1.3 Выбор средств разработки

В ходе разработки потребуются:

1) работа с нейронными сетями;

2) разметка данных и ее обучение;

3) написание пользовательского интерфейса;

4) написание API.

Нейронные сети используются для решения сложных задач, которые требуют аналитических вычислений подобных тем, что делает человеческий мозг. Самыми распространенными применениями нейронных сетей является:

1. 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 т. д.

2. Предсказание — возможность предсказывать следующий шаг. Например, рост или падение акций, основываясь на ситуации на фондовом рынке.

3. 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.

Основная задача нейронной сети в данной работе — это распознавание и классификация объектов. В связи с чем была выбрана архитектура сверточной нейронной сети. На данный момент сверточная нейронная сеть и ее модификации считаются лучшими по точности и скорости алгоритмами нахождения объектов.

В настоящее время для написания нейронных сетей больше всего подходят такие языки программирования, как Python и R.

R — интерпретируемый язык программирования, основным способом работы с которым является командный интерпретатор. Главным недостатком данного языка является то, что он сложен в обучении и достаточно медленный [8].

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен, однако стандартная библиотека включает большой объём полезных функций. Основное преимущество языка Python в том, что он является универсальным и многоцелевым языком программирования [9]. Вследствие чего для работы с нейронными сетями был выбран язык программирования Python.

Для написания интерфейса была выбрана Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств [10]. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения и веб-службы [10]. А точнее Visual Studio Community — полнофункциональная, расширяемая и бесплатная интегрированная среда разработки для создания современных приложений Android, iOS и Windows, а также веб-приложений и облачных служб. Любой индивидуальный разработчик может создавать бесплатные или платные приложения с помощью Visual Studio Community [10].

В качестве языка был выбран соответственно объектно-ориентированный язык программирования C#.

Для разработки API были выбраны технология ASP.NET Core Web API: ASP.NET Core представляет собой переписанную и расширенную версию фреймворка ASP.NET, которая работает на платформе .NET Core. ASP.NET Core Web API предоставляет средства для создания легковесных и высокопроизводительных API. Он также поддерживает различные форматы данных и предоставляет гибкую конфигурацию и маршрутизацию.

2 Обзор моделей нейронных сетей

Нейронная сеть - математическая модель, а также ее программные и аппаратные реализации, предназначенные для поиска неких полезных представлений данных в пространстве представлений.

Нейронные сети состоят из следующих компонентов:

* данные – входные и выходные (иногда только входные);
* модель нейронной сети, состоящая из слоев;
* весовые параметры - те параметры, которые непосредственно изменяются в процессе обучения;
* функция потерь - определяет, каким образом будут обновляться весовые параметры сети для достижения желаемых результатов;
* оптимизатор - непосредственно отвечает за обновление весовых коэффициентов сети.

Процесс обновления весовых параметров нейронной сети, при котором ошибка сети, характеризующаяся функцией потерь, стремится к минимуму, называется обучением сети

Полносвязные нейронные сети могут выступать в качестве средства для решения задачи классификации изображений, однако на данный момент доминирующей архитектурой для решения этой задачи являются сверточные нейронные сети (CNN), которые и будут использоваться в данной работе.

Классическая нейронная сеть для классификации изображении состоит и следующих компонентов:

* сверточный слой;
* пулинговый слой (или слой субдискретизации);
* слой дропаута (или прореживания);
* блок линейных слоев.

Сверточный слой реализует операцию свертки. Она состоит в следующем: на вход слою подается изображение (исходное или уже обработанное), каждый фрагмент каждого канала изображения умножается на ядро сверткии поэлементно, результаты по фрагменту суммируются и определяются в позицию, аналогичную таковой у исходного фрагмента. Таким образом формируется карта признаков.

