Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Студент Федоров Виктор Павлович
фамилия, имя, отчество

Группа ИУ7-63Б

Тип практики Технологическая

Название предприятия ООО ПолисСОФТ

Студент _____ _____
подпись, дата *фамилия, и.о.*

Руководитель практики _____ _____
подпись, дата *фамилия, и.о.*

Оценка _____

2021 г.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Разработать программный модуль, отвечающий за классификацию объектов на взлётно-посадочной полосе, а также предоставить программный интерфейс для взаимодействия с модулем. Данный модуль рассматривается в качестве составной части приложения, решающего следующую задачу: разработать WEB-приложение, позволяющее обнаружить и классифицировать мусор на взлётно-посадочной полосе по видеоматериалам, взятым с камер аэропорта.

Введение

Ежедневно в мире совершается около 100 тысяч гражданских авиарейсов, однако каждый полёт обязательно связан с рисками и опасностями. Одним из источников опасности является наличие мусора на взлётно-посадочной полосе. Кроме того, объекты на ВПП могут нанести существенный материальный ущерб из-за задержки рейсов и повреждения самолётов.

Многие аэропорты решают проблему обнаружения и классификации объектов на ВПП путем привлечения к этому работников аэропорта, однако в таком случае ошибка, связанная с человеческим фактором, может стоить жизней многих людей. С развитием современных технологий стали появляться системы автоматического обнаружения и классификации мусора, которые позволили снизить риск возникновения опасных ситуаций в аэропорту. В большинстве случаев подобные системы базируются на использовании камер.

Целью данной практики является разработка WEB-приложения, которое позволит по видеоинформации с камеры аэропорта обнаружить и классифицировать мусор на ВПП. Для достижения поставленной цели необходимо решить следующие задачи:

- провести анализ существующих алгоритмов обнаружения и классификации объектов по видеоинформации;
- разработать архитектуру приложения и схему взаимодействия модулей;
- разработать приложение в соответствии с выбранными архитектурой и алгоритмами обнаружения и классификации объектов;

1. Аналитический раздел

1.1 Обзор существующих решений

1.1.1 FODetect

Автоматизированная система FODetect обеспечивает непрерывный мониторинг взлетно-посадочных полос аэропорта для быстрого удаления посторонних предметов с полосы в любых погодных условиях. FODetect использует сочетание радаров миллиметрового диапазона с изображениями в высоком разрешении для большей эффективности обнаружения. Из недостатков данной системы стоит отметить использование радаров на взлетной полосе. Радары излучают электромагнитные волны, что может нарушить функционирование других технических средств аэропорта и самолётов.

1.1.2 FOD Finder XM

Система предназначена для обнаружения посторонних предметов, находящихся в любом месте в пределах обозначенных для наблюдения границ аэродрома. Недостатками этой системы является использование радаров на ВПП и отсутствие в системе камер, что делает невозможным классификацию обнаруженных объектов, а также оперативную проверку обнаружения человеком.

1.2 Анализ задачи классификации объектов

Задача классификации объектов сводится к выделению нескольких классов объектов, которые могут быть обнаружены на входных данных, и соотнесение каждого изображения с одним из выделенных классов. Имеется множество объектов (ситуаций), разделенных некоторым образом на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется *обучающей выборкой.* Классовая принадлежность остальных объектов не известна. Требуется построить

алгоритм, способный классифицировать произвольный объект из исходного множества.

Классифицировать объект - значит, указать номер (или наименование класса), к которому относится данный объект. Классификация объекта - номер или наименование класса, выдаваемый алгоритмом классификации в результате его применения к конкретному объекту. Задача классификации относится к разделу обучения с учителем.

1.3 Выделение классов объектов

По мере эксплуатации на взлетно-посадочной полосе могут появляться различные дефекты, которые должны быть своевременно устранены, а также различные посторонние объекты: болты, гайки, инструменты, обрывки шин, бутылки, куски покрытия и другое. Такие объекты являются источником опасности для воздушных судов.

Целью проекта как раз и является обнаружение и классификация подобных объектов. Для классификации объектов необходимо выделить соответствующие классы. Было выделено 6 классов объектов, которые могут встречаться на взлетно-посадочной полосе. Это:

- металл;
- стекло;
- пластик;
- бумага;
- объекты живой природы (животные и птицы);
- мусор (общий класс для всего остального).

