«Система распознавание лиц для пропускного пункта школы»

Выполнила: Мягкова Ирина Олеговна

МБОУСОШ №38 11В класс

2022 год, г. Воронеж

***Содержание***

* Актуальность темы…………………………………………………... 3
* Обоснование выбора темы……....……………………………………4
* Цель и задачи…………………………………………………………..5
* Исследование темы проекта…….……………………………………6-8
* Оборудование и программное обеспечение.…………………………9

**Технологический этап**……………………………………………10-23

* План работы……………………………………………………………10
* Ход работы………………………..………………………………….11-13
* Калибровка…………………………………………………………….13

**Заключительный этап**…………………………………………...14-16

* Система распознавания лиц..…………………………………………14
* Вывод………………..………………………………………………….15
* Используемая литература……………………..………………………16

***Актуальность***

В современном мире безопасность выходит на первый план. Практически в каждом офисе, школе или университете существует пропускная система. В большинстве таких учреждений нужны пропускные карточки или пропуски. Но иногда эти маленькие пластиковые прямоугольники теряются или забываются в другой сумке или куртке. В некоторых местах без пропуска невозможно попасть внутрь, а где-то всё равно пропустят. И в том, и в том случае система не эффективна. В первом случае работник/ученик не могут выполнять свою работу от чего страдают сами, а также учреждение. Во втором случае страдает безопасность, ведь охранники не могут знать всех в лицо, а значит они могут пропустить нарушителя. Возникает вопрос. Что же мы не можем потерять и что может служить нашим личным идентификатором. Ответ прост. Наше лицо. Лицо нельзя забыть или потерять (в прямом смысле). Так и появилась идея создания пропускной системы с использованием распознавания лиц.

***Обоснование выбора темы***

В моей школе существует система пропусков через турникет, но школьники часто забывали свой пропуск или теряли, а восстановление не только стоит денег, но и достаточно долгий процесс. Из-за этого многие ученики просили друзей пропускать их. В связи с этим в бозе данных один человек мог входить в школу несколько раз подряд, а другой не появляться неделями. Вся суть этой системы потерялась и её наличие утратило смысл, а ведь сама идея пропускного пункта не только выступает в качестве безопасного барьера, но и может следить за опоздавшими и фиксировать количество учеников в школе каждый день. В связи с этим я решила посвятить тему своего проекта именно созданию системы распознавания лиц для пропускных пунктов

***Цель и задачи***

**Цель:** создать систему распознавания лиц

**Задачи:**

* Собрать и разметить dataset для задач object detection
* Обучить нейронную сеть с помощью yolov3
* Изучить возможные способы совмещения весов нейронный сети и камеры
* Разобраться с выводом видео в реальном времени одновременно с определением лиц
* Протестировать систему
* Сделать вывод о её эффективности

***Исследование***

**Нейронная сеть** — это последовательность нейронов, соединенных между собой синапсами. Структура нейронной сети пришла в мир программирования прямиком из биологии.  
Нейронные сети используются для решения сложных задач, которые требуют аналитических вычислений подобных тем, что делает человеческий мозг. Самыми распространенными применениями нейронных сетей является:

* **Классификация** — распределение данных по параметрам. Например, на вход дается набор людей и нужно решить, кому из них давать кредит, а кому нет. Эту работу может сделать нейронная сеть, анализируя такую информацию как: возраст, платежеспособность, кредитная история и тд.
* **Предсказание** — возможность предсказывать следующий шаг. Например, рост или падение акций, основываясь на ситуации на фондовом рынке.
* **Распознавание** — в настоящее время, самое широкое применение нейронных сетей. Используется в Google, когда вы ищете фото или в камерах телефонов, когда оно определяет положение вашего лица и выделяет его и многое другое.

**Нейрон** — это вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше. Они делятся на три основных типа: входной, скрытый и выходной. В том случае, когда нейросеть состоит из большого количества нейронов, вводят термин слоя. Соответственно, есть входной слой, который получает информацию, n скрытых слоев (обычно их не больше 3), которые ее обрабатывают и выходной слой, который выводит результат. У каждого из нейронов есть 2 основных параметра: входные данные (input data) и выходные данные (output data). В случае входного нейрона: input=output. В остальных, в поле input попадает суммарная информация всех нейронов с предыдущего слоя, после чего, она нормализуется, с помощью функции активации и попадает в поле output.  
**Синапс** — это связь между двумя нейронами. У синапсов есть 1 параметр — вес. Благодаря ему, входная информация изменяется, когда передается от одного нейрона к другому. Допустим, есть 3 нейрона, которые передают информацию следующему. Тогда у нас есть 3 веса, соответствующие каждому из этих нейронов. У того нейрона, у которого вес будет больше, та информация и будет доминирующей в следующем нейроне

**Функция активации** — это способ нормализации входных данных.То есть, если на входе у вас будет большое число, пропустив его через функцию активации, вы получите выход в нужном вам диапазоне. Функций активации достаточно много поэтому мы рассмотрим самые основные: Линейная, Сигмоид (Логистическая) и Гиперболический тангенс. Главные их отличия — это диапазон значений.

**YOLOv3** — это очень популярная на текущий момент архитектура CNN, которая используется для распознавания множественных объектов на изображении. Это усовершенствованная версия архитектуры YOLO. Она состоит из 106-ти свёрточных слоев и лучше детектирует небольшие объекты по сравнению с её предшественницей YOLOv2. Основная особенность YOLOv3 состоит в том, что на выходе есть три слоя каждый из которых рассчитан на обнаружения объектов разного размера. Оригинальная архитектура YOLOv3 реализована с помощью framework[Darknet](https://github.com/pjreddie/darknet).

Сейчас очень много задач в области компьютерного зрения решаются с помощью свёрточных нейронных сетей (Convolutional Neural Networks). Благодаря своему строению они хорошо извлекают признаки из изображения. CNN используются в задачах классификации, распознавания, сегментации и еще во множестве других.  
  
Популярные архитектуры CNN для распознавания объектов:

* R-CNN. Можно сказать первая модель для решения данной задачи. Работает как обычный классификатор изображений. На вход сети подаются разные регионы изображения и для них делается предсказания. Очень медленная так как прогоняет одно изображение несколько тысяч раз.
* **Fast R-CNN**. Улучшенная и более быстрая версия R-CNN, работает по похожему принципу, но сначала все изображение подается на вход CNN, потом из полученного внутреннего представления генерируются регионы. Но по прежнему довольно медленная для задач реального времени.
* **Faster R-CNN**. Главное отличие от предыдущих в том, что вместо selective search алгоритма для выбора регионов использует нейронную сеть для их «заучивания».
* **YOLO**. Совсем другой принцип работы по сравнению с предыдущими, не использует регионы вообще. Наиболее быстрая. Более подробно о ней пойдет речь в статье.
* **SSD**. По принципу похожа на YOLO, но в качестве сети для извлечения признаков использует VGG16. Тоже довольная быстрая и пригодная для работы в реальном времени.
* **Feature Pyramid Networks (FPN)**. Еще одна разновидность сети типа Single Shot Detector, из за особенности извлечения признаков лучше чем SSD распознает мелкие объекты.
* **RetinaNet**. Использует комбинацию FPN+ResNet и благодаря специальной функции ошибки (focal loss) дает более высокую точность (аccuracy).

[**TensorFlow**](https://neurohive.io/ru/tutorial/tensorflow-tutorial-tenzory-i-vektory/)**—** это[ML](https://neurohive.io/ru/osnovy-data-science/vvedenie-v-mashinnoe-obuchenie-kto-ego-primenjaet-i-kak-stat-razrabotchikom/)**-**framework от Google, который предназначен для проектирования, создания и изучения моделей глубокого обучения. Глубокое обучение — это область машинного обучения, алгоритмы в которой были вдохновлены структурой и работой мозга. Вы можете использовать TensorFlow, чтобы производить численные вычисления. Само по себе это не кажется специфичным, однако эти вычисления производятся с помощью data-flow графов. В этих графах вершины представляют собой математические операции, в то время как ребра представляют собой данные, которые обычно представляются в виде многомерных массивов или тензоров, которые сообщаются между этими ребрами.

**Anaconda** - Дистрибутив языков программирования Python и R, включающий набор популярных свободных библиотек, объединённых проблематиками науки о данных и машинного обучения. Основная цель - поставка единым согласованным комплектом наиболее востребованных соответствующим кругом пользователей тематических модулей с разрешением возникающих зависимостей и конфликтов, которые неизбежны при одиночной установке.

***Оборудование, программное обеспечение***

* Ноутбук/компьютер
* Камера
* Google Colab
* Интернет
* Терминал
* Githab
* CVAT

***План работы***

1. Снять на видео людей для первого обучения
2. Разметить видео и скачать dataset
3. Загрузить нужные файлы на google диск
4. Обучить модель
5. Проверить модель
6. Подключить модель в веб камере

***Ход работы***

1. Первым делом я сняла видео с тремя своими одноклассницами. Я решила, что четыре видео (включая моё видео) будет достаточно для прототипа. Видео снимались таким образом, чтобы девочки немного меняли причёски, закрывали какие-то части лица и меняли эмоции (это было нужно для разнообразия датасета).
2. Далее, с помощью программы CVAT, я разделила видео на кадры и на каждом кадре выделила лицо девочек. Это было долгое занятие, но мне нравится размечать данные, это довольно медитативный процесс. Вообще от разметки данных зависит очень много, поэтому этот этап очень важен. Одно из основных правил выделения объектов в разметке заключается в том, что если выделять объект квадратом, то стороны этого квадрата должны быть как можно ближе к объекту.
3. Третий этап – подготовка файлов. Для начала я конвертировала все файлы из формата PNG в JPG. После этого я переместила фотографии и текстовые данные со зачем расположения объектов на изображении в одну папку и сжала её. Далее мне нужно было изменить под себя файлы obj.name, obj.data, yolov3\_custom.cfg.  
   Что это за файлы? В obj.name находятся названия наших объектов -labels. В моём случае это имена учеников. Важно расположить эти имена в порядке того, как были пронумерованы labels в CVAT.  
   В obj.data хранятся пути сохранения весов и текстовых файлов теста и valid, а таже количество классов, его нужно было изменить на 4, так как мне нужно, чтобы нейронная сеть определяла 4 класса.   
   И последний файл yolov3\_custom.cfg. Там содержится вся важная информация для обучения. Сначала я сняла комментарий со строк для обучения. Потом изменила значение max\_batches (количество проходимых эпох обучения) по формуле количество классов \* 2000 (4\*2000=8000). Также немного ниже нужно изменить значение steps. Эти два значения должны быть равны 80% и 90% от max\_batches (в моём случае 6400 и 7200). В этом файле присутствует 3 идентичных части под название [yolo]. Во всех этих частях нужно поменять количество классов (для меня 4) и значение filters (над блоком [yolo]) по формуле количество (классов + 5) \*3 ((4+5)\*3=27)
4. Теперь переходим к обучению. Я обучала с помощью Google colab. Почему? Google colab даёт доступ к gpu и вообще этот сервер отлично подходит для машинного обучения. Единственный минус, что пользоваться gpu можно 6 часов в 24 часа, поэтому обучение затягивается на несколько дней. Я обучала по уже готовому коду для обучения yolov3   
    1. Сначала я обучила нейронную сеть на своих фотографиях, да, она определяла меня, но, по всей видимости, нейронная сеть запомнила не какие-то особенные черты моего лица, а просто вид человеческого лица, потому что она определяла всех, как меня.  
    2. Далее я обучала нейронную сеть на dataset со всеми участниками , но я поступила опрометчива и не проверила количество фотографий каждой, поэтому dataset получился неравномерный и нейронная сеть неправильно определяла лица. Так я подумала первоначально. Когда я запускала программу повторно, то делала это при ярком освящении и тогда нейронная сеть определила меня правильно, а чужого человека не определила. Но даже тут была небольшая ошибка, моих родителей программа определяла так же, как меня. Такой результат меня не устроил, и я решила заново обучить нейронную сеть.  
    3. Учитывая предыдущие ошибки, я сравняла количество фотографий, насколько это было возможно и дополнила dataset обработанными фотографиями (изменение контрастности, чёрно-белый эффект, рябь, зелёный фильтр, красный фильтр, синий фильтр). На новом наборе данных я обучила нейронную сеть. Теперь при плохом свете лица определяются лучше, но проблему с определением родственников решить не удалось.
5. Последний этап – вывод видео с камеры совместно с нейронной сетью. Над этим пунктом я думала долго, изначально я планировала использовать yolov5 и flask, но, к сожалению, моих знаний не хватило для реализации моей идеи именно этим способом. Поэтому я начала поиски решения. На просторах интернета я нашла способ вывода видео с помощью терминала с использованием Anaconda и [TensorFlow](https://neurohive.io/ru/tutorial/tensorflow-tutorial-tenzory-i-vektory/)(см. Исследование). Для начала я скачала все нужные библиотеки, а также набор файлов с GitHab для реализации. После этого я перевела файлы из формата weights в формат tf (формат для работы [TensorFlow](https://neurohive.io/ru/tutorial/tensorflow-tutorial-tenzory-i-vektory/)) с помощью команды   
     
