«Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)

Направление	09.03.04 Программная инженерия
Профиль	Разработка программно-информационных систем
Факультет	КТИ
Кафедра	МО ЭВМ

К защите допустить

И.о. зав. кафедрой А.А. Лисс

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

ТЕМА: РАЗРАБОТКА СРЕДСТВА ВИЗУАЛИЗАЦИИ И МОНИТОРИНГА КРИМИНАЛЬНОГО КОНТЕНТА ИЗ ТУМАННОЙ ВЫЧИСЛИТЕЛЬНОЙ СРЕДЫ

Студент		_____ *подпись*	Л.С. Рослова
Руководитель	к.т.н., доцент	_____ *подпись*	А.А. Лисс
Консультанты	к.т.н., доцент	_____ *подпись*	А.Ю. Первицкий
		_____ *подпись*	А.Н. Субботин
	к.т.н.	_____ *подпись*	М.М. Заславский

Санкт-Петербург

2023

ЗАДАНИЕ

НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Утверждаю

И.о. зав. кафедрой МО ЭВМ

_____ А.А. Лисс

«___»_____2023 г.

Студент Рослова Л.С. Группа 9304

Тема работы: Разработка средства визуализации и мониторинга криминального контента из туманной вычислительной среды

Место выполнения ВКР: Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

Исходные данные (технические требования):

ОС Windows, ОС Linux

Содержание ВКР:

Введение, Обзор предметной области, Функциональные требования, Архитектура системы, Тестирование программной системы, Безопасность жизнедеятельности, Заключение

Перечень отчетных материалов: пояснительная записка, иллюстративный материал.

Дополнительные разделы: Безопасность жизнедеятельности.

Дата выдачи задания «_04_»__апреля_____2023 г.

Дата представления ВКР к защите «9»__июня___2023 г.

Студент _____ Л.С. Рослова

Руководитель к.т.н., доцент _____ А.Ю. Первицкий

Консультант _____ А.Н. Субботин

КАЛЕНДАРНЫЙ ПЛАН ВЫПОЛНЕНИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Утверждаю
И.о. зав. кафедрой МО ЭВМ
_____ А.А. Лисс
«___»_____2021 г.

Студент	Рослова Л.С.		Группа	9304
Тема работы: Разработка средства визуализации и мониторинга криминального контента из туманной вычислительной среды				

№ п/п	Наименование работ	Срок выполнения
1	Обзор литературы по теме работы	04.04 – 03.05
2	Обзор предметной области	04.05 – 06.05
3	Постановка задачи и формализация функциональных требований	06.05 – 10.05
4	Архитектура программной реализации	10.05 – 12.05
5	Тестирование разработанной системы	12.05 – 22.05
6	Безопасность жизнедеятельности	22.05 – 23.05
7	Оформление пояснительной записки	23.05 – 31.05
8	Оформление иллюстративного материала	01.06 – 04.06
9	Предзащита	01.06

Студент _____ Рослова Л.С.

Руководитель к.т.н., доцент _____ А.Ю. Первицкий

Консультант _____ А.Н. Субботин

РЕФЕРАТ

Пояснительная записка 65 стр., 19 рис., 16 табл., 24 ист.

КРИМИНАЛЬНЫЙ КОНТЕНТ, МОНИТОРИНГ, АНАЛИЗ, ИНТЕРНЕТ-РЕСУРС, МОДЕЛЬ, НЕЙРОННАЯ СЕТЬ, ТУМАННЫЕ ВЫЧИСЛЕНИЯ

Объектом исследования являются интернет-ресурсы, содержащие криминальный контент.

Предметом исследования являются методы анализа контента веб-сайтов с использованием машинного обучения.

Цель работы: разработка архитектуры расширения для браузера анализирующего посещаемый веб-ресурс на криминальный контент с применением средств машинного обучения и с возможностью минимизации задержек при обработке данных.

Результатом данной работы является создание расширение для браузера, которое анализирует изображения на интернет-ресурсе моделями глубокого обучения и предоставляет информацию о том, к какой категории оно относится. Для этого использовалась готовые программные библиотеки, позволяющие строить и обучать нейронной сети с целью автоматической классификации образов. В работе описывается архитектура расширения, состоящая из двух основных компонентов: модуля загрузки изображений и отображения результатов анализа, модуля загрузки и подключения модели к расширению.

ABSTRACT

The result of this work is the creation of a browser extension that analyzes images on an Internet resource with deep learning models and provides information about which category it belongs to. To do this, we used ready-made software libraries that allow us to build and train neural networks in order to automatically classify images. The paper describes the architecture of the extension, consisting of two main components: the module for loading images and displaying analysis results, the module for loading and connecting the model to the extension.

Содержание

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ЯП – язык программирования;

HTML – стандартизированный язык разметки документов для просморта веб-страниц в браузере;

Датасет – набор данных для обучения нейронной сети;

JSON – текстовый формат обмена данными;

Тензор – обобщение векторов и матриц на более высокие измерения.

Автоэнкодеры - нейронные сети прямого распространения, которые восстанавливают входной сигнал на выходе.

ReLU (Rectified Linear Unit) - это наиболее часто используемая функция активации при глубоком обучении. Данная функция возвращает 0, если принимает отрицательный аргумент, в случае же положительного аргумента, функция возвращает само число.

Свёрточный слой - это основной блок свёрточной нейронной сети. Слой свёртки включает в себя для каждого канала свой фильтр, ядро свёртки которого обрабатывает предыдущий слой по фрагментам (суммируя результаты поэлементного произведения для каждого фрагмента).

Полносвязный слой - этот слой обрабатывает каждый элемент предыдущего слоя, выполняя матричное перемножение этих элементов со своими весами.

Dropout - метод регуляризации искусственных нейронных сетей, предназначен для уменьшения переобучения сети за счет предотвращения сложных коадаптаций отдельных нейронов на тренировочных данных во время обучения.

Пулинг - понижение размерности изображения, применяемой во избежание переобучения модели нейронной сети.

Batch Normalization - метод, при котором некоторым слоям нейронной сети на вход подаются данные, предварительно обработанные и имеющие нулевое математическое ожидание и единичную дисперсию.

CNN (англ. convolutional neural network) – свёрточная нейронная сеть.

Аугментация - это методика создания дополнительных данных из имеющихся данных.

DOM (англ. Document Object Model) — это независящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML-, XHTML- и XML-документов, а также изменять содержимое, структуру и оформление таких документов.

ВВЕДЕНИЕ

Согласно данным из отчёта ‘Global Digital 2022’[1] процент интернет-пользователей в 2022 году перешёл черту в 62% от населения мира. При этом каждый имеет возможность выкладывать в сеть новый материал. Особенно активно в сеть загружаются изображения, которые практически никогда не проходят проверку на криминальное содержимое перед публикацией. Актуальность расширения для браузера, анализирующего такой материал на криминальное содержание с помощью нейронной сети, обусловлена необходимостью защиты пользователей от нежелательного контента в интернете. В современном мире существует большое количество сайтов, на которых имеется деструктивное содержание, такое как порнография, алкоголь, наркотики, оружие и т.д. Этот контент может быть вредным для психического и эмоционального здоровья пользователей, особенно детей и подростков.

Традиционные методы блокировки нежелательного контента, такие как фильтры URL и списки блокировки сайтов, не всегда эффективны и могут пропустить определенные типы деструктивного материала. Однако использование методов машинного обучения, в том числе нейронных сетей, позволяет более точно и надежно определять деструктивное содержание на изображениях.

Таким образом, разработка расширения для браузера, анализирующего изображения на криминальный контент с помощью нейронной сети, является актуальной задачей, которая позволит пользователям защититься от негативного контента на посещаемых веб-ресурсах, а также представляет возможность использования машинного обучения и туманных вычислений для решения задач анализа изображений в реальном времени.

Цель работы: разработка архитектуры расширения для браузера анализирующего посещаемый веб-ресурс на криминальный контент с применением средств машинного обучения и туманных вычислений для минимизации задержек.

Задачи данной работы:

1. Изучение методов и технологий машинного обучения, необходимых для анализа изображений.
2. Разработка архитектуры расширения для браузера, которое будет обращаться к туманным сервисам для анализа изображений на криминальный контент.
3. Реализация механизмов, обеспечивающих сбор и анализ данных о посещаемых веб-ресурсах.
4. Интеграция средств машинного обучения для обработки и анализа данных на наличие криминального контента.
5. Тестирование и оптимизация работы расширения для браузера, с целью повышения скорости работы продукта.

Объектом исследования являются интернет-ресурсы, содержащие криминальный контент.

Предметом исследования являются методы анализа контента веб-сайтов с использованием машинного обучения.

Практическая ценность работы:

1. Повышение безопасности пользователей в интернете. Расширение может обнаруживать наличие криминального контента на посещаемых веб-ресурсах, что поможет избежать неприятных ситуаций и предотвратить потенциальные угрозы для пользователей.
2. Уменьшение распространения криминального контента в интернете. Расширение может обнаруживать криминальный контент на посещаемых веб-ресурсах и предупреждать пользователей о его наличии. Это может помочь снизить распространение криминального контента в интернете и сделать его менее доступным.
3. Упрощение родительского контроля. Родители могут использовать расширение для браузера, чтобы отслеживать посещаемые их детьми веб-ресурсы и обнаруживать наличие криминального контента на этих ресурсах.