Также хотелось выделить в отдельный класс камни (например, куски асфальта), но найти существующий датасет не получилось, а для создания его с нуля не хватило ресурсов.

1.4 Анализ методов классификации объектов

Задача классификации является классической задачей обучения с учителем. В ходе изучения методов классификации объектов был проведен сравнительный анализ и выделены преимущества и недостатки тех или иных алгоритмов. Сравнивались простые метрические классификаторы и сверточные нейросети. Среди простых метрических классификаторов выделяется метод ансамбля деревьев. Идея данного алгоритма заключается в том, чтобы использовать много простых классификаторов, например, 50 простых деревьев, и агрегировать результат.

Таблица 1: Сравнение метрических классификаторов

	Нечувствительность к выбросам	**Построение модели на обучающих примерах**	**Мультиклассовая классификация**	**Отсутствие необходимости в кросс-валидации**
Дерево решений	- (нужно стричь дерево или строить не полностью)	+	+	-
Метод ближайших соседей	- (если соседей мало)	-(если данных много)	+	-
Метод опорных векторов	+	+	-+	-
Логистическая регрессия	+	+	-+	+ (метод сам содержит максимизацию точности)
Ансамбль деревьев	+	+	+	+ (точность достигается за счёт усреднения)

1.5 Сравнение сверточных нейронных сетей

При более глубоком изучении задачи было установлено, что на данный момент самыми популярными и надежными методами классификации объектов на фото и видео являются сверточные нейронные сети. Сверточная нейронная сеть - это нейронная сеть, в которой присутствует слой свертки (convolutional

layer). Обычно в сверточных нейронных сетях также присутствует слой субдискретизации (pooling layer) и полносвязный слой (fully connected layer).

1.5.1 ResNet

Победителем ILSVRC 2015 с ошибкой в 3,57 % стал ансамбль из шести сетей типа ResNet (Residual Network), разработанный в Microsoft Research.

Авторы ResNet заметили, что с повышением числа слоев сверточная нейронная сеть может начать деградировать — у неё понижается точность на валидационном множестве. Так как падает точность и на тренировочном множестве, можно сделать вывод, что проблема состоит не в переобучении сети.

Было сделано предположение, что если свёрточная нейронная сеть достигла своего предела точности на некотором слое, то все следующие слои должны будут выродиться в тождественное преобразование, но из-за сложности обучения глубоких сетей этого не происходит. Для того чтобы «помочь» сети, было предложено ввести пропускающие соединения (Shortcut Connections), изображённые на рисунке 2.

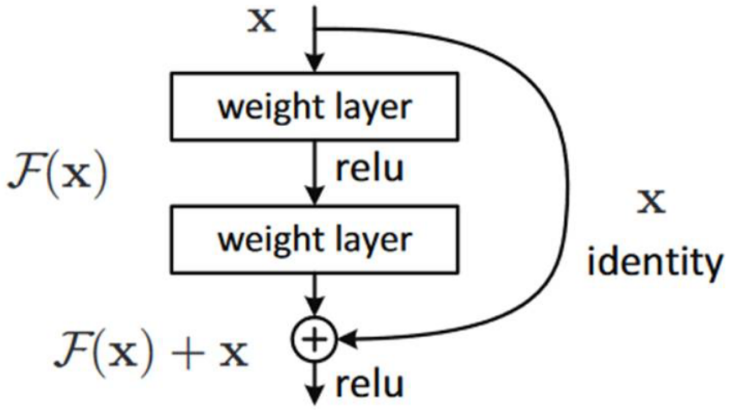

Рис. 2. Пропускающее соединение

На рисунке 3 приведена архитектура модификации нейронной сети ResNet с 32 слоями.

