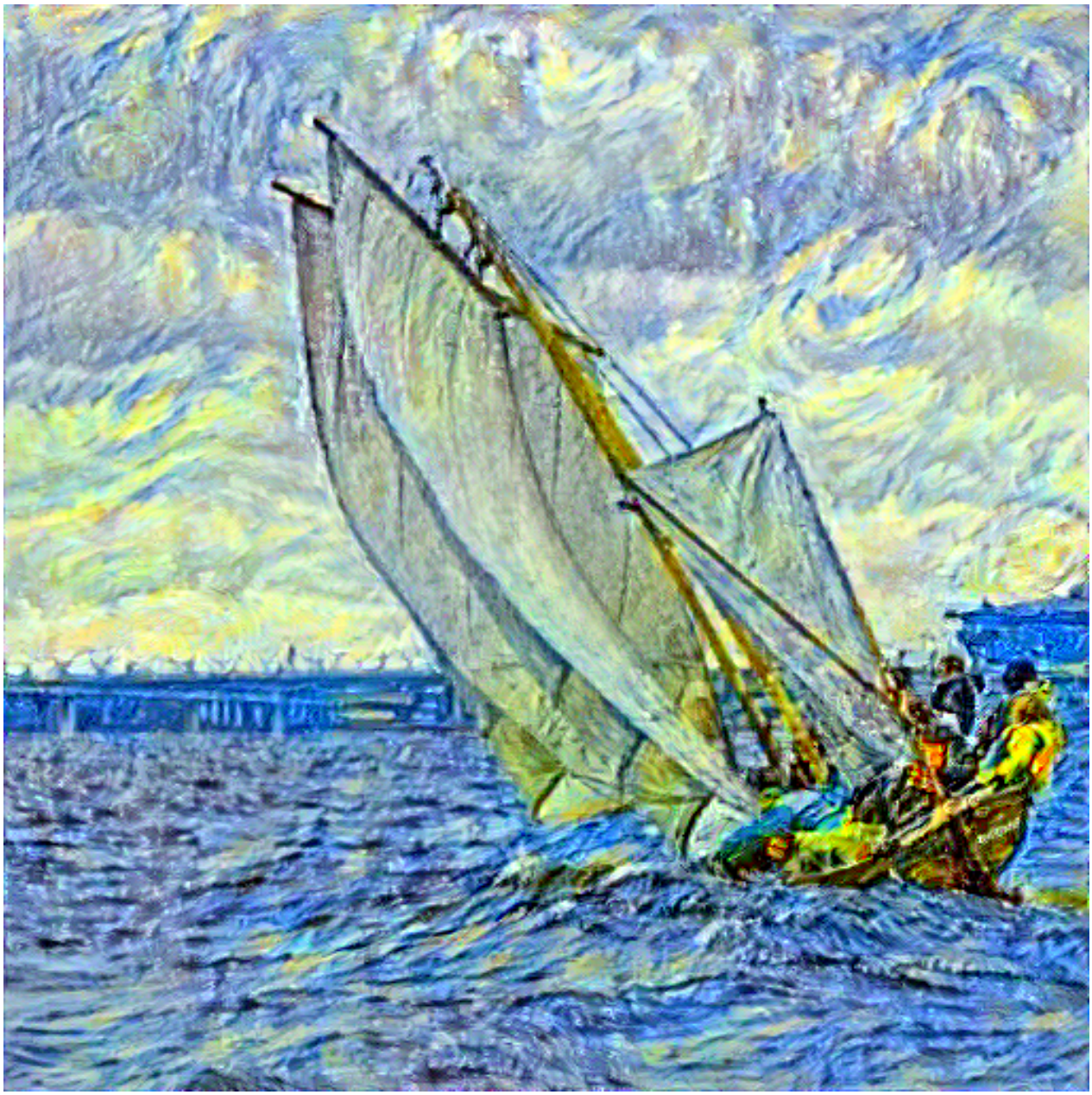
- пересчитывает время ожидания, в зависимости от того сколько человек находится в очереди на обработку (когда заработает асинхронность)

Пока не получилось:

Не получилось справиться с блокировкой бота на время работы модели. Пробовал модули threading и multiprocess, но бот все равно блокировался. Удалось добиться от Multiprocess, чтобы он запускал отдельный интерпритатор с моделью и считал в отдельном процессе, но не помогло. В принципе, появилось понимание как это можно сделать через очередь, но времени реализовать и протестировать не хватило - надеюсь скоро допилю.

На реализацию трансфера с помощью GAN тоже не хватило времени, да и ресурсов требовалось в разы больше - так что пока только ознакомился и погонял в колабе немного.

ПРИМЕР РАБОТЫ:



DEPLOY

В процессе деплоя пришлось столкнуться с некоторым количеством сложностей, часть уже была описана - самая большая это запихнуть сетку на 1GB и 1 ядро - в итоге все получилось. Изображения получаются 128 на 128, за 10 мин. , 256 - уже больше 40 мин. - это, конечно, боль. Дома на 4-х ядрах - менее 2-х мин.

ВЫВОДЫ

Самое ценное - это то, что удалось пройти, так сказать, весь pipeline от идеи до готовой реализации в продакшене, попутно освоив на практике множество процессов и технологий. В процессе работы появились идеи новых фичей и доработок - надеюсь в будущем довести до более-менее законченного продукта. Спасибо куратору и всем авторам и преподавателям курса!

СПИСОК ИСТОЧНИКОВ:

1. Учебник по aiogram и FSM: https://mastergroosha.github.io/telegram-tutorial-2/
2. Сайт TG чата российского aiogram сообщества (оч.помогло): https://telegra.ph/aiogram-ru-01-28
3. Статья по StyleTransfer на PyTorch: https://kushaj.medium.com/all-you-need-for-photorealistic-style-transfer-in-pytorch-acb099667fc8