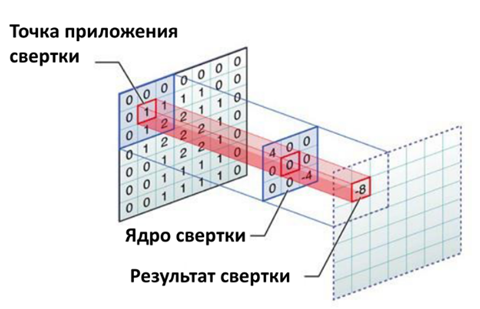


Рисунок 2.1 – Формирование карты признаков сверточным слоем

Пулинговый слой (или субдискретизации) уплотняет поступающие на него карты признаков, при этом группы пикселей сжимаются путем нелинейного преобразования, обычно таким преобразованием является максимизация.

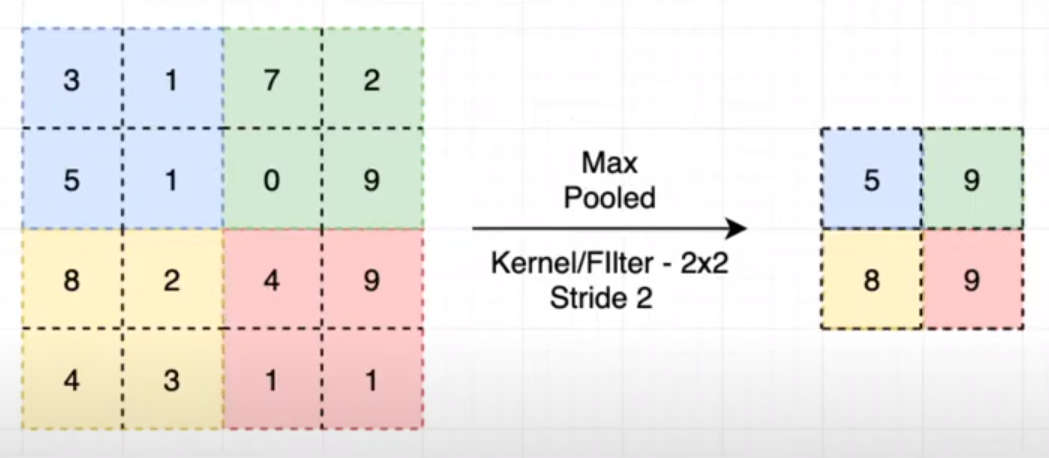


Рисунок 2.2 – Работа пулингового слоя

Слой дропаута обнуляет в процессе обучения некоторый случайный набор параметров сети. Применение этого слоя положительно влияет на склонность сверточной сети к обобщению данных.

Рассмотрим сверточные нейронные сети и выберем наиболее подходящие для задач данной работы.

2.1 Выбор нейросетевой модели для обнаружения человека

Faster R-CNN (Region-based Convolutional Neural Network) - архитектур нейронных сетей для обнаружения объектов. Она использует двухэтапный подход, включающий обнаружение предлагаемых областей и классификацию объектов внутри этих областей. Faster R-CNN обеспечивает высокую точность обнаружения.

SSD (Single Shot MultiBox Detector).

SSD предлагает более быстрый одноэтапный подход, обнаруживая объекты на разных масштабах и различных уровнях свертки. SSD хорошо подходит для задач реального времени.

Основные преимущества нейронной сети SSD:

1. Архитектура SSD позволяет детектировать объекты в реальном времени.

2. Качество работы близко к Faster R-CNN.

3. Используется большое количество default box-ов, покрывающих входное изображение на разных масштабах.

4. На этапе Inference архитектура SSD осуществляет детектирование 7308 объектов, большая часть из которых впоследствии фильтруется.

YOLOv4 (You Only Look Once) — очень популярная архитектура сверточных нейронных сетей, которая используется для распознавания множественных объектов на изображении. Главная особенность этой архитектуры по сравнению с другими состоит в том, что большинство систем применяют сверточные нейронные сети несколько раз к разным регионам изображения, в YOLO CNN применяется один раз ко всему изображению сразу. Сеть делит изображение на своеобразную сетку и предсказывает ограничивающие рамки и вероятности того, что там есть искомый объект для каждого участка.

