СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc41554441)

[1 СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ 5](#_Toc41554442)

[2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ](#_Toc41554443) 7

[2.1 Проектирование архитектуры ПО](#_Toc41554447) 7

[2.2 Структура программного, аппаратного и информационного обеспечения системы](#_Toc41554447) 13

[2.3 Макетирование пользовательского интерфейса](#_Toc41554447) 15

[3 РЕАЛИЗАЦИЯ СИСТЕМЫ](#_Toc41554446) 16

[4 ТЕСТИРОВАНИЕ](#_Toc41554449) 17

[4.1 Описание входных и выходных данных](#_Toc41554450) 17

[4.2 Результаты тестирования](#_Toc41554450) 19

[ЗАКЛЮЧЕНИЕ](#_Toc41554452) 24

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ](#_Toc41554453) 25

ПРИЛОЖЕНИЕ А ТЕКСТ ПРОГРАММЫ

ПРИЛОЖЕНИЕ Б ГРАФИЧЕСКИЙ МАТЕРИАЛ

# Введение

На сегодняшний момент искусственный интеллект прочно вошел в нашу жизнь и помогает в решении очень большого количества задач. Одним из самых перспективных направлений в искусственном интеллекте, которое приближает будущее, отраженное в всякого рода фантастических фильмах, являются нейронные сети.

Изучение и использование искусственных нейронных сетей, в принципе, началось уже достаточно давно – в начале 20 века, но по-настоящему широкую известность они получили несколько позже. Связано это, в первую очередь, с тем, что стали появляться продвинутые (для того времени) вычислительные устройства, мощности которых были достаточно велики для работы с искусственными нейронными сетями. По сути, на данный момент можно легко смоделировать нейронную сеть средней сложности на любом персональном компьютере.

Успех нейронных сетей обусловлен следующим:

* **Богатые возможности.**  [Нейронные сети](http://statsoft.ru/home/textbook/glossary/gloss_n.html#Neural%20Networks)– исключительно мощный метод моделирования, позволяющий воспроизводить чрезвычайно сложные зависимости.
* **Простота в использовании.**  Нейронные сети учатся на примерах. Пользователь нейронной сети подбирает представительные данные, а затем запускает алгоритм обучения, который автоматически воспринимает структуру данных. При этом от пользователя, конечно, требуется какой-то набор эвристических знаний о том, как следует отбирать и подготавливать данные, выбирать нужную архитектуру сети и интерпретировать результаты, однако уровень знаний, необходимый для успешного применения нейронных сетей, гораздо скромнее, чем, например, при использовании традиционных методов статистики.

Искусственные нейронные сети построены по принципу биологических, однако с рядом допущений. Подобно человеческому мозгу данные сети способны обучаться. Под процессом обучения подразумевается процесс настройки структуры связей между нейронами и влияющих на сигналы коэффициентов для эффективного решения поставленной задачи.

Все задачи, которые могут решать нейронные сети, так или иначе связаны с обучением. Среди основных областей применения нейронных сетей — прогнозирование, принятие решений, распознавание образов, оптимизация, анализ данных.

В данном курсовом проекте будет рассмотрена тема детекции и идентификации человека с помощью нейронных сетей, которая в настоящее время получила широкое распространение.

# 1. СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ

В последние годы биометрия все более активно проникает в нашу жизнь. Ведущие страны мира уже ввели в оборот или в ближайшее время планируют ввести электронные паспорта, содержащие сведения о биометрических характеристиках своего владельца; многие офисные центры внедрили биометрические сенсоры в корпоративные системы контроля доступа; ноутбуки уже давно оснащаются средствами биометрической аутентификации пользователя; на вооружении служб безопасности появляются современные средства выявления любого разыскиваемого преступника в толпе людей.

К слову технология распознавания лица активно используется в государственном секторе, медицинских компаниях, сфере безопасности и многих других. Распознавание человека по изображению лица выделяется среди биометрических систем тем, что, во-первых, не требует специального дорогостоящего оборудования. Для большинства приложений достаточно персонального компьютера и обычной видеокамеры. Во-вторых, отсутствует физический контакт человека с устройствами. Не надо ни к чему прикасаться или специально останавливаться и ждать срабатывания системы. В большинстве случаев достаточно просто пройти мимо или задержаться перед камерой на несколько секунд.

К недостаткам распознавания человека по изображению лица следует отнести то, что сама по себе такая система не обеспечивает 100%-ной надёжности идентификации. Там, где требуется высокая надёжность, применяют комбинирование нескольких биометрических методов.

На данный момент проблеме распознавания человека по изображению лица посвящено множество работ, однако в целом она ещё далека от разрешения. Основные трудности состоят в том, чтобы распознать человека по изображению лица независимо от изменения ракурса и условий освещённости при съёмке, а также при различных изменениях, связанных с возрастом, причёской и т.д.

На сегодняшний день направление биометрики является как никогда актуальным, так как пандемия COVID-19 стала двигателем существенного роста бесконтактных биометрических технологий. Мировые и российские фирмы (NtechLab, ЦРТ, Tevian и др.) смогли быстро настроить (с использованием нейросетей) свои алгоритмы так, чтобы человек уверенно распознавался даже в том случае, если он в медицинской маске.

Очень интересным проектом в направлении биометрики по-моему мнению является проект NtechLab – платформа мультиобъектной видеоаналитики FINDFACE Multi. В основе платформы NtechLab лежит технология, обеспечивающая рекордную точность и высокую скорость распознавания лиц, силуэтов и действий людей, а также детектирование автомобилей. Мультиобъектность и аналитические возможности FindFace Multi открывают новые возможности для решения широкого спектра задач во всех областях применения видеоаналитики. Разработчики и ученые компании NtechLab постоянно совершенствуют алгоритмы распознавания, повышая скорость и точность их работы. Меньше секунды требуется системе на платформе FindFace Multi, чтобы выявить лицо в видеопотоке, распознать его и произвести поиск по нему в базе с миллиардами изображений. Алгоритмы максимально точно распознают появление одного и того же человека на разных изображениях и видео, даже если его внешность изменилась, большого расстояния видимости и на видео низкого качества.

Так же в FindFace Multi интегрирована система антиспуфинга, которая работает на любых камерах и позволяет отличить живого человека от изображения. Нейросеть анализирует сразу несколько кадров и фиксирует изменения на лице. Это гарантирует, что перед камерой живой человек, и исключает возможность мошенничества с использованием изображений на бумаге или экранах мобильных устройств.

FindFace Multi не только детектирует наличие маски и ее правильное ношение, но и с высокой точностью распознает лица, наполовину закрытые маской.

На основании выше сказанного можно сделать вывод что такое направление как детекция и идентификация человека на фото- и видеоизображениях является перспективным направлении в биометрике. По всему миру наблюдается рост данного рынка и спрос на экспертов в этой области. Так что разработка программного продукта, связанного с распознаванием человека в рамках данного курсового проекта является целесообразным.

Задачей данного курсового проекта является разработка оконного приложения, позволяющее распознавать лицо человека и самого человека на фото – и видео изображениях.

Программный продукт должен выполнять следующие функции:

1. Детекция лиц и их идентификация на фотографиях и видео;
2. Детекция целевого человека на серии фотографий;
3. Для случая обработки видео процесс детекции/идентификации осуществлять с отображением в реальном времени;
4. Оконный интерфейс

Так же для реализации поставленной задачи рекомендуется использование предобученной модели распознавания.

# 2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ

### **2.1 Проектирование архитектуры ПО**

Наилучшие результаты в области распознавания лиц показала Convolutional Neural Network или сверточная нейронная сеть. Анализ показывает, что среди наиболее подходящих для решения задачи детектирования СНС предпочтение сегодня следует отдать сетям класса YOLO. Эти сети обеспечивают большие точность и скорость детектирования объектов на изображениях, чем сети других классов.

YOLO или You Only Look Once — это очень популярная на текущий момент архитектура CNN, которая используется для распознавания множественных объектов на изображении.

Главная особенность этой архитектуры по сравнению с другими состоит в том, что большинство систем применяют CNN несколько раз к разным регионам изображения, в YOLO CNN применяется один раз ко всему изображению сразу. Сеть делит изображение на своеобразную сетку и предсказывает bounding boxes и вероятности того, что там есть искомый объект для каждого участка.

Алгоритм YOLO выгодно использовать по следующим причинам:

1. **Скорость:** этот алгоритм повышает скорость обнаружения, поскольку он может предсказывать объекты в реальном времени.
2. **Высокая точность:** YOLO - это метод прогнозирования, который обеспечивает точные результаты с минимальными фоновыми ошибками.
3. **Возможности обучения:** алгоритм обладает отличными возможностями обучения, которые позволяют ему изучать представления объектов и применять их при обнаружении объектов.

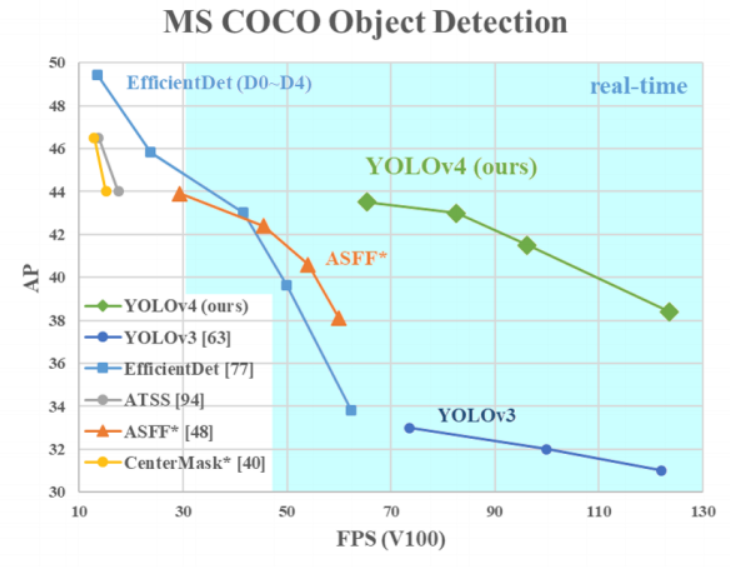


Рисунок 2.1 – График сравнения

Как видно из графика, YOLO работает достаточно быстро и точно.

Архитектура YOLO выглядит следующим образом:

1. Подается картинка/кадр на вход
2. Кадр прогоняется через CNN для формирования feature maps (правда в YOLO используется своя CNN под названием Darknet-53)
3. Feature maps определенным образом анализируются, выдавая на выходе позиции и размеры bounding boxes и классы, которым они принадлежат.

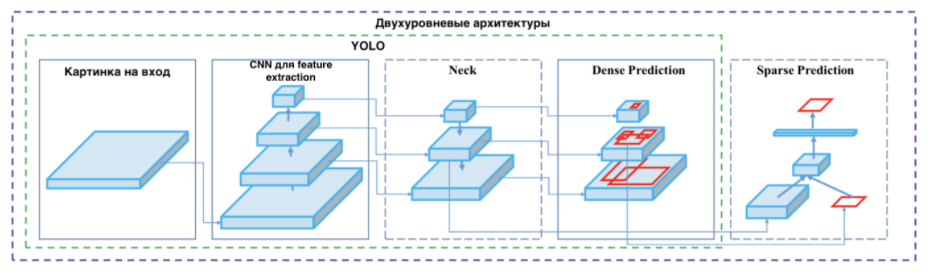


Рисунок 2.2 – Архитектура YOLO

Алгоритм YOLO работает с использованием следующих трех методов:

1. Остаточные блоки
2. Регрессия ограничивающей рамки
3. Пересечение через объединение (IOU)

#### Остаточные блоки:

#### Сначала изображение разбивается на различные сетки. Каждая сетка имеет размер S x S. На следующем изображении показано, как входное изображение делится на сетки.



Рисунок 2.3 – Разбиение на сетки

На изображении выше есть много ячеек сетки одинаковой размерности. Каждая ячейка сетки будет обнаруживать объекты, которые появляются внутри них. Например, если центр объекта появляется в определенной ячейке сетки, то эта ячейка будет отвечать за его обнаружение.

#### Регрессия ограничивающей рамки:

#### Ограничивающая рамка - это контур, выделяющий объект на изображении. Каждый ограничивающий прямоугольник на изображении состоит из следующих атрибутов:

* Ширина (bw)
* Высота (bh)
* Класс (например, человек, автомобиль, светофор и т. д.) - обозначается буквой c.
* Центр ограничивающей рамки (bx, by)

На следующем изображении показан пример ограничивающей рамки. Ограничивающая рамка представлена ​​желтым контуром.

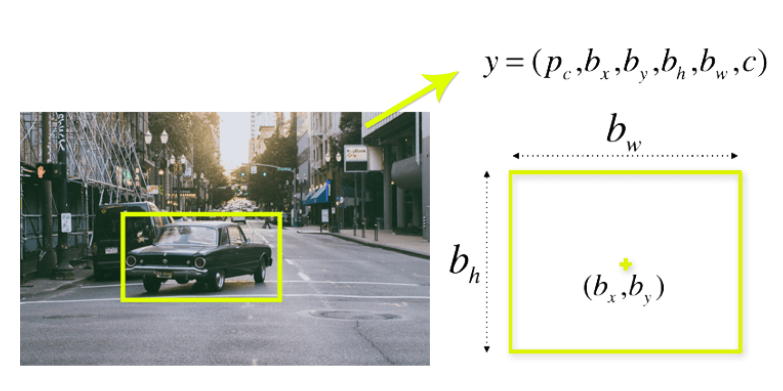


Рисунок 2.4 – Ограничивающая рамка

YOLO использует регрессию одного ограничивающего прямоугольника для прогнозирования высоты, ширины, центра и класса объектов. На изображении выше представляет вероятность появления объекта в ограничивающей рамке.

#### Пересечение через объединение(IOU):

#### Пересечение через объединение (IOU) - это явление при обнаружении объектов, которое описывает, как блоки перекрываются. YOLO использует пересечение через объединение для создания окна вывода, которое идеально окружает объекты. Каждая ячейка сетки отвечает за прогнозирование ограничивающих прямоугольников и их оценок достоверности. IOU равен 1, если прогнозируемый ограничивающий прямоугольник такой же, как и реальный прямоугольник. Этот механизм устраняет ограничивающие прямоугольники, которые не равны реальному прямоугольнику.

#### На следующем изображении представлен простой пример того, как работает пересечение через объединение.



Рисунок 2.5 – Пересечение через объединение

На изображении выше есть две ограничивающие рамки, одна зеленая, а другая синяя. Синий прямоугольник - это прогнозируемый прямоугольник, а зеленый - настоящий. YOLO гарантирует, что два ограничивающих прямоугольника равны.

На следующем изображении показано, как три метода применяются для получения окончательных результатов обнаружения.

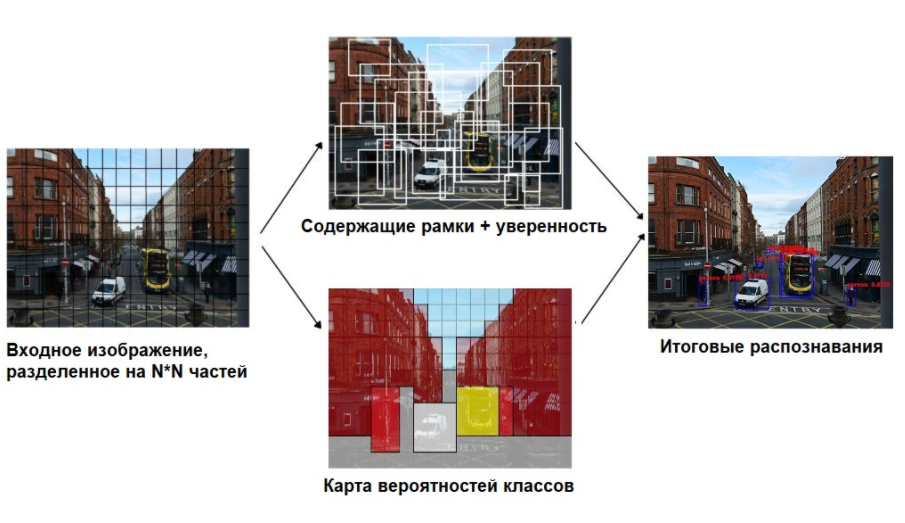


Рисунок 2.6 – Визуализация процесса

Сначала изображение разбивается на ячейки сетки. Каждая ячейка сетки прогнозирует ограничивающие прямоугольники и предоставляет их оценки достоверности. Ячейки предсказывают вероятности классов для установления класса каждого объекта. Все прогнозы делаются одновременно с использованием единой сверточной нейронной сети.

Пересечение через объединение гарантирует, что предсказанные ограничивающие прямоугольники равны реальным прямоугольникам объектов. Это явление устраняет ненужные ограничивающие рамки, которые не соответствуют характеристикам объектов (например, высоте и ширине). Окончательное обнаружение будет состоять из уникальных ограничивающих рамок, идеально подходящих к объектам.

При анализе выше приведенной информации, было принято решение использовать именно YOLO для реализации курсового проекта.

Общий алгоритм работы программного продукта будет следующим:

Шаг1. Получаем на вход фотоизображение, видеоизображение или изображение с камеры.

Шаг2. Происходит процесс обнаружения человека/лица.

В итоге, на выходе мы получаем фотоизображение, видеоизображение или изображение с камеры, на котором будут идентифицированы тело и лицо человека.

### **2.2 Структура программного, аппаратного, информационного обеспечения системы**

Для решения поставленной задачи в данном проекте будет использован язык Python, так как он кроссплатформенный – хорошо работает на всех популярных ОС и переносится на них без каких-либо доработок, а также он обладает такими качествами как простота, лаконичность и выразительность, позволяющие с минимальными затратами времени и сил разрабатывать сложные алгоритмы.

Для обнаружения целевого человека, будет использована предобученная модель yolov4, которая имеет 129 предварительно обученных сверточных слоев.

Что касается среды разработки, то среди большого количества инструментального ПО была выбрана среда разработки JetBrains PyCharm. PyCharm делает разработку максимально продуктивной благодаря функциям автодополнения и анализа кода, мгновенной подсветке ошибок и быстрым исправлениям. С PyCharm удобно работать с ноутбуками Jupyter, запускать команды в интерактивной консоли Python, подключать библиотеки Anaconda, а также работать с другими библиотеками для научных вычислений и анализа данных, включая Matplotlib и NumPy. А так же PyCharm – настраиваемая кросс-платформенная IDE, а значит PyCharm можно установить на нескольких компьютерах в операционных системах Windows, Mac OS или Linux.

Кроме интерпретатора python будет использована библиотека глубокого обучения Keras. Keras – это минималистичная библиотека, основанная на Python, которая может запускаться поверх TensoFlow, Theano или CNTK. Keras поддерживает широкий спектр слоев нейронных сетей, таких как сверточные слои, рекуррентные или плотные. Этот фреймворк хорош в кейсах для перевода, распознавании изображений, речи и т.п.

Преимущества:

1. Прототипирование действительно быстрое и простое.

2. Он достаточно маловесный для построения моделей глубокого обучения для множества слоев.

3. Имеет полностью конфигурируемые модули.

4. Имеет простой и интуитивно-понятный интерфейс, соответственно, хорош для новичков.

5. Имеет встроенную поддержку для обучения на нескольких GPU.

6. Может быть настроен в качестве оценщиков для TensorFlow и обучен на кластерах GPU на платформе Google Cloud.

## Недостатки:

## 1. Может оказаться слишком высокоуровневым и не всегда легко

## кастомизируется.

## 2. Ограничен бэкэндами Tensorflow, CNTK и Theano.

## Также для реализации поставленной задачи будут использованы следующие библиотеки:

OpenCV — это open source библиотека компьютерного зрения, которая предназначена для анализа, классификации и обработки изображений. Широко используется в таких языках как C, C++, Python и Java.

NumPy — это библиотека языка Python, добавляющая поддержку больших многомерных массивов и матриц, вместе с большой библиотекой высокоуровневых (и очень быстрых) математических функций для операций с этими массивами.

Imutils — этот пакет включает ряд удобных функций OpenCV +, которые выполняют базовые задачи, такие как перевод, поворот, изменение размера и скелетонизация.

Что касается аппаратного обеспечения, то курсовой будет выполнен на ПК, который имеет процессор Intel Core i5-8300H CPU с частотой 2.3 ГГц, ОЗУ 8 ГБ, а также тип операционной системы: 64-разрядная OC Windows10.

### **2.3 Макетирование пользовательского интерфейса**

Для разработки макетов к курсовому проекту, была выбрана Figma из-за уже имеющегося небольшого опыта работы с этим сервисом. Figma — онлайн-сервис для разработки интерфейсов и прототипирования с возможностью организации совместной работы в режиме реального времени.

Интерфейс проекта будет выглядеть следующим образом:

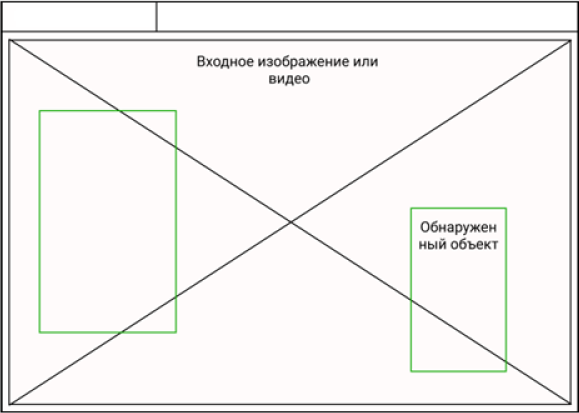
****

Рисунок 2.7 – Макет внешнего вида системы

# 3 РеАЛИЗАЦИЯ СИСТЕМЫ

В таблице 3.1 описаны основные модули программы.

Таблица 3.1 – Описание модулей программы

|  |  |  |
| --- | --- | --- |
| **Название модуля** | **Методы модуля** | **Описание модуля** |
| train.py | train()  parse\_opt()  main()  run() | Утилита для обучения нейронной сети. |
| detect.py | run()  main()  parse\_opt() | Утилита для распознавания тела и лица человека на основе обученной модели. |
| coco.yaml | - | Утилита для формирования описания модели  нейронной сети. |
| coco.names | - | Утилита для формирования описание названий классов. |
| yolov5s.yaml | - | Утилита для формирования описание модели. |
| best.pt | - | Утилита для описания весов, полученных в ходе обучения. |

Описание аргументов модуля train.py приведены в таблице 3.2.

Таблица 3.2 – Аргументы модуля train.py

|  |  |
| --- | --- |
| **Аргументы** | **Пояснение** |
| train | Функция для обучения нейронной сети. |
| parse\_opt | Функция для формирования аргументов для утилиты. |
| main | Основная функция, в которой формируются графики и происходит вызов функции обучения нейронной сети. |

Описание аргументов модуля detect.py приведены в таблице 3.3.

Таблица 3.3 – Аргументы модуля detect.py

|  |  |
| --- | --- |
| **Аргументы** | **Пояснение** |
| parse\_opt | Функция для формирования аргументов для утилиты. |
| main | Основная функция, в которой происходит вызов функции для распознавания тела и лица человека. |
| run | Функция применения обученной модели для распознавания тела и лица человека. |

# 4 ТЕСТИРОВАНИЕ

### **Описание входных и выходных данных**

В качестве обучающей выборки был использован набор данных в количестве 215 изображений, созданный с помощью Roboflow, который позволяет создать собственную обучающую выборку. Создание собственной выборки для обучения заключалось в следующих этапах:

1. Загрузка изображений;
2. Создание классов для объектов распознавания;
3. Выделение объекта на изображении и определение его в соответствующий класс;
4. Сохранения архива, в котором находятся изображения и лейблы;

Обучение проводилось на выборке по несколько часов с разным количеством эпох. В итоге была замечена зависимость между количеством эпох обучения и точность обнаружения: чем больше эпох, тем выше точность.

Входными данными являются подготовленные изображения (рис. 4.1) и лейблы (рис. 4.2) (текстовые файлы в которых содержаться точки выделенного объекта, а также номер класса, к которому относится выделенный объект).

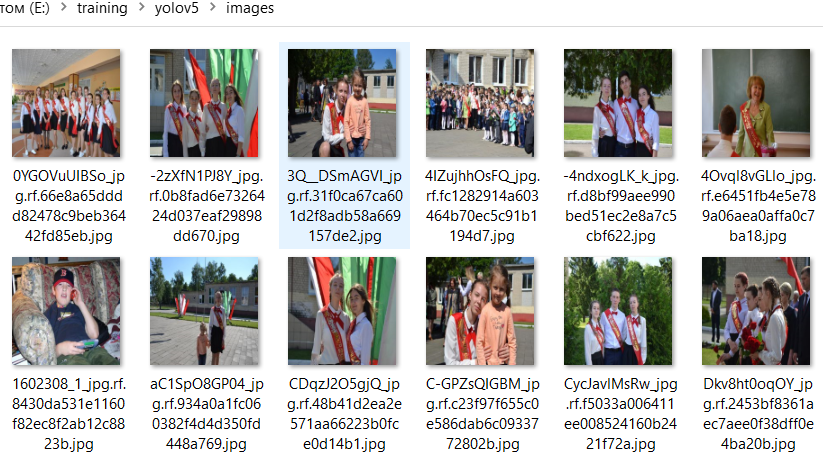


Рисунок 4.1 – Изображения обучающей выборки



Рисунок 4.2 – Лейблы обучающей выборки

Выходными данными являются результаты обучения (файл с весами, изображения обучающей выборки с выделенными на них объектами обнаружения, обученная модель, а также графики, описывающие процесс обучения модели) и результаты детекции (изображения или видеофайл с выделенными на них объектами обнаружения).

### **Результаты тестирования**

Среда тестирования – ПК, который имеет процессор Intel Core i5-8300H CPU с частотой 2.3 ГГц, ОЗУ 8 ГБ, а также тип операционной системы: 64-разрядная OC Windows10.

Обучение и тестирование производилось на базе PyCharm.

**Тест 1:** «Обучение модели нейронной сети»

Описание: обучить разработанную модель для распознавания лица и целого человека (рис. 4.3). В ходе обучения следует внимательно следить за тем, чтобы с

прохождением каждой эпохи функция распространения ошибки loss уменьшалась (рис. 4.4). Это говорит о том, что процесс обучения модели происходит верно. Также в результате этого теста будет сформирован график точности распознавания нейронной сети (рис. 4.5).

Полученный результат:

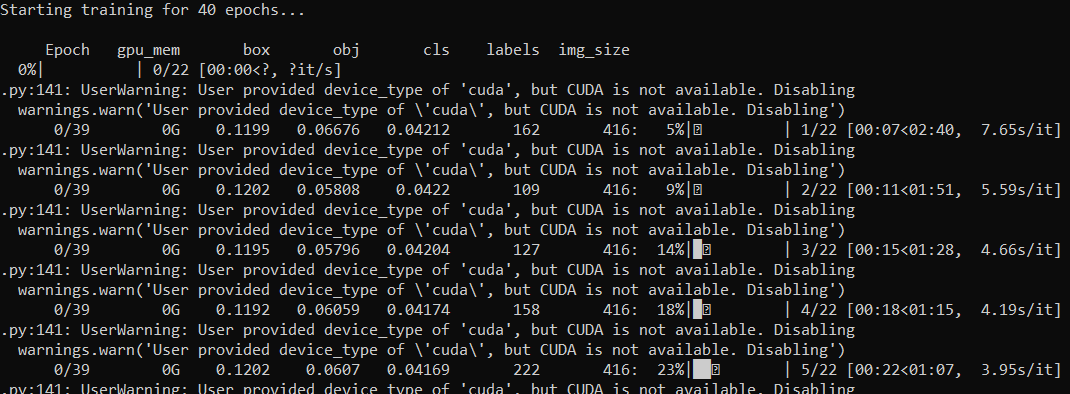


Рисунок 4.3 – Обучение нейронной сети

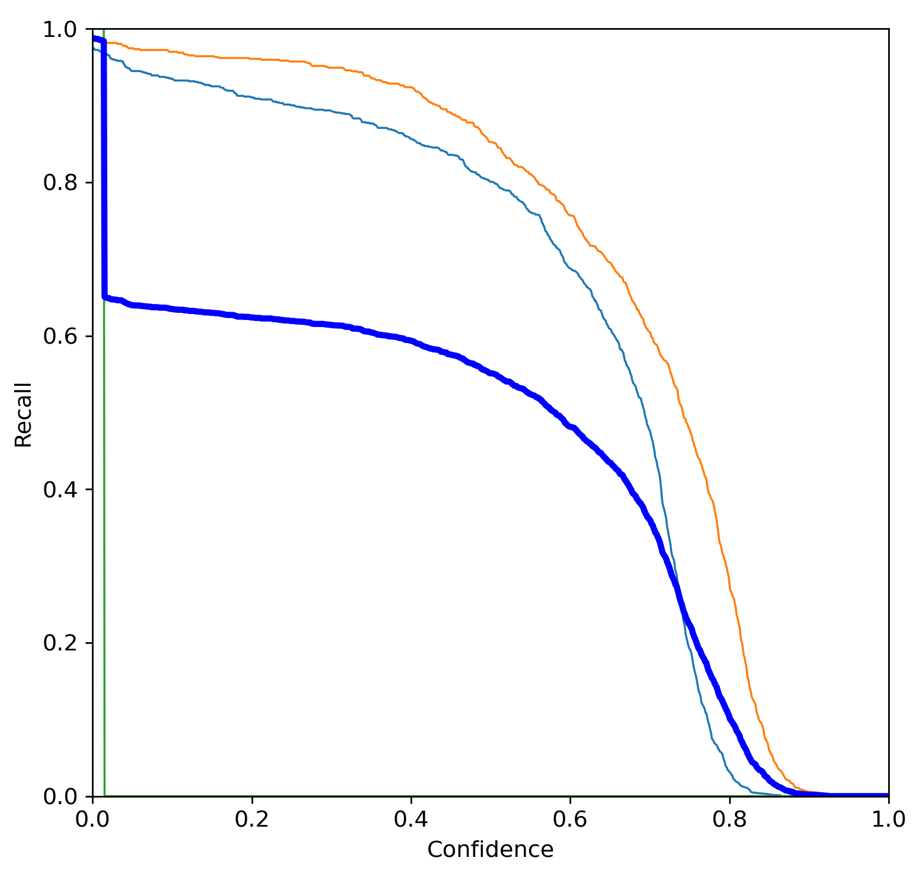


Рисунок 4.4 – График распространения ошибки

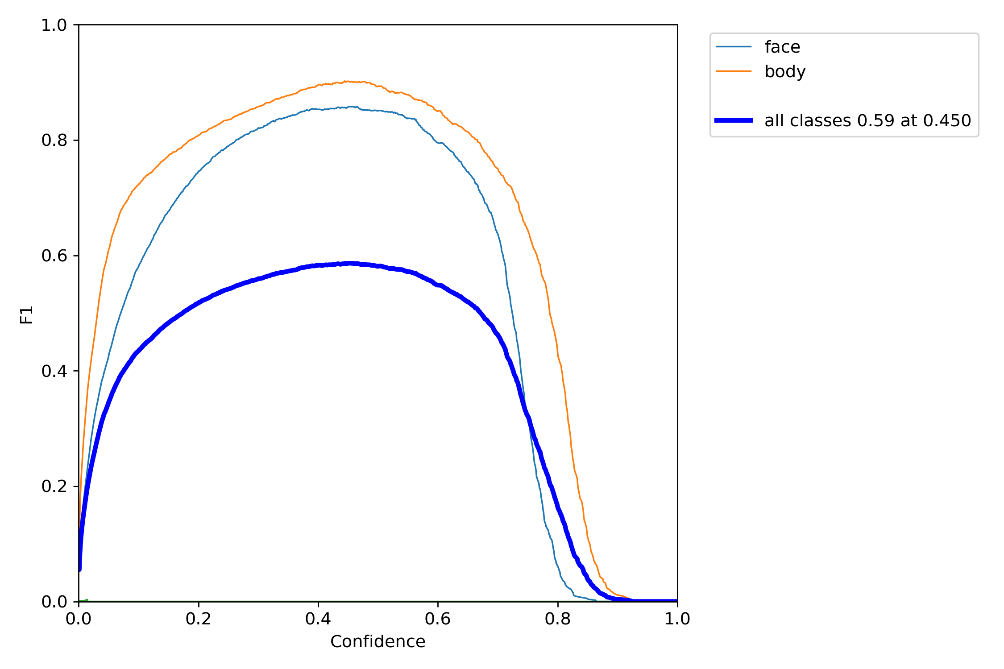


Рисунок 4.5 – График точности распознавания

Вывод: в результате эксперимента была обучена и сохранена нейронная сеть для детекции и идентификации человека на фото- и видеоизображении.

**Тест 2:** «Тестирование разработанной модели на фотоизображении»

Описание: необходимо протестировать обученную модель для детекции и идентификации человека на фотоизображении (рис. 4.6) (рис. 4.7).

Полученный результат:

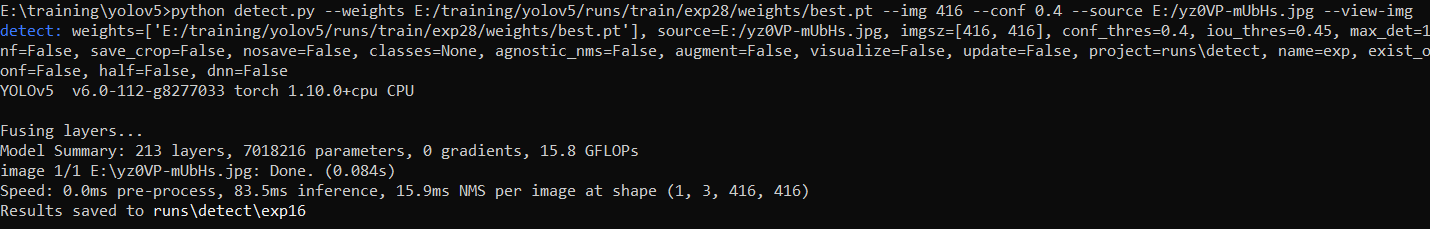


Рисунок 4.6– Запуск программы

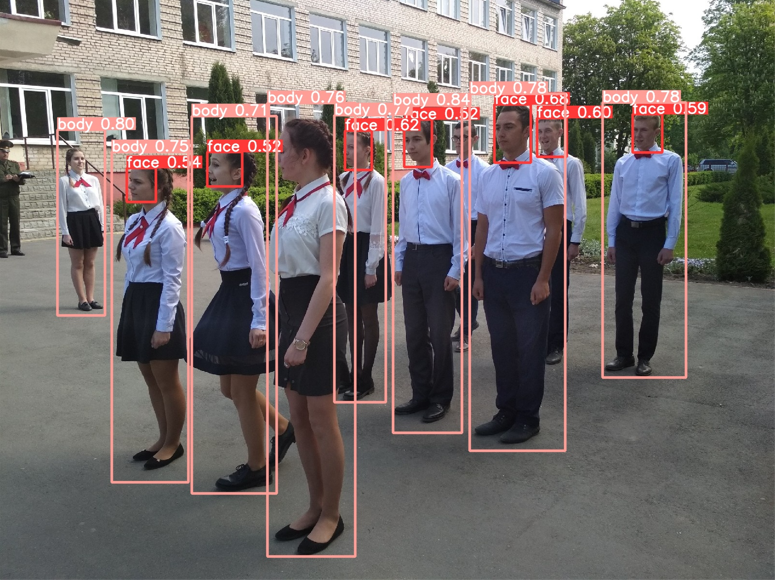


Рисунок 4.7– Распознавание на фотоизображении

Вывод: в результате эксперимента была протестирована модель для детекции и идентификации человека на фотоизображении. Она формирует не очень точные предсказания, так как точность распознавания нейронной сети меньше 0.95-0.98. Если увеличить объем тестовой выборки, подобрать более подходящие параметры для нейронной сети, можно получить более точные результаты распознавания речи. Но для этого нужны более мощные вычислительные ресурсы.

**Тест 3:** «Тестирование разработанной модели на видеоизображении»

Описание: необходимо протестировать обученную модель для детекции и идентификации человека на видеоизображении (рис. 4.8) (рис. 4.9).

Полученный результат:

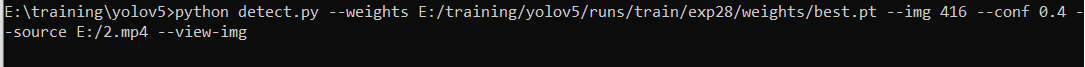


Рисунок 4.8– Запуск программы

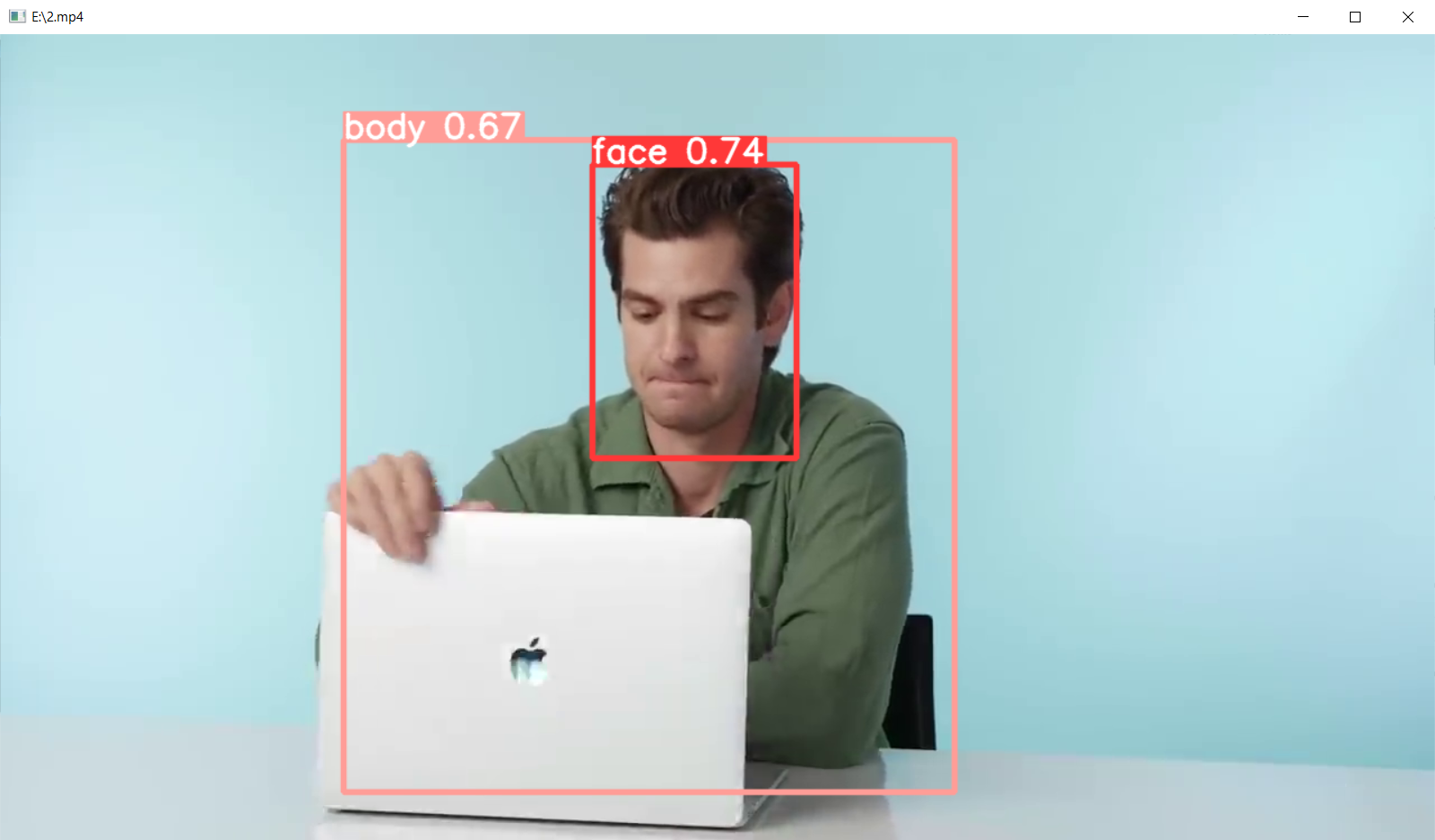


Рисунок 4.9– Распознавание на видеоизображении

Вывод: в результате эксперимента была протестирована модель для детекции и идентификации человека на видеоизображении. Она формирует не очень точные предсказания, так как точность распознавания нейронной сети меньше 0.95-0.98. Если увеличить объем тестовой выборки, подобрать более подходящие параметры для нейронной сети, можно получить более точные результаты распознавания речи. Но для этого нужны более мощные вычислительные ресурсы.

**Тест 4:** «Тестирование разработанной модели на веб-камере»

Описание: необходимо протестировать обученную модель для детекции и идентификации человека на веб-камере (рис. 4.10) (рис. 4.11).

Полученный результат:



Рисунок 4.10 – Запуск программы

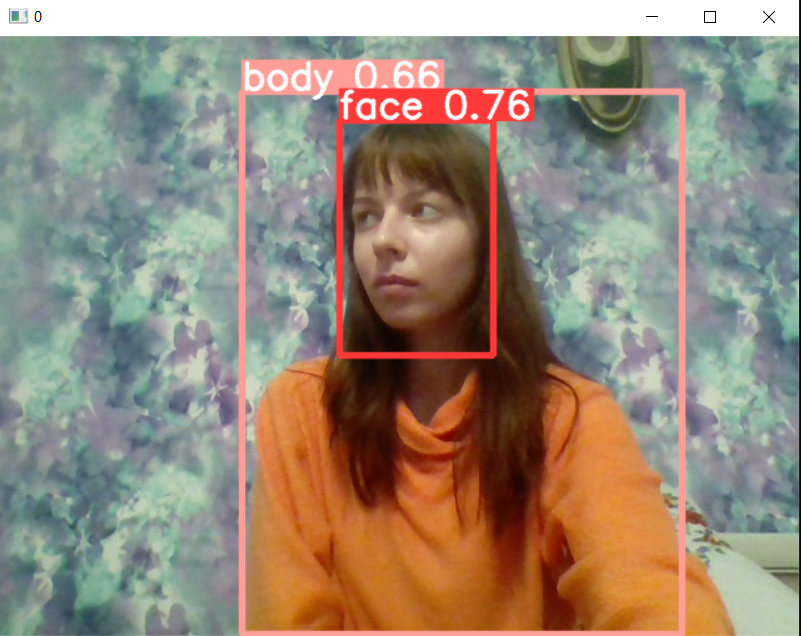


Рисунок 4.11– Распознавание на веб-камере

Вывод: в результате эксперимента была протестирована модель для детекции и идентификации человека на веб-камере. Она формирует не очень точные предсказания, так как точность распознавания нейронной сети меньше 0.95-0.98. Если увеличить объем тестовой выборки, подобрать более подходящие параметры для нейронной сети, можно получить более точные результаты распознавания речи. Но для этого нужны более мощные вычислительные ресурсы.

# Заключение

В ходе выполнения данного курсового проекта была разработана нейронная сеть на базе архитектуры YOLOV5s, обученная на наборе данных, сделанном с помощью Roboflow.

Полученная и используемая в проекте модель позволяет детектировать и идентифицировать лицо и целого человека на фото- и видеоизображениях, что может быть использовано в разных сферах человеческой деятельности. В ходе тестирования данного проекта было выяснено, что сеть такого типа, можно и дальше улучшать, подбирая необходимые условия и расширяя обучающую выборку.

Главным минусом нейросетевых моделей является повышенная временная сложность процесса обучения, высокая зависимость результата от начальной инициализации весовых коэффициентов нейронов, высокие требованиями к мощности вычислительных ресурсов и репрезентативности обучающего множества. Все это обуславливает необходимость наличия определенных навыков в использовании НС при решении практических задач подобного класса.

Программное обеспечение работает под управлением любой популярной операционной системой: Windows, Linux или MacOS с предустановленным интерпретатором Python и соответствующими библиотеками для него.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 2.105-95. Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам.
2. ГОСТ 19.504-79. Единая система программной документации ЕСПД. Руководство программиста. Требования к содержанию и оформлению.
3. ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения.
4. ГОСТ 19.005-85. ЕСПД.

Р-схемы алгоритмов и программ. Обозначения условные графические и правила выполнения.

1. ГОСТ 19.101-77. ЕСПД. Виды программ и программных документов.
2. ГОСТ 19.102-77. ЕСПД. Стадии разработки.
3. ГОСТ 19.103-77. ЕСПД. Обозначения программ и программных документов.
4. ГОСТ 19.401-78. ЕСПД. Текст программы. Требования к содержанию и оформлению.
5. ГОСТ 19.402-78. ЕСПД. Описание программы.
6. ГОСТ 7.1-2003. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления.
7. Roboflow [Электронный ресурс]. – 2021. – Режим доступа:

https://app.roboflow.com/gritsak-alexandra/face-and-body/1– Дата доступа: 12.12.2021.

1. PyTorch [Электронный ресурс]. – 2021. Режим доступа:

https://pytorch.org/tutorials/– Дата доступа: 12.12.2021.

13. Custom Object Detection with YOLOV5 [Электронный ресурс]. – 2021. Режим доступа: https://data-flair.training/blogs/custom-object-detection-yolo/– Дата доступа: 12.12.2021.