1 ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Обработка и анализ изображений

Существует множество способов обработки и анализа изображений, включая:

1. Преобразование изображений: изменение размера, поворот, перспективная коррекция, цветокоррекция, обрезка и т.д.
2. Фильтрация изображений: удаление шума, повышение резкости, фильтры низких и высоких частот, усреднение и т.д.
3. Сегментация изображений: разделение изображения на отдельные объекты или регионы, разделение переднего плана и фона, определение контуров и т.д.
4. Оптическое распознавание символов (OCR): преобразование текста на изображении в машинно-читаемый текст.
5. Классификация изображений: определение объектов, сцен, лиц и т.д. на изображении.
6. Детектирование объектов: обнаружение объектов на изображении с помощью методов компьютерного зрения.
7. Распознавание лиц: обнаружение и идентификация лиц на изображении.
8. Машинное обучение: использование алгоритмов машинного обучения для реализации упомянутых выше методов.

В данной работе рассмотрен последний подход с применением машинного обучения, так как он имеет хорошую скорость обработки данных и высокую точность.

1.1.1 Возможности обработки изображений моделями машинного обучения

Существует несколько подходов к обработке изображений моделями машинного обучения:

- Классификация изображений: подход, при котором изображения разбиваются на классы, каждый из которых соответствует определенной категории. Обычно используются сверточные нейронные сети, которые позволяют автоматически извлекать признаки из изображений.
- Детектирование объектов на изображениях: подход, при котором модель машинного обучения позволяет обнаруживать и выделять объекты на изображении. Для этого используются различные архитектуры нейронных сетей, такие как Faster R-CNN или SSD.
- Сегментация изображений: подход, при котором каждый пиксель изображения относится к определенному классу. Обычно используются семантические сегментационные модели, которые позволяют разделять изображение на отдельные части.
- Генерация изображений: подход, при котором модель машинного обучения может генерировать новые изображения на основе имеющихся данных. Для этого используются генеративно-состязательные сети (GAN), автоэнкодеры и другие модели.
- Реконструкция изображений: подход, при котором модель машинного обучения позволяет восстанавливать недостающие части изображения или улучшать его качество. Для этого используются различные архитектуры нейронных сетей, такие как автоэнкодеры или глубокие сверточные сети.

Все имеющиеся возможности можно объединять друг с другом для решения необходимых задач. Классификация изображений является наиболее подходящим подходом для разрабатываемой системы, так как она предназначено для анализа содержимого веб-ресурсов на наличие криминального контента, в частности на наличие изображений, связанных с алкоголем, наркотиками и т.д. Преимуществом использования данного подхода является способность к обнаружению сложных иерархических зависимостей между признаками изображений.

Для решения задачи обнаружения деструктивного материала необходимо провести классификацию изображений на пять основных категорий: «алкоголь», «наркотики», «порнография», «оружие», «нормальное изображение». Для этого можно обучить нейронную сеть на большом количестве изображений, размеченных на представленные группы. Затем, после обучения, модель можно интегрировать в расширение браузера, чтобы автоматически сканировать веб-ресурсы на предмет наличия негативного контента.

1.1.2 Нейронные сети для классификации изображений

Нейронные сети для классификации изображений - это одна из наиболее актуальных областей машинного обучения. Они широко применяются в различных областях, таких как компьютерное зрение, медицинская диагностика, робототехника, автономные транспортные средства, контроль качества в производстве и многих других. Для классификации изображений, нейронные сети обучаются на большом наборе данных, размеченных на различные категории. В процессе обучения, модели постепенно определяют характеристики, которые помогают им правильно классифицировать материал.

Существует множество различных видов нейронных сетей, которые используются для классификации изображений. Некоторые из них включают в себя:

- Сверточные нейронные сети (Convolutional Neural Networks, CNN) - это наиболее распространенный тип нейронных сетей для классификации изображений. Они специально разработаны для обработки многомерных данных, таких как изображения.
- Рекуррентные нейронные сети (Recurrent Neural Networks, RNN) - это тип нейронных сетей, который хорошо подходит для работы с последовательными данными, такими как речь или текст. Они также могут использоваться для классификации изображений с использованием предварительно обученных сверточных нейронных сетей.

- Многослойные персептроны (Multilayer Perceptrons, MLP) - это тип нейронных сетей, который имеет несколько скрытых слоев и может быть использован для классификации изображений.
- Глубокие вероятностные модели (Deep Probabilistic Models) - это класс моделей машинного обучения, который использует вероятностные методы для классификации изображений.
- Автоэнкодеры (Autoencoders) - это нейронные сети, которые могут использоваться для обучения скрытых представлений изображений, которые затем могут быть использованы для классификации изображений.
- Глубокие более широкие сети (Deep Wide Networks) - это тип нейронных сетей, который сочетает в себе свойства глубоких и широких сетей, что позволяет обрабатывать данные более эффективно.

Каждый из этих типов нейронных сетей имеет свои преимущества и недостатки в зависимости от конкретного применения, и выбор наиболее подходящей модели будет зависеть от многих факторов, таких как объем данных и доступность вычислительных ресурсов.

1.1.3 VGG-16

VGG16 — модель сверточной нейронной сети, предложенная K. Simonyan и A. Zisserman из Оксфордского университета в статье “Very Deep Convolutional Networks for Large-Scale Image Recognition” [2]. Модель достигает точности 92.7% — топ-5, при тестировании на датасете ImageNet в задаче распознавания объектов на изображении. Этот набор данных состоит из более чем 14 миллионов изображений, принадлежащих к 1000 классам.

Архитектура VGG16 представлена на рисунке 1.

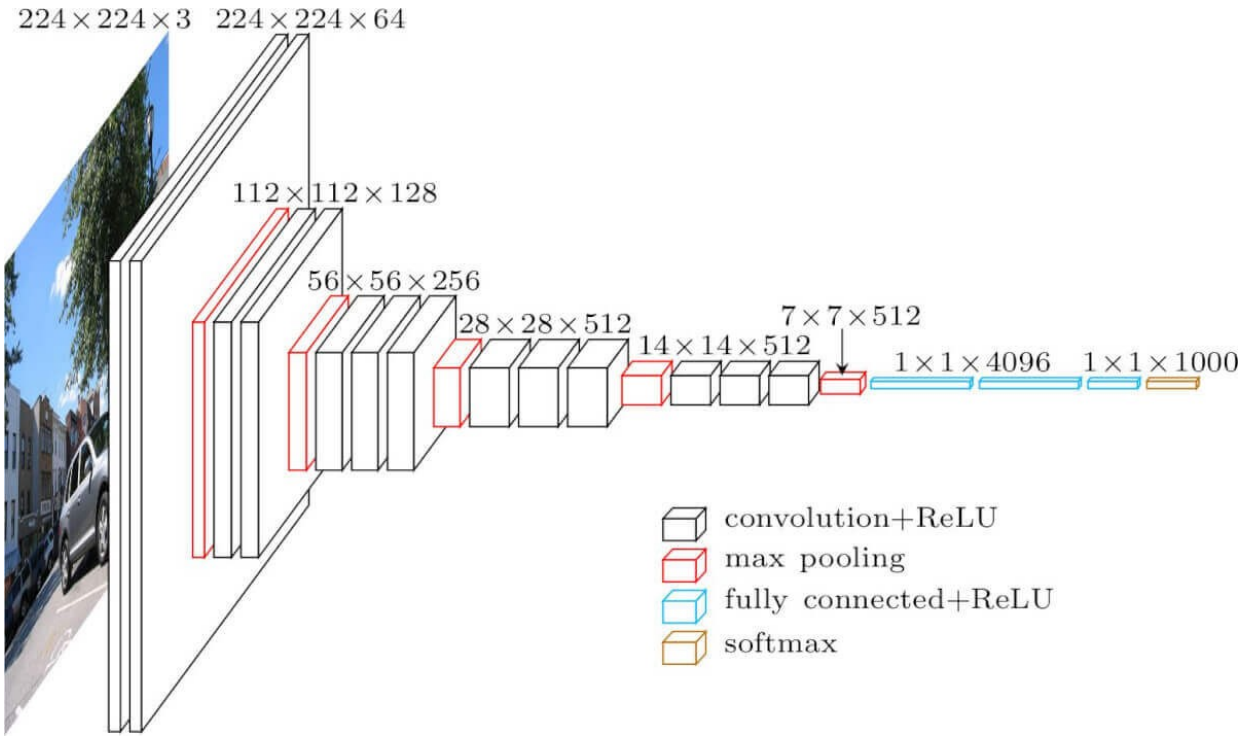

Рисунок 1 - Архитектура VGG16

Нейронная сеть VGG16 состоит из 16 слоев, включая 13 сверточных слоев и 3 полносвязных слоя. Она имеет фиксированный размер входного изображения 224x224 пикселя. Сверточные слои применяют фильтры размером 3x3 и используют функцию активации ReLU. После каждых двух сверточных слоев следует слой максимальной подвыборки размера 2x2. В конце сети расположены три полносвязных слоя, где последний слой используется для классификации на заданное количество классов. VGG16 имеет более 138 миллионов обучаемых параметров.

1.1.4 InceptionV3

InceptionV3 [3] - это сверточная нейронная сеть, разработанная компанией Google, которая является улучшенной версией модели InceptionV1 и InceptionV2. Она использует специальные блоки, называемые "Inception modules", которые объединяют в себе несколько операций свертки с различными размерами ядер и операций пулинга для извлечения признаков из изображений. Благодаря этому, InceptionV3 имеет высокую точность в задачах

классификации изображений, а также может быть использована для решения задач сегментации и детектирования объектов. InceptionV3 имеет более чем 20 слоев и обучается на большом количестве данных для достижения высокой точности в классификации. Архитектура InceptionV3 представлена на рисунке 2.

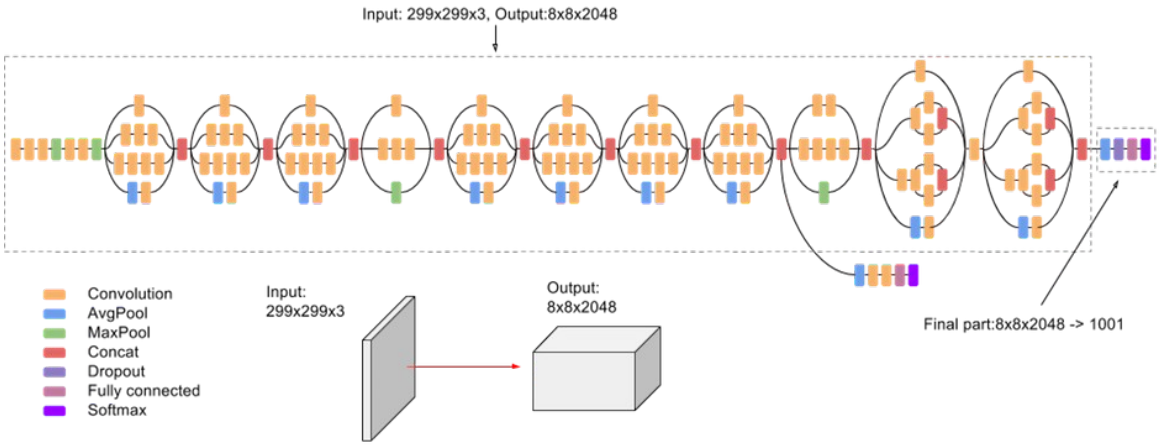

Рисунок 2 - Архитектура InceptionV3

Архитектура InceptionV3 состоит из сверточных слоев, блоков "Inception modules" и полносвязных слоев. Внутри "Inception modules" используются несколько операций свертки с различными размерами ядер и операций пулинга. Архитектура имеет более чем 20 слоев и заканчивается полносвязными слоями для классификации. InceptionV3 также использует некоторые техники регуляризации, такие как Dropout и L2 регуляризация, чтобы избежать переобучения.

1.1.5 MobileNetV2

MobileNetV2 [4] - это сверточная нейронная сеть, оптимизированная для мобильных устройств с ограниченными вычислительными ресурсами. Архитектура состоит из нескольких блоков, которые включают в себя последовательность слоев "depthwise separable convolution", которые заменяют стандартные сверточные слои. Схема данной структуры представлена на рисунке 3.

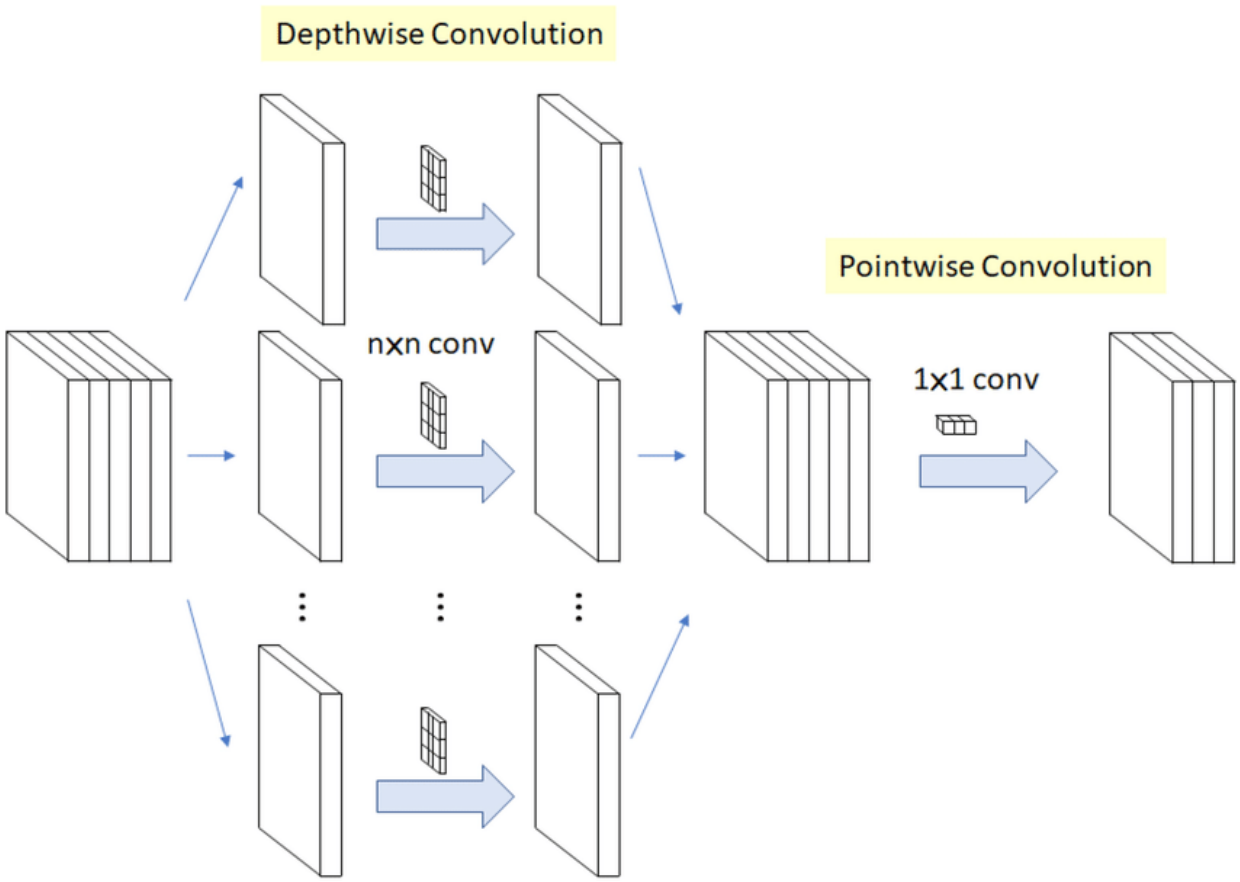

Рисунок 3 - Схема слоя «depthwise separable convolution»

Эти блоки уменьшают количество параметров и операций с плавающей запятой, что позволяет сети работать на мобильных устройствах с ограниченной вычислительной мощностью. MobileNetV2 также использует слои Batch Normalization и ReLU, чтобы ускорить сходимость и обеспечить нелинейность в сети. Она достигает высокой точности на задачах классификации изображений, при этом имея небольшой размер и низкий уровень потребления энергии. Архитектура сети представлена на рисунке 4.

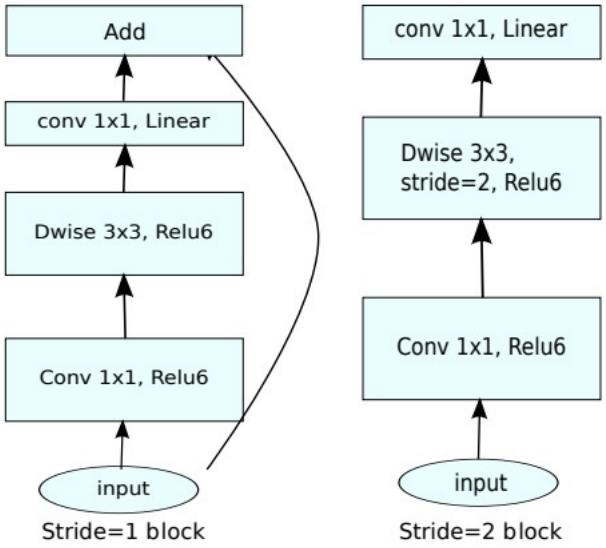

Рисунок 4 - Архитектура MobileNetV2

Блоки включают в себя несколько последовательных операций. Сначала входные данные проходят через сверточный слой с ядром размера 3x3, затем применяется функция активации ReLU6. Затем следует операция "бутстрапинга" - сверточный слой с ядром 1x1, который увеличивает количество каналов признаков. После этого следует еще один сверточный слой с ядром 3x3, снова с функцией активации ReLU6. Благодаря такой конфигурации блоков, MobileNetV2 обладает высокой точностью на задачах классификации изображений при относительно низком количестве параметров и быстродействии.

1.1.6 ResNet50

ResNet — это сокращенное название для Residual Network (дословно — «остаточная сеть») [5]. Это глубокая нейронная сеть, предназначенная для классификации изображений. Её основной принцип состоит в том, чтобы добавить "сквозные" соединения между блоками сверточных слоев, что позволяет сети изучать глубокие представления изображений. Архитектура данной модели представлена на рисунке 5.

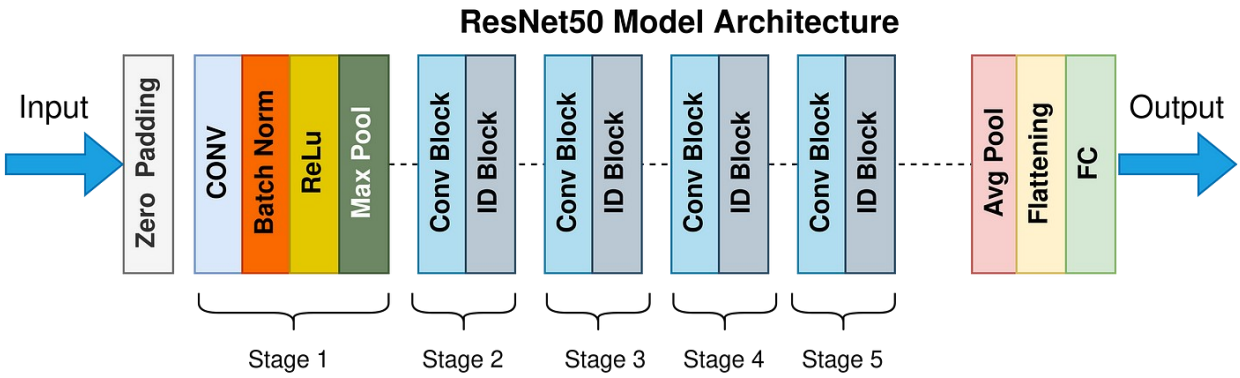

Рисунок 5 - Архитектура ResNet50

Модель состоит из 50 сверточных слоев и нескольких полносвязных слоев. Внутри каждого блока ResNet50 использует сквозные соединения для суммирования входных и выходных данных, чтобы увеличить скорость обучения и улучшить точность предсказаний. ResNet50 также использует сверточные слои с ядрами 1x1 и 3x3, а также групповую нормализацию, чтобы добиться лучшей точности на задачах классификации изображений.