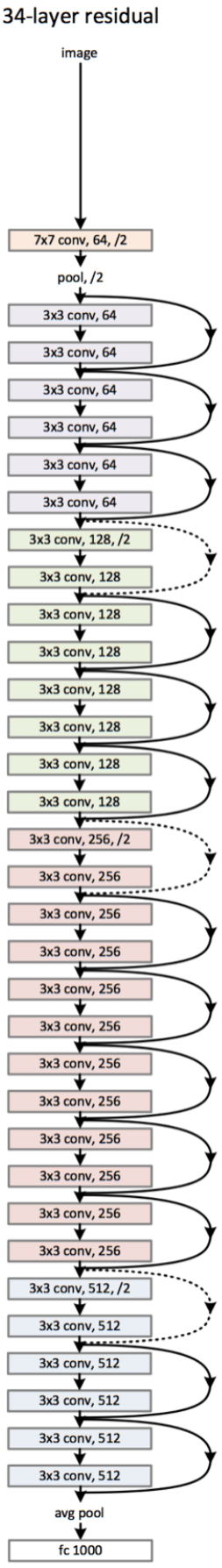

Рис. 3. Архитектура ResNet

1.5.2 Inception

Inception-v1 — победитель ILSVRC 2014 с top-5 ошибкой 6,7 %, также известный как GoogLeNet. Создатели этой сети во главе с Christian Szegedy исходили из факта, что после каждого слоя сети необходимо сделать выбор — будет ли следующий слой свёрткой с фильтром 3x3, 5x5, 1x1 или же слоем субдискретизации.

Каждый из таких слоёв полезен — фильтр 1x1 выявляет корреляцию между каналами, в то время как фильтры большего размера реагируют на более глобальные признаки, а слой субдискретизации позволяет уменьшить размерность без больших потерь информации.

Вместо того чтобы выбирать, какой именно слой должен быть следующим, предлагается использовать все слои сразу, параллельно друг другу, а затем объединить полученные результаты в один. Чтобы избежать роста числа параметров, перед каждым слоем свёртки используется свёртка 1x1, которая уменьшает число карт признаков. Такой блок слоев назвали модулем Inception. Она представлен на рисунке 1.

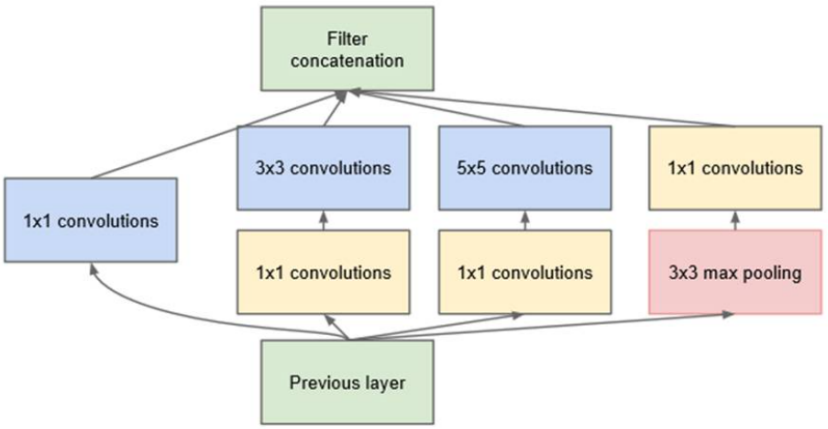

Рис. 1: Модуль Inception

1.5.3 AlexNet

В 2012 году на конкурсе ILSVRC по классификации изображений впервые победила нейронная сеть — AlexNet, достигнув top-5 ошибки 15,31 %. Для сравнения, метод, не использующий свёрточные нейронные сети, получил ошибку 26,1 %. В AlexNet были собраны новейшие на тот момент техники для улучшения работы сети.

Обучение AlexNet из-за количества параметров сети происходило на двух GPU, что позволило сократить время обучения в сравнении с обучением на CPU. Также оказалось, что использование функции активации ReLU (Rectified Linear Unit) вместо более традиционных функций сигмоиды и гиперболического тангенса позволило снизить количество эпох обучения в шесть раз.

Формула ReLU следующая: $y(x) = max(0, x)$. ReLU позволяет побороть проблему затухания градиентов, свойственную другим функциям активации.

Таблица 2: Сравнение нейронных сетей

Нейронная сеть	**top-1**	**top-5**
ResNet	19,38%	4,49%
AlexNet	39,00%	17%
Inception	29,00%	9,20%

top-5 - одна из основных оценок ошибки алгоритма. Считается, что алгоритм не ошибся, если правильная категория объекта находится среди пяти категорий, выданных алгоритмом, как наиболее вероятные.

По сравнению с простыми классификаторами сверточные нейронные сети показывают очень высокую точность результатов. Однако почти все нейросети требуют огромных вычислительных мощностей.

Выводы

В результате изучения алгоритмов классификации объектов были выбраны сверточные нейронные сети (CNN) и нейронная сеть ResNet18 в частности. Она позволяет добиться качественных результатов, и при этом не требует высоких вычислительных мощностей.

1.6 Выбор архитектуры приложения

При проектировании системы необходимо выбрать между монолитной и микросервисной архитектурой приложения.

1.6.1 Монолитная архитектура

Монолитная архитектура предполагает наличие единой платформы, на которой сконцентрированы все компоненты приложения. Такой подход имеет ряд преимуществ:

- быстрая разработка за счёт более простого межмодульного взаимодействия;
- простота реализации;
- простой запуск программы.

Однако у монолитной архитектуры есть ряд существенных недостатков:

- при обновлении одного из модулей возникает необходимость разворачивать заново всё приложение;
- возникновение ошибки в одном из модулей может привести к остановке всего приложения, а не одного компонента;
- «перегруженность» кода за счёт нескольких взаимосвязанных компонентов, границы между которыми размыты;

Таким образом, монолитная архитектура подходит для быстрой разработки небольших продуктов, долгосрочная поддержка и модификация которых не предполагается.

1.6.2 Микросервисная архитектура

Все перечисленные недостатки монолитной архитектуры решает микросервисная архитектура, которая предполагает наличие нескольких платформ, на каждой из которых разворачивается небольшое приложение, соответствующее определенному компоненту программного продукта. Эти приложения работают на разных серверах и взаимодействуют друг с другом по сети. К преимуществам данной архитектуры можно отнести:

- обновление одного сервиса не затрагивает другой;
- четкое разграничение ответственности и сохранение модульности;
- возможна разработка сервисов на разных языках;
- ошибка в одном сервисе не приведёт к остановке всего приложения;
- возможность масштабирования модулей независимо от других;

Вывод: микросервисная архитектура является более гибкой по сравнению с монолитной и имеет более существенные преимущества, поэтому для решения задачи практики была выбрана микросервисная архитектура.

2. Конструкторский раздел

2.1 Диаграмма компонентов

Диаграмма компонентов приложения выглядит следующим образом:

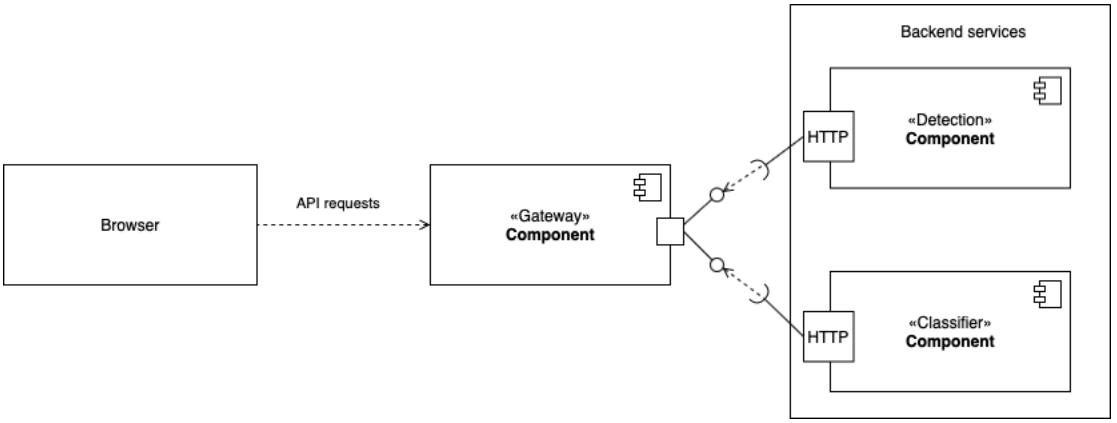

Рисунок 2 Диаграмма компонентов приложения

На диаграмме компонентов фигурируют браузер (интерфейсная часть приложения) и 3 сервиса:

- browser (интерфейс) отображает данные в виде, понятном пользователю, а также отправляет видеоматериалы, полученные от пользователя, на сервис Gateway;

- сервис Gateway отвечает за координацию запросов и подготовку данных для остальных сервисов и интерфейса;

- сервис Detection получает изображения от сервиса Gateway и возвращает ему фрагменты кадра, на которых были обнаружены объекты;

- сервис Classification получает фрагмент кадра с обнаруженным объектом от сервиса Gateway и возвращает ему класс этого объекта;

Сервисы общаются друг с другом посредством HTTP-запросов с поддержкой REST стиля.

2.2 Описание сервиса классификации

В аналитическом разделе были рассмотрены методы классификации объектов. Основная работа строится вокруг модели для предсказания класса объектов. Модель получает на вход обработанное изображение и возвращает класс объекта в качестве результата.

Основная сложность заключается в обучении модели. Для обучения использовался датасет, состоящий из объектов 6 классов (пластик, металл, бумага, стекло, животные и «остальное»). Каждый класс объектов был представлен 1500 картинками.

Отдельно стоит отметить классификацию пустых изображений без объектов. Относить их к классу “остальные” некорректно, так как непосредственно объектов на изображении нет. Для решения этой проблемы было установлено минимальное значение вероятности, при котором сервис классифицировал объект, в противном случае фрагмент помечался, как «unknown».

Модель для классификации обучалась и тестировалась с помощью трех выборок из датасета: обучающей выборки (training sample), валидационной выборки (validation sample) и тестовой или контрольной выборки (test sample).

В качестве метрики для проверки качества полученной модели была выбрана *accuracy* (процент верных предсказаний). Данная метрика определяется, как отношение количества правильных ответов алгоритма к их общему числу. Метрика *accuracy* хорошо подходит для задачи, так как каждый класс объектов является одинаково важным.

Вывод: в данном разделе была приведена архитектура ПО и указано место модуля классификации в нём.

3. Технологический раздел

3.1 Выбор языка программирования

Для программной реализации сервиса обнаружения был выбран язык программирования Python как наиболее удобное средство для создания серверной части приложения. Основные его архитектурные черты – это динамическая типизация, автоматическое управление памятью, механизм обработки исключений, высокоуровневые структуры данных. Немаловажно, что Python поддерживает объектно-ориентированное программирование.

К плюсам языка Python относятся его простота, большое количество библиотек для использования в самых различных областях. Основным минусом языка является низкая скорость выполнения программ по сравнению с низкоуровневыми языками.

3.2 Использование сторонних библиотек и фреймворков

Для реализации межмодульного взаимодействия был использован фреймворк Flask как наиболее простое и удобное средство для создания веб-приложений. С его помощью сервисы общаются между собой посредством HTTP-запросов. Кроме того, для отправки HTTP-запросов используется библиотека requests.

Для работы с изображениями было решено использовать библиотеку OpenCV. OpenCV предоставляет более 2500 оптимизированных алгоритмов, которые включают полный набор как классических, так и самых современных алгоритмов компьютерного зрения, машинного обучения и обработки изображений.

Заключение

В проделанной работе был создано веб-приложение, позволяющее обнаруживать и классифицировать объекты на взлетно-посадочной полосе аэропорта. В ходе работы был проведен сравнительный анализ алгоритмов для обнаружения и классификации объектов, было получено базовое представление о методах решения классической задачи машинного обучения, задачи классификации. Также были получены базовые знания об устройстве нейронных сетей и глубоком обучении в целом. Также в ходе практики были усовершенствованы навыки работы в команде.

Список использованных источников

1.Сверточные нейронные сети [Электронный ресурс]. URL: https://paperswithcode.com/sota/image-classification-on-imagenet

2. Метрики качества в задачах классификации [Электронный ресурс]. URL: https://pythonru.com/baza-znanij/metriki-accuracy-precision-i-recall

3. Задача обучения с учителем [Электронный ресурс]. URL: http://www.machinelearning.ru/

4. Датасеты [Электронный ресурс]. URL: https://www.kaggle.com/