Преимуществом данного подхода является то, что сеть смотрит на все изображение сразу и учитывает контекст при детектировании и распознавании объекта. Используя эту систему, нужно всего один раз пропустить изображение через нейронную сеть, чтобы предсказать объекты и их расположение. Это обеспечивает YOLO высокую скорость по сравнению с другими методами обнаружения объектов.

После проведенного анализа была выбрана архитектура модели нейронной сети YOLOv4 по следующим критериям:

1. Данная модель лучше детектирует небольшие объекты по сравнению с Faster R-CNN и SDD.

2. Имеет больше FPS (кадров в секунду) по сравнению с Faster R-CNN и проще в использовании по сравнению с SSD.

3. Более эффективная в режиме реального времени по сравнению с Faster R-CNN.

2.2 Выбор нейросетевых моделей для обнаружения средств индивидуальной защиты

Для задач классификации рассмотрим такие нейронные сети, как MixNets и EfficientNets.

MixNets и EfficientNets - это два разных подхода к созданию нейросетей, которые могут быть эффективны в различных задачах машинного обучения.

MixNets имеет следующие преимущества по сравнению с EfficientNets:

* меньшее число параметров (MixNets использует меньше параметров, чем EfficientNets, что делает ее более легкой и быстрой в работе);
* повышенная точность при обучении на маленьких наборах данных (MixNets показала лучшие результаты в классификации изображений на меньших наборах данных, т.к. использует оптимизированные функции свертки);
* более легкая архитектура (MixNets использует свертки с меньшим числом фильтров, что позволяет ей работать с менее ресурсной аппаратурой).

Исходя из данных преимуществ будем использовать MixNets:

1. MixNet\_S – для классификации правильности применения и обнаружения красной каски;

2. MixNet\_L – для классификации CИЗ и положения человека (лицом или спиной к камере).

MixConvs - концепция объединения ядер нескольких размеров в один слой, которая является основным строительным блоком для MixNets. На рисунке 2.3 приведена архитектура MixNet-M, показывающая различные группы ядер.

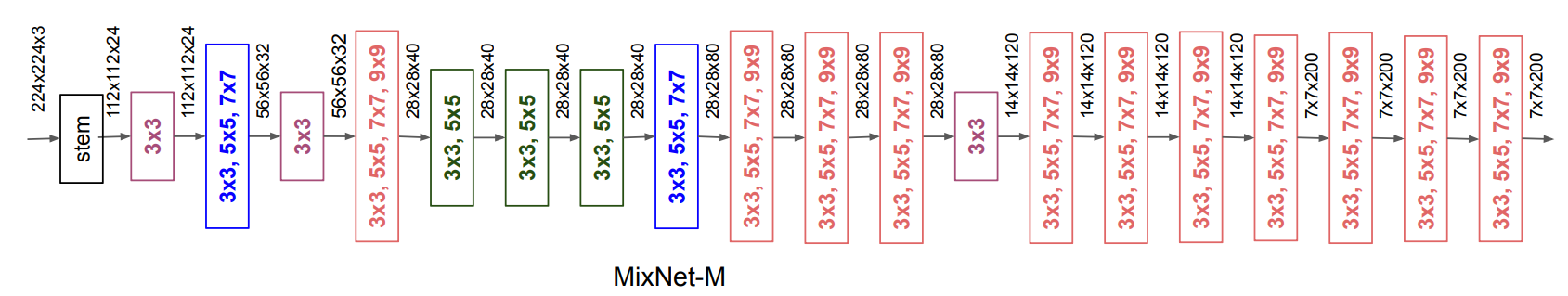


Рисунок 2.3 – Архитектура сети MixNet-M

Вначале используются меньшие ядра, затем более крупные ядра постепенно интегрируются по мере прохождения данных через слои. MixNet-L - это просто MixNet-M с коэффициентом глубины 1,3.

Коэффициент глубины является параметром, который используется для сокращения числа параметров и операций в сверточных слоях. Это позволяет ускорить вычисления и снизить требования к памяти.

Установка коэффициента глубины в 1 соответствует применению стандартных сверточных операций. При увеличении коэффициента глубины увеличивается количество параметров и вычислений, что может улучшить точность, но также сделать нейронную сеть более трудоемкой.