1.1.7 EfficientNet

EfficientNet - это архитектура сверточной нейронной сети и метод масштабирования, который равномерно масштабирует все измерения глубины/ширины/разрешения с использованием составного коэффициента [6]. Разработчики предложили следующие параметры для коэффициента:

- Глубина = 1.20
- Ширина = 1.20
- Разрешение = 1.15

Данные значения не являются константными и пользователь может переопределить их под свои задачи.

Архитектура модели представлена на рисунке 6.

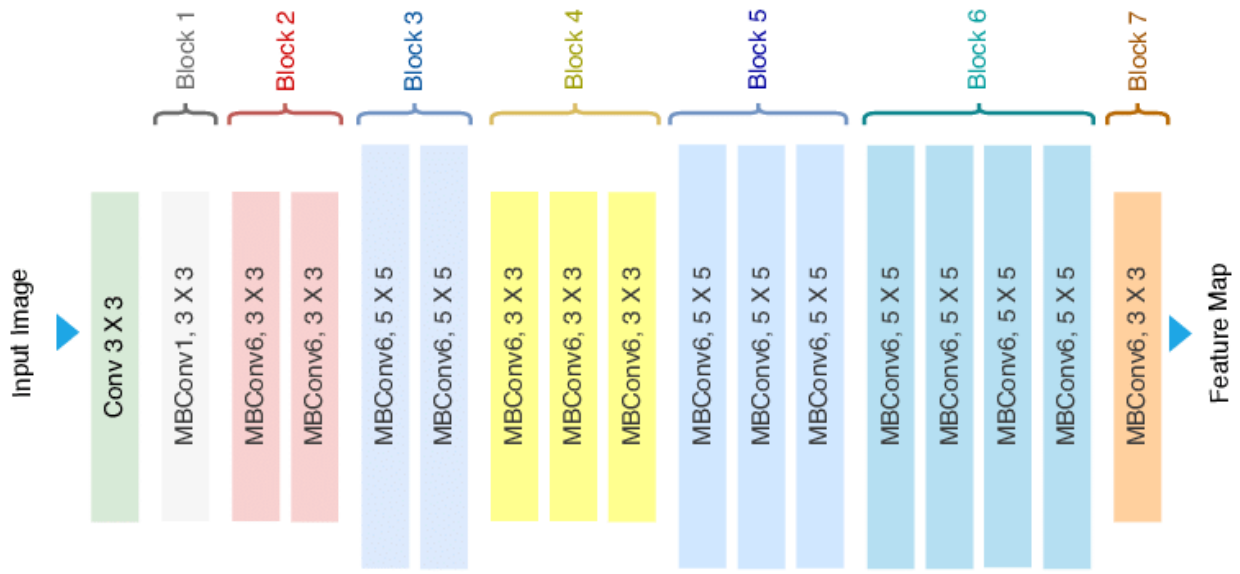

Рисунок 6 - Архитектура EfficientNet

EfficientNet использует блоки Fused MBConv, которые объединяют свертки 3x3 и свертки с ядрами 1x1, что увеличивает скорость обучения и эффективность работы сети. MBConv означает мобильное инвертированное узкое место Convolution (аналогично MobileNetv2). Внутри каждого блока модель использует механизмы подавления сигнала, такие как Squeeze-and-Excitation структуры, которые позволяют сети лучше фокусироваться на важных объектах изображений. Нейронная сеть также использует аугментацию данных и Dropout для уменьшения переобучения и повышения точности.

1.2 Сравнение моделей машинного обучения

После того, как были рассмотрены основные модели машинного обучения для анализа изображений, необходимо сравнить их между собой. Это может помочь выбрать наиболее эффективный и точный алгоритм для решения задачи распознавания изображений, что особенно важно в случае, когда точность классификации является ключевым фактором в принятии решений. Кроме того, сравнение моделей также может помочь в оптимизации производительности и ресурсов, необходимых для работы модели.

Для сравнения рассмотренных нейронных сетей можно использовать следующие критерии:

- Количество параметров: чем меньше параметров, тем более легковесной и быстрой является модель.
- Точность (accuracy) на тестовых данных: чем выше точность, тем лучше модель распознает изображения.
- Скорость предсказаний (вычисления на новых данных): чем быстрее модель может производить вычисления на новых данных, тем более эффективной она является для реальных приложений.
- Размер модели: чем меньше размер модели, тем меньше занимаемое ею место на диске или в памяти устройства, что может быть важным для некоторых приложений.

Результаты сравнения нейронных сетей по выделенным критериям представлены в таблице 1.

Таблица 1 – Сравнение моделей машинного обучения

Модель	Размер (мб)	Топ-1 точность (%)	Топ-5 точность (%)	Параметры (млн)	Время вывода на CPU (мс)	Время вывода на GPU (мс)
VGG16	528	71.3	90.1	138.4	69.5	4.2
InseptionV3	92	77.9	93.7	23.9	42.2	6.9
ResNet50	98	74.9	92.1	25.6	58.2	4.6
MobileNetV2	14	71.3	90.1	3.5	25.9	3.8
EfficientNet	220	85.3	97.4	54.4	1578.9	61.6

Поскольку эта работа направлена на анализ веб-ресурсов на криминальный контент, то показатель точности модели является главным приоритетом. Опираясь на результаты сравнения, можно выделить нейронную сеть EfficientNet как наиболее успешную в классификации изображений. Однако

она также имеет и самое высокое время вывода результата, что можно нивелировать применением туманных вычислительных сред.

1.3 Туманные вычисления

Туманные вычисления (англ. Fog computing) – это модель распределенных вычислений, которая предназначена для обработки данных в облачных сетях ближе к месту их создания и использования [7]. В отличие от классической облачной вычислительной модели, где данные обрабатываются на удаленных серверах, туманные вычисления предлагают обрабатывать данные на более близких к конечным пользователям устройствах (например, на борту транспорта, в здании или на устройствах IoT), что позволяет снизить задержку и увеличить быстродействие приложений. Таким образом, такой подход является дополнением к облачным вычислениям, предлагая новую модель распределенной обработки данных в цифровых сетях. Архитектура туманной среды представлена на рисунке 7.

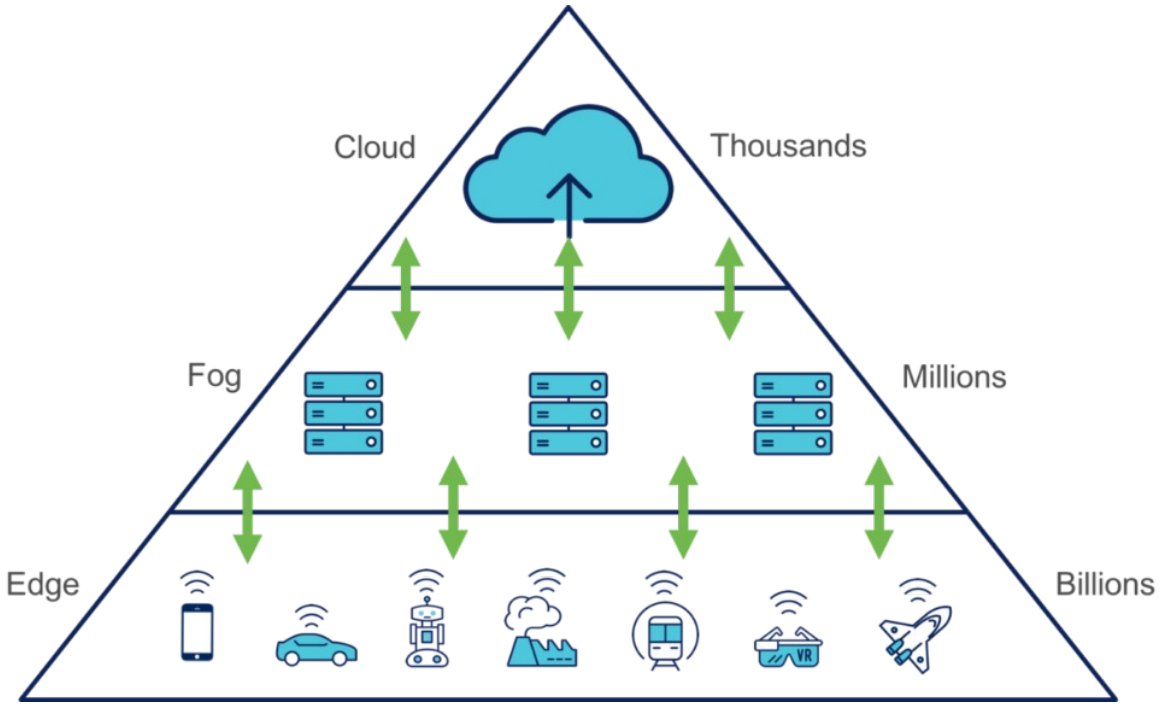

Рисунок 7 - Архитектура туманной вычислительной среды

Туманные среды выполняют следующие функции:

- Обеспечение доступа к вычислительным ресурсам и программному обеспечению через сеть Интернет, что позволяет предоставлять вычислительные ресурсы пользователям на удаленных серверах.
- Автоматическое масштабирование ресурсов для удовлетворения изменяющихся требований к вычислениям, что позволяет быстро реагировать на изменения в объеме вычислений.
- Обеспечение безопасности данных и приложений, которые используются в туманных вычислениях, с помощью механизмов шифрования и других мер защиты.
- Управление ресурсами и производительностью, что позволяет более эффективно использовать ресурсы и повышать производительность приложений.
- Расширение возможностей для использования Интернета вещей и других технологий, которые требуют быстрого доступа к большому количеству данных и вычислительной мощности.

Применение туманных сред в разработке расширения для браузера может помочь в следующих аспектах:

- Расширение может быть развернуто на удаленных серверах в туманной среде, что позволяет снизить нагрузку на локальные ресурсы устройства пользователя.
- Туманные вычисления могут предоставить расширению доступ к большим вычислительным ресурсам и библиотекам машинного обучения, которые могут быть недоступны на локальном устройстве пользователя.
- При использовании туманных вычислений расширение может быстро анализировать большое количество данных, что позволяет улучшить качество и точность классификации.

1.4 Вывод по разделу

В данном разделе были рассмотрены различные подходы к обработке изображений, включая классические методы и нейронные сети. Были описаны такие архитектуры нейронных сетей, как VGG16, InceptionV3, MobileNetV2, ResNet50 и EfficientNet. Также в результате сравнения упомянутых моделей была выявлена наиболее точная. Для минимизации её задержек при работе предложен вариант интеграции машинного обучения в туманные вычислительные среды.

Использование машинного обучения для обработки изображений имеет множество преимуществ перед ручной обработкой, включая высокую точность и автоматическую обработку больших объемов данных. Кроме того, применение нейронных сетей может ускорить процесс разработки расширения для браузера, которое будет использоваться для анализа изображений на криминальный контент. Также использование туманных вычислений позволит снизить нагрузку на локальные ресурсы компьютера и обеспечить более быструю и эффективную обработку данных.

2 ФОРМУЛИРОВКА ТРЕБОВАНИЙ К РЕШЕНИЮ И ПОСТАНОВКА ЗАДАЧИ

2.1 Постановка задачи

Необходимо разработать программную систему визуализации и мониторинга криминального контента в сети Интернет для анализа сайтов на наличие деструктивного содержания. Продукт должен обеспечивать анализ изображений за минимальное время за счёт возможности развёртывания модуля машинного обучения в облаке или тумане с большими вычислительными мощностями. Затем необходимо провести оценку разработанной системы — оценить время анализа изображений при работе на разных компьютерных мощностях.

2.2 Требования к решению

Разрабатываемый продукт должен удовлетворять следующим требованиям:

1. Высокая точность: расширение должно иметь высокую точность в определении криминального контента на сайтах.
2. Высокая скорость: расширение должно работать быстро и эффективно, чтобы пользователь мог использовать его без критических задержек.
3. Гибкость: должна быть возможность подключения и использования нескольких моделей машинного обучения в зависимости от задач.
4. Простота использования: расширение должно быть легко и удобно в использовании для пользователей всех уровней.

2.2 Обоснование требований к решению

Вышеописанные требования к расширению для анализа сайтов на криминальный контент с применением средств машинного обучения и туманных вычислений могут быть обоснованы следующим образом:

1. Надежность: требуется высокая точность и надежность анализа сайтов на криминальный контент, поскольку неверные результаты могут при-

вести к серьезным последствиям, таким как блокировка незаконных контент-провайдеров или неправомерное обвинение законопослушных пользователей.

2. Высокая производительность: требуется высокая скорость анализа сайтов, поскольку пользователи не будут ждать длительное время загрузки страниц, а также для того, чтобы снизить затраты на обработку данных.
3. Адаптивность: расширение должно быть способным адаптироваться к изменяющимся требованиям и условиям, например появление новых видов криминального контента или более усовершенствованных нейронных сетей.
4. Простота использования: программа должна быть простой в использовании для даже для тех пользователей, которые не имеют опыта работы с подобными приложениями.

2.2 Выводы

В данном разделе были выдвинуты требования к разрабатываемому расширению для браузера, анализирующему сайты на криминальный контент средствами машинного обучения и туманных вычислений. Требования включают в себя: высокую точность анализа, быстродействие, возможность адаптации к изменяющимся требованиям и условиям, а также наличие простого интерфейса пользователя.

Основная задача состоит в разработке архитектуры расширения, которая будет соответствовать всем вышеперечисленным требованиям и обеспечивать эффективный анализ веб-ресурсов на наличие криминального контента с помощью методов машинного обучения и туманных вычислений.

3 АРХИТЕКТУРА ПРОГРАММНОЙ СИСТЕМЫ

3.1 Используемые технологии

Для решения поставленной задачи по разработке расширения браузера, удовлетворяющего вышеописанным требованиям были выбраны следующие технологии и цели их использования:

- JavaScript [8] - ЯП, который позволит реализовать само расширение браузера. На его основе будет производится работа с DOM-элементами [16] для получения изображения на анализ, его предобработки перед анализом нейронной сетью, а также вывод результата предсказания категории на экран.
- Python [9] - ЯП, на котором будет реализована нейронная сеть. Также с его помощью написан скрипт для создания датасета криминальных изображений, на котором будет обучена модель.
- Туманные вычисления или fog computing [10] - технология, которая повысит производительность и уменьшит задержки при работе приложения.
- Tensorflow [11] - это библиотека для машинного обучения, группы технологий, которая позволяет обучать искусственный интеллект решению разных задач. Благодаря ей будет реализована и обучена нейронная сеть, отличающая изображения от деструктивных и приемлемых.
- Express.js [12] - это фреймворк для backend-разработки на JavaScript внутри среды исполнения Node.js [13]. На его основе будет реализован сервер, в котором расположиться нейронная сеть JSON-формата.

3.2 Трансферное обучение модели EfficientNet

Трансферное обучение (transfer learning) - это метод обучения глубоких нейронных сетей, который позволяет использовать знания, полученные при обучении на одной задаче, для решения другой задачи [17]. Вместо обучения сети с нуля на новом наборе данных, трансферное обучение позволяет использовать заранее обученную модель как базовую, затем модифицировать

ее для решения новой задачи. Это позволяет существенно уменьшить время и количество данных, необходимых для обучения новой модели, а также улучшить ее качество.

Такой подход был использован для обучения глубокой нейронной сети EfficientNet классифицировать изображения по 5 категориям:

1. Алкоголь
2. Наркотики
3. Порнография
4. Оружие
5. Нормальное изображение

Для начала необходимо было создать датасет, содержащий изображения с криминальным контентом. Готовых наборов данных найти не удалось, поэтому было принято решение написать скрипт на Python. Он принимает на вход строку запроса и необходимое количество иллюстраций к скачиванию. В скрипте средствами Selenium [18] эмитируется взаимодействия пользователя с браузером, в результате чего происходит поиск и скачивание изображений по указанном ранее значениям.

Затем собранные данные необходимо привести к структуре, указанной на рисунке 8. Количество папок внутри архива должно соответствовать количеству классов, которым необходимо обучить сеть. В данной работе каждая директория имеет по 3 тысячи изображений с соответствующим содержанием.

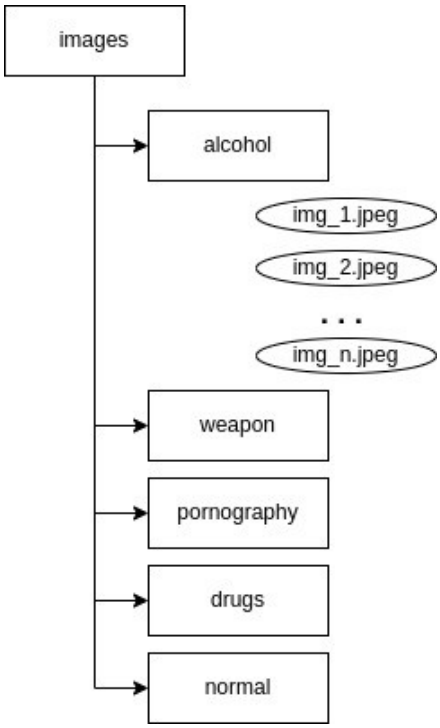

Рисунок 8 - Структура датасета

Далее был реализован второй Python скрипт, для импорта готовой и обученной модели EfficientNet из Keras Applications [14]. Эта библиотека также предоставляет методы для конвертирования датасетов в нужный для нейронных сетей формат. Так, собранный архив изображений методом tf.keras.utils.image_dataset_from_directory() преобразуется в тензор, поступающий на вход модели на обучение. Изображения при этом сжимаются до размеров 180 x 180 пикселей.

После обучения на датасете нейронная сеть показала точность в 92% на тестовых данных. Затем модель была сохранена в формате HDF5 (.h5).

Чтобы использовать натренированную сеть в браузере, была установлена библиотека Tensorflow.js, которая предоставляет скрипт конвертации моделей из HDF5 в Layers. Целевой формат TensorFlow.js Layers — это каталог, содержащий файл model.json и набор разделенных файлов веса в двоичном формате. Файл model.json содержит как топологию модели, так и манифест файлов весов.

3.3 Архитектура системы

Программная реализация расширения подразумевает клиент-серверную архитектуру. Всего имеется 3 модуля:

1. Расширение:
   - Расширение будет реализовано как часть браузера и будет иметь доступ к загружаемым в браузер страницам.
   - Расширение будет содержать код для сбора информации о загруженной странице и изображении, с которым произошло взаимодействие пользователя. Далее эта информация передаётся сервис-воркеру для последующей обработки.
   - Расширение будет иметь интерфейс для пользователя, который будет отображать результаты анализа страницы на наличие криминального контента.
2. Сервис-воркер:
   - Сервис-воркер будет предоставлять API клиенту для связи с сервером.
   - Сервис-воркер позволяет кэшировать контент и контролировать работу, когда нет доступа к сети или интернет очень медленный.
3. Сервер:
   - Сервер будет хранить модели машинного обучения и использовать их при анализе страниц.
   - Сервер будет обрабатывать изображение, что бы его можно было проанализировать средствами машинного обучения.

Все компоненты будут взаимодействовать между собой посредством API. Расширение будет считывать нужное изображение и отправлять сообщения сервис-воркеру для координации запросов к серверу. Сервер произведёт необходимою обработку и анализ, а затем передаст результат работы в расширение для вывода. Архитектура описанной системы представлена на рисунке 8.

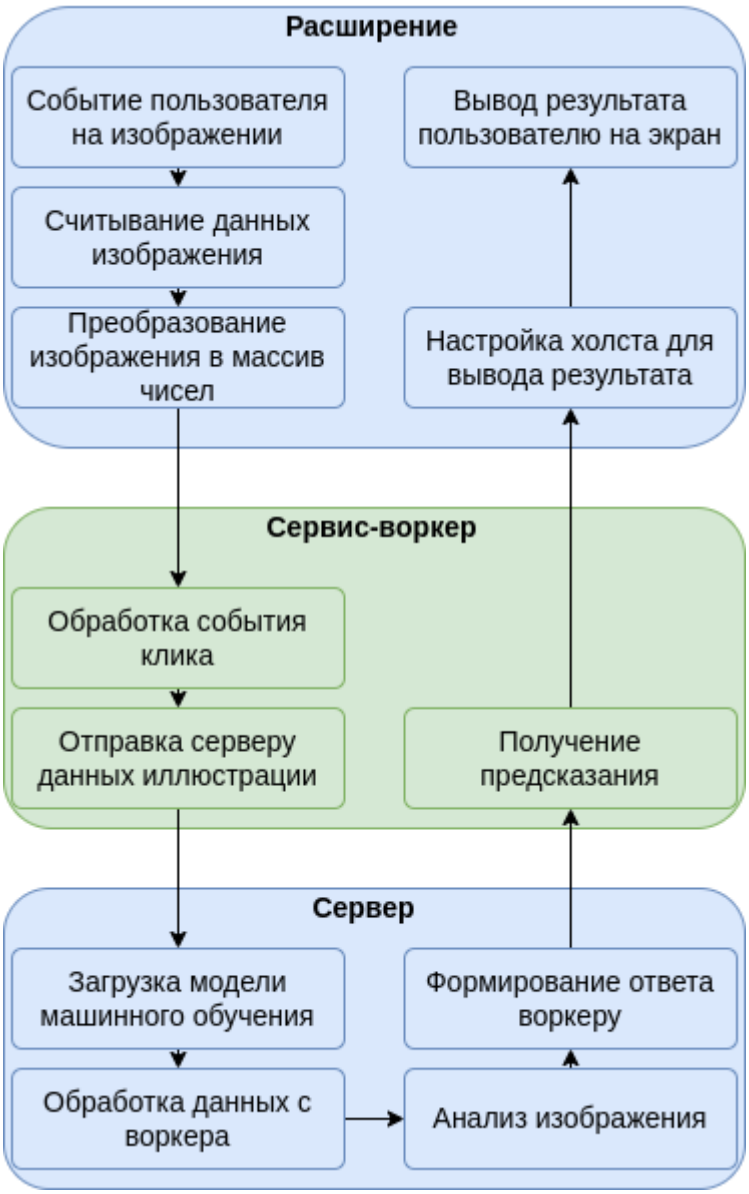

Рисунок 8 - Архитектура расширения

Сервер реализованной системы может находится на удаленных узлах, поэтому при таком подходе возможно применение туманных технологий для уменьшения задержек и увеличения производительности при вычислениях. Ниже каждый модуль будет рассмотрен более подробно.

3.3.1 Расширение Chrome

Для прослушивания событий пользователя в браузере, а также вывода конечного результата работы программы на экран было разработано расширение к браузеру Chrome. Оно реализовано двумя файлами:

- Файл манифеста — описывает структуру расширения и зависимости, необходимые для загрузки и запуска программы в браузере. Имеет формат JSON.
- Скрипт на JavaScript — в нём прописана логика модуля.

В манифесте было указано подключение к сервис-воркеру в фоновом режиме. Это позволяет функционировать расширению сразу после установки его в браузер и не совершать дополнительных манипуляций.

Благодаря тому, что сервис-вокер создаёт пункт контекстного меню для взаимодействия с системой, скрипт в расширении может прослушивать клики по изображениям на веб-странице. Когда пользователь выбирает изображение на сайте и способ взаимодействия с ним, обратный вызов сообщает расширению идентификатор текущей вкладки в браузере и URL-адрес изображения, по которому щелкнули правой кнопкой мыши. Скрипт начинает обрабатывать событие и считывает данные иллюстрации, а затем конвертирует их в численный массив для передачи сервис-воркеру. После этого расширение переходит в режим ожидания результата анализа переданного изображения.

Результатом приходит объект с двумя полями:

- index — индекс класса, к которому нейронная сеть отнесла изображение;
- percent — процент, с которым анализируемое изображение относится к данному классу.

Эти данные передаются функции, ответственной за вывод результата на экран. В ней создаётся HTML [15] контейнер div в котором оформляется результат анализа. Для вывода были установлены следующие стили:

- container.style.position = 'relative';
- container.style.textAlign = 'center';
- container.style.color = 'white';
- text.style.position = 'absolute';
- text.style.top = '50%';

- text.style.left = '50%';
- text.style.transform = 'translate(-50%, -50%)';
- text.style.fontSize = '34px';
- text.style.fontFamily = 'Google Sans,sans-serif';
- text.style.fontWeight = '700';
- text.style.color = 'white';
- text.style.lineHeight = '1em';
- text.style['-webkit-text-fill-color'] = 'white';
- text.style['-webkit-text-stroke-width'] = '1px';
- text.style['-webkit-text-stroke-color'] = 'black'.

Данный контейнер с помощью скрипта встраивается в DOM-дерево веб-страницы поверх изображения, которое проходило анализ. Таким образом пользователь получает вывод о наличии криминального содержания на иллюстрации.

3.3.2 Сервис-воркер

Связующим звеном между клиентом и сервером выступает модуль сервис-воркера. По сути он действует как прокси-сервер. Модуль представляет из себя JavaScript файл, запускаемый в фоновом режиме окна браузера.

При инициализации расширения в браузере, сервис-воркер добавляет в контекстное меню правого клика мыши пункт анализа изображения. Затем модуль переходит в режим ожидания до тех пор, пока не получит массив данных изображения от расширения.

Как только сервис-воркер получает ожидаемые данные, их конвертируют в формат Uint8ClampedArray - массив, в котором каждый элемент представляет собой 8-битное целое число без знака. Полученный результат передаётся POST запросом на сервер для дальнейшей обработки и анализа, а модуль переходит в режим ожидания результатов.

При получения предсказания от сервера, сервис-воркер проверяет его целостность и отправляет данные расширению на дальнейший вывод пользователю.

3.3.3 Сервер

Сервер выступает хранилищем моделей машинного обучения в формате JSON и сопутствующих им артефактов бинарного типа. Модуль основан на Express.js - веб-фреймворке для Node.js, который позволяет создавать серверные приложения и API с минимальным количеством кода. Он обеспечивает простой интерфейс для создания маршрутов HTTP, подходит для раздачи статических файлов и прост в кофигурации.

На сервере храниться директория models, в которой содержаться папки различных моделей машинного обучения импортированные в JSON формат для использования в браузере. Эти данные находятся в открытом доступе и для скачивания требуется обратиться по необходимому маршруту к нужному файлу.

Для анализа изображения на сервере средствами машинного обучения был реализован класс ImageClassifier, имеющий два асинхронных метода:

- loadModel() - функция для загрузки требуемой модели, принимающая аргументом путь к файлу JSON на сервере.
- analyzeImage() - функция классификации изображения. Принимает тензор данных изображения и передаёт его на предсказание загруженной модели нейронной сети. Результатом классификации является массив из пяти чисел, потому что в данной работе модель обучена различать пять видов изображений. Значения отражают уверенность модели в том, насколько иллюстрация относится к каждой группе. Среди полученных значений выбирается наивысшее и передаётся обратно в модуль воркера вместе с индексом класса.

Полученные с воркера данные, перед тем как пройти анализ на криминальное содержание, проходят процедуру сжатия и методами из библиотеки

tensorflow.js конвертируются в тензор. Это позволяет передать информацию о каждом пикселе изображения на исследование нейронной сети.

3.3.4 Интерфейс расширения

Интерфейс расширения - это то, как пользователь может взаимодействовать с продуктом в браузере. На рисунке 9 представлена карточка программной системы в библиотеке расширений Chrome.

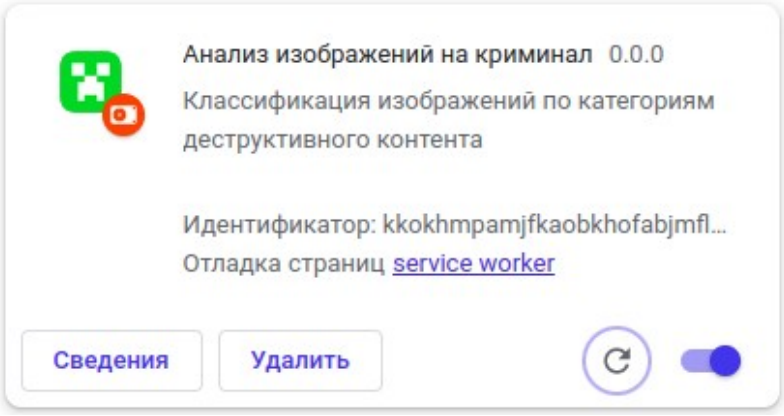

Рисунок 9 - Карточка расширения в Chrome

Чтобы начать анализировать изображения на сайте, нужно выбрать любую иллюстрацию и нажать правой кнопкой мыши по ней. Пользователю откроется контекстное меню, в котором необходимо выбрать пункт «Анализировать изображение». Меню представлено на рисунке 10.

Рисунок 10 - Контекстное меню

На рисунках 11-12 показаны результаты работы расширения.

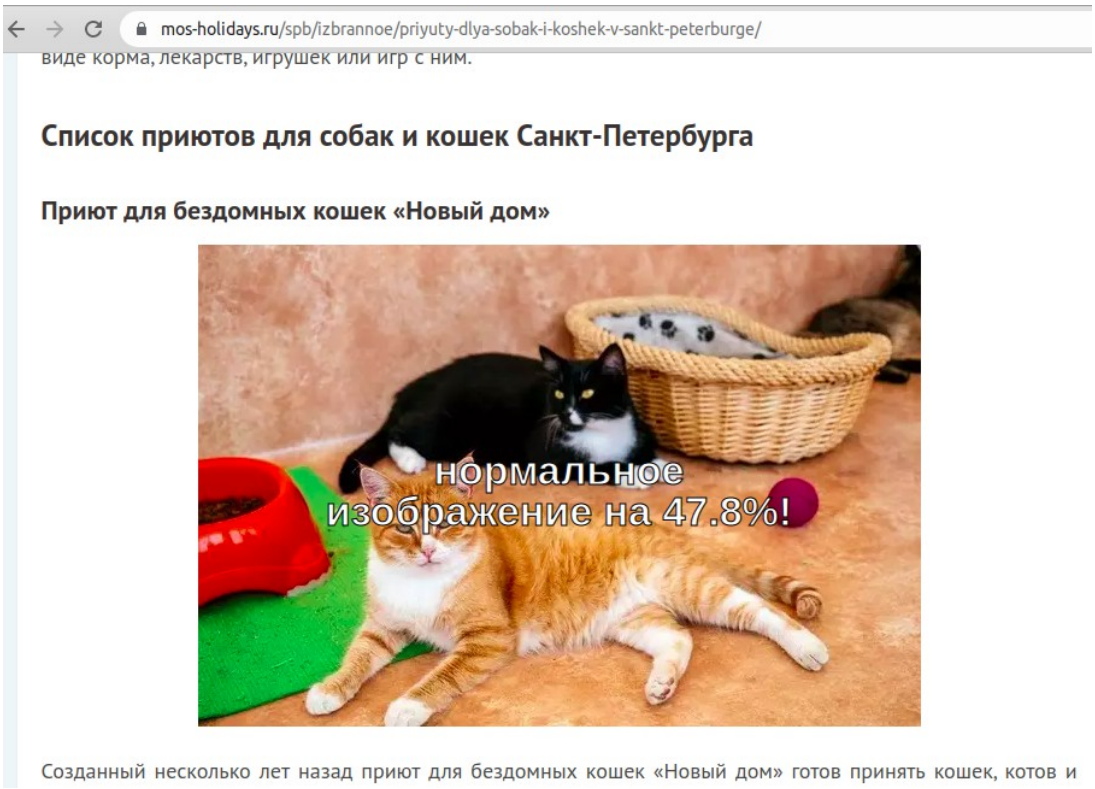

Рисунок 11 - Результат классификации не деструктивного изображения

Рисунок 12 - Результат классификации деструктивной иллюстрации

Если изображение для анализа слишком маленькое, приложение выдаёт предупреждение, как показано на рисунке 13.

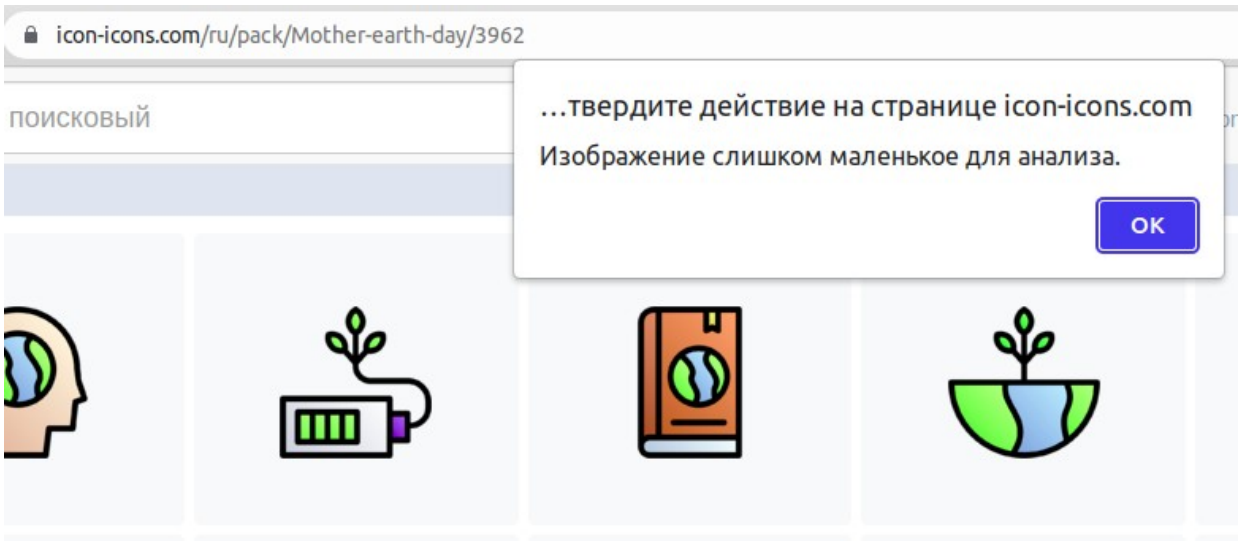

Рисунок 13 - Предупреждение о размере изображения

Если изображение для анализа по каким-то причинам не может быть загружено, расширение также об этом сообщает, как показано на рисунке 14.

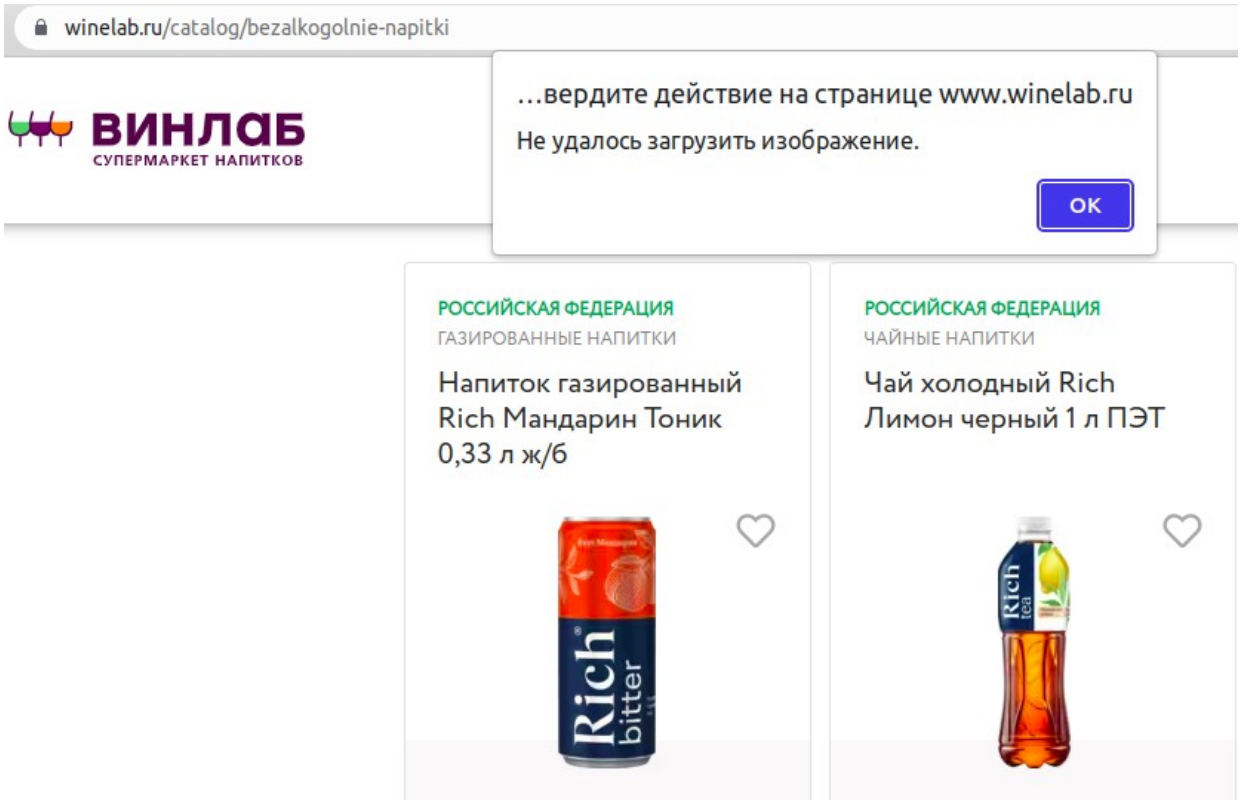

Рисунок 14 - Ошибка загрузки изображения

3.4 Выводы

В данном разделе была представлена архитектура программной системы, разрабатываемой для анализа веб-ресурсов на криминальный контент с использованием методов машинного обучения. Разработанная архитектура включает в себя расширение для браузера, сервис-воркер и сервер.