   python load\_weights.py –weights './weights/yolov3\_custom\_last.weights' --output './weights/yolov3\_custom.tf' --num\_classes '4'  
     
   Потом, когда файлы приобрели нужно расширение я ввожу команду, для вывода видео с видео камеры, а также записи этого видео. Зачем я решила записывать видео? Это довольно просто, пусть по статистике нейронные сети ошибаются меньше людей (см. Исследование), но всё же ошибаются, поэтому наверняка понадобятся записи в случае непоняток, а также это просто как видео наблюдение. Я думаю, что такая функция довольно полезна.  
     
   python detect\_video.py --weights 'weights/ [yolov3\_custom.tf](http://yolov3.tf/)' --output 'data/video/video.avi' --video 0 -num\_classes '4' --classes 'data/labels/obj.names'
6. Когда мы вывели видео, нужно проверить определяет ли он нужных нам людей. После завершения и проверки можно с гордостью заявить, что прототип системы распознавания лиц для пропускного пункта школы готов!

***Калибровка***

Самая большая проблема у меня возникла с dataset-ом. Сначала я неправильно расположила lebels и поэтому с одним номером у меня были разные люди. Эта проблема решается новой разметкой с правильным расположением классов. Следующая проблема заключалась с обучение из-за неравномерности dataset-а, но эту проблему и её решение я рассмотрела в ходе работы, пункте 6.2.

***Система распознавания лиц***

Изображение выглядит как текст, человек, внутренний, оранжевый

Автоматически созданное описание

Изображение выглядит как текст, человек, оранжевый

Автоматически созданное описание***Вывод***

В результате работы был создан прототип системы распознавания лиц для пропускного пункта школы. Эта система способна распознавать лица учеников, выводить их имя, фамилию и класс, а также сохранять видео с распознаванием. В моём прототипе есть ряд недочётов, которые я буду исправлять в дальнейшем. Первое – определение родственников ученика, как его самого. Второе - при дальнем расположении ученика при определении могут происходить ошибки. Для решения этой проблемы около информации об ученике есть процент верности, поэтому если процент меньше 85, то нужно подойти ближе к камере или сделать свет ярче для точного распознавания. Эти недочёты я планирую решать с помощью расширения dataset-а (дальнее и ближнее расположения человека)

***Используемая литература***

Сайты:

<https://habr.com/ru/post/460869/>

<https://github.com/>

<https://www.machinelearningmastery.ru/pedestrian-tracking-in-real-time-using-yolov3-33439125efdf/>

<https://www.youtube.com/>

<https://neurohive.io/ru/tutorial/tensorflow-tutorial-tenzory-i-vektory/>

<https://habr.com/ru/post/312450/>

**Спасибо за внимание!**