СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Во время обучения студентам регулярно приходится писать отчеты к различным видам работ (курсовые, лабораторные, научно-исследовательские работы и т. д.), при этом оформление работ должно соответствовать ГОСТ, что необходимо своевременно проверить и при необходимости отправить отчет на доработку, однако, количество студентов намного превышает количество нормоконтроллеров. Для ускорения процесса проверки возможно использование автоматических систем.

Для достижения цели научно-исследовательской работы требуется решить следующие задачи:

— проанализировать существующие виды PDF-документов и связанных с ними ограничений;

— классифицировать типовые требования и ошибки при оформлении отчётов: текста, рисунков, графиков, схем алгоритмов, таблиц, списка источников и т. д.;

— проанализировать существующие решения и разработать алгоритм выделения составных частей (элементов) отчёта, представленного в формате PDF, в соответствии с ГОСТ 7.32 (фрагменты текста, рисунки, графики, схемы алгоритмов, источники, таблицы и пр.) для дальнейшего анализа с использованием средств компьютерного зрения и автоматического анализа текста;

— проанализировать существующие решения и разработать алгоритм проверки рисунков на соответствие ГОСТ 7.32 и дополнительным требованиям;

— проанализировать существующие решения и разработать алгоритм классификации рисунков по содержанию: графики, схемы алгоритмов, UML-диаграммы, IDEF0, BPMN2.0 и прочие изображения;

— проанализировать существующие решения и разработать алгоритм проверки схемы алгоритма на соответствие ГОСТ 7.32 и дополнительным требованиям;

— проанализировать существующие решения и разработать алгоритм проверки текста и списка используемых источников на соответствие ГОСТ 7.32 и дополнительным требованиям;

— реализовать предложенные алгоритмы в едином ПО для целевой ОС «Astra Linux» и «ROSA Linux».

1 Аналитический раздел

В данной части работы, будет рассмотрена структура PDF файла, основные ошибки студентов, допускаемые при написании отчетов, а также рассмотрены различные виды PDF файлов.

В современном мире одними из самых популярных форматов электронных документов являются формат docx и PDF, ввиду этого именного в данных форматах отчеты представляются на проверку [1]. Формат файлов PDF имеет статус международного стандарта (закреплён ISO/IEC 32000-1:2008) [2], различные типы формата PDF поддержиываются различными международными стандартами (например PDF/A-1 поддерживается стандартом ISO 19005-1, PDF/E стандартом ISO 24517 и т. д.) [3; 4].

1.1 Структура PDF файла

Для получения и анализа данных из файла формата PDF необходимо рассмотреть его структуру.

1.1.1 Разделы PDF файла

Структура PDF файла включает 4 раздела:

1) заголовок;

2) тело;

3) таблица перекрестных ссылок;

4) хвост [5].

Заголовок

Заголовком называется первая строка файла, он содержит информацию о версии PDF [5], пример: `%PDF-1.5`.

Тело

Все содержимое документа находится в теле файла. Информация, которая отображается пользователю представлена восемью типами данных:

1) булевы значения. Принимают значения true или false;

2) числа. integer (целочисленный) и real (вещественный), дробная часть в вещественных числах отделяется точкой;

3) имена. Последовательность ASCII символов, начинаются со слеша, который не входит в имя, вместо непосредственно символов могут включать их шестнадцатеричные коды, начинающиеся с символа #;

4) строки. Ограничены длиной в 65535 байтов, записываются в круглых либо треугольных скобках, могут быть представлены как ASCII символами, так и шестнадцатеричными или восьмеричными кодами.

5) массивы. Могут содержать любые PDF-объекты, элементы разделяются пробелом и заключаются в квадратные скобки;

6) словари. Представляют коллекцию пар ключ-значение. Ключом должно быть имя, а значением может быть любой объект. Запись словаря начинается с символов «, а заканчиваются — »;

7) потоки. Потоки содержат неограниченные последовательности байтов. В них содержится основное содержимое документов. Поток начинается с ключевого слова `stream` и заканчивается словом `endstream`. Перед началом потока записывается словарь с мета-информацией, включающей данные о количестве байтов, фильтре применимом их к обработке и т. д.

8) null-объекты. Представляются ключевым словом `null` [5].

PDF объектом является любой вышеперечисленный тип, содержащий информацию [5].

Хвост

Данный раздел начинается с ключевого слова `trailer` и содержит несколько значений.

1) Словарь, содержащий:

   — данные о количестве объектов (ключевое слово `Size`);

   — ссылки на каталог документа (ключевое слово `Root`);

   — информационный словарь (ключевое слово `Info`);

— идентификатор файла (ключевое слово `ID`) [5].

2) Смещение относительно таблицы перекрестных ссылок (англ. cross-reference table);

3) Маркер конца файла `%%EOF`.

Пример «хвоста» PDF файла приведен в листинге А.2.

Таблица перекрестных ссылок

Cross-reference table состоит из нескольких секций, каждая из которых соответствует новой версии документа, и позволяет получать произвольный доступ к любому объекту в файле, данная таблица начинается с ключевого слова `xref`, так что иногда ее называют xref таблицей [6].

Листинг 1.1 – Пример таблицы перекрестных ссылок

```
xref
0 44
0000000000 65535 f
0000000361 00000 n
0000000257 00000 n
0000000015 00000 n
```

Любой PDF-объект может быть помечен уникальным идентификатором и использоваться как ссылка, называемые косвенными, данные объекты начинаются с идентификатора, номера поколения и ключевого слова `obj`, а заканчиваются словом `endobj`. На такие объекты можно ссылаться в таблице cross-reference table и любом другом объекте (для этого используется символ R) [6].

1.1.2 Хранение страниц

С точки зрения организации страниц PDF документ имеет иерархическую структуру (дерево), корнем которого является словарь Catalog, содержащий ссылку на дерево страниц и ссылку на словарь ссылок на иные элементы документа. Каждый элемент дерева страниц хранит в себе всю информацию, отображаемую на странице: выравнивание страницы, картинки, текст и т. д., данный тип хранения позволяет ускорить доступ к объектам на странице [5; 7].

1.2 Виды PDF форматов

Существует несколько различных видов PDF документов, каждый из которых имеет свои особенности и ограничения:

1) PDF/A;
2) PDF/X;
3) PDF/E;
4) PDF/UA.

1.2.1 PDF/A

Предназначен для долгосрочного хранения документов. Он обеспечивает сохранность и неприкосновенность содержимого. Данный формат также разделяется на несколько подклассов: PDF/A-1, PDF/A-2, PDF/A-3, PDF/A-4 [8].

Также вводится новое понятие уровня соответствия, оно накладывает дополнительные требования на классы PDF/A, для предоставления дополнительных возможностей.

1) Уровень b (Basic). Цель: обеспечение надёжного воспроизведения внешнего вида документа. Распространяется на файлы формата: PDF/A-1b, PDF/A-2b, PDF/A-3b;
2) уровень a (Accessible). Цель: обеспечение возможности поиска и преобразования содержимого документа. Включает все требования уровня b и дополнительно требует, чтобы была включена структура документа. Также вводит требования:
    1) Содержимое должно быть помечено деревом иерархической структуры, что означает, что такие элементы, как порядок чтения, рисунки и таблицы, явно идентифицируются с помощью метаданных.
    2) Должен быть указан естественный язык документа.
    3) Изображения и символы должны иметь альтернативный описательный текст. Файл должен включать сопоставление символов с Unicode.

Распространяется на файлы формата: PDF/A-1a, PDF/A-2a, PDF/A-3a;

3) уровень u (Unicode). Распространяется на файлы формата: PDF/A-2u, PDF/A-3u. Требуется сопоставление символов с Unicode. Изменения: отбрасываются требования уровня a, включая встроенную логическую структуру (т. е. теги и дерево структур);

4) уровень f (Format). Распространяется на файлы формата: PDF/A-4f. Изменения: позволяет встраивать типы файлов любого другого формата;

5) уровень e (Engineering). Распространяется на файлы формата: PDF/A-4e. Изменения: поддержка аннотаций типов RichMedia и 3D [8].

PDF/A-1

PDF/A-1 самый распространенный формат оригинального PDF/A на сегодняшний день. Он основан на PDF 1.4 и является наиболее ограниченным, так как не поддерживает JPEG 2000, вложения, слои и прозрачность. Часть 1 стандарта была опубликована 28 сентября 2005 года и определяет два уровня соответствия для файлов PDF: PDF/A-1b и PDF/A-1a [9].

