**Документация к решению команды SPMI-IST22-ATS**

Данный файл представляет собой описание работы решения, предложенного нашей командой. Прежде чем перейти к подробному описанию работы каждой из частей решения, необходимо упомянуть важные моменты. Нейронная сеть и веб-интерфейс работают независимо, так как для нашей команды — это первое выступление в хакатонах и всеми необходимыми знаниями мы не обладаем, из-за чего после нескольких предпринятых попыток нам удалось поднять веб-сервер на Flask, способный принимать изображение и сохранять его в директорию. К нашему большому сожалению, финальная цель достигнута не была. Нейронная сеть может быть улучшена и нуждается в дообучении. Наша модель была обучена на датасете из 10,000 изображений в течение 45 итераций. Такое количество итераций обусловлено лимитами в Google Collaboratory. Обучение на десктопной видеокарте было нецелесообразно ввиду того, что на одну итерацию уходило 45 минут. Касаемо датасета – в нём содержатся изображения пистолетов, ножей, мечей, шпаг, топоров и бит. Большая часть изображений – пистолеты, мечи и ножи, так как на сайте roboflow были найдены готовые размеченные датасеты, содержащие необходимое оружие. Остальные виды вооружения были размечены вручную.

Средний показатель map50-95 модели составляет 0,719, box\_loss – 0,794, cls\_loss – 0,854. Естественно, при большем количестве итераций эти показатели могут быть значительно улучшены. Модель прекрасно детектирует вышеперечисленные виды вооружения на фотографиях, в то время как на видео можно встретить минимальное количество ложных срабатываний. Далее будет описана работа Веб-сервиса и скрипта с нейронной сетью.

**Web-сервис**

Текущая реализация web интерфейса поддерживает обработку изображений, следующий этап нашей разработки - добавление возможности загрузки видео через WEB-интерфейс(не вошел в данный релиз). Для работы необходимо запустить скрипт «object\_detector.py».

Пользователю предоставляется возможность обратиться по адресу http://localhost:8080. Дальнейшее использование регламентированно ТЗ: выбор файла - фотография - ответ от сервера с выделенным оружием на фото.

Принцип работы основан на обращении к странице index1.html. С помощью библиотеки Flask создается request на получение изображения, которому впоследствии присваивается id="uploadInput".

После обработки на изображение накладываются боксы с с предполагаемым местом нахождения оружия и подписью в виде класса этого оружия. Это изображение сворачивается в файл с расширением .json и отправляется на сервер.

В файле «1231231231» предложена реализация боксов хранения видео для загрузки и выгрузки, внедрение которых предполагается в следующем релизе. Код написан на языке node.js, который является серверным.

**Нейронная сеть**

Библиотеки:

* Ultralytics
* cv2
* shutil
* math
* os

Описание работы программы:

В переменную model записывается путь к обученной модели.

classNames содержит названия детектируемых классов

videopath - переменная, содержащая путь к обрабатываемому видео.

save\_path - путь для сохранения кадров, где было обнаружено оружие.

Перед запуском самой нейронной сети происходит проверка директории, куда сохраняются размеченные изображения. Если в неё содержатся файлы с прошлых запусков они будут удалены.

Создаётся множество unique\_id, куда будут записываться номера уникальных кадров.

cap - переменная, инициализирующая запуск модели cv2. Пока она запущена - происходит чтение кадров с видео. Если кадр содержит вооруженных людей или оружие в целом - считываются основные интересующие параметры (id, класс, уверенность - confidence score и массив с координатами интересующего объекта для выделения его в рамку - bounding box). Если кадр не встречался до этого - он добавляется в множество unique\_id. Далее происходит сохранение кадра в нужную директорию с помощью модуля OS. Выделение объекта вынесено в отдельный блок и реализовано посредством функций cv2.rectangle() и cv2.putText(), куда передаются данные, записанные на этапе обработки кадра. Помимо сохранения, в скрипте предусмотрен вывод видео на экран пользователя, чтобы пользователь мог в целом оценить качество модели и увидеть возникающие ошибки. За это отвечает функция cv2.imshow(). Остановка работы нейронной сети происходит нажатием клавиши 'q', либо остановкой самого скрипта в запускающей его среде.

Первичный запуск скрипта может потребовать больше времени ввиду того, что происходит инициализация. После первичного запуска активация нейронной сети и вывод видео будет занимать считанные секунды. Скачать обученную модель можно [тут](https://drive.google.com/file/d/1xEZTA2AX9OmhQms5xjYJF-c0UO9-WV1T/view?usp=drive_link).