Расширение, работая внутри браузера, предоставляет возможность анализировать загруженные веб-страницы на наличие криминального контента. Оно использует предобученные модели машинного обучения для анализа изображений. Расширение также обменивается данными с сервис-вокером и отправляет запросы на анализ иллюстраций.

Сервис-вокер является промежуточным звеном между расширением и сервером. Он предоставляет API для обработки запросов на анализ управляется событиями расширения. Сервис-вокер также взаимодействует с сервером для поставки данных на анализ и получения результатов мониторинга.

Сервер хранит модели машинного обучения и производит классификацию изображений по категориям средствами нейронных сетей. Также этот модуль обрабатывает иллюстрации, конвертируя их в необходимый для вычислений формат.

Такая архитектура позволяет эффективно анализировать веб-ресурсы на криминальный контент, используя мощности машинного обучения. Также она позволяет использовать облачные и туманный вычисления для повышения производительности продукта. Архитектура обеспечивает быстрый и точный анализ изображений и обновление моделей без необходимости обновления самого расширения.

4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА

4.1 Нагрузочное тестирование

В данном разделе была исследована производительность обработки видео на различных вычислительных мощностях.

Используемые вычислительные мощности:

- Acer Aspire E5-573, ОЗУ 8 Гб, CPU 4;
- Облако на платформе Yandex, ОЗУ 20 Гб, CPU 20;
- Облако на платформе Yandex, ОЗУ 12 Гб, CPU 14;
- Платформа Google Colab ОЗУ 12.7, GPU T4;

В данном исследовании производится измерение времени отклика сервера, на котором производятся основные вычисления системы. В первом случае модуль запускается на локальном компьютере, а во втором - на дистанционном сервере. Для анализа производительности в воркере был использован метод performance.now(), который возвращает временную метку, измеряемую в миллисекундах. Результаты измерений для модели представлены в таблице 2.

Таблица 2 – Среднее время ответа, модель EfficientNet

Компьютер	Обработка изображения (мс)	Подключение к модели (мс)	Классификация изображения (мс)
Acer Aspire E5-573	18 500	32	18 700
Облако Yandex ОЗУ 12	10 000	18	11 300
Облако Yandex ОЗУ 20	800	5	6 500
Google Colab	600	5	1300

Поскольку модель имеет большую глубину и содержит более 50 миллионов параметров, было ожидаемым получить большую задержку вычислений на фоне высокой точности модели. По результатам видно, что использование

облачных технологий заметно увеличивает производительность системы по сравнению с личным компьютером.

Нераскрытый до конца потенциал вычислительных мощностей облачных технологий был выявлен с помощью платформы Google Colab, в которой была запущена среда выполнения с ОЗУ 12.7 ГБ и GPU T4. Предобработка данных и предсказание категории контента моделью EfficientNet составила примерно 2 секунды, что в 1.8 раза быстрее облака на 20 ядрах процессора и в 10 раз быстрее персонального компьютера. Это доказывает, насколько использование облачных технологий с высокими характеристиками оптимизирует работу системы.

Также в рамках тестирования были подтверждены границы в 90-92% точности, полученные после обучения модели EfficientNet на собранном датасете криминального содержания.

4.2 Выводы

В результате нагрузочного тестирования были определены возможности работы системы в нескольких конфигурациях с разными вычислительными мощностями. Было продемонстрировано ускорение обработки данных и анализа иллюстраций за счёт использования облачных технологий. С самой дорогой конфигурацией вычислительных мощностей можно уменьшить задержки в 10 раз по сравнению с персональным компьютером.

Рисунок 1 – Пример рисунка

Каждая таблица должна иметь ссылку в тексте (таблица 1).

Таблица 1 – Пример таблицы

Показатель	Единица измерения	Значение
Время работы	сек.	159
Нагрузка на ЦПУ	процент	95
Стоимость работ	тыс. руб.	100

5 БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

5.1 Госты

5.2 Выводы

ЗАКЛЮЧЕНИЕ

В заключение данной работы, было представлено разработанное расширение для браузера, которое осуществляет анализ изображений на наличие негативного контента с использованием методов машинного обучения. Реализованный инструмент имеет возможность размещения модуля обработки информации в облаке и тумане, что значительно сокращает задержки в работе продукта.

Для достижения поставленной цели были выполнены все необходимые задачи:

1. Изучение методов и технологий машинного обучения, необходимых для анализа изображений.
2. Разработка архитектуры расширения для браузера, которое будет обращаться к туманным сервисам для анализа изображений на криминальный контент.
3. Реализация механизмов, обеспечивающих сбор и анализ данных о посещаемых веб-ресурсах.
4. Интеграция средств машинного обучения для обработки и анализа данных на наличие криминального контента.
5. Тестирование и оптимизация работы расширения для браузера, с целью повышения скорости работы продукта.

В рамках сравнительного анализа была выбрана нейронная сеть, обладающая наибольшей точностью предсказаний. Такой выбор обусловлен тем, что эта работа направлена на анализ веб-ресурсов на криминальный контент и показатель точности модели является главным приоритетом.

В процессе работы была разработана архитектура программной системы, включающая расширение для браузера, сервис-воркер и сервер. Расширение осуществляет считывание изображений, сервис-воркер управляет запросами между клиентом и сервером, а сам сервер обрабатывает и анализирует иллюстрации.

Тестирование показало, что разработанное расширение обладает высокой точностью и эффективностью в обнаружении негативного контента на изображениях. Оно может служить полезным инструментом для пользователей, помогая им избегать ресурсов, которые могут быть вредным или нежелательным.

Существует несколько способов дальнейшего развития разработанного расширения для браузера. Некоторые из них включают:

1. Расширение функциональности: можно добавить дополнительные возможности и функции в расширение. Будет полезным внедрить возможности анализа текстового контента и мониторинг нескольких изображений за единицу времени. Также можно предоставить пользователю возможность настройки параметров мониторинга и выбора категорий контента для анализа.
2. Улучшение точности и производительности: работа над оптимизацией моделей машинного обучения и алгоритмов анализа может помочь улучшить точность обнаружения негативного контента и снизить ложные срабатывания. Также можно работать над оптимизацией производительности для ускорения процесса анализа изображений и улучшения отзывчивости расширения.
3. Расширение поддерживаемых платформ и браузеров: расширение может быть адаптировано и расширено для поддержки разных браузеров и платформ.
4. Улучшение пользовательского интерфейса: оптимизация и улучшение пользовательского интерфейса расширения поможет сделать его более интуитивно понятным и удобным в использовании. Это может включать переработку макета, добавление визуальных индикаторов.
5. Обновление моделей: регулярное обновление моделей машинного обучения и базы данных негативного контента поможет улучшить работу расширения и его способность обнаруживать новые типы негативного содержания.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Обзор Digital 2022 [Электронный ресурс] URL: https://www.web-canape.ru/business/statistika-interneta-i-socsetej-na-2022-god-cifry-i-trendy-v-mire-i-v-rossii/ (дата обращения: 23.12.2022)
2. K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition // conference paper at ICLR 2015 С. 2-3.
3. Докуметация InceptionV3 [Электронный ресурс] URL: https:// docs.exponenta.ru/deeplearning/ref/inceptionv3.html (дата обращения: 08.05.2023)
4. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L. Chen, MobileNetV2: Inverted Residuals and Linear Bottlenecks // 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) С. 4-7.
5. ResNet (34, 50, 101): «остаточные» CNN для классификации изображений [Электронный ресурс] URL: https://neurohive.io/ru/vidy-nejrosetej/ resnet-34-50-101/ (дата обращения: 08.05.2023)
6. M. Tan, Quoc V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks // Proceedings of the 36th International Conference on Machine Learning, ICML 2019 С. 16-24.
7. Jonathan Bar-Magen Numhauser. Fog Computing- Introduction to a new Cloud evolution // Proceedings from the CIES III Congress, January 2012 С. 3-4.
8. JavaScript документация [Электронный ресурс] URL: https:// www.javascript.com/ (дата обращения: 09.05.2023)
9. Python документация [Электронный ресурс] URL: https:// www.python.org/ (дата обращения: 09.05.2023)
10. Туманные вычисления (fog computing) [Электронный ресурс] URL: https://www.techtarget.com/iotagenda/definition/fog-computing-fogging (дата обращения: 09.05.2023)

11. Tensorflow документация [Электронный ресурс] URL: https://www.tensorflow.org/js/tutorials/conversion/import_keras?hl=ru (дата обращения: 09.05.2023)
12. Фреймворк Express.js [Электронный ресурс] URL: https://expressjs.com/ru/ (дата обращения: 09.05.2023)
13. Среда выполнения Node.js [Электронный ресурс] URL: https://nodejs.org/en/about (дата обращения: 09.05.2023)
14. Keras документация [Электронный ресурс] URL: https://keras.io/ (дата обращения: 09.05.2023)
15. HTML документация [Электронный ресурс] URL: https://www.w3schools.com/html/ (дата обращения: 09.05.2023)
16. DOM-дерево [Электронный ресурс] URL: https://learn.javascript.ru/dom-nodes (дата обращения: 09.05.2023)
17. Трансферное обучение [Электронный ресурс] URL: https://www.tensorflow.org/js/tutorials/transfer/what_is_transfer_learning?hl=ru (дата обращения: 09.05.2023)
18. Проект автоматизации браузера [Электронный ресурс] URL: Selenium https://www.selenium.dev/documentation/ (дата обращения: 09.05.2023)

ПРИЛОЖЕНИЕ А