PDF/A-2

PDF/A-2 дополняет A-1 новыми функциями:

1) поддерживает сжатие JPEG2000, что особенно полезно для отсканированных документов (карты, книги), а также документов с цветным содержимым (чеки или паспорта);

2) вложенные файлы PDF/A через коллекции: Acrobat позволяет пользователям создавать коллекции (иногда также называемые «портфелями»), где несколько документов PDF/A объединяются в один «контейнерный» документ PDF;

3) поддерживает необязательное содержимое (слои): необязательное содержимое, иногда также называемое слоями, полезно для приложений картографии или инженерных чертежей, где отдельные слои могут быть показаны или скрыты в соответствии с требованиями просмотра;

4) поддерживает новый уровень соответствия PDF/A-2u — «u» для Unicode, упрощающий поиск и копирование текста Unicode для цифровых PDF-документов и PDF-документов, которые были отсканированы с последующим оптическим распознаванием символов (OCR) [9];

5) поддерживает хранение метаданных на уровне объекта XMP: PDF/A-2 определяет требования к настраиваемым метаданным XMP;

PDF/A-3

PDF/A-3 расширяет PDF/A-2, поддерживая добавления любых файлов, а не только PDF типа A, однако не гарантирует валидность их прочтения в будущем [9].

Также адаптирован для использования в электронном документообороте [10].

PDF/A-4

Основное отличие данного вида, является отказ от уровней соответствия a, b и c. PDF/A-4 требует отображения в юникоде для всех шрифтов [11].

1.2.2 PDF/X

Формат, разработанный специально для обмена и печати документов в издательской отрасли. Он обеспечивает точность цветов и расположения элементов страницы, что особенно важно при печати [12].

1.2.3 PDF/E

Формат, предназначенный для обмена и хранения документов в инженерной отрасли. Он поддерживает вставку трехмерных моделей, векторных изображений и технической документации [12].

1.2.4 PDF/UA

Формат, предназначенный для создания доступных документов для пользователей с ограниченными возможностями. Он обеспечивает структурированное представление контента и поддержку технологий чтения вслух и распознавания речи [12].

1.3 Основные ошибки в отчетах

В данном разделе будут рассмотрены наиболее часто встречающиеся ошибки, которые совершают студенты при написании различных отчетов.

В целях выявления наиболее часто встречающихся ошибок были опрошены преподаватели, работа которых непосредственно связана с проверкой отчетов студентов.

1.3.1 Общие ошибки

В ГОСТ 7.32 указаны следующие размеры полей: левое — 30 мм, правое — 15 мм, верхнее и нижнее — 20 мм [13]. Выход за границы листа является одной из самых распространенных ошибок.

Каждый объект (таблица, рисунок, схема алгоритма, формула) должен быть подписан и пронумерован, однако более подробно подписи к каждому из них будут рассмотрены в следующих подразделах.

Если таблицу или схему не удается разместить на одной странице, то следует разбить данный объект на несколько частей, каждая из которых должна быть подписана.

1.3.2 Ошибки в тексте

Слова в тексте должны быть согласованы в роде, числе и падеже.

Страницы отчета должны быть пронумерованы, однако, номер на титульном листе не ставится, но он является первой страницей, что означает, что следующая страница должна иметь номер 2.

Ненумерованный заголовок (введение, список литературы, оглавление и т. п.) должен быть выравнен по центру, при этом он состоит только из прописных букв (пример представлен в приложении Б.1), другие варианты оформления являются не соответствующими стандарту.

Абзацный отступ должен быть одинаковым по всему тексту отчета и равен 1,25 см [13]. Любые другие варианты оформления считаются ошибочными.

Возможна потеря научного стиля и переход к публицистике, что является ошибкой, текст работы должен быть написан на государственном языке в научном стиле.

1.3.3 Ошибки в рисунках

Частой ошибкой является неправильное оформление рисунков. Каждый рисунок должен быть подписан, при этом подпись должна располагаться строго по центру, внизу рисунка. Другое оформление считается ошибочным.

Использование рисунков низкого разрешения является ошибкой. Все рисунки должны быть выполнены в высоком качестве, если обратное не требуется в самой работе.

Некорректный поворот рисунка считается ошибкой. Если рисунок не удается разместить на странице, то допускается повернуть его таким образом, чтобы верх рисунка был ближе к левой части страницы (см. рисунок Б.2).

Ошибки в графиках

Для каждого графика должна существовать легенда, для оформления которой существует два варианта:

— в одном из углов графика находится область, в которой указаны все обозначения;

— в подписи к графику описано каждое обозначение;

другое оформление является ошибкой.

Часто на графиках отсутствуют единицы измерения, что является ошибкой. Должны быть подписаны единицы измерения каждой из осей графика, даже в том случае, если на графике оси подписываются словами, например, если измерение идет в штуках или на оси обозначены времена года (см. рисунок Б.3).

Отчеты могут быть напечатаны в черно-белом варианте, поэтому на графиках должны быть маркеры, которые позволят отличить графики друг от друга даже не в цветом варианте. Отсутствие маркеров считается ошибкой.

При большом количестве графиков на одном рисунке возможна ситуация, при которой невозможно отличить один график от другого, что является ошибкой.

Ошибки в схемах алгоритмов

Если схему не удается разместить на одной странице, то она разбивается на несколько частей, каждая из которых должна быть подписана. Для разделения схемы алгоритма на части используется специальный символ-соединитель, который отображает выход в часть схемы и вход из другой части этой схемы, соответствующие символы-соединители должны содержать одно и то же уникальное обозначение, любые другие варианты оформления являются ошибочными.

Часто вместо символа начала или конца алгоритма используют овал, однако в этом случае должен быть использован прямоугольник с закругленными углами (см. рисунок Б.4).

При использовании символа процесса (прямоугольник) часто используют прямоугольник с закругленными углами (см. рисунок Б.5), что является ошибкой.

При соединении символов схемы алгоритмов не нужны стрелки, если они соединяют символы в направлении слево-направо или сверху-вниз, в остальных случаях символы должны соединяться линиями со стрелкой на конце, отсутствие требуемых стрелок считается ошибкой.

При использовании символа процесса-решение как минимум одна из соединительных линий должна быть подписана (см. рисунок Б.6), однако возможен также вариант, когда подписаны обе линии. Отсутствие пояснений к выходам данного символа является ошибкой.

Часто пояснительный текст пересекается с символами, использующимися для составления схем, что является ошибкой.

1.3.4 Ошибки в таблицах

Каждая таблица должна быть подписана. Наименование следует помещать над таблицей слева, без абзацного отступа в следующем формате: Таблица Номер таблицы - Наименование таблицы. Наименование таблицы приводят с прописной буквы без точки в конце [13]. Другие варианты оформления считаются не соответствующими стандарту.

Таблицу с большим количеством строк допускается переносить на другую страницу. При переносе части таблицы на другую страницу слово «Таб-

лица», ее номер и наименование указывают один раз слева над первой частью таблицы, а над другими частями также слева пишут слова «Продолжение таблицы» и указывают номер таблицы [13]. Любое другое оформление считается ошибочным.

1.3.5 Ошибки в формулах

Каждая формула должна быть пронумерована вне зависимости от того, существует ли ссылка на нее. Нумерация может осуществляться в двух вариантах:

— сквозная нумерация (номер формулы не зависит от раздела, в котором она находится);

— нумерация, зависящая от раздела (в том случае номер формулы начинается с номера раздела);

другое оформление считается ошибкой.

Отсутствие знака препинания после формулы является ошибкой. После каждой формулы должен находиться знак препинания (точка, запятая и т. п.), зависящий от контекста. Если в формуле содержится система уравнений, то после каждого из них (за исключением последнего) ставится запятая, а после последнего — точка, либо запятая (см. рисунок Б.7).

Номер формулы должен быть выравнен по правому краю страницы и находиться по центру формулы (в вертикальной плоскости). Другое оформление нумерации формул считается не соответствующим стандарту.

Если формула вставляется в начале страницы, то часто перед ней может присутствовать отступ, которого быть не должно.

1.3.6 Ошибки в списках

Ненумерованные списки должны начинаться с удлиненного тире (см. рисунок Б.8), другое оформление является ошибочным.

В нумерованных списках после номера пункта обязательно должна стоять скобка (см. рисунок Б.10), использование другого знака считается ошибкой.

В конце каждого пункта списка должен быть знак препинания, от которого зависит первая буква первого слова следующего пункта (см. рисунок Б.9):

— если пункт заканчивается на точку, то первое слово следующего пункта должно начинаться на прописную букву;

— если пункт заканчивается запятой или точкой с запятой, то следующий первое слово следующего слово должно начинаться со строчной буквы;

другое оформление является ошибочным.

1.3.7 Ошибки в списке литературы

Часто при описании одного из источников не указывается одна из составных частей (автор, издательство и т. п.), что является ошибкой.

Также нередко встречаются ссылки на так называемые «препринтовские» издательства (статья еще не вышла), однако была использована в отчете, это считается ошибкой.

1.4 Библиотеки по работе с PDF-файлами

В данной части работы будут сравниваться существующие Python-библиотеки для извлечения данных из PDF-файлов, охватывая из возможности с точки зрения извлечения текста, изображений и таблиц, скорости выполнения и обширности функциональности.

1.4.1 PyPDF2

PyPDF2 - это библиотека на чистом Python, которая позволяет читать PDF-файлы и манипулировать ими. Хотя она в основном ориентирована на извлечение текста, она также предоставляет ограниченную поддержку для извлечения изображений. Однако извлечение таблиц не является встроенной функцией. PyPDF2 получил широкое распространение благодаря небольшой, но достаточной, функциональности и обширной документации.

1.4.2 pdfminer.six

pdfminer.six - это поддерживаемая сообществом библиотека Python, основанная на оригинальном проекте PDFMiner. Она предлагает расширенные возможности для извлечения текста из PDF-файлов, включая возможность извлекать информацию о макете текста. Однако она не обеспечивает прямой поддержки извлечения изображений или таблиц. pdfminer.six известен своей

точностью при извлечении текста, так как была специально разработана для его извлечения из PDF-файлов.

1.4.3 PyMuPDF

PyMuPDF - это привязка Python для библиотеки MuPDF, которая известна своими высокопроизводительными возможностями рендеринга и синтаксического анализа. PyMuPDF предлагает обширные возможности для извлечения как текста, так и изображений из PDF-файлов. Хотя он не обеспечивает встроенного извлечения таблиц, он обеспечивает прочную основу для реализации пользовательских алгоритмов извлечения таблиц. PyMuPDF полностью документирован и предоставляет богатый набор функциональных возможностей.

2 Конструкторский раздел

В неавтоматизированной системе проверки отчетов на соответствие ГОСТ и дополнительным требованиям присутствуют две роли: студент, выполняющий некоторую работы, которая подразумевает написание отчета и нормоконтроллер, принимающий экспертное решение о соответствии предоставленного ему отчета необходимым требованиям.

Использование автоматической проверки отчетов на соответствие ГОСТ и дополнительным требованиям сократит временные затраты на проверку отчетов.

2.1 Описание системы автоматической проверки отчета на соответствие ГОСТ и дополнительным требованиям

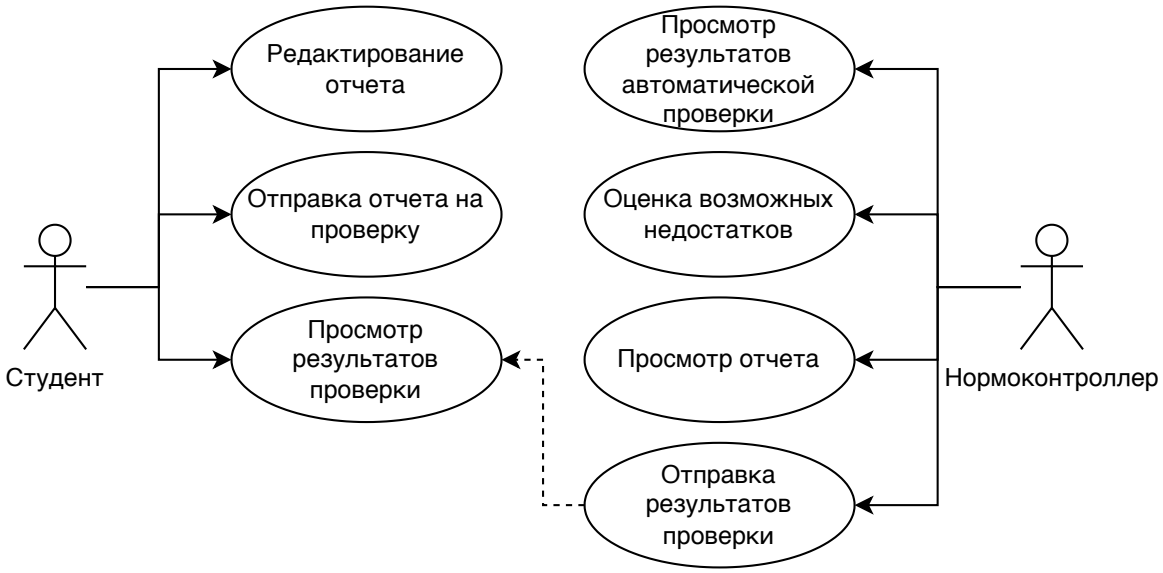


Рисунок 2.1 – Диаграмма вариантов автоматической проверки отчета

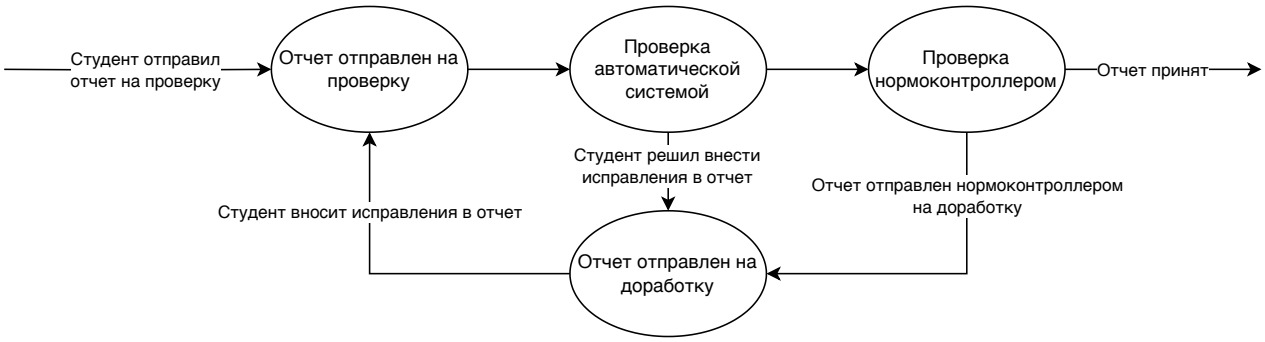


Рисунок 2.2 – Диаграмма состояний проверки отчета

С помощью использования автоматической проверки отчета возможно сократить временные ресурсы, выделяемые нормоконтроллером на проверку отчетов студентов, однако, полностью отказаться от финального контроля результатов человеком невозможно, таким образом существует две роли при проверки отчета на соответствие ГОСТ, а именно: студент и нормоконтроллер.

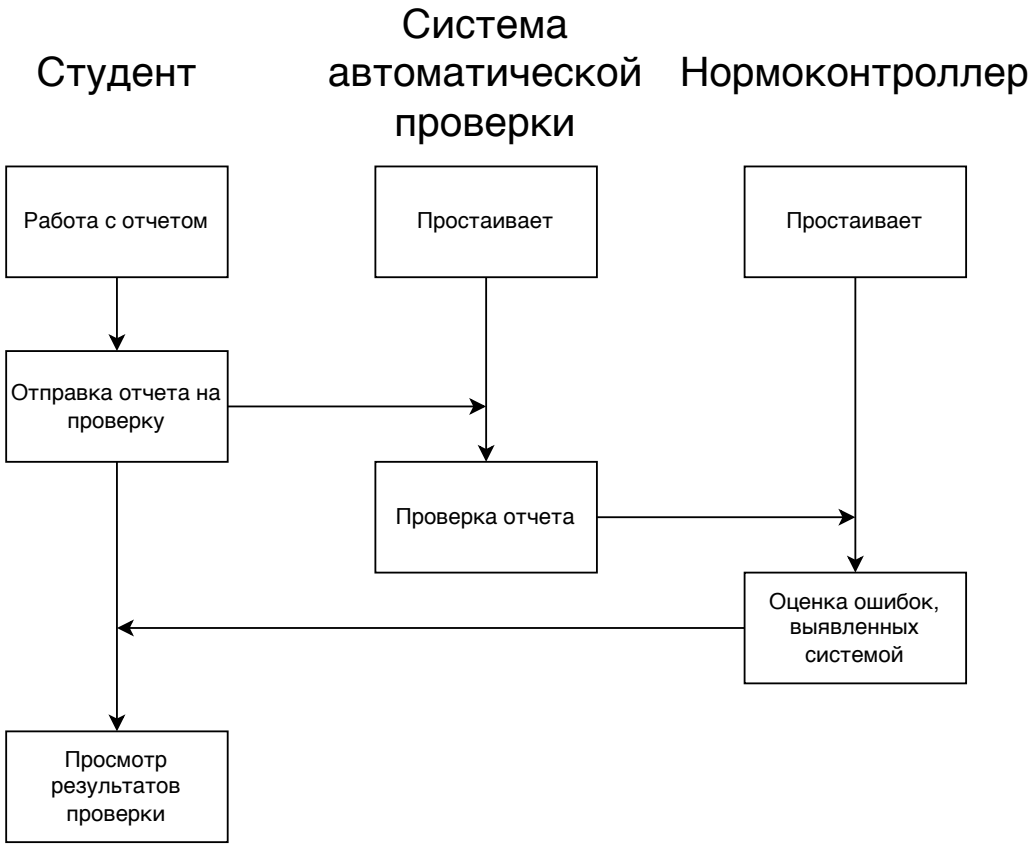


Рисунок 2.3 – Диаграмма последовательности действий

Студент отправляет отчет на проверку, а затем получает результат со списком ошибок (если имеются). Нормоконтроллер же анализирует отчет, составленный автоматической системой проверки, и при необходимости может внести необходимые правки.

Использование автоматической проверки отчетов на соответствие ГОСТ позволяет эффективнее использовать временные ресурсы нормоконтроллера.

2.2 Разработка автоматической системы проверки отчетов

Ввиду отсутствия функции получения векторных изображений из pdf файла в рассмотренных ранее парсерах, использование поиска по шаблону не является возможным.

Системе автоматической проверки необходимо проверить на правильность составные части отчета, что подразумевает детекцию рисунков, графиков, схем алгоритмов, списка используемых источников, а также формул.

На рисунках 2.4–2.5 представлены результаты разработки системы детекции составных частей.

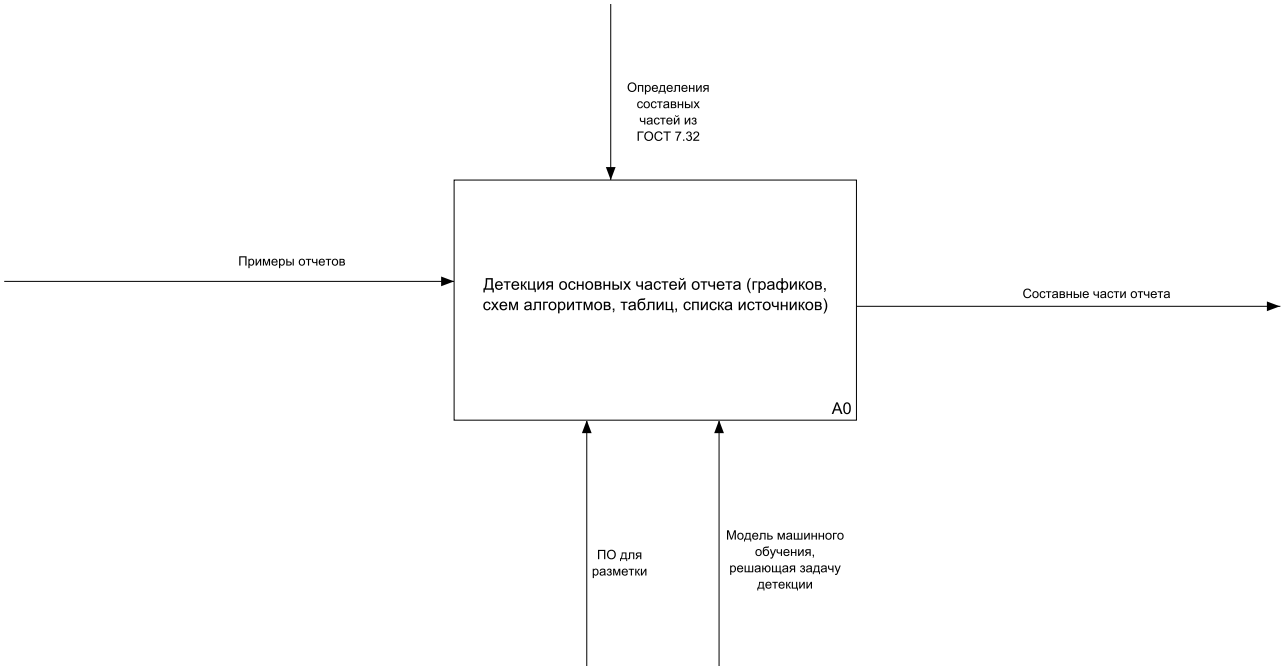


Рисунок 2.4 – IDEF0 обнаружение составных частей отчет 0 уровня

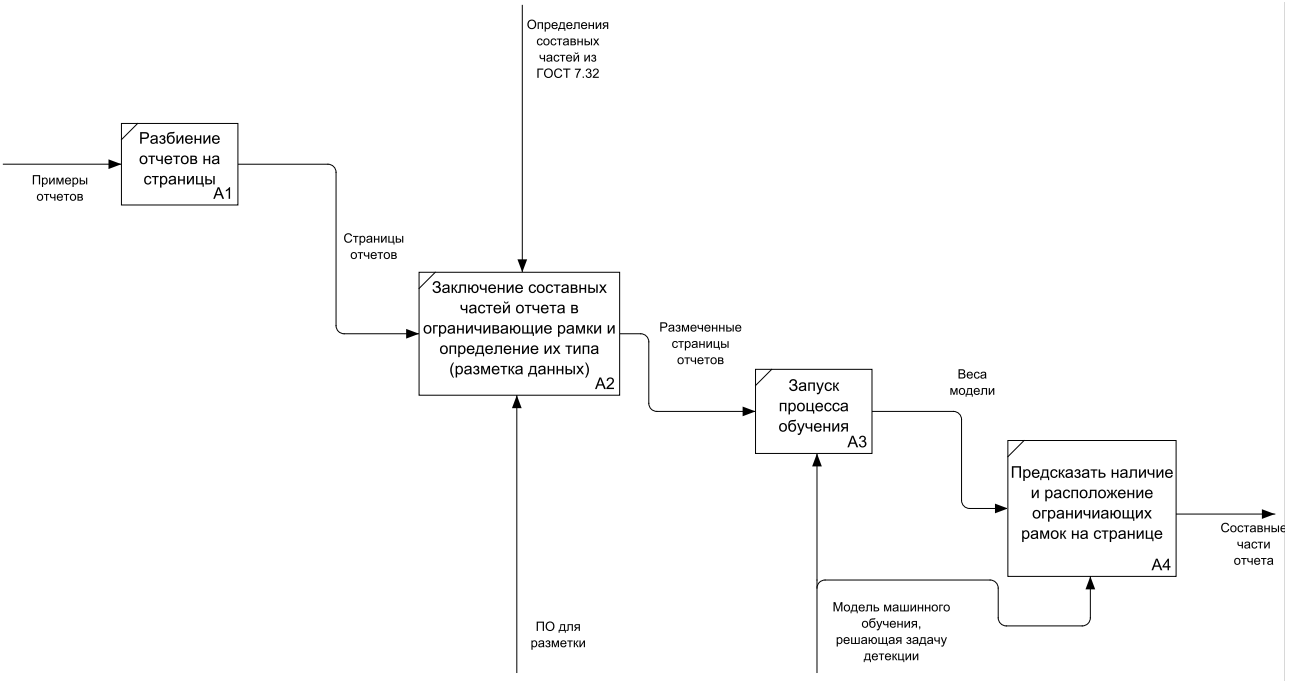


Рисунок 2.5 – IDEF0 обнаружение составных частей отчета 1 уровня

После решения задачи детекции необходимо проверить составные части

на соответствие ГОСТ 7.32, однако, финальный вердикт должен выноситься экспертом. На рисунках 2.6–2.7 представлены результаты разработки данной системы.

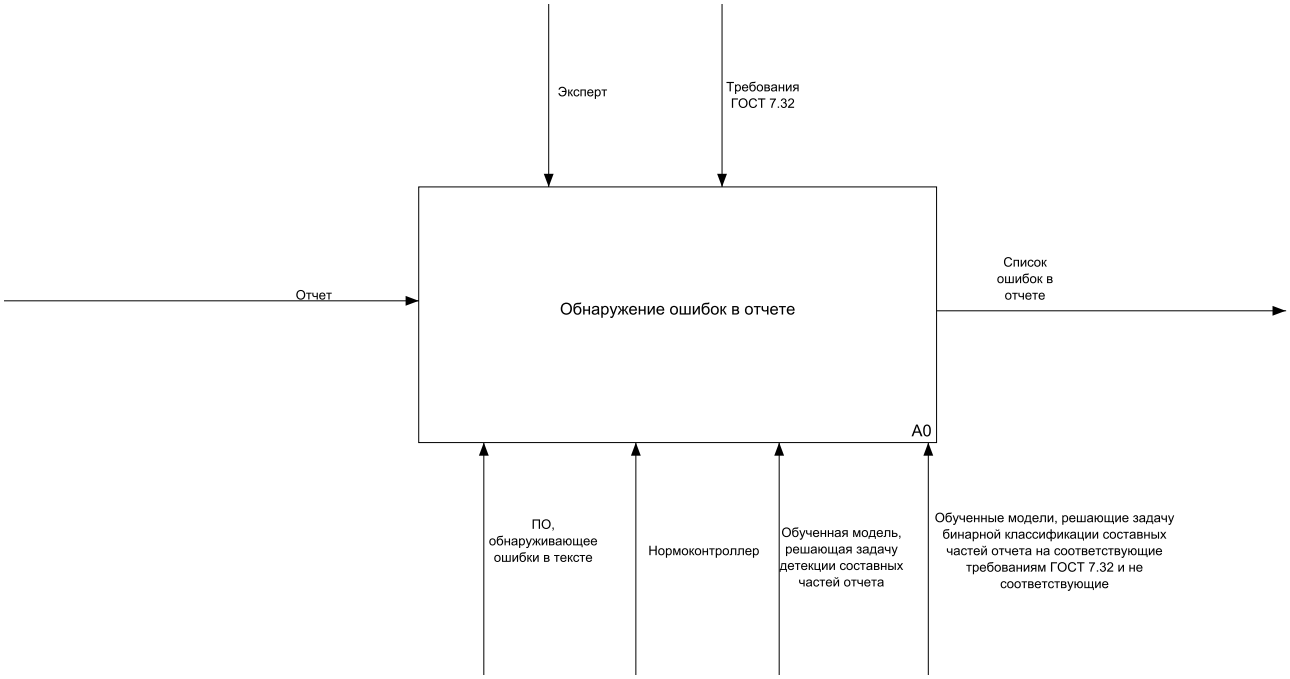


Рисунок 2.6 – IDEF0 проверки составных частей отчета на соответствие ГОСТ 7.32 0 уровня

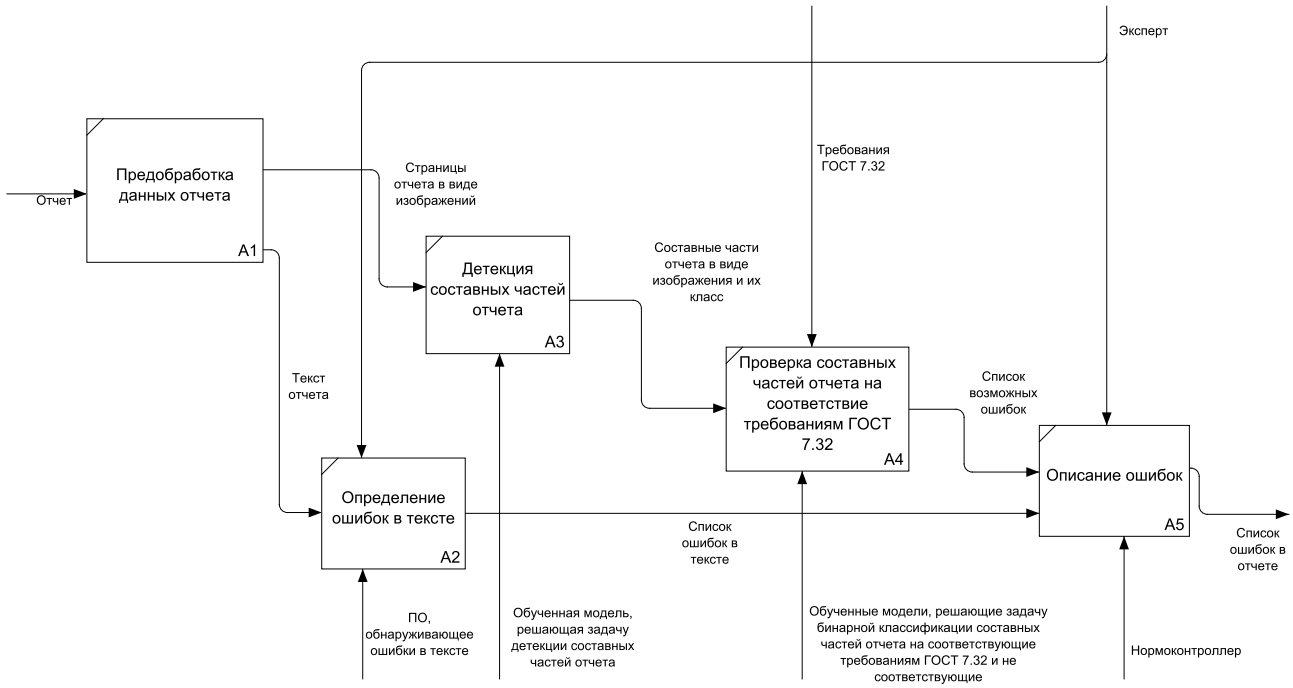


Рисунок 2.7 – IDEF0 проверки составных частей отчета на соответствие ГОСТ 7.32 1 уровня

Вывод

В данном разделе была описана система автоматической проверки отчетов студентов на соответствие ГОСТ и дополнительным требованиям, а также спроектировано программное обеспечение, выполняющее проверку отчета.

3 Технологический раздел

В данной части работы, будут приведены инструменты, использующиеся для реализации спроектированных систем и оценки их эффективности.

3.1 Средства реализации

Ввиду необходимости использования разрабатываемых систем на ОС «Astra Linux», рассматриваются только инструменты, работа которых возможна на данной ОС.

3.1.1 OpenCV

OpenCV — это библиотека программного обеспечения для компьютерного зрения и машинного обучения с открытым исходным кодом.

Библиотека содержит более 2500 оптимизированных алгоритмов, которые включают в себя полный набор как классических, так и самых современных алгоритмов компьютерного зрения и машинного обучения. Эти алгоритмы могут быть использованы для обнаружения и распознавания лиц, идентификации объектов, классификации действий человека в видео, отслеживания движений камеры, отслеживания движущихся объектов, поиска похожих изображений из база данных изображений, распознавание пейзажа и т. д. [14].

Данная библиотека реализуют следующий функционал:

1) поиск по шаблону (англ. template matching), данная функция позволяет находить на изображении большего размера шаблон меньшего размера и выделять его, данная функция упростит поиск геометрических примитивов на изображении [15];

2) классификация изображений из модуля глубоких нейронных сетей (англ. dense neural networks module) позволит разбивать изображения на необходимые подклассы [16].

3) Благодаря оптическому распознаванию текста (англ. optical character recognition) возможно определение местоположения и получение информации о содержании текста [17].

3.1.2 YOLO

Популярная модель обнаружения объектов и сегментации изображений YOLO (англ. необходимо посмотреть один раз) была разработана Джозефом Редмоном и Али Фархади из Вашингтонского университета. YOLOv8, использующаяся в данной работе является эволюцией серии моделей YOLO [18].

1) Модель YOLOv2, выпущенная в 2016 г., была усовершенствована за счет использования пакетной нормализации, якорных блоков и размерных кластеров [19].

2) YOLOv3, выпущенная в 2018 году, позволила еще больше повысить производительность модели за счет использования более эффективной опорной сети, множества якорей и объединения пространственных пирамид [20].

3) YOLOv4, выпущенная в 2020 году, обзавелась такими инновациями, как увеличение данных Mosaic, и «голова», работающая без якорей и новая функция потерь [21].

4) YOLOv5 позволила еще больше повысить производительность модели, в этой версии быди добавлены такие новые возможности, как оптимизация гиперпараметров, интегрированное отслеживание обучения [22].

5) YOLOv6 была открыта компанией Meituan в 2022 году и используется во многих автономных роботах-доставщиках компании [23].

6) В YOLOv7 добавлены дополнительные задачи, такие как оценка позы по набору данных COCO keypoints [24].

7) YOLOv8 — это последняя версия YOLO на сегодняшний день от Ultralytics, поддерживающая обнаружение, сегментацию, оценку положения, отслеживание и классификацию [18].

Для решения задачи детекции изображений также существуют альтернативные модели [25]: GroundingDINO, Faster R-CNN.

GroundingDINO — модель, использующая трансформеры и двухшаговый подход, заключающийся в извлечении визуальных и текстовых представлений, а затем выравнивании этих представлений.

Faster R-CNN — представляет собой метод обнаружения объектов, объединяющий конвейер из двух основных компонентов: сети глубокого обучения для извлечения признаков и регионального генератора для предложения областей, предположительно содержащих объекты.

3.2 Метрики

Для оценки успешности работы модели были рассмотрены метрики:

1) средняя усредненная точность (англ. mean average precision, сокращенно mAP);
2) точность (англ. precision);
3) отзыв (англ. recall).

При решении задачи детекции, необходимо также решить задачу классификации, для оценки успешности классификации изображений была использованы метрики precision и recall, для оценки точности выделения нужных объектов используется метрика mAP.

3.2.1 Точность

Точность для данного класса в многоклассовой классификации — это доля экземпляров, правильно классифицированных как принадлежащие к определенному классу, из всех экземпляров, которые модель предсказала как принадлежащие к этому классу [26].

3.2.2 Отзыв

Отзыв в многоклассовой классификации — это доля экземпляров в классе, которые модель правильно классифицировала, из всех экземпляров в этом классе [26].

3.2.3 Усреднение

Так как классификация многоклассовая, в случае YOLOv8 приведенные выше метрики рассчитываются отдельно для каждого класса, после усредняются, используя макроусреднение (англ. macro-averaging) [18]. При ис-

пользовании данного метода вычисляется среднее метрик по каждому классу [26]. Данная метрика рассчитывается согласно следующим формулам:

$$Precision = \frac{Precision_{ClassA} + Precision_{ClassB} + ... + Precision_{ClassN}}{N} \quad (3.1)$$

$$Recall = \frac{Recall_{ClassA} + Recall_{ClassB} + ... + Recall_{ClassN}}{N} \quad (3.2)$$

3.2.4 Средняя усредненная точность

При решении задач детекции, необходимо заключить требуемый объект в ограничивающую рамку (англ. bounding box). Для поиска требуемого значения вводится еще одна метрика «пересечение перед объединением» (англ. intersection over union, сокращенно iou), для ее подсчета необходимо разделить площадь пересечения (англ. area of overlap), предсказанных и размеченных ограничивающих рамок на площадь их объеденения (англ. area of union):

$$Intersection\ over\ Union\ (iou) = \frac{Area\ of\ Overlap}{Area\ of\ Union}. \quad (3.3)$$

После введения «пересечения перед объединением» (англ. iou) вводится нижний «порог» значений этой метрики, значение данного порога может быть любым. В случае, если предсказанная ограничивающая рамка имеет значение «пересечения перед объединением» с размеченной рамкой меньше, чем «порог», то объект относится к неверно определенным значениям (англ. false positive), иначе относится к верно определенным значениям (англ. true positive), стоит также уточнить, что на одном изображении может быть предсказано несколько объектов, при этом предсказанные ограничивающие рамки сортируются по «уверенности» модели для данного результата. После разделения изображений на неверно определенные и верно определенные значения, вычисляются описанные ранее метрики точность и отзыв. После чего усредненная точность вычисляется подсчетом площади под кривой точность-отзыв. Например, на рисунке 3.1 метрика среднего значения (англ. average precision, сокращенно AP) будет рассчитана по формуле (3.4) [27].

$$AP = \frac{1}{11} \sum_{Recall_i} Precision(Recall_i) = \frac{1}{11} \cdot (1 \cdot 6) + (0 \cdot 5) = 0.545 \qquad (3.4)$$

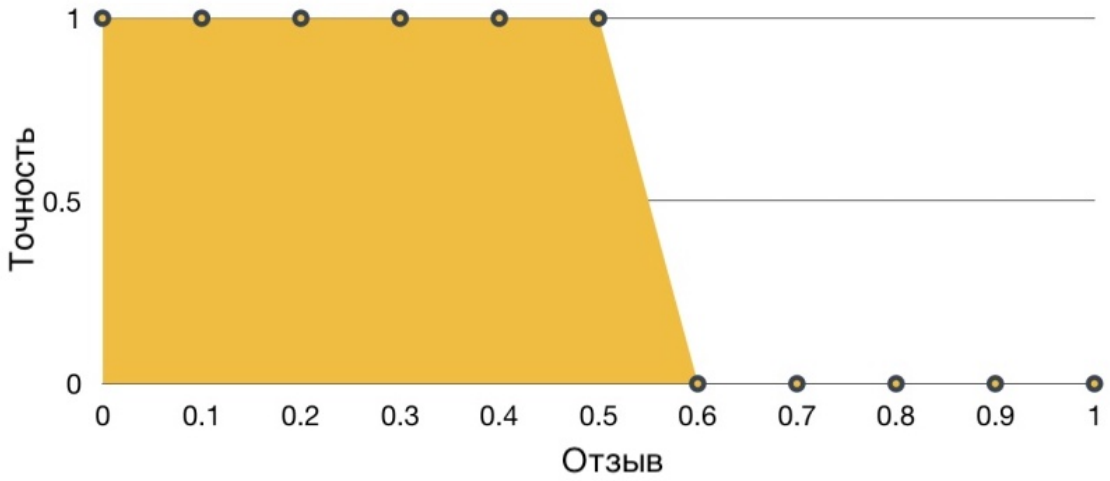


Рисунок 3.1 – Примера расчета AP

После получения метрики AP для каждого класса, значения AP усредняются, результатом усреднения AP по всем классам является требуемое значение mAP.

3.3 Этапы обучения модели

Процесс обучения модели состоит из нескольких этапов [28]:

1) разметка данных;
2) предобработка данных;
3) обучение модели;
4) анализ результатов обучения модели;

Разметка данных

Перед запуском процесса обучения необходимо соотнести данные с требуемыми значениями с рамках поставленной задачи. В случае задачи классификации необходимо сопоставить каждый объект одному из классов. На данном этапе необходимо собрать максимальное количество данных для наиболее эффективного обучения модели [28].

Предобработка данных

На данном этапе необходимо привести данные в соответствующий для модели формат. Также производится анализ количества классов, и собранный набор данных разделяется на тренировочную выборку и валидационную выборку. На тренировочной выборке модель будет «обучаться», с помощью валидационной выборки можно оценить точность работы модели. Также на данном этапе определяются метрики для оценки качества модели [28].

Обучение модели

На тренировочной выборке запускается процесс обучения модели, с определенными гиперпараметрами. Модели обучаются несколько эпох, каждая эпоха — полный «проход» модели по тренировочной выборке в процессе обучения [29].

Анализ результатов обучения модели

После окончания процесса обучения, рассматриваются значения выделенных метрик и делаются выводы о точности работы модели, принимается решение о выборе иной модели или других значений гиперпараметров [28].

3.4 Использование Astra Linux

Astra Linux позволяет использовать библиотеки глубокого обучения, такие как «TensorFlow» и «Keras» [30]. Данные библиотеки позволяют решать задачу детекции изображений с помощью модели YOLOv8, что дает возможность реализовать обучение на данной операционной системе [31].

4 Исследовательский раздел

В данном разделе будут рассмотрены методы классификации и проверки выбранных объектов документа на валидность.

4.1 Анализ изображений

Для анализа изображений (таблиц, схем, списка информационных ресурсов) необходимо получить их представление из отчета. Так как изображения могут быть представлены в векторном формате, то необходимо решать задачу детекции изображений.

4.1.1 Использование соответствия по шаблону

Предположение: наличие отличительных объектов на картинке, не встречающихся в других (например, оси для графиков) позволит классифицировать объект. Для поиска объектов используется поиск по шаблону [15].

Однако объект, представленный на изображении, может находиться в любом положении и под любым наклоном, таким образом для поиска соответствия необходимо рассматривать все возможные повороты шаблона. Например при необходимости поиска изображения стрелки в изображении 4.1 с использованием шаблона 4.2. При использовании метода **CV-TM-CCOEFF-NORMED** — корреляция Пирсона, результаты представлены на изображении 4.3, перед использованием шаблона изображение было переведено в вид оттенков серого (один канал). Было найдено только одно изображение стрелки, без учета ее поворота, что не позволяет использовать данный метод при всевозможных ее поворотах.

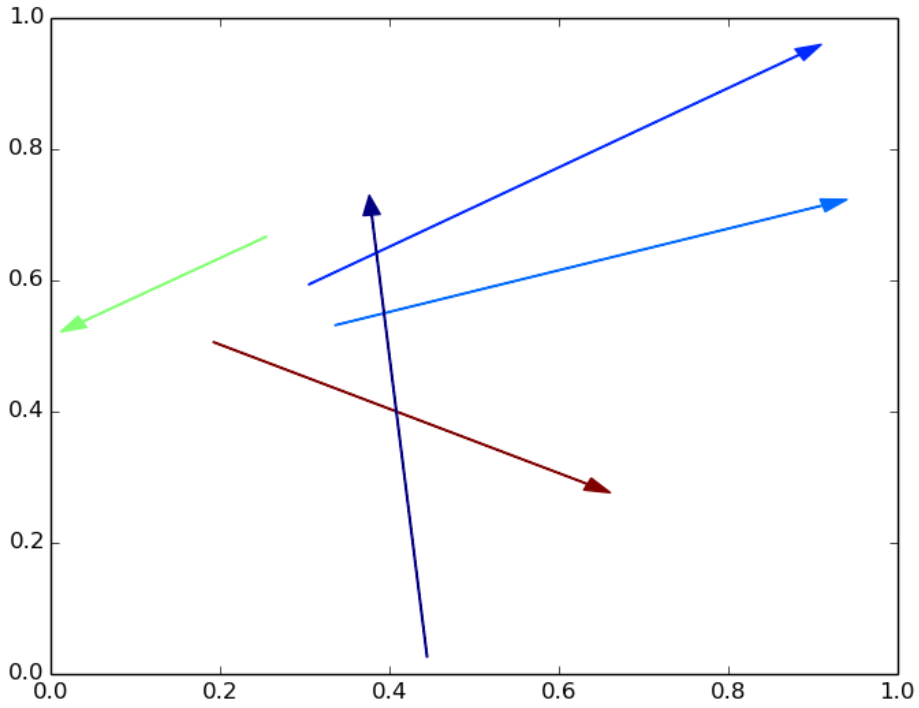


Рисунок 4.1 – Пример изображения для поиска шаблона

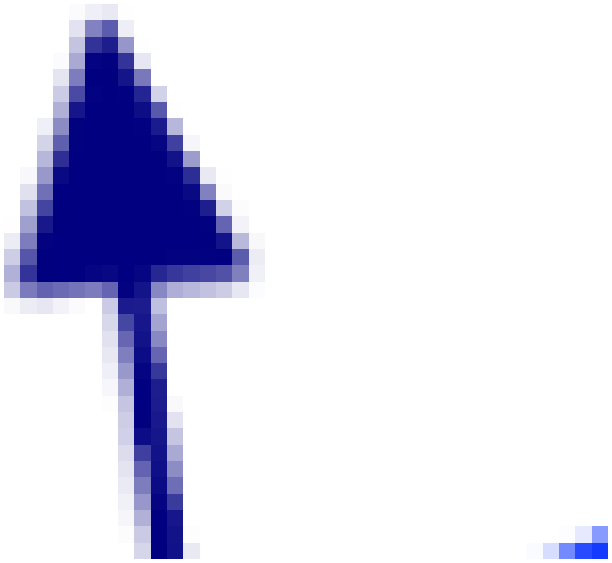


Рисунок 4.2 – Шаблон изображения для поиска

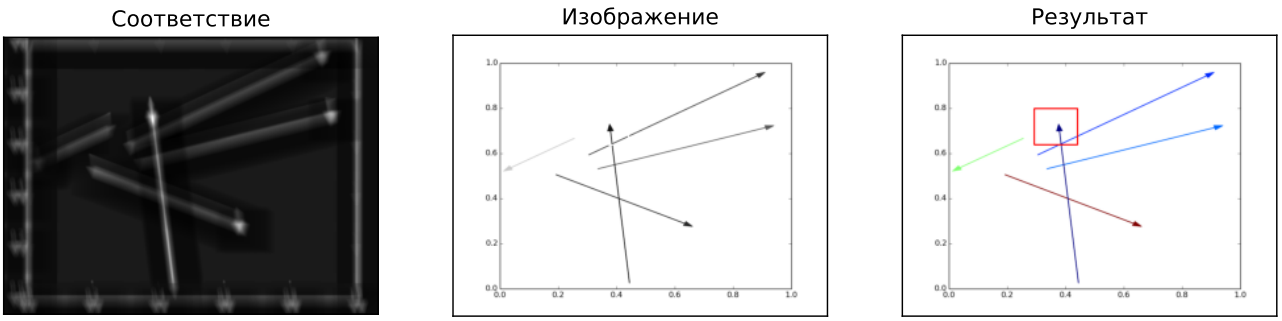


Рисунок 4.3 – Результаты применения шаблона

4.1.2 Использование YOLOv8 для детекциии изображений

Для детекции изображений была использована модель YOLOv8 [18]. Для разметки изображений был использован labelimg [32]. Данные для разметки были взяты из отчетов студентов по предмету «Анализ Алгоритмов». Было выделено 5 классов изображений:

1) формулы (имеют метку eq);
2) схемы (имеют метку scheme);
3) таблицы (имеют метку table);
4) графики (имеют метку graph);
5) списки информационных ресурсов (имеют метку lit);

Обучение на 10 эпохах

На 267 изображениях была обучена модель YOLOv8, с гиперпараметрами обучения $iou = 0.5$, $conf = 0.001$ на 10 эпохах, 30 изображений было выделено в валидационную выборку. Результаты полученных изображений приведены на рисунке 4.4. На рисунках 4.5–4.7, представлены метрики после обучения на 10 эпохах, после каждой эпохи метрики вычислялись на валидационной выборке, число 50 после mAP означает порог iou в 50 процентов. Значения метрик на валидационной выборке из 30 изображений при 10 эпохах обучения:

1) $precision = 0.217$;

2) $recall = 0.216$;

3) $mAP = 0.284$.

Результаты работы данной модели на валидационной выборке представлены на картинке B.1.

Рисунок 4.4 – Результаты использования модели при обучении на 10 эпохах

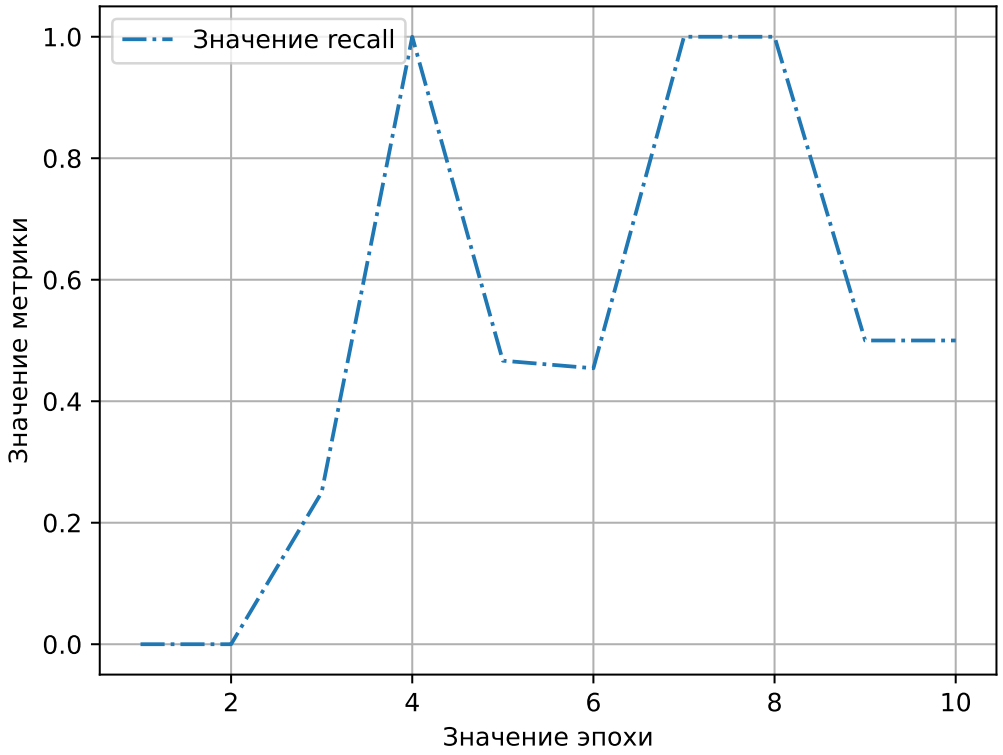


Рисунок 4.5 – Значение метрики recall при обучении на 10 эпохах

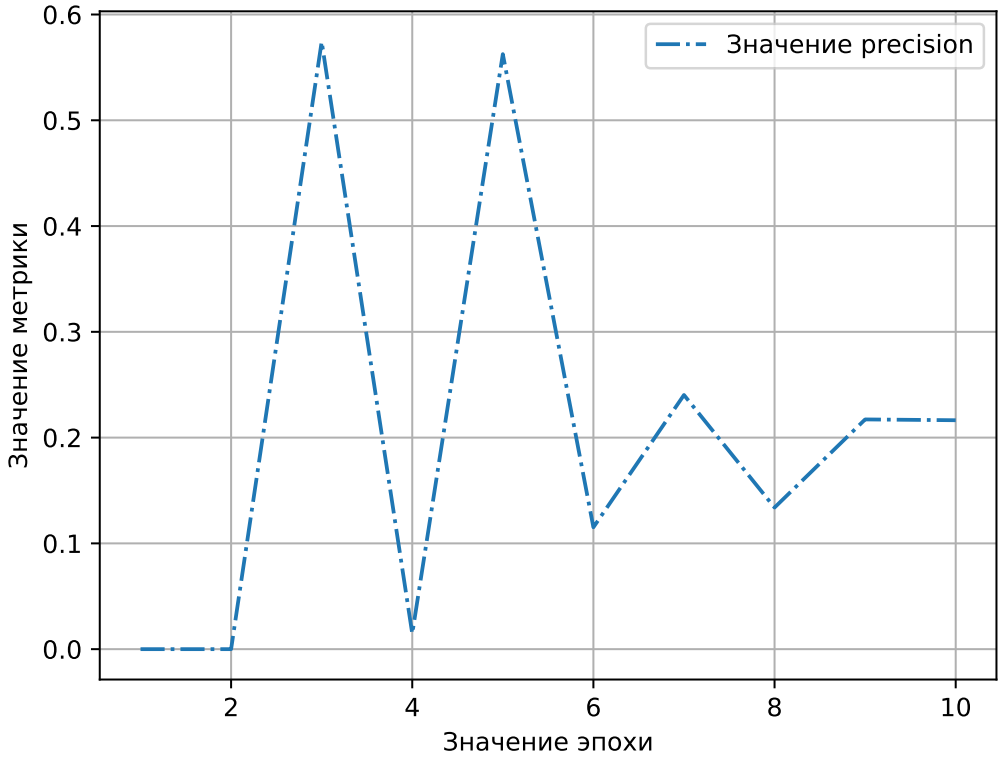


Рисунок 4.6 – Значение метрики precision при обучении на 10 эпохах

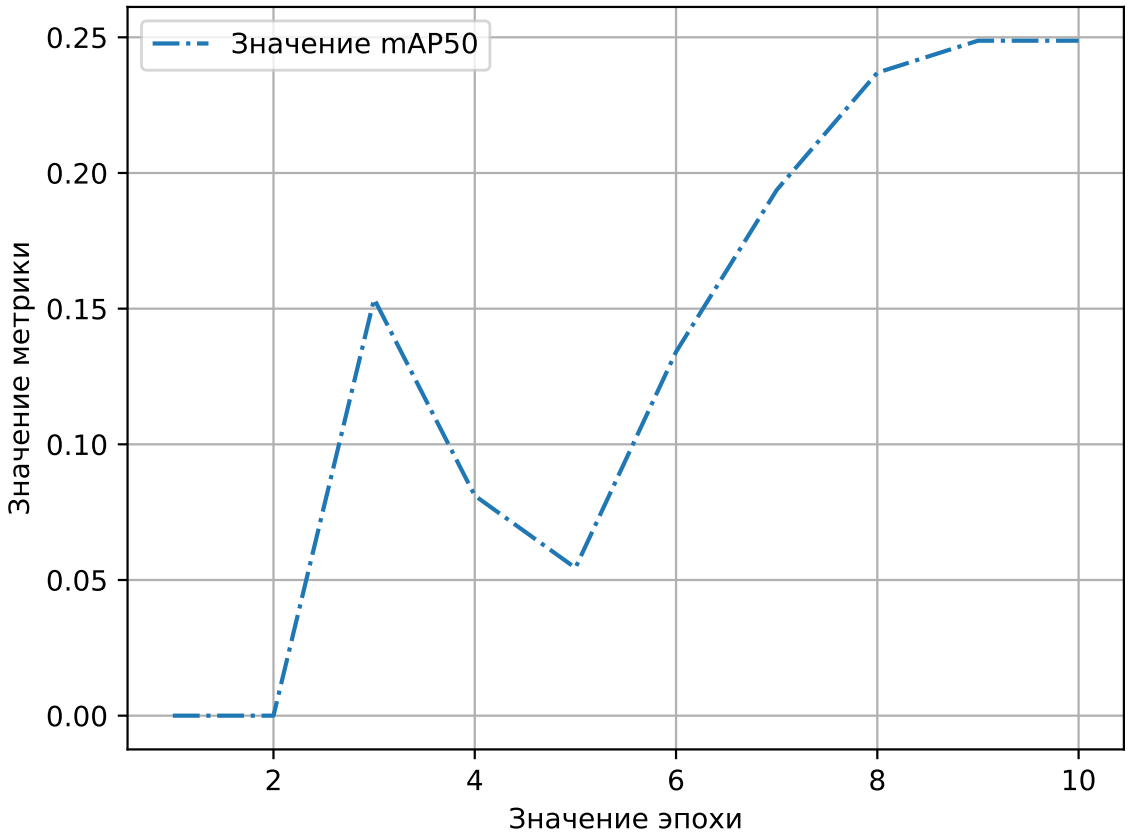


Рисунок 4.7 – Значение метрики mAP50 при обучении на 10 эпохах

Обучение на 100 эпохах

Также была поптыка обучения на 100 эпохах без изменения гиперпараметров, однако обучение было остановлено на 73 эпохе оптимизатором yolov8, так как предсказания модели не улучшились за последние 50 эпох, наилучшие результаты предсказаний были получены на 23 эпохе. Результаты работы данной модели на валидационной выборке представлены на рисунке В.2. Результаты полученных изображений приведены на рисунке 4.8. На рисунках 4.9–4.11, представлены метрики после обучения на 73 эпохах. Значения метрик на валидационной выборке из 30 изображений при 23 эпохах обучения:

1) $precision = 0.364$;

2) $recall = 0.9603$;

3) $mAP = 0.74$.

Actual image

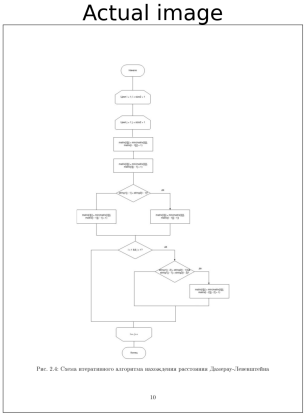
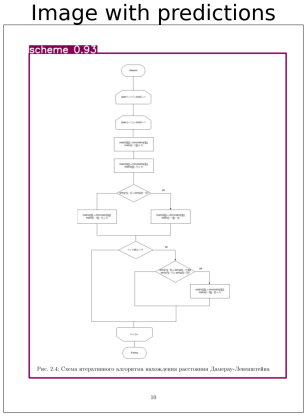


Image with predictions



Actual image

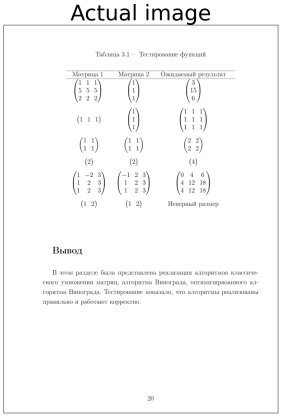


Таблица 3.1 – Тестирование функций

Матрица 1	Матрица 2	Ожидаемый результат
$\begin{pmatrix} 1 & 1 & 1 \\ 5 & 5 & 5 \\ 2 & 2 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 3 \\ 15 \\ 6 \end{pmatrix}$
$\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$
$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$
(2)	(2)	(4)
$\begin{pmatrix} 1 & -2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} -1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} 0 & 4 & 6 \\ 4 & 12 & 18 \\ 4 & 12 & 18 \end{pmatrix}$
(1 2)	(1 2)	Неверный размер

Вывод

В этом разделе были представлены реализации алгоритмов классического умножения матриц, алгоритма Винограда, оптимизированного алгоритма Винограда. Тестирование показало, что алгоритмы реализованы правильно и работают корректно.

Image with predictions

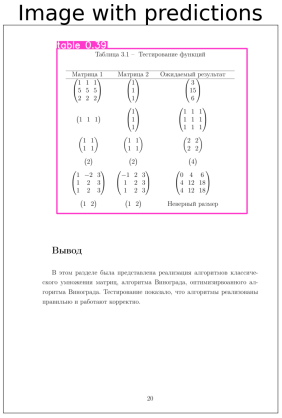


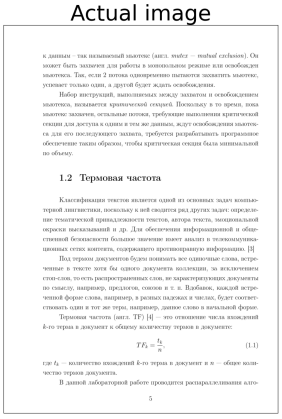
Таблица 3.1 – Тестирование функций

Матрица 1	Матрица 2	Ожидаемый результат
$\begin{pmatrix} 1 & 1 & 1 \\ 5 & 5 & 5 \\ 2 & 2 & 2 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 3 \\ 15 \\ 6 \end{pmatrix}$
$\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$
$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}$
(2)	(2)	(4)
$\begin{pmatrix} 1 & -2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} -1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$	$\begin{pmatrix} 0 & 4 & 6 \\ 4 & 12 & 18 \\ 4 & 12 & 18 \end{pmatrix}$
(1 2)	(1 2)	Неверный размер

Вывод

В этом разделