МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
Доцент кафедры экологии и
химических технологий
ФГАОУ ВО «ЮУрГУ», к.г.н.

_____ О.Ю. Ленская

«___»_____ 2024 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка нейросетевой модели для обнаружения дефектов видеотрансляции

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.04.04.2024.308-1479.ВКР

Научный руководитель,
профессор кафедры СП,
д.г.н., к.ф.-м.н.
_____ С.М. Абдуллаев

Автор работы,
студент группы КЭ-229
_____ Ю.В. Петраш

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы магистранта
студенту группы КЭ-229
Петрашу Юрию Валерьевичу,
обучающемуся по направлению
09.04.04 «Программная инженерия»
(магистерская программа «Искусственный интеллект и инженерия данных»)

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)
Разработка нейросетевой модели для обнаружения дефектов видеотрансляции.
2. **Срок сдачи студентом законченной работы:** 20.05.2024 г.
3. **Исходные данные к работе**

3.1. Poynton C. Digital Video and HDTV. // Morgan Kaufmann Publishers, 2002. – 700 с.

3.2. Коул А. Искусственный интеллект и компьютерное зрение / А. Коул, С. Ганджу, М. Казам. // Санкт-Петербург: Питер, 2023. – 608 с.

3.3. Goodfellow I., Bengio Y., Courville A. Deep Learning (Adaptive Computation and Machine Learning). – MIT Press, 2016. – 800 p.

3.4. Wang F., Beladev M., Kleinfeld O., Frayerman E., Shachar T., Fainman E., Assaraf K., Mizrachi S. Text2Topic: Multi-Label Text Classification System for Efficient Topic Detection in User Generated Content with Zero-Shot Capabilities

[Электронный ресурс] // arXiv.org, 2024. Дата обновления: 23.10.2023 г. URL: https://arxiv.org/abs/2310.14817 (дата обращения: 10.02.2024 г.).

4. Перечень подлежащих разработке вопросов

4.1. Провести анализ предметной области.

4.2. Изучение технологий цифрового видеовещания.

4.3. Собрать набор данных.

4.4. Выбор обучение нейросетевой модели.

4.5. Реализовать метод вызова нейронной сети для анализа видеофайла.

5. Дата выдачи задания: 29.01.2024 г.

Научный руководитель,
профессор кафедры СП, д.г.н., к.ф.-м.н. С.М. Абдуллаев

Задание принял к исполнению Ю.В. Петраш

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

Современные источники информации вытеснили телевидение из графика повседневной жизни. Сейчас вместо аналогового телевидения преимущество используют цифровое. Протоколы передачи аудио и видеоданных в цифровых пакетах продолжают активно использоваться не только в цифровом телевидении, но и в других сферах.

Цифровое телевидение (включая более качественное HDTV) было введено в большинстве развитых стран в начале 2000-х годов. Цифровое видео используется в современных мобильных телефонах и системах видеоконференцсвязи. На рисунке 1 показана схема использования цифрового телевидения в домашней сети. Цифровое видео используется для Интернет-распространения мультимедиа, включая потоковое видео и одноранговое распространение фильмов.

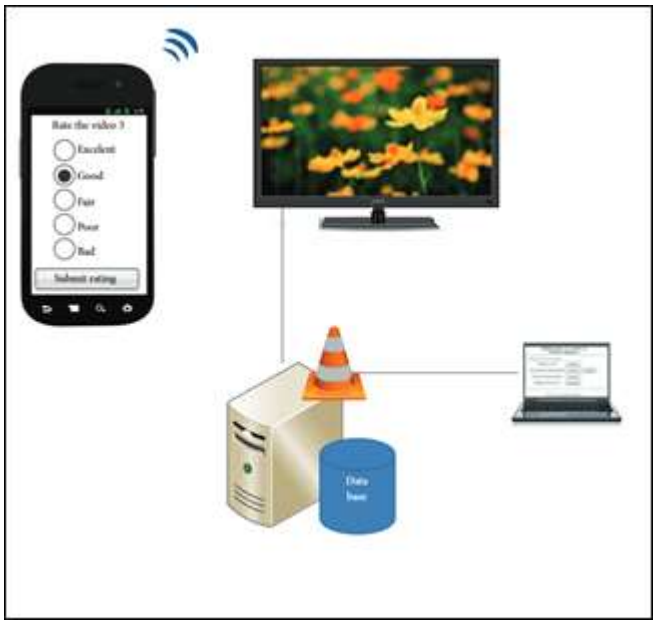

Рисунок 1 – Схема сети цифрового телевидения через домашнюю сеть

При передаче медиаконтейнера по транспортному потоку могут произойти ошибки, смещения пакетов, которые несут в себе медиаданные. Стоит отметить, что решение задачи детектирования ошибок видеопотока

актуально не только для пользователей телевидения, но и для других платформ. Современные видеохостинги также могут сталкиваться с этой проблемой. Система, способная правильно детектировать ошибки кадра, может позволить провайдерам видеосигнала получить данные об ошибках, которые могут быть полезны для последующего анализа причин их возникновения. На рисунке 2 показан график отношения качества телевещательной картинки к количеству поврежденных пакетов, опубликованный в международном журнале цифрового мультимедийного вещания.

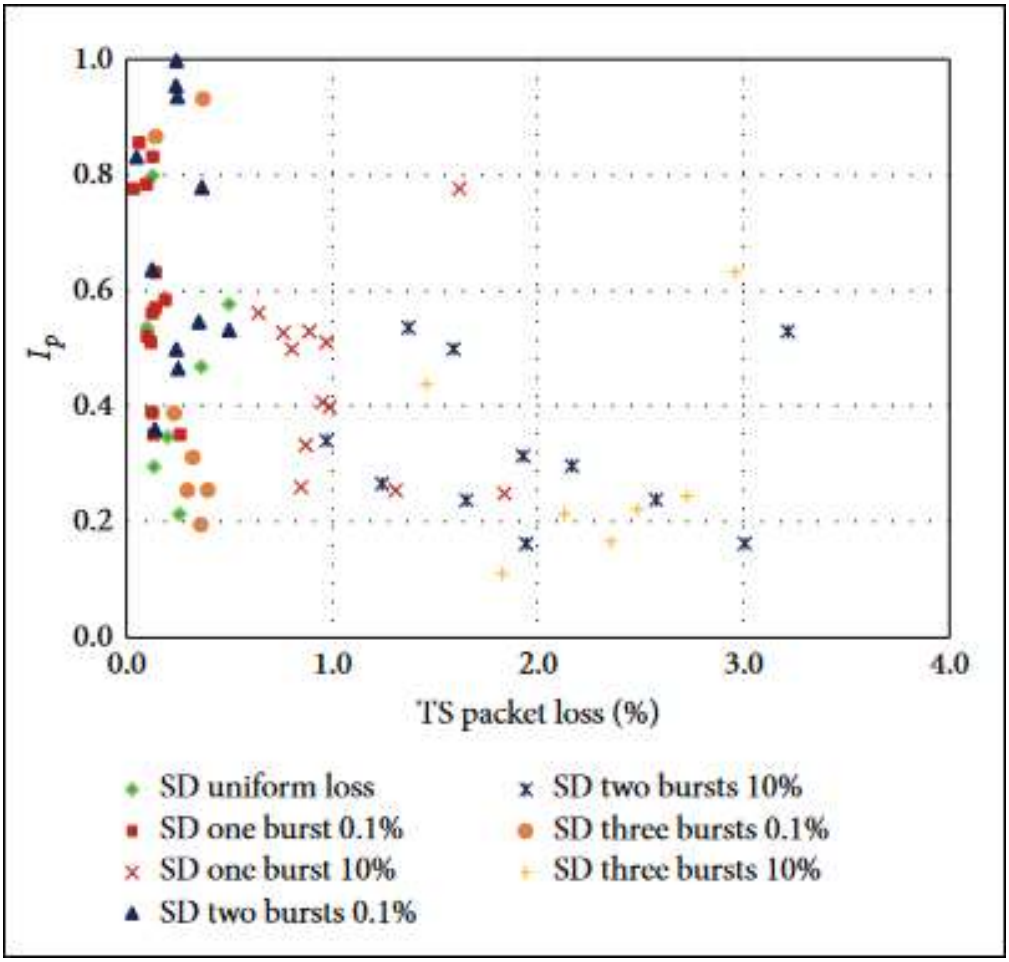

Рисунок 2 – График отношения коэффициента качества картинки к количеству поврежденных пакетов

Постановка задачи

Целью выпускной квалификационной работы является разработка нейросетевой модели для обнаружения дефектов видеотрансляции. Для достижения поставленной цели необходимо решить следующие задачи:

1) провести анализ предметной области;

2) изучение технологий цифрового видеовещания;

3) собрать набор данных;

4) выбор нейросетевой модели для обучения;

5) получить результаты обучения нейросетевой модели;

6) реализовать метод вызова нейронной сети для анализа видеофайла.

Структура и содержание работы

Работа состоит из введения, пяти глав, заключения и списка литературы и приложения. Объем работы составляет 53 страницы, объем списка литературы – 32 источника.

В первой главе описывается обзор на существующие научные работы по анализу видео на обнаружение дефектов.

Вторая глава посвящена технологиям цифрового вещания.

В третьей главе описывается о создании датасета, распределению данных по выборкам

В четвертой главе описывается обучения нейросетевых моделей, получение результатов обучения.

Пятая глава посвящена реализации метода использования обученной нейронной сетевой модели

В приложениях содержится дополнительная графическая информация и листинг кода для вызова нейронной сети.

1. ОБЗОР НА СУЩЕСТВУЮЩИЕ НАУЧНЫЕ РАБОТЫ ПО АНАЛИЗУ ТЕЛЕВЕЩАНИЯ

1.1. Описание предметной области

Цифровое видео – это электронное представление движущихся визуальных изображений в виде закодированных цифровых данных.

Цифровое телевизионное вещание (ЦТВ) имеет ряд преимуществ перед устаревшим аналоговым. Основное преимущество заключается в помехоустойчивости цифровых сигналов. Кроме того, цифровая обработка сигналов значительно увеличивает возможности воздействия на обрабатываемый сигнал, получая преобразования, которые тяжело получить аналоговым способом. Базовые требования к системам цифрового телевидения были разработаны группой DVB Project и были отражены в стандартах: DVB-S, DVB-C, DVB-T. На рисунке 3 приведена структурная схема цифровой передачи сигнала конечному пользователю. Цифровая форма любого сигнала всегда является лишь промежуточной его формой, предназначенной для улучшения передачи и обработки этого сигнала. На конечном участке линии связи сигнал должен быть опять представлен в аналоговой форме. Цифровой поток подается на декодер, который уже преобразует данные с пакетов в формат возможный для показа на используемом устройстве. Сегодня декодером может быть не только аппаратное устройство, но и программа. Например, VLC Media Player имеет функции декодера цифрового видеопотока и возможность его показа на ПК.

Цифровой поток подается на декодер, который уже преобразует данные с пакетов в формат возможный для показа на используемом устройстве. Сегодня декодером может быть не только аппаратное устройство, но и программа. Например: VLC Media Player имеет функции декодера цифрового видеопотока и возможность его показа на ПК.

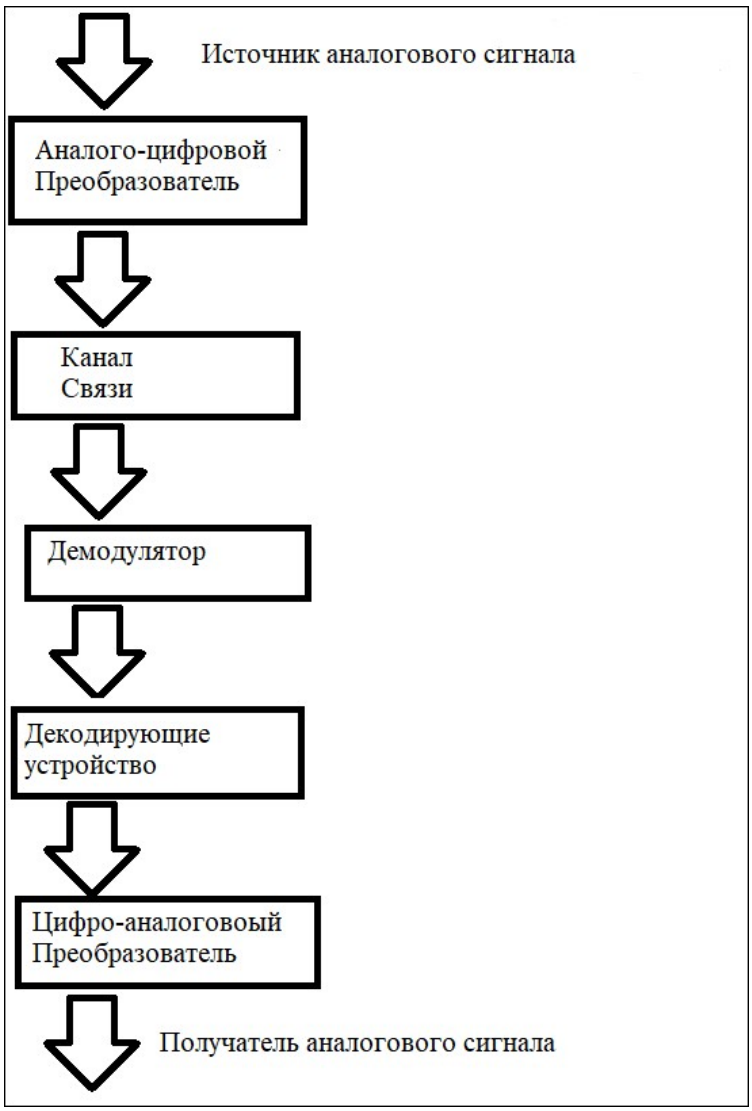

Рисунок 3 – Структурная схема цифровой передачи сигнала

Телевидение по Интернет-протоколу (IPTV) – это доставка телевизионного контента по сетям Интернет-протокола (IP). Это контрастирует с доставкой через традиционные форматы наземного, спутникового и кабельного телевидения. В отличие от загруженного мультимедиа, IPTV предлагает возможность непрерывной потоковой передачи исходного мультимедиа. В результате, клиентский медиаплеер может начать воспроизведение контента (например, телеканала) практически сразу. Это известно, как потоковое мультимедиа. На рисунке 4 показана схема использования IPTV в домашней сети.

Рисунок 4 – Упрощенная схема сети IPTV

Хотя IPTV использует интернет-протокол, оно не ограничивается потоковым телевидением из Интернета (интернет-телевидением). IPTV широко распространено в абонентских телекоммуникационных сетях с высокоскоростными каналами доступа в помещения конечного пользователя через приставки или другое оборудование, устанавливаемое в помещениях клиента. IPTV также используется для доставки мультимедиа по корпоративным и частным сетям. IPTV на телекоммуникационной арене отличается продолжающимся процессом стандартизации.

Прямая трансляция – еще один важный аспект интернет-видео. Это когда определенные события транслируются с использованием онлайн-видео в реальном времени. На данный момент, все большое количество людей пользуются этой услугой через такие площадки как: YouTube, GoodGame, Twitch, Amazon, AppleTV и т.д.

Видео по запросу (VOD) – это система распространения мультимедиа, которая позволяет пользователям получать доступ к видео, телешоу и фильмам без традиционного устройства воспроизведения видео и типичного статического расписания вещания. В XX веке эфирное вещание было наиболее распространенной формой распространения средств массовой информации. По мере развития технологий Интернета и IPTV в 1990-х годах потребители начали тяготеть к нетрадиционным способам потребления контента, кульминацией чего стало появление VOD на телевизорах и персональных компьютерах. Телевизионные системы VOD могут передавать контент либо через традиционную телеприставку, либо через удаленные устройства, такие как компьютеры, планшеты и смартфоны. Пользователи VOD могут навсегда загружать контент на такое устройство, как компьютер, цифровой видеомагнитофон (DVR) или портативный медиаплеер, для дальнейшего просмотра. Большинство провайдеров кабельного и телефонного телевидения предлагают потоковое видео по запросу, при котором пользователь выбирает видеопрограмму, которая начинает воспроизводиться немедленно, или загружается на видеорегистратор, арендованный или приобретенный у провайдера, или на ПК или портативное устройство для просмотра. отложенный просмотр. Потоковое мультимедиа становится все более популярным средством предоставления VOD. Клиентские приложения для настольных компьютеров, такие как интернет-магазин контента Apple iTunes, и приложения Smart TV, такие как Amazon Prime Video, позволяют временно брать напрокат и покупать развлекательный видеоконтент. Другие интернет-системы VOD предоставляют пользователям доступ к пакетам развлекательного видеоконтента, а не к отдельным фильмам и шоу. Наиболее распространенные из этих систем – Netflix, Hulu, Disney+, Peacock, Max и Paramount+ – используют модель, которая требует от пользователей ежемесячной платы за доступ к подборке фильмов, телешоу и оригинальных сериалов.

1.2. Обзор научной литературы

Первые научные работы по детектированию артефактов телевизионной картинки появились относительно недавно. В исследовательской работе [2] приведены системы измерений искажений и обнаружений искаженного кадра в реальном времени. На рисунке 5 показана структурная схема прохождения телесигнала от студии к потребителю и типы возникающих ошибок на этих стадиях. В данной работе не используют нейросетевые технологии. Авторы работ используют бинарные алгоритмы для анализа выведенной картинки на основе величин градиентов отдельных блоков изображения.

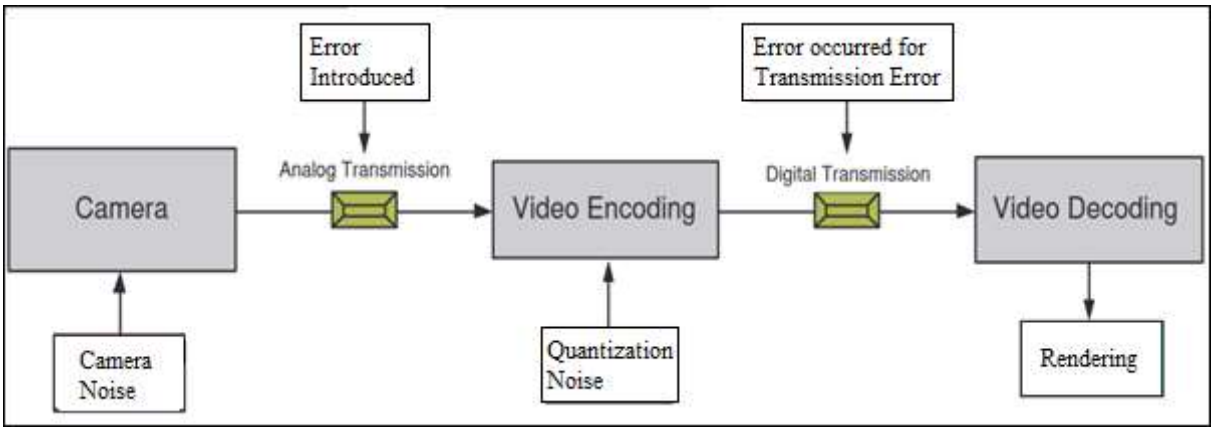

Рисунок 5 – Структурная схема возникновения ошибок в системе телевещания

В приведенной научной работе были использованы алгоритмы инвариативности, в результате чего артефакты на картинке находились независимо от угла поворота изображения на экране или относительно самого артефакта. Планируется возможность повторения такого же умения с использованием нейросетевых технологий.

В статье «No-reference real-time video transmission artifact detection for video signals» [3] предлагается алгоритм обнаружения артефактов видеопередачи на основе пикселей без привязки (NR), называемый алгоритмом измерения площади потери пакетов (PLAM). При обнаружении артефактов передачи видео алгоритм PLAM учитывает пространственную и временную

информацию видеосигнала. Производительность предложенного алгоритма PLAM была сравнена с характеристиками трех существующих различных алгоритмов обнаружения PL на широком наборе существенно отличающихся видеосигналов из двух общедоступных баз данных видео. Результаты показывают, что алгоритм PLAM очень надежен при обнаружении артефактов передачи видео в видеосигналах различного содержания, с различными уровнями деградации и методами сокрытия ошибок PL, используемыми при постобработке декодера. На рисунке 6 показана структурная схема обработки кадра данным алгоритмом. Благодаря своей низкой вычислительной сложности алгоритм PLAM способен обрабатывать видеосигналы Full HD и Ultra HD с частотой кадров до 100 кадров в секунду (fps) соответственно в режиме реального времени, в случае использования высокопроизводительного процессора. На рисунке 7 представлены графики количества артефактов на кадр (нарисован красным цветом) и график количества артефактов на кадр измеренный методом PALM (нарисован зеленым цветом).

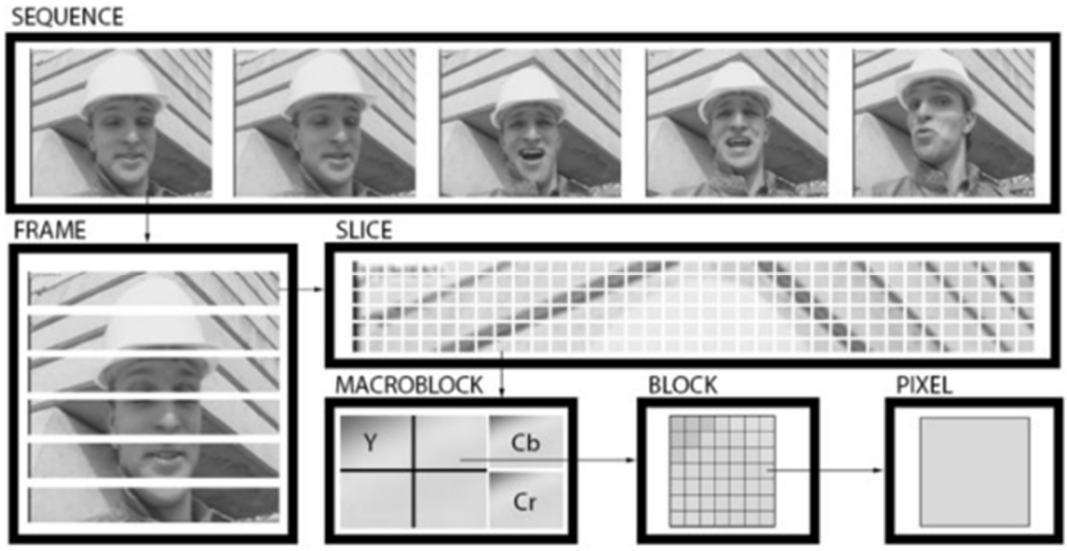

Рисунок 6 – Обработка кадра методом PLAM

В научной работе [4] приведены данные об оценках кадра MAV (mean annoyance values) при воздействии на сигнал различными видами помех. На рисунке 8 приведены графики результатов измерений раздраженности кад-

ров от типа помехи оказываемой на сигнал. Полезность данной статьи заключается в том, что оценка раздраженности кадра происходит без сравнивания его с эталонным сигналом.

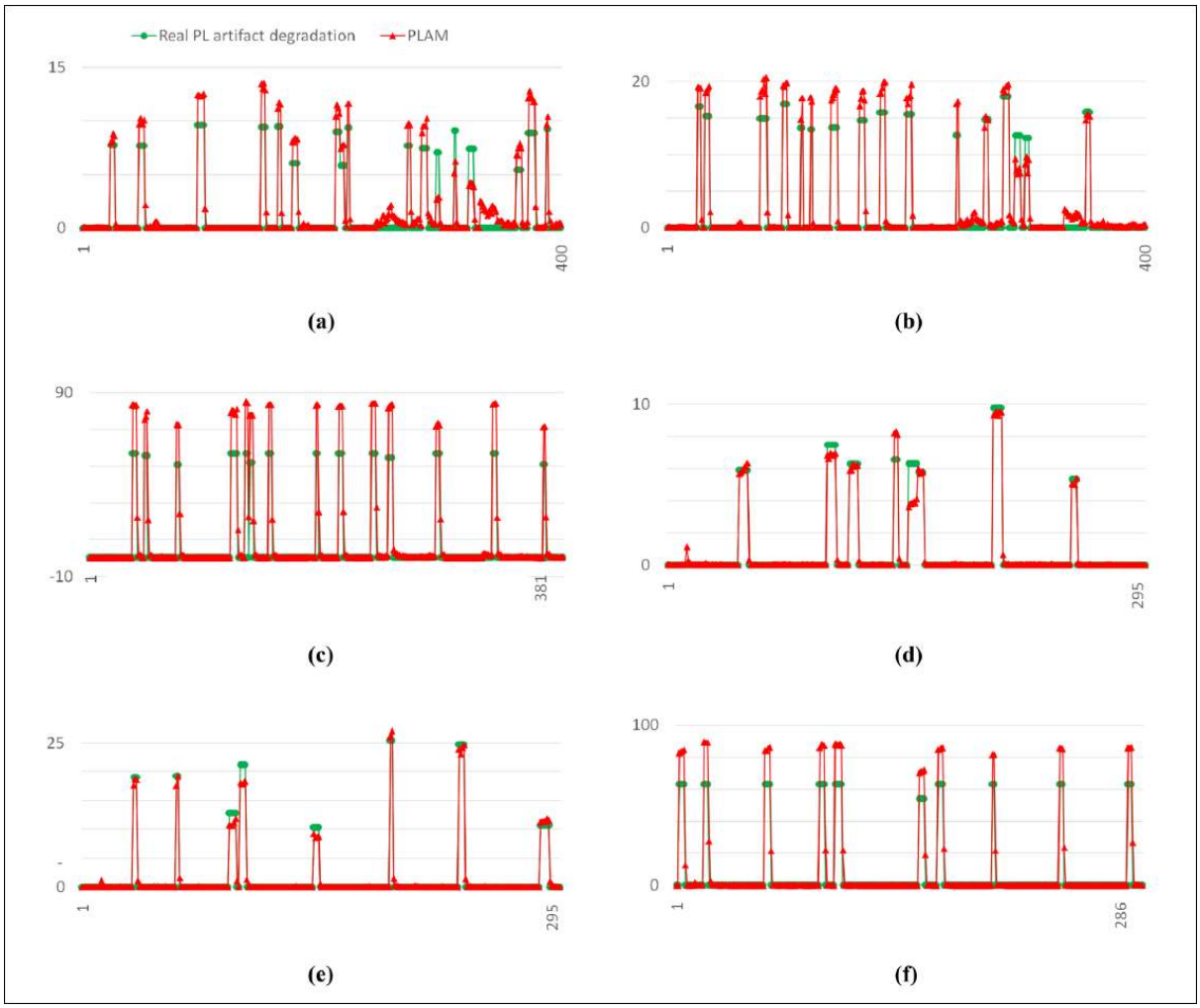

Рисунок 7 – Реальное количество артефактов и количество артефактов, измеренное PLAM при разных методах деградации кадра

В статье [5] приводятся характеристики различных типов артефактов, обычно встречающихся в сжатых видео. Создаются артефактные сигналы, которые преимущественно воспринимаются как блочные, размытые, звенящие и шумные, и комбинируем их в различных пропорциях. Полученные результаты можно использовать для создания датасета для обучения нейронной сети поиска различных типов артефактов кадра. На рисунке 8 представлен основной рисунок и его копии, которые были подвержены различными видами и разной степенью деградации.

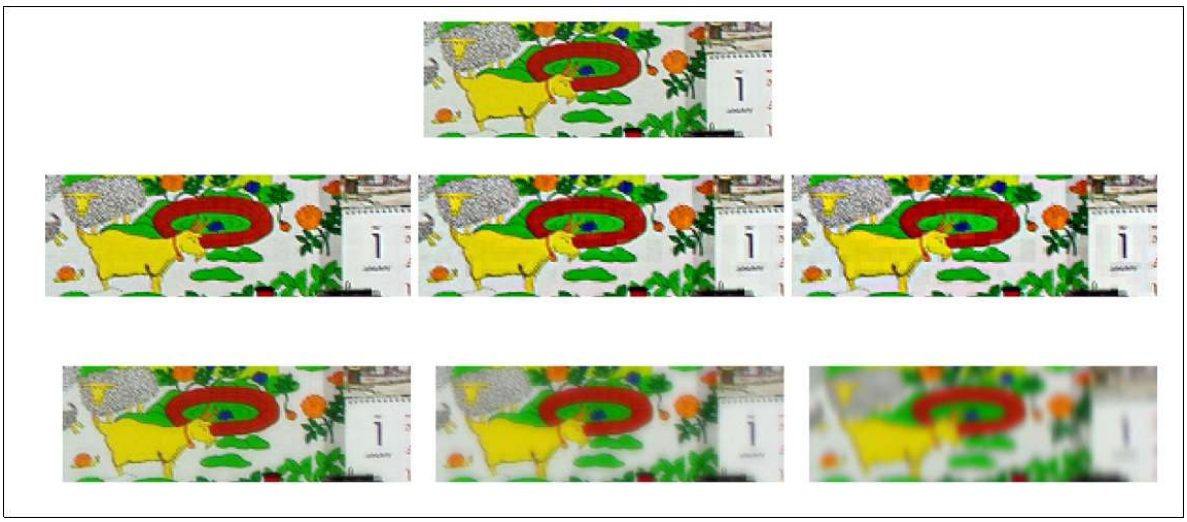

Рисунок 8 – Пример использования различных типов артефактов на одном кадре

В работе «Automatic Soccer Video Analysis and Summarization» [6] предлагается система анализа и обобщения футбольных материалов с использованием кинематографических и объектных признаков. Система выдает сводки по видеозаписи. Преимущество данной системы заключается в том, что нет необходимости вычислять объектно-ориентированные признаки, когда достаточно кинематографичных признаков. Под кинематографическими понимаются признаки, обусловленные общими правилами композиции и производства видео, такими как типы кадров и повторы. Объекты описываются их пространственными, например, цветом, текстурой и формой, а также пространственно-временными признаками, такими как движения и взаимодействия объектов.

На рисунке 9 показан пример пространственной композиции «золотого сечения». Признаки, основанные на объектах, позволяют проводить высокоуровневый анализ домена, но их извлечение может быть вычислительно затратным для реализации в реальном времени. С другой стороны, кинематографические признаки обеспечивают хороший компромисс между вычислительными требованиями и получаемой семантикой.

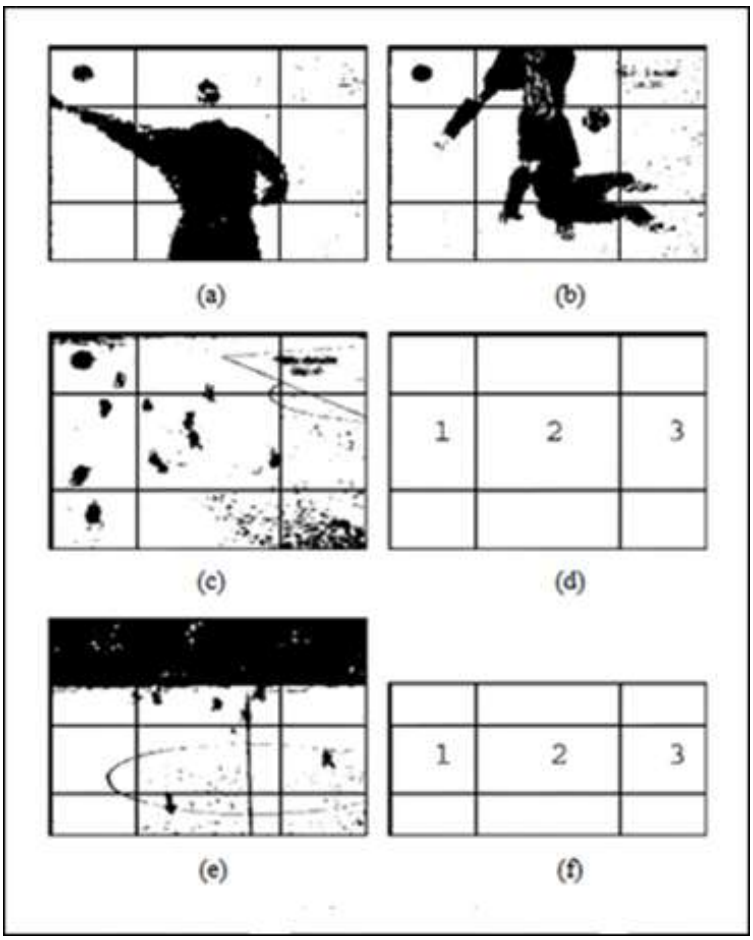

Рисунок 9 – Примеры пространственной композиции «золотого сечения» на видах

В научной работе «Применение нейронной сети YOLO для распознавания дефектов» [7] затрагивается схожая проблема детектирования дефектов на изображении. В ней описывается использование нейронной сети YOLO для обнаружения образов дефектов на изображении, полученном в процессе ультразвукового контроля дифракционно-временным методом. Для достижения данной цели в работе было создано 110 гистограмм, содержащих изображения дефектов, 100 изображений для обучающей выборки и 10 для тестовой выборки. Пример работы нейронной сети в данной работе показан на рисунке 10.

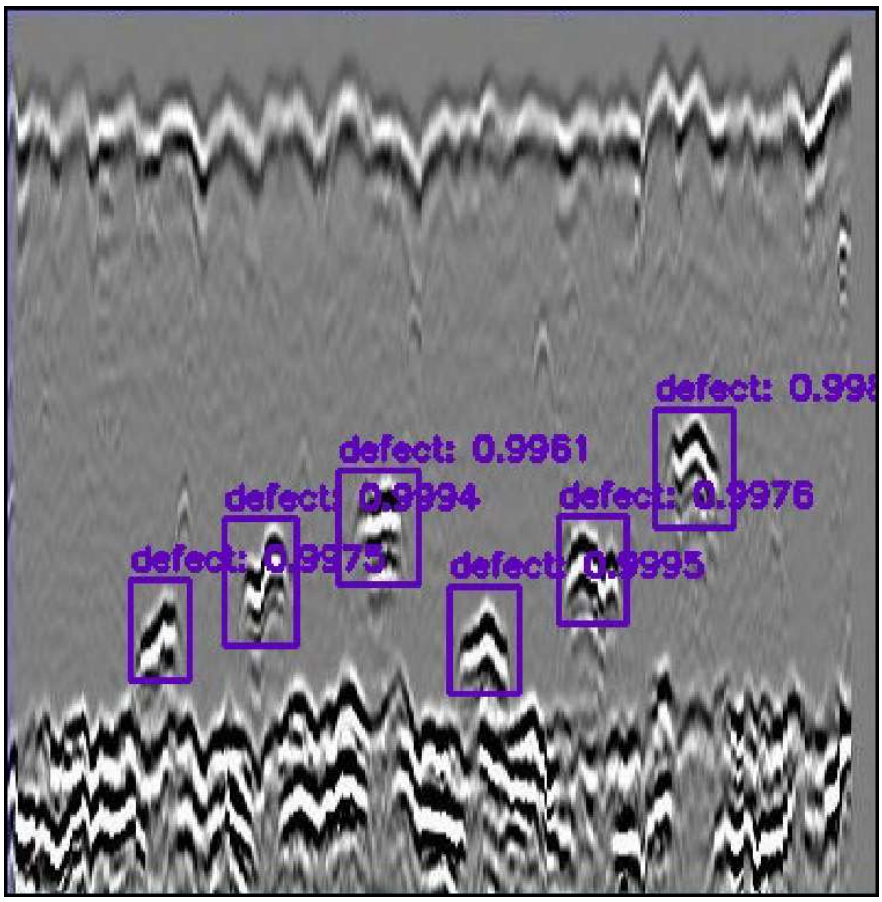

Рисунок 10 – Пример работы нейронной сети YOLO для обнаружения дефектов

Из прочитанной работы было полезно узнать, какую модель нейронной сети можно использовать для решения задачи по обнаружению дефектов на изображениях.

Выводы по первой главе

В ходе анализа предметной области был проведен обзор литературы. Было выявлено, что исследования в области применения нейросетевых технологий для анализа видео крайне актуальны. В данной главе был проведен анализ предметной области и обзор литературы по теме работы. Были рассмотрены уже существующие алгоритмы детектирования артефактов в кадре телевизионного сигнала, системы оценки раздраженности кадра.

2. ТЕХНОЛОГИИ ЦИФРОВОГО ВЕЩАНИЯ

2.1. Стандарт сжатия видео и цифрового потока MPEG

MPEG (Moving Picture Experts Group) является серией стандартов для сжатия и кодирования аудио и видео контента. Практически все современные системы сжатия видеоданных, так или иначе, связаны с MPEG. На его основе функционируют системы цифрового вещания по стандартам DVB и ATSC, системы бытовой и профессиональной видеозаписи DVC, DVCPro, Digital-S, DVD, SX, а также многие другие системы. Схема генеологии стандарта MPEG указана в приложении А.

Поток видеоданных представляет собой иерархическую структуру, элементы которой строятся и объединяются друг с другом в соответствии с определенными синтаксическими и семантическими правилами. Существует 6 типов элементов этой иерархической структуры: Видеопоследовательность, группа изображений, изображение, срез, макроблок, блок.

Один из основных принципов сжатия – это передача только той информации, которая изменяется от кадра к кадру. Изображение типа I кодируется с использованием только той информации, которая содержится в нем самом (I – Intra-coded picture). В нем устраняется только пространственная избыточность. При кодировании P и B изображений используется межкадровое кодирование. При кодировании изображения типа P формируется разность между исходным изображением и предсказанием, полученным на основе предшествующего или последующего изображения типа I (P – Predictive-coded picture). Изображение типа B – это изображение, при кодировании которого используется предсказание, сформированное на основе предшествующего и последующего изображений типа I или P (B – Bidirectionally-predicted-coded picture). В изображениях типа P и B устраняется и пространственная, и временная избыточность. Пример видеопоследовательности с различными типами изображений показан на рисунке 11.

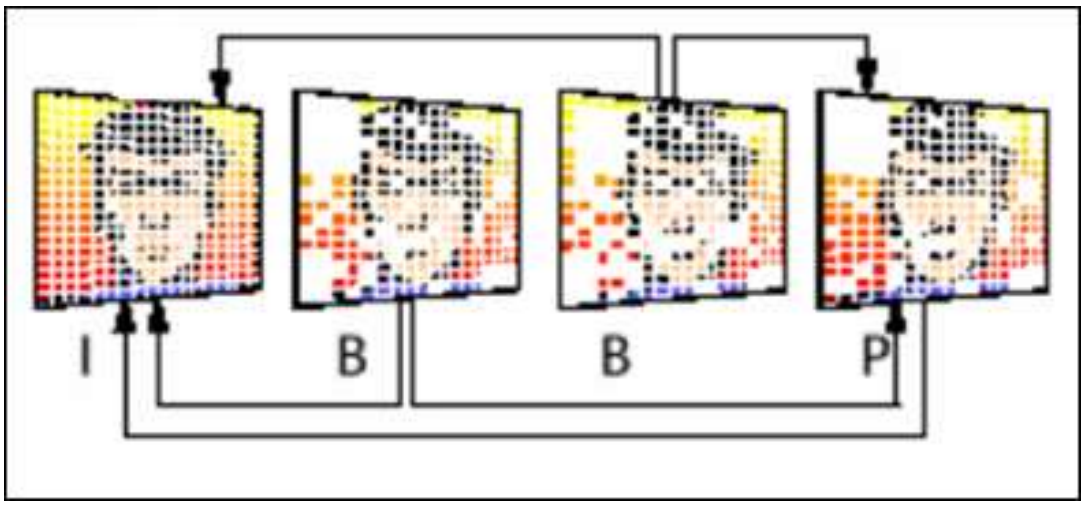

Рисунок 11 – Видеопоследовательность с различными типами изображений

Потоки цифровых данных разделены во времени на пакеты, выстроенные в определенной последовательности. Каждый пакет содержит заголовок и некоторое количество байтов данных, одного и только одного элементарного потока. Стандарт описывает несколько видов потока. Элементарный поток – это кодированный по стандарту MPEG поток видео, звука, или вспомогательных данных, относящихся к одной ТВ программе. Пакетированный элементарный поток (PES) включает метки времени и заголовки. Программный поток (PS) представляет собой сборку элементарных потоков, относящихся к одной ТВ программе. Они объединяются общими часами. Такой поток состоит из пакетов переменной длины и предназначен для записи или передачи по каналам с малой вероятностью ошибки. В частности, именно программные потоки записываются на диски CD-ROM и DVD. Транспортный поток (TS) – другой вид объединения элементарных потоков. Он может содержать несколько программ с независимыми часами, и даже различными тактовыми частотами. Транспортные потоки состоят из пакетов фиксированной длины (188 байт) и предназначены для передачи по каналам с ошибками, в частности для телевещания. На рисунке 12 показана структурная схема пакета транспортного потока.

Термин пакет, применительно к различным типам потоков, используется для обозначения сильно различающихся объектов. В программном потоке пакет неопределенной длительности содержит одну единицу воспроизведения данных определенного типа, или несколько таких единиц. Единица воспроизведения видеопотока – одно изображение (один видеокадр), а единица воспроизведения звукопотока – один звуковой кадр. В транспортном потоке, напротив, пакет имеет жестко определенную длительность, но может содержать данные любого типа. Ни один из потоков не является подмножеством, либо компонентом другого.

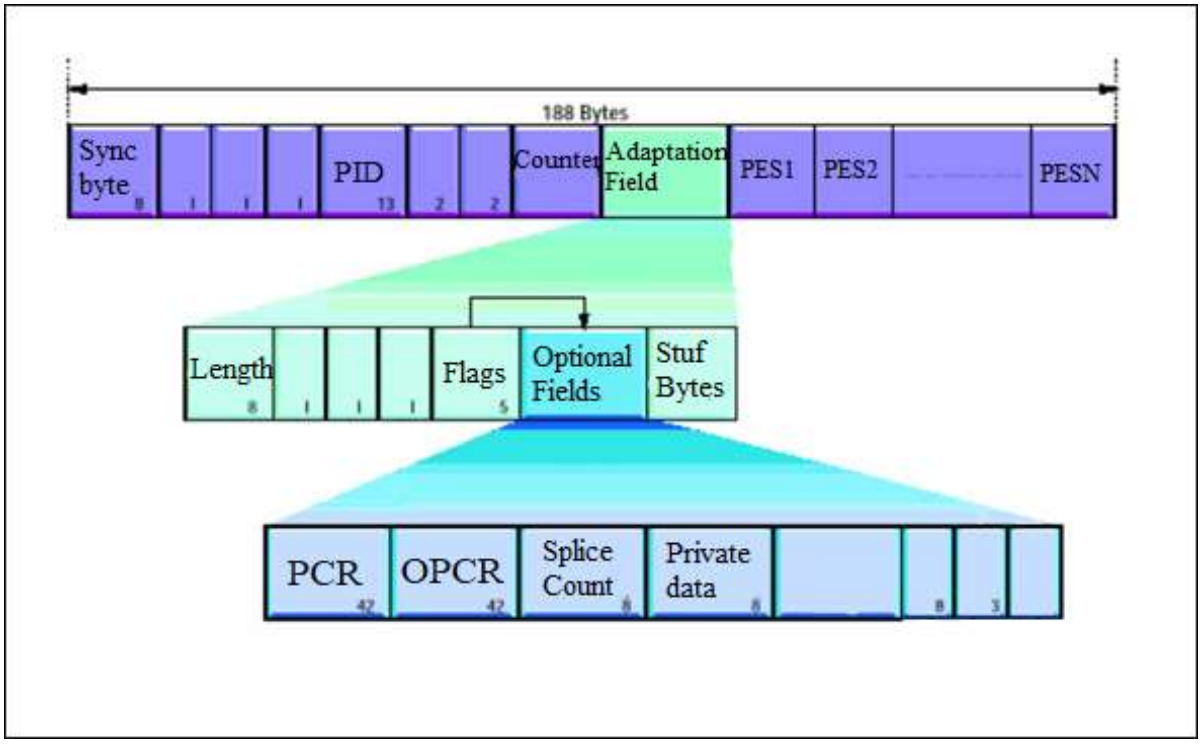

Рисунок 12 – Структура пакета транспортного потока

2.2. Форматы потокового видео

Наиболее популярным на сегодняшний день форматом файла хранения видео является формат MP4 (MPEG-4). Основное преимущество формата – его компактные размеры, благодаря чему открыть файл MP4 можно практически на любом устройстве, их просто пересылать и хранить. Изначально не предназначен для потокового вещания, но такая возможность имеется.

Формат файла TS (Transport Stream) используется для передачи и транспортировки аудио, видео и других данных в телекоммуникационных системах, таких как цифровое телевидение и видеостриминг. Он часто используется для трансляции видеопотоков и обеспечивает эффективное и надежное вещание мультимедийного контента. Формат транспортного потока требует больше места для того же содержимого поэтому файлы этого формата весят значительно тяжелее других форматов видео.

Несмотря на то, что цифровой видеопоток более качественный чем аналоговое видео, в нем тоже возникают ошибки, эти ошибки делят на три по важности приоритета.

1. Приоритет первого уровня: жизненно важные, результаты которых могут довести ситуацию до потери декодирования вещательной службы.

2. Приоритет второго уровня: результаты которых могут повлиять на функционирование системы в целом.

3. Приоритет третьего уровня: результаты которых отражают соответствие правилам DVB использованных средств MPEG.

Ошибки первого и второго приоритета прекратят трансляцию цифрового видеопотока, пользователь просто ничего увидит. Ошибки третьего приоритета искажают изначальную картинку видеопотока, пользователь видит смещение кадров, стоячие кадры, искажение цвета.

2.3. Описание требований для решения поставленной задачи

На основе изученного материала можно сделать перечень необходимых требований для реализации детектирования нейросетевой моделью дефектов видеотрансляции.

Для выполнения поставленной задачи необходимо выполнить следующие пункты.

1. Сделать набор изображений, содержащих кадры с дефектами трансляции

2. Разметить полученный набор изображений на типы ошибок, изображенные на кадре

3. Выбрать существующую нейросетевую модель и обучить ее на созданном датасете.

4. Написать приложение, которое будет использовать обученную модель для обнаружения дефектов на видео.

Для реализации текущих требований необходимо также использовать дополнительное ПО. Для воспроизведение потокового видеотрансляции будет использован VLC mediaplayer [8]. Данный медиаплеер имеет имеет возможность воспроизведения файлов потокового вещания, проигрывает большинство кодеков, имеет открытый исходный код. Для обучения нейронной сети необходимо создать датасет с изображениями дефектов видеотрансляции. Для создания изображений с дефектами нужно получить файл видеотрансляции с приоритетом ошибок пакетов третьего уровня. Для внесения в пакет ошибок будет использовано ПО TSDuck [9]. Чтобы получать техническую информацию о файле содержащем видеотрансляцию, будет использовано приложение Mediainfo [10]. Для вырезания кадров из видео используется программа Kdenlive [11].

Выводы по второй главе

В ходе анализа технологий цифрового вещания из стандартов MPEG сжатия и вещания, были получены знания о структуре пакетов цифрового видеопотока и о ошибках цифровой видеотрансляции. На основе изученного материала были описаны требования к разработке нейросетевой модели и подобрано ПО для достижения поставленных задач.

3. НАБОР ИСПОЛЬЗУЕМЫХ ДАННЫХ

Подбор видео для создания набора данных

Для обучения нейросетевой модели необходимо иметь датасет с изображениями дефектов трансляции. Было принято решение создать свой датасет данных используя свободные источники. Были скачаны видеозаписи из свободных источников, которые транслируются по цифровому телевидению. Были скачаны видеотрансляции цифровых телепередач следующих форматов: мультфильмы, спорт, новости, наука, детективы, произведения художественной культуры, передачи о животных, реклама, ситкомы и комедийные шоу. На каждый формат было скачано около 10 видеозаписей. В сумме, было скачано 100 видео файлов. Все видеозаписи были конвертированы в формат потокового вещания «.TS» с использованием программы VLCMediaplayer.

Создание первичного набора данных

Для создания первичного набора данных используется ПО TSDuck. Управление программы осуществляется через терминал. Используемая программа в качестве входных данных использует файл формата «.TS» и файл параметров. В таблице 1 приведены параметры внесения ошибок в файл с пакетами потокового вещания.

Таблица 1 – Описание параметров программы TSDuck

№	Параметр	Описание параметра
1	`PID value`	PID элементарного потока видео
2	`random_packet_err`	Внесение ошибок в поток случайным образом путем искажения транспортного пакета
3	`random_bit_err`	Искажаются произвольные биты в потоке
4	`packet_err`	Параметр, в который вносится способ внесения ошибок с атрибутом value
5	`continuity_count_error`	Внесение искажений в битовую маску пакета
6	`tel`	Установление определенной маски в заголовке пакета
7	`drop`	Удаление определенного пакета
8	`offset pkt_cnt`	Смещение от начала потока
9	`max_pkt_err`	Максимальное количество битовых ошибок на один пакет
10	`err_interval`	Внесение ошибок через заданный интервал

Используемая программа вносит ошибки в пакеты файла видеотрансляции. Для демонстрации работы программы будут использованы рекомендованные параметры, которые были предоставлены самими разработчиками.

1. Смещение от начала потока (`offset pkt_cnt="10"`).
2. Количество ошибочных пакетов на миллион (`random_packet_err err_cnt="1000"`).
3. Способ внесения ошибок (`packet_err value="stuffing" burst="false"`).

Для первого тестового запуска было использовано рекламное видео. На рисунке 13 показан пример кадра видеотрансляции с ошибками в пакетах. На рисунке 14 показан тот же кадр, но не подвергнутый ошибкам в пакетах.

Рисунок 13 – Кадр видеотрансляции с ошибками в пакетах

Рисунок 14 – Кадр видеотрансляции без ошибок в пакетах

Сравнивая рисунки 13 и 14, можно сделать заключение, что ошибки в пакетах данной видеотрансляции привели к ошибкам в макроблоках кадра и задержке кадра на 1 секунду.

Экспериментальным путем было выявлено, что для получения максимального количества дефектов в видеопотоке, необходимо увеличить максимальное количество битовых ошибок на пакет (`max_pkt_err`). Для каждой видеозаписи параметры из таблицы 1 подбирались индивидуально, т.к. некоторые значения одного видео могут сделать следующий видеофайл не пригодным для воспроизведения.

В все файлы видеотрансляций были внесены ошибки с помощью утилиты TSDuck. Примеры полученных ошибок видеотрансляции указаны в приложении Б.

Чтобы перейти к разметке датасета, необходимо из файлов видеотрансляции вырезать кадры, содержащие дефекты в картинке. Программа для видеомонтажа, Kdenlive, имеет инструментарий для обработки файлов видеотрансляции «TS». С помощью данного ПО было вырезано 1000 кадров с разрешением высокой четкости (1280x720) в формате «.png». Изображения с низким качеством были удалены

3.1. Разметка первичных данных

Для разметки изображений была использована программа Label Studio [12]. Label Studio — это инструмент для разметки данных с открытым исходным кодом. Поддерживает несколько типов аннотаций.

Для обучения нейросетевой модели вырезанный набор кадров с дефектами был размечен в программе Label Studio. Для каждого типа дефекта был определен класс. Метод разметки сегментов – Bounding boxes. Краткое описание классов приведен в таблице 2.

Таблица 2 – Описание классов разметки

№	Название класса	Описание класса разметки
1	zone_error	Область кадра со всеми ошибками, которые изображены на кадре
2	microblock_error	Область кадра с ошибками микроблока(пиксельный-шум,)
3	macroblock_error	Область кадра в которой находится единичная ошибка макроблока, которые по размеру не больше объекта на кадре.
4	zone_macroblock_error	Область кадра с ошибками макроблока, которая по размеру равна целому объекту на кадре или больше его.
5	zinzone_col_error	Область кадра с ошибками макроблока, которая по размеру равна целому объекту на кадре. Макроблоки радикально отличаются цветом.
6	microblock_col_error	Область кадра с ошибками микроблока(пиксельный-шум,)
7	macroblock_col_error	Область кадра в которой находится единичная ошибка макроблока. Макроблоки радикально отличаются цветом
8	line_string_error	Область кадра, на котором находится ошибки из макроблоков, которые расположены строго по ряду от начала и до конца области ошибки. Каждый ряд макроблоков имеет один цвет.
9	line_colstring_error	Область кадра, на котором находится ошибки из макроблоков, которые расположены строго по ряду от начала и до конца области ошибки. Каждый ряд макроблоков имеет один цвет. Цвет ряда макроблоков радикально отличаются
10	hor_line_error	Область кадра с ошибками макроблоков. Примерно похож на линию в полную ширину кадра. Может быть разной толщины.
11	hor_coline_error	Область кадра с ошибками макроблоков. Примерно похож на линию в полную ширину кадра. Может быть разной толщины. Цвет макроблоков радикально отличается.

Такое большое количество классов разметки обусловлено тем, что неизвестно какой тип ошибок будет обнаруживаться лучше – после обучения об качестве обнаружения видов ошибок были сделаны выводы. Основная задача нейросетевой модели – обнаружение только кадра с дефектом. Пример разметки кадра изображен на рисунке 15

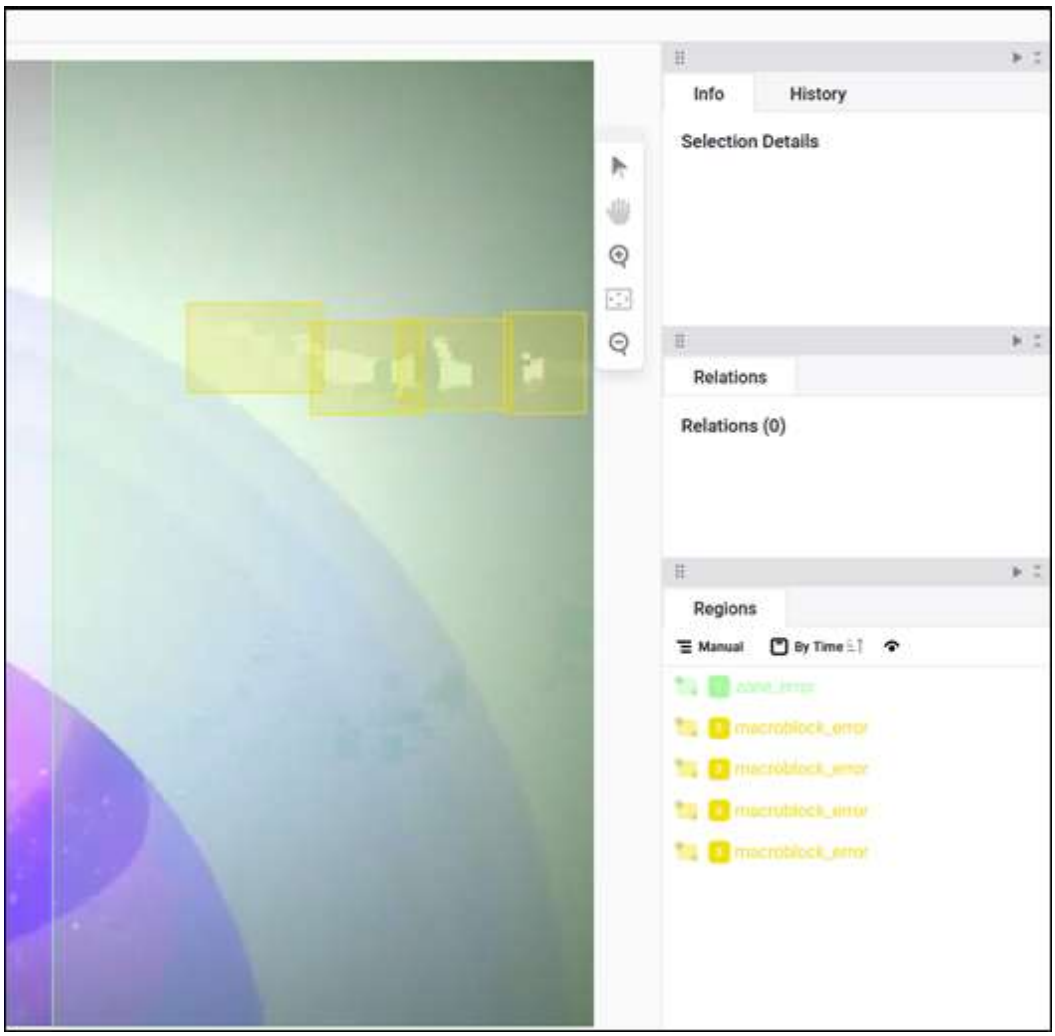

Рисунок 15 – Пример разметки изображения с дефектом кадра в рабочем окне программы Label Studio

3.2. Сбор тренировочной, тестовой, валидационной выборки

В работе нейросети был использован самостоятельно подготовленный набор данных на основе файлов видеотрансляций с дефектами кадров. Датасет состоит из 1016 изображений: 816 изображений с дефектами и 200 изображений с обычными кадрами видеотрансляции, которые не содержат дефектов. Количество единиц размеченных классов представлен в таблице 3.

Таблица 3 – Количество размеченных единиц каждого класса

№	Название класса	Количество размеченных сегментов класса
1	zone_error	816
2	microblock_error	421
3	macroblock_error	10 116
4	zone_macroblock_error	1 154
5	zinzone_col_error	433

Окончание таблицы 3

№	Название класса	Количество размеченных сегментов класса
6	microblock_col_error	257
7	macroblock_col_error	2 155
8	line_string_error	3 144
9	line_colstring_error	173
10	hor_line_error	390
11	hor_coline_error	105

Из данных, содержащихся в таблице 3, можно сделать вывод, что не все ошибки трансляции одинаково распространены. Это обусловлено тем, чтобы получить дефект макроблока достаточно небольшого количества ошибок в пакете видеотрансляции, а чтобы получить дефект с ошибками в цвете, необходима определенная последовательность ошибок в нескольких пакетах, что не так просто получить.

Также на одном кадре могут быть сразу несколько типов ошибок, некоторые ошибки могут наслаиваться друг на друга, получается. Пример кадра с несколькими ошибками представлен на рисунке 16.

Рисунок 16 – Изображение содержащие кадр с несколькими типами ошибок

В данном случае, чтобы нейросеть правильно обучилась обнаруживать ошибки видеотрансляции, необходимо распределить изображения на тренировочную, тестовую и валидационную выборку так, чтобы количество изображений с определенным классом примерно удовлетворяла соотношению: 70x20x10. После распределения данных, получилось: 686 изображений в тренировочной выборке, 225 в тестовой, 105 в валидационной.

Также, необходимо сбалансировано распределить количество размеченных ошибок датасета по выборкам, чтобы нейросетевая модель, чтобы обучаемая модель корректно обучилась. При сильном дисбалансе количества размеченных классов в выборках, модель может проигнорировать некоторые кадры для обучения

Выводы по третьей главе

В данной главе были описаны необходимые действия для получения видеофайла с дефектами видеопотока. На основе полученного первичного набора данных были получены кадры с ошибками изображения. Классам разметки были присвоены определения для понимания типа объекта обнаружения. Была произведена разметка изображений, основываясь на типах классифицированных дефектов. Сбалансированно распределены данные на тренировочную, тестовую и валидационную выборку в программе Label Studio. Полученный, размеченный датасет был подготовлен для последующего обучения.

4. РЕАЛИЗАЦИЯ НЕЙРОСЕТЕВОЙ МОДЕЛИ

4.1. Среда выполнения и обучение нейросетевой модели

В качестве основного языка программирования для разработки был выбран язык Python версии 3.12.0 [13]. Для данного языка существует множество библиотек, упрощающих работу с данными и машинным обучением.

Для обучения моделей машинного обучения была выбрана среда Google Colaboratory [14]. Google Colaboratory – облачный сервис, предоставляющий бесплатный доступ к вычислительным ресурсам GPU и TPU, работающий на основе Jupyter Notebook. Выбранная среда позволяет запускать код в отдельных ячейках, что позволяет быстро отлаживать работу разрабатываемых приложений. Также в Google Colaboratory предустановлены многие популярные библиотеки для работы с данными и машинным обучением.

Для обучения была выбрана модель обнаружения объектов, YOLO [15]. YOLO – это система (сеть) обнаружения объектов. Модели YOLO подходят для задач: обнаружения объектов, сегментации экземпляров, классификации, ориентированному обнаружению. Наибольшим преимуществом YOLO над другими архитектурами является скорость. Модели семейства YOLO исключительно быстры и намного превосходят R-CNN (Region-Based Convolutional Neural Network). Это позволяет добиться обнаружения объектов в режиме реального времени, что крайне необходимо для обнаружения дефектов видеотрансляции.

В качестве среды разработки выбрана программа Visual Studio Code [16]. Данный редактор кода способен работать с Jupyter Notebook, позволяет запускать код в отдельных ячейках.

CNN

Сверточная нейронная сеть (CNN) – архитектура искусственной нейронной сети, предназначенная для обработки пространственных отношений, используя сверточные слои [17]. Данная архитектура нейронных сетей является крайне эффективным инструментом для задач обнаружения визуальных шаблонов.

Архитектура сверточной нейронной сети (CNN) состоит из следующих слоев.

1. Сверточный слой (Convolution layer). Он обнаруживает локальные шаблоны во входном изображении. Применяются обучаемые фильтры (ядра) для обнаружения схожих признаков входного изображения.

2. Слой подвыборки (Pooling Layer). Этот слой обобщает при признаки объекта, делает представление об объекте более устойчивым ко сдвигам и деформациям.

3. Полносвязный слой (Fully Connected Layer) принимает двумерные карты признаков и преобразует их в вектор признаков фиксированной длины, классификация изображений на основе признаков.

4. Слой активации добавлет нелинейность к выходу предыдущего слоя, позволяя сети более точно предсказывать результат.

5. Слой нормализации пакета (Batch Normalization Layer) нормализует входные данные для каждого признака во время обучения, давая возможность использовать более высокие скорости обучения.

Схема архитектуры сверточной нейронной сети представлена на рисунке 17.

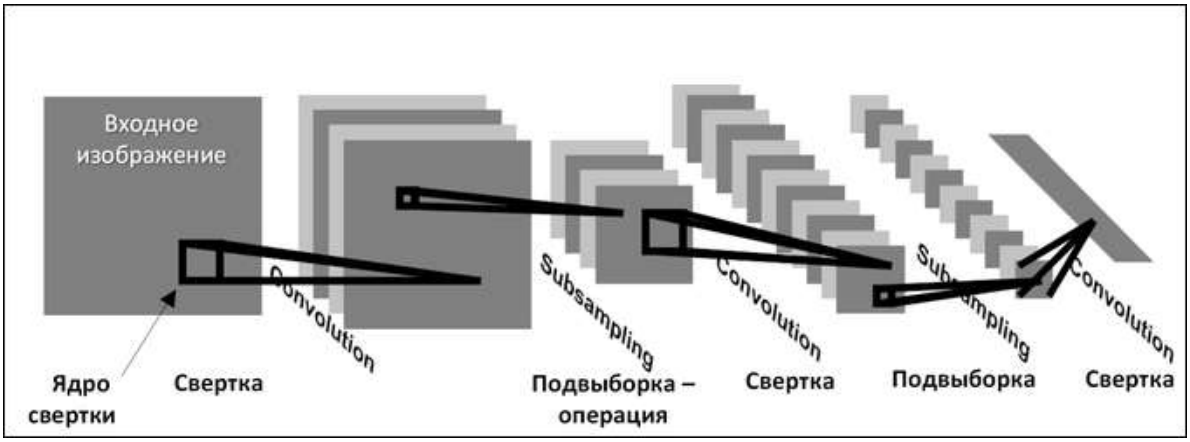

Рисунок 17 – Архитектура сверточной нейронной сети

Обучение сверточной нейронной сети происходит с использованием алгоритма обратного распространения ошибки на размеченных наборах

данных. Процесс обучения медом обратного распространения ошибки включает в себя следующие шаги.

1. На входной слой нейронной сети подаются веса.
2. Последовательно вычисляются выходные сигналы для каждого слоя.
3. Вычисление ошибки. Выходные нейроны сравниваются с истинными метками классов, и вычисляется ошибка (функция потерь, категориальная кросс-энтропия)
4. Обратное распространение. Значение ошибки распространяется обратно через сеть, начиная с выходного слоя. Для каждого слоя вычисляются градиенты ошибки по отношению к весам слоя.
5. Обновление весов. Веса сети обновляются пропорционально градиентам ошибки для ее минимизации. Достигается это с помощью оптимизатора.
6. Повторение предыдущих шагов для каждого изображения в наборе данных в течение нескольких эпох.

Первое обучение модели YOLO

YOLO – проект сверточной нейронной сети компании ultralytics, который находится в свободном доступе. Официально доступно несколько версий проекта, каждая версия имеет отдельные преимущества. Для обнаружения дефектов видеотрансляции были выбраны версии: YOLOv5, YOLOv8. YOLOv5 является легкой нейросетевой моделью. Главное отличие этой версии – способность принимать самое большое разрешение изображения (1280x1280) по сравнению с другими версиями. YOLOv8 – последняя версия этого проекта на данный момент. Она имеет самую высокую скорость и точность работы. Каждая версия YOLO имеет несколько моделей для обнаружения, сегментации и классификации. Версии, имеющие суффикс «n», имеют наименьшее количество параметров и слоев, что делает их быстрыми. Версии с суффиксом «x» имеют большее количество слоев, что

делает их более точными. На рисунке 18 показан пример нескольких моделей для версии YOLOv8. Промежуточные модели предлагают баланс между точностью и скоростью.

Рисунок 18 – Модели YOLOv8

Для последующего обучения были выбраны самая быстрая и самая точная модель версий YOLOv5 и YOLOv8, чтобы узнать какая из моделей продемонстрирует наилучший результат. В таблице 4 представлены несколько выбранных моделей YOLO и их показатели.

Таблица 4 – Показатели моделей YOLO

№	Наименование модели YOLO	Максимальный размер изображения	Скорость обработки одной фотографии (мс)	mAPval (средняя точность обнаружения объектов)
1	yolov5x6u.pt	1280	1,06	56,8
2	yolov5n6u.pt	1280	3,81	42,1
3	YOLOv8x.pt	640	3,53	53,9
4	YOLOv8n.pt	640	0,99	37,3

Обучение моделей YOLO происходило в среде Google Colaboratory. Код для обучения нейронной сети представлен в листинге 1. В таблице 5 описаны используемые переменные для листинга 1. Для обучения была использована библиотека google.colab для управления файлами в используемой среде.

Листинг 1 – Обучение моделей YOLO

```
from google.colab import drive
drive.mount('/content/drive')

ROOT_DIR = '/content/drive/MyDrive/Colab_Notebooks/Diplom/data'

!pip install ultralytics
import os
from ultralytics import YOLO

model = YOLO("yolov5n6u.pt")
result = model.train(data=os.path.join(ROOT_DIR, "Data_T100.yaml"),
epochs=50, imgsz=200)
```

Таблица 5 – Описание переменных

№	Название переменной	Описание переменной
1	`ROOT_DIR`	Директория хранения обучающей, тестовой и валидационной выборки
2	`model`	Используемая модель для тренировки
3	`epochs`	Количество эпох
4	`imgsz`	Разрешение изображений
5	`batch`	Количество изображений, обрабатываемых за один проход

Чтобы понять какие нужны параметры, необходимо было провести первое оценочное обучение. Была выбрана модель с самой большой скоростью обучения – YOLOv8n.pt, с количеством эпох – 50, и разрешением изображений – 400x400. Результаты обучения представлены на рисунке 19 и 20. Из рисунка 19 и 20 можно сделать вывод, что модель не отличает ошибки видеотрансляции от обычных объектов, находящихся на кадре, но находит зону кадра, в которой находятся ошибки.

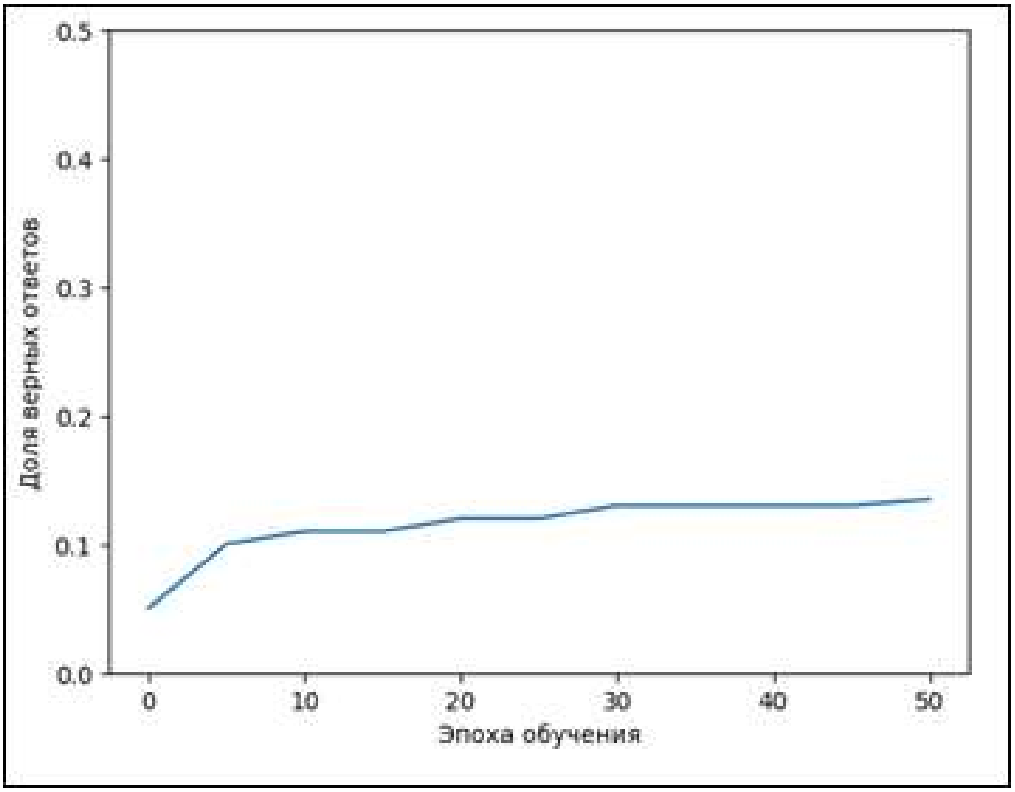

Рисунок 19 – Результат обучения модели

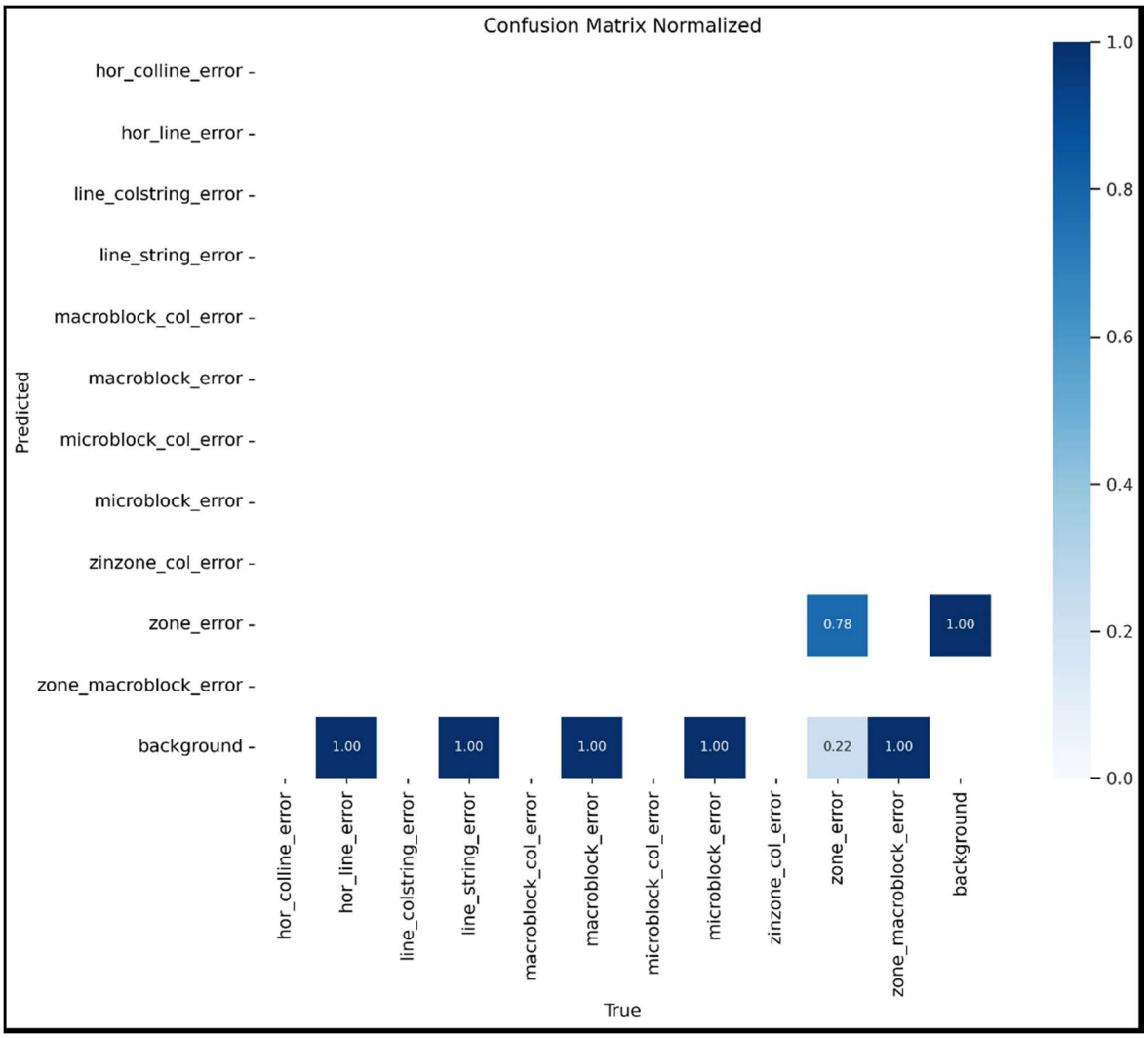

Рисунок 20 – Матрица ошибок первого обучения

Для повторного тестового обучения изменим показатель разрешения картинки до максимума (640). Количество эпох обучения остается неизменным, для более точных моделей количество эпох будет уменьшены, в связи с ограничением на вычислительные ресурсы. Модель обучения не меняется. На рисунках 21 и 22 показаны результаты повторного тестового обучения модели.

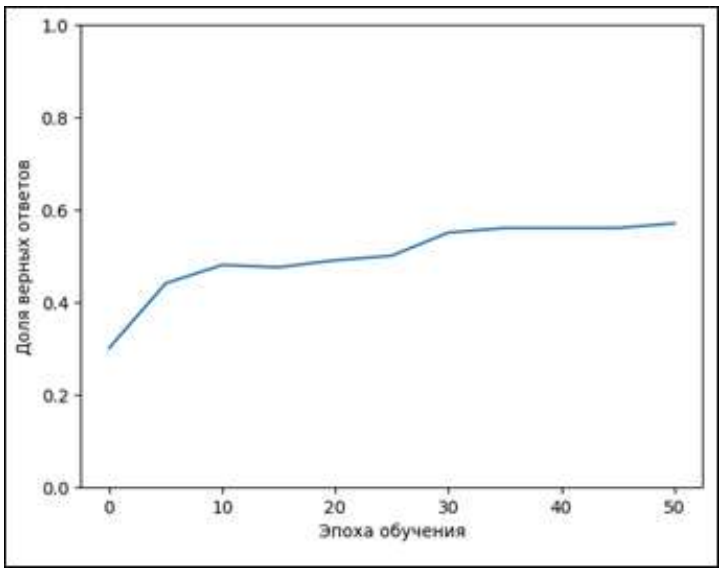

Рисунок 21 – Результат второй попытки обучения модели

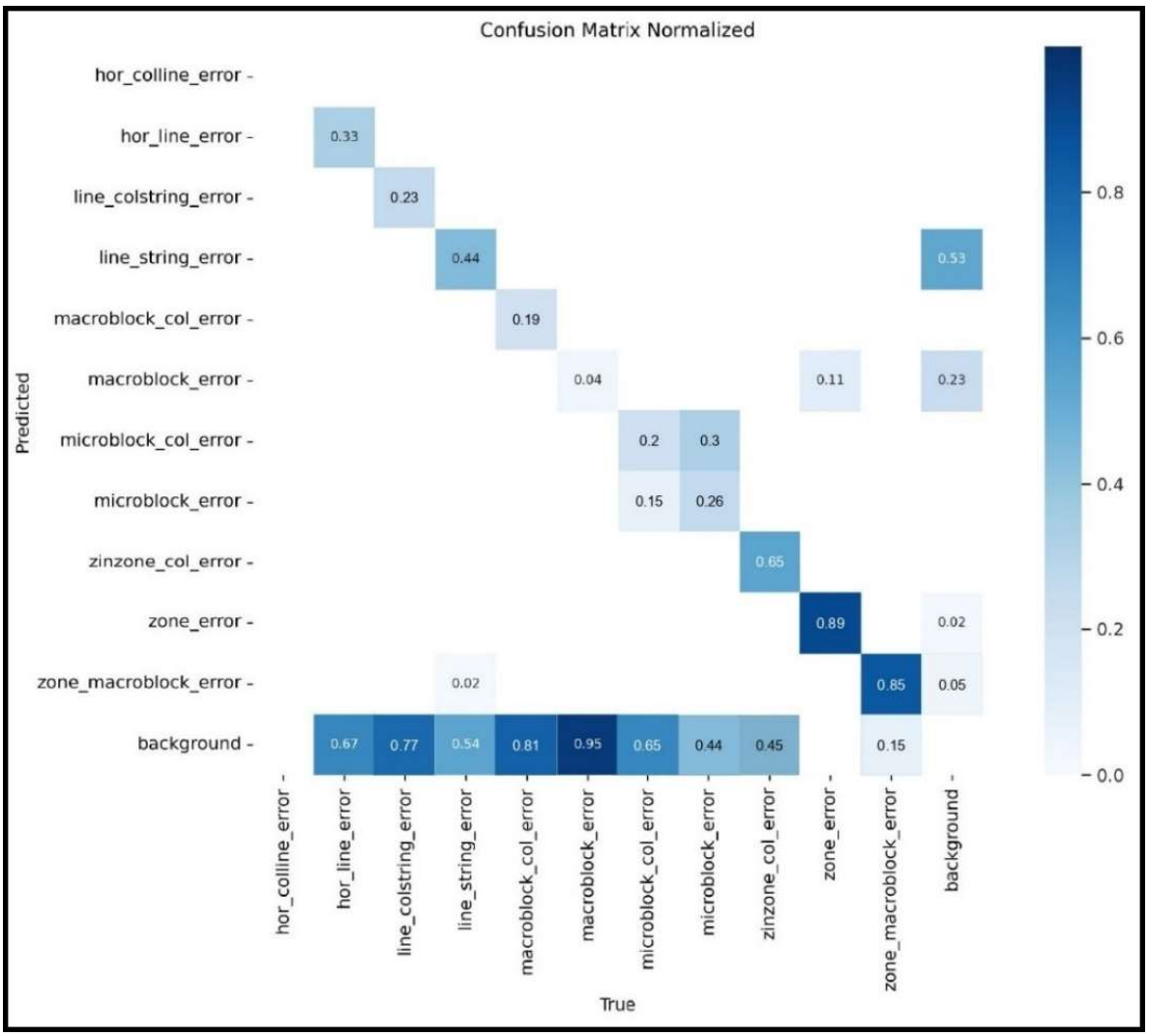

Рисунок 22 – Матрица ошибок второй попытки обучения

4.2 Обучение моделей YOLO и результаты обучения

На основе первых тестовых попытках обучения можно сделать вывод, что разрешение изображений – крайне важный параметр. Если качество картинки слишком низкое, то нейросеть не сможет отличить мелкие ошибки изображения от объектов расположенных на картинке, таким образом, необходимо установить максимально возможное разрешение для обучения каждой модели. Также для получения наилучших результатов изменялись гиперпараметры моделей. Во время обучения моделей было замечено, что на итоговый результат влияет значение гиперпараметра «batch» (количество изображений, обрабатываемых за один проход). Гиперпараметры – это высокоуровневые, структурные настройки алгоритма. Они задаются перед этапом обучения и остаются неизменными во время него. Нейросетевые модели YOLO защищены от эффекта переобучения – это достигается автоматическим сохранением параметров, при которых нейронная сеть имеет минимальное значение ошибки.

В таблице 6 приведены финальные значения переменных для обучения каждой модели.

Таблица 6 – Значения параметров для обучения моделей

№	Модель YOLO	Максимальный размер изображения	Скорость обработки одной фотографии (мс)	mAPval (средняя точность обнаружения объектов)	Количество изображений, обрабатываемых за один проход
1	yolov5x6u.pt	1280	1.06	56.8	8
2	yolov5n6u.pt	1280	3.81	42.1	4
3	YOLOv8x.pt	640	3.53	53.9	16
4	YOLOv8n.pt	640	0.99	37.3	12

Наилучшие результаты обученных моделей обучения приведены в приложении В. Из полученных результатов можно сделать вывод, что наибольшей точностью в обнаружении дефектов обладает обученная модель YOLOv8x.pt. После получения наилучших результатов обучения моде-

лей было выявлено, что наибольшее разрешение влияет на качество обнаружения ошибок только до достижения определенного значения. На данной выборке обучения достаточным разрешением для детектирования изображений является 540x540 точек.

Вычислительные ресурсы сервиса Google Colaboratory ограничены для бесплатного использования, поэтому для моделей YOLOv8x.pt и yolov5x6u.pt были сокращено количество эпох обучения. Даже с меньшим количеством эпох обучения модель YOLOv8x.pt смогла показать наилучший результат. Для последующего использования будет использоваться обученная нейросетевая модель YOLOv8x.pt.

Выводы по четвертой главе

В данной главе описано обучение выбранных моделей сверточных нейронных сетей, были подобраны оптимальные параметры для обучения каждой модели нейронной сети, были получены наилучшие результаты для каждой выбранной нейросетевой модели, на основе этих результатов была выбрана модель для дальнейшей работы. Основываясь на полученных результатах можно сказать, что нейросетевая модель очень хорошо обнаруживает объекты на кадре содержащие несколько дефектов, очень хорошо обнаруживает область поражения, отличает кадр с дефектом от кадра без дефекта, но дает много ложных предсказаний для обнаружения конкретного дефекта макроблока или микроблока, также нейросетевая модель плохо различает дефекты с искажением цвета от дефектов без искажения.

5. РЕАЛИЗАЦИЯ МЕТОДА ВЫЗОВА НЕЙРОННОЙ СЕТИ

Для того, чтобы полученная нейросетевая модель могла обрабатывать новые файлы видеотрансляции, необходимо создать метод, который будет подавать на вход обученной модели видеопоток для обнаружения ошибок видеотрансляции.

В качестве основного языка программирования для разработки был выбран язык Python версии 3.12.0. В качестве среды разработки выбрана программа Visual Studio Code. Для достижения необходимого результата воспроизведения видеозаписи и отрисовывания рамок, обозначающих положение дефекты, была использована библиотека OpenCV [18]. В приложении Г представлен код для вызова нейронной сети в среде Visual Studio Code для анализа видеофайла на обнаружение дефектов. Реализованный метод вызывает окно для воспроизведения в нем необходимого видео, рисуя ячейки на кадрах, в которых, по оценке нейронной сети, находится дефект. Метод позволяет проверять видеофайлы на наличие ошибок, можно настраивать ячейки обводки для более удобной работы

Выводы по пятой главе

В данной главе был реализован метод вызова нейронной сети для анализа видеофайла на наличие дефектов. Данный метод работает в среде Visual Studio Code для более гибкой настройки работы. Проверив работу метода на видеопотоке с дефектами, можно сказать, что нейросетевая модель справляется с обнаружением кадра с дефектами, области с ошибками, объекта с дефектами, но плохо угадывает мелкие ошибки микроблоков.

ЗАКЛЮЧЕНИЕ

В рамках данной работы было разработана нейросетевая модель для анализа видео. При этом были решены следующие задачи.

1. Проведен анализ предметной области.
2. Изучены технологии цифрового вещания.
3. Был собран набор данных.
4. Выбрана нейросетевая модель для обучения.
5. Получены результаты обучения нейросетевой модели.
6. Реализован метод вызова нейронной сети для анализа видеофайла.

В настоящий момент проводится дополнительные тестирование разработанной модели. В будущем планируется продолжать разработку и улучшение качества работы нейросетевой модели, в частности обучить модель на новой выборке, использовав иной метод разметки, чтобы повысить долю обнаруживаемых ошибок микроблоков, а также расширить выборку большим количеством размеченных участков кадров. Также планируется улучшить метод вызова нейронной сети для анализа требуемого видеопотока, сделав расширение программного или аппаратного декодера, для более простого и быстрого способа анализа видео. Планируется провести сравнение результатов обученной нейросетевой модели с другими методами детектирования ошибок.

ЛИТЕРАТУРА

1. International Journal of Digital Multimedia Broadcasting. [Электронный ресурс] URL: https://www.hindawi.com/journals/ijdmb/contents/ (дата обращения: 10.02.2024 г.).

2. Mehedi H., Tasneem R., Kiok A., Oksam C. Artifacts Detection and Error Block Analysis from Broadcasted Videos. // arXiv preprint, 2018 г. – 14 с.

3. Glavota I., Kaprocki Z., Vranjes M., Herceg M. No-reference real-time video transmission artifact detection for video signals. // Journal of Real-Time Image Processing, 2020. – С. 799–821.

4. Morals D., Silva A.F., Mylene C.Q., A Correlation-Based No-Reference Packet-Loss. // Conference Paper, 2016. – 6 с.

5. Mylene C.Q. Perceptual contributions of blocky, blurry, noisy, and ringing synthetic artifacts to overall annoyance. // Journal of Electronic Imaging, 2012. – 23 с.

6. Ahmet E., Tekalp M. A., Automatic Soccer Video Analysis and Summarization. // IEEE Transactions on Image Processing, 2003 г. – С. 795–807.

7. Брехт Э.А., Коншина В.Н. Применение нейронной сети YOLO для распознавания дефектов. // Журнал Интеллектуальные технологии на транспорте, 2022 г. – С. 41–47.

8. VLC mediaplayer. [Электронный ресурс] URL: https://www.videolan.org/vlc/index.ru.html (дата обращения: 10.02.2024 г.).

9. TSDuck. [Электронный ресурс] URL: https://github.com/tsduck/tsduck (дата обращения: 10.02.2024 г.).

10. MediaInfo. [Электронный ресурс] URL: https://mediaarea.net/en/MediaInfo (дата обращения: 10.02.2024 г.).

11. Kdenlive. [Электронный ресурс] URL: https://kdenlive.org/en/ (дата обращения: 10.02.2024 г.).

12. Label Studio. [Электронный ресурс] URL: https://labelstud.io/ (дата обращения: 10.02.2024 г.).

13. Python. [Электронный ресурс] URL: https://www.python.org/ (дата обращения: 10.02.2024 г.).

14. Google Colaboratory. [Электронный ресурс] URL: https://colab.google/ (дата обращения: 10.02.2024 г.).

15. YOLO. [Электронный ресурс] URL: https://docs.ultralytics.com/ (дата обращения: 10.02.2024 г.).

16. Visual Studio Code – Code Editing. Refined. [Электронный ресурс] URL: https://code.visualstudio.com/ (дата обращения: 10.02.2024 г.).

17. Yoon K. Convolutional Neural Networks for Sentence Classification [Электронный ресурс] // arXiv.org, 2014. Дата обновления: 03.09.2014 г. URL: https://arxiv.org/abs/1408.5882 (дата обращения: 10.02.2024 г.).

18. OpenCV – open-source library. [Электронный ресурс] URL: https://opencv.org/ (дата обращения: 10.02.2024 г.).

19. Воробьев М.С. Приемные распределительные системы телевидения. – Челябинск: Татьяна Лурье, 2002. – 240 с.

20. Зима З.А. Системы кабельного телевидения. – Москва: МГТУ им. Н.Э. Баумана, 2004. – 616 с.

21. Телевидение: учебник для вузов / В.Е. Джакония, А.А. Гоголь, Я.В. Друзин, Н.А. Ерганжиев, С.Э. Коганер, П.М. Копылов, В.И. Лисогурский, О.В. Украинский. – Москва: Радио и Связь, 2003. – 616 с.

22. Локшин, Б.А. Цифровое вещание: от студии к телезрителю. – Москва: Сайрус системс, 2001. – 446 с.

23. IPTV Challenges and metrics | Technonline [Электронный ресурс] URL: https://www.techonline.com/tech-papers/iptv-challenges-and-metrics/ (дата обращения: 10.02.2024 г.).

24. Considerations for Measurement of PCR Jitter // pixelmetrix: [Электронный ресурс] URL: https://www.pixelmetrix.com/datasheets/AppNote/AN101 PCR Jitter.pdf (дата обращения: 10.02.2024 г.).

25. Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks // Based Networks telcogroup: [Электронный ресурс] URL: https://telcogroup.ru/files/materials-pdf/DVB_standards/IPTV/ts_102034_mpeg2_ts_over_ip.pdf (дата обращения: 10.02.2024 г.).

26. Мониторинг современных систем цифрового телевидения. [Электронный ресурс] URL: https://deps.ua/ua/
(дата обращения: 10.02.2024 г.).

27. Charles, P. Digital video and HDTV / P Charles. – San Francisco: Morgan Kaufmann, 2001. – 700 с.

28. Introducing AI-Assisted Application Screening: Transform your grant review process with intelligent pre-screening. [Электронный ресурс] URL: https://www.smartsimple.com/blog/introducing-ai-assisted-application-screening (дата обращения: 10.02.2024 г.).

29. Jonathan O., Misra S., Osamor V. Comparative analysis of machine learning techniques for network traffic classification. // IOP Conference Series: Earth and Environmental Science, 2021. – Т. 655. – №. 1. – С. 12–25.

30. NLTK: пакет библиотек| Github. [Электронный ресурс] URL: https://github.com/nltk/nltk (дата обращения: 10.02.2024 г.).

31. Chen J. S. An Agile and Low Cost FPGA Implementation of MPEG-2 TS Remultiplexer for CATV Head-end Equipment // The 10th International Symposium on Pervasive Systems, 2009 г. – С. 722–726.

32. Transport of MPEG-2 TS Based DVB Services over IP Based Networks. [Электронный ресурс] URL: https://dvb.org/?standard=transport-of-mpeg-2-ts-based-dvb-services-over-ip-based-networks-and-associated-xml (дата обращения: 10.02.2024 г.).

ПРИЛОЖЕНИЯ

Приложение А. Древо формата MPEG

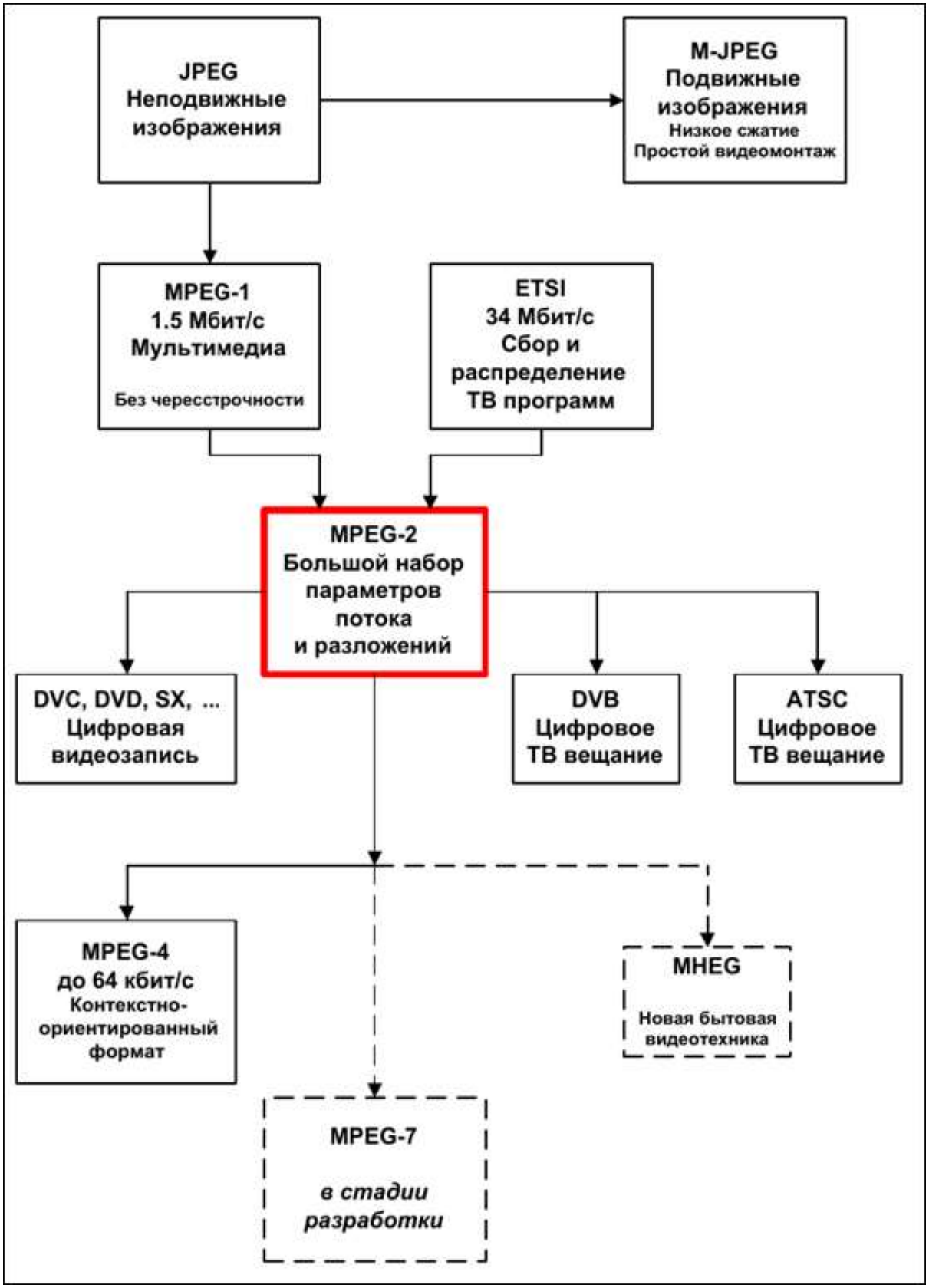

Рисунок 1 – Генеалогия формата MPEG

Приложение Б. Примеры ошибок в файлах видеотрансляции

Рисунок 2 – Кадр с ошибками макроблока

Рисунок 3 – Кадр с объектом с ошибками макроблока

Рисунок 4 – Кадр с ошибками микроблока

Рисунок 5 – Кадр с ошибками макроблока с искажением цвета

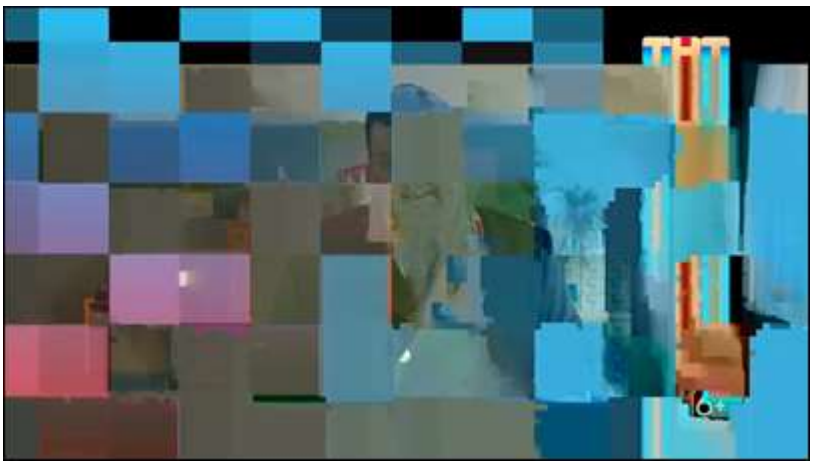

Рисунок 6 – Кадр с объектом с ошибками макроблока с искажением цвета

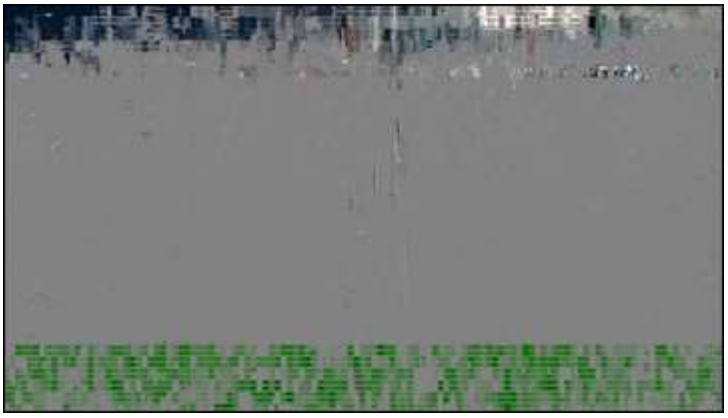

Рисунок 7 – Кадр с ошибками микроблока с искажением цвета

Рисунок 8 – Кадр с ошибками макроблока во всю ширину кадра

Рисунок 9 – Кадр ошибками макроблока по вертикали

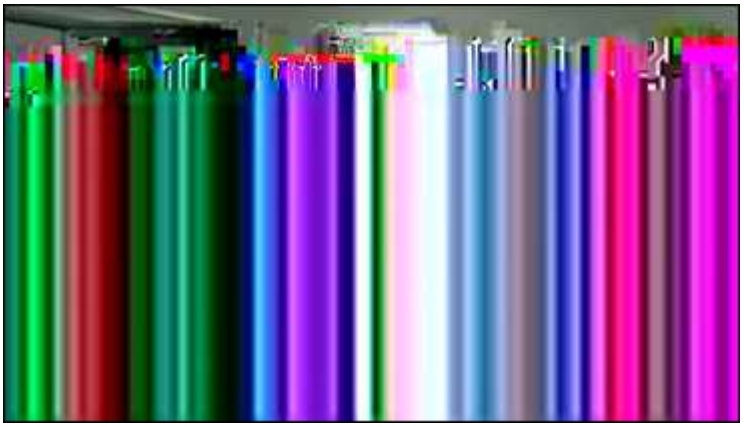

Рисунок 10 – Кадр с ошибками макроблока во всю ширину
с искажением цвета

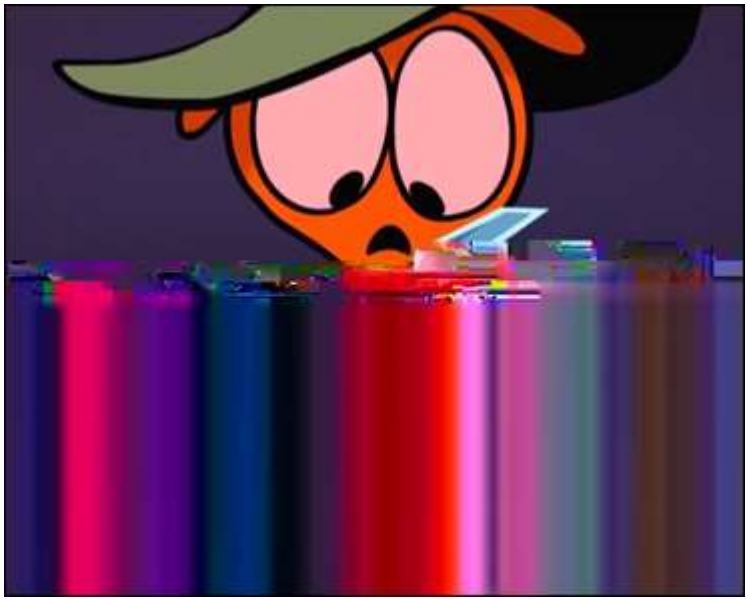

Рисунок 11 – Кадр с ошибками макроблока по вертикали
с искажением цвета

Приложение В. Результаты обучения нейростевых моделей

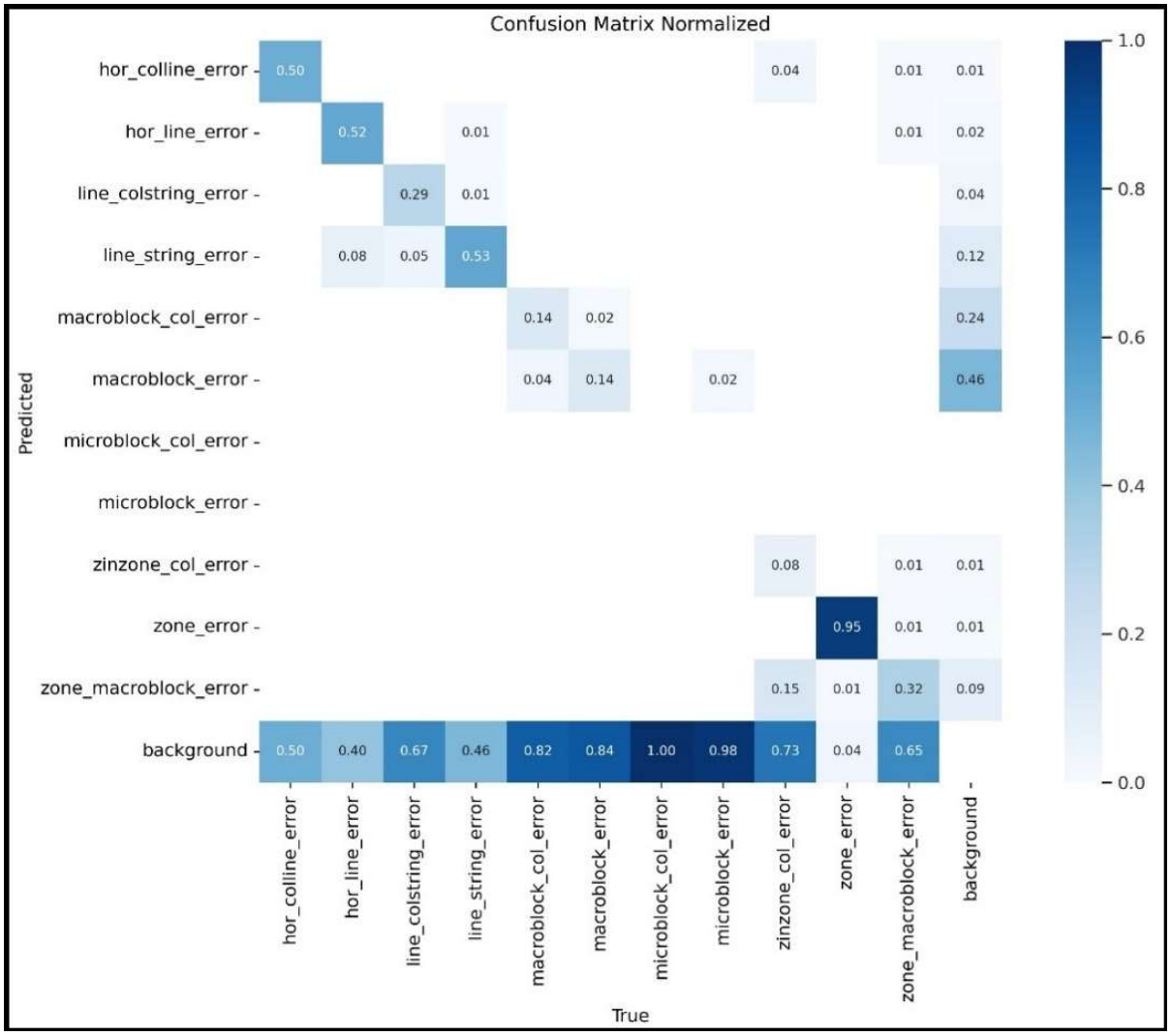

Рисунок 12 – Наилучший результат матрицы ошибок модели yolov5x6u.pt

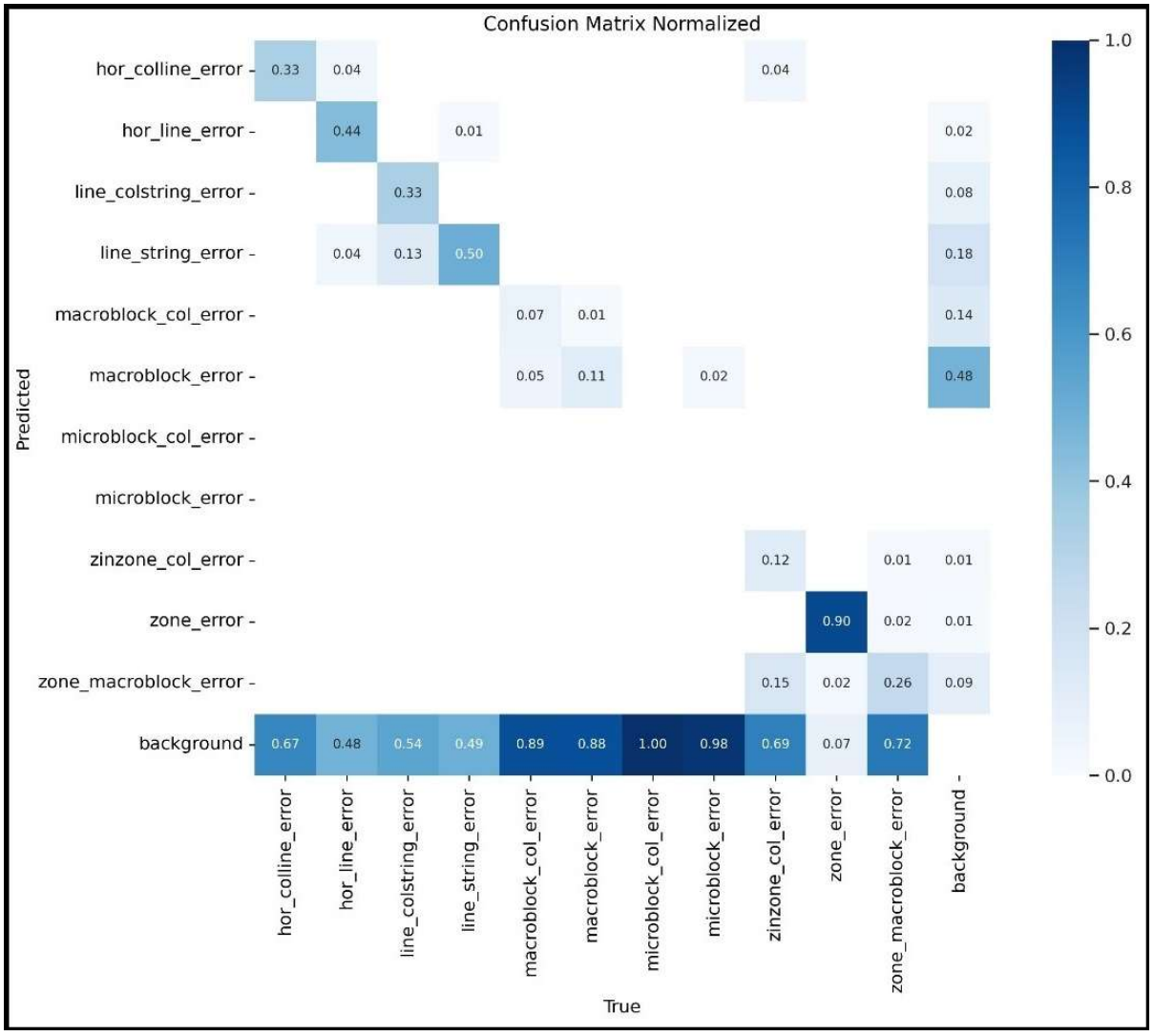

Рисунок 13 – Наилучший результат матрицы ошибок модели yolov5n6u.pt

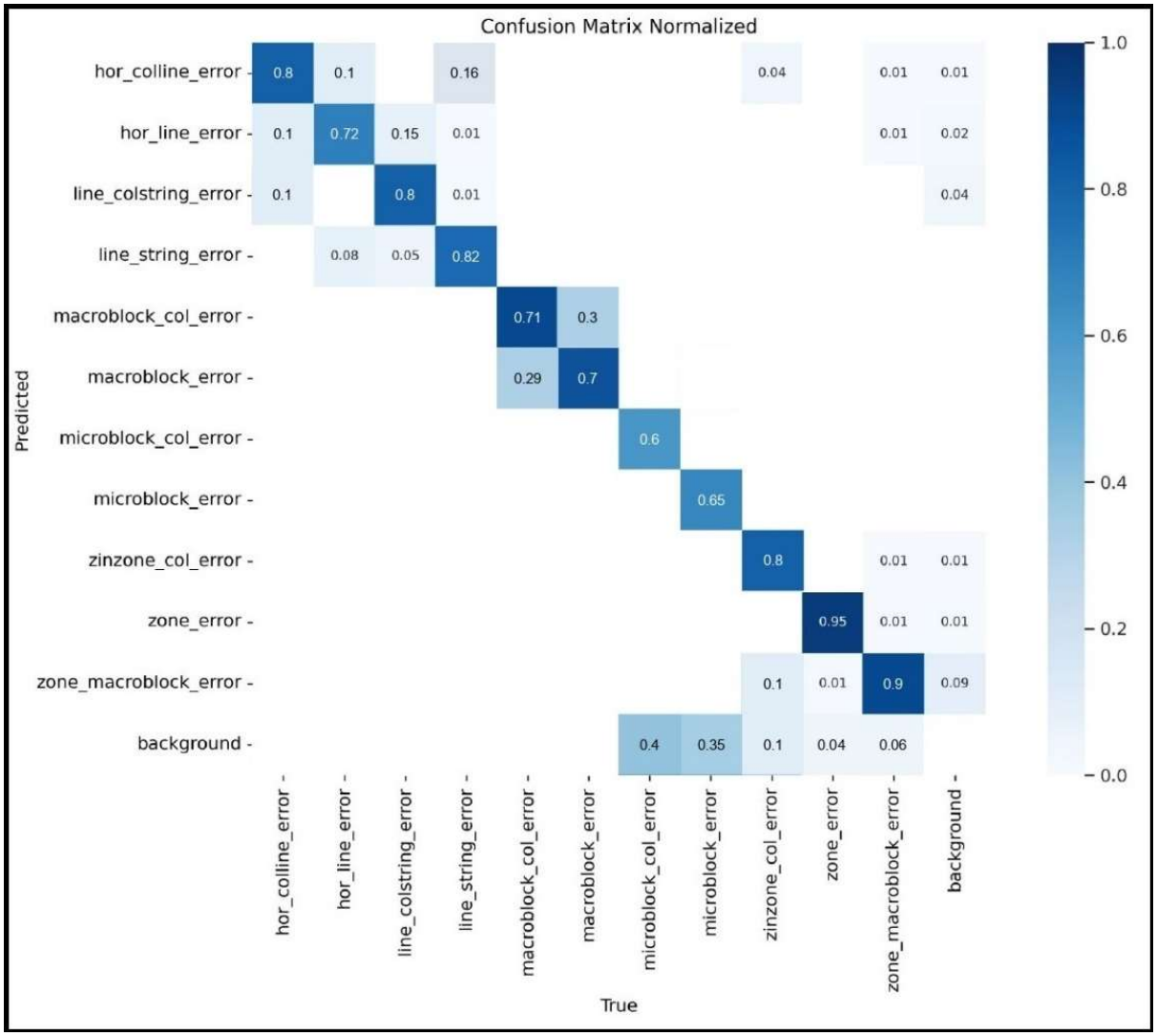

Рисунок 14 – Наилучший результат матрицы ошибок модели YOLOv8x.pt

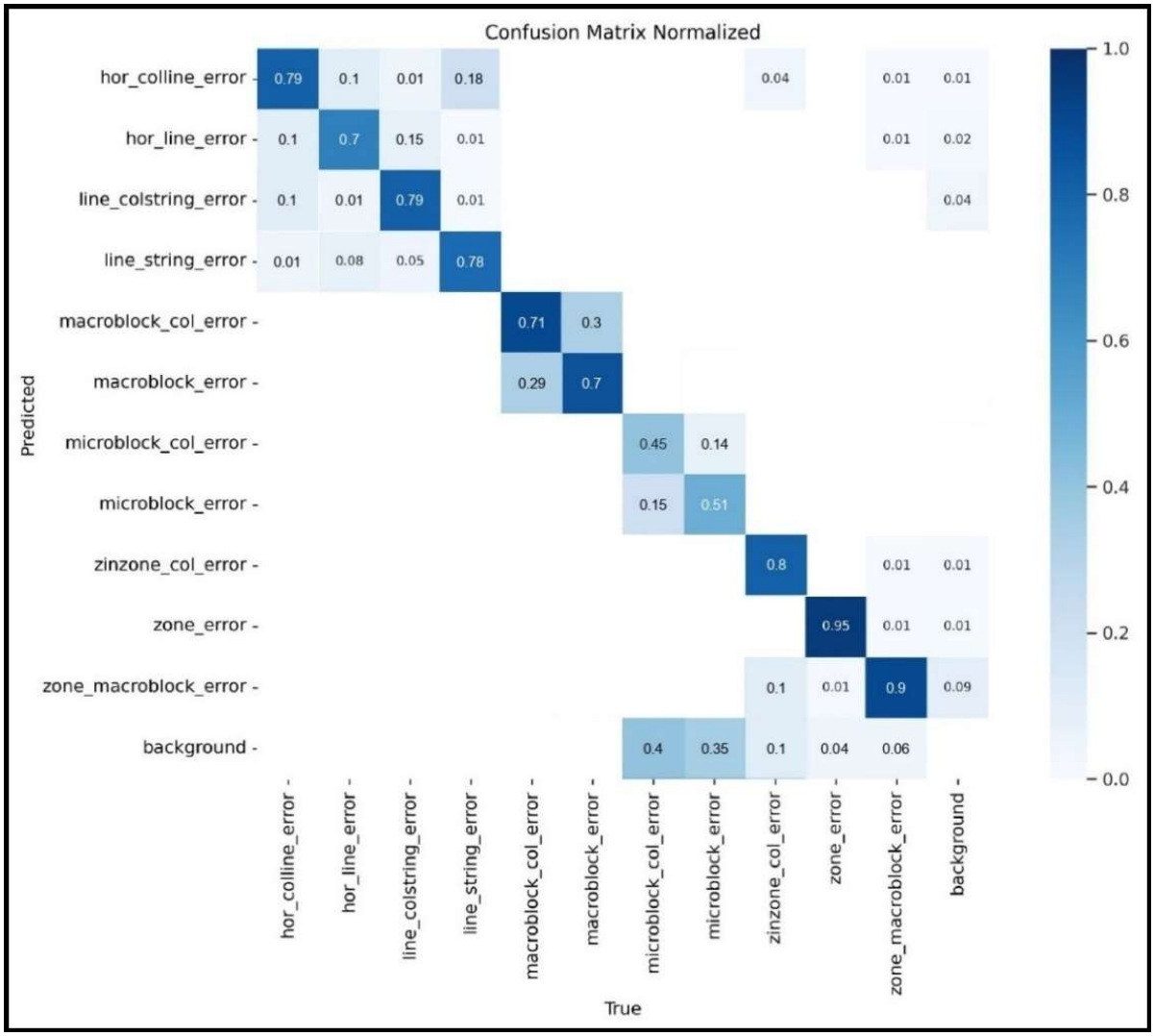

Рисунок 15 – Наилучший результат матрицы ошибки модели YOLOv8n.pt

Приложение Г. Листинг кода вызова нейронной модели

Лиистинг 1 – Код, запускающий нейросетевую модель

```
import cv2
from ultralytics import YOLO
import random

def process_video_with_tracking(model, input_video_path, show_video=True,
save_video=False, output_video_path="output_video.mp4"):
    cap = cv2.VideoCapture(input_video_path)
    if not cap.isOpened():
        raise Exception("Error: Could not open video file.")
    fps = int(cap.get(cv2.CAP_PROP_FPS))
    frame_width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frame_height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
    if save_video:
        fourcc = cv2.VideoWriter_fourcc(*'mp4v')
        out = cv2.VideoWriter(output_video_path, fourcc, fps, (frame_width,
frame_height))
    while True:
        ret, frame = cap.read()
        if not ret:
            break
        results = model.track(frame, iou=0.4, conf=0.5, persist=True,
imgsz=608, verbose=False, tracker="botsort.yaml")
        if results[0].boxes.id != None: # this will ensure that id is not
None -> exist tracks
            boxes = results[0].boxes.xyxy.cpu().numpy().astype(int)
            ids = results[0].boxes.id.cpu().numpy().astype(int)
            for box, id in zip(boxes, ids):
                random.seed(int(id))
                color = (random.randint(0, 255), random.randint(0, 255),
random.randint(0, 255))

                cv2.rectangle(frame, (box[0], box[1]), (box[2], box[3],),
color, 2)
                cv2.putText(
                    frame,
                    f"Id {id}",
                    (box[0], box[1]),
                    cv2.FONT_HERSHEY_SIMPLEX,
                    0.5,
                    (0, 255, 255),
                    2,
                )

        if save_video:
            out.write(frame)
        if show_video:
            frame = cv2.resize(frame, (0, 0), fx=0.75, fy=0.75)
            cv2.imshow("frame", frame)
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break
    cap.release()
    if save_video:
        out.release()
    # Close all OpenCV windows
    cv2.destroyAllWindows()
model = YOLO('runs/detect/train/weights/best.pt')
model.fuse()
process_video_with_tracking(model, "test.mp4", show_video=True,
save_video=False, output_video_path="output_video.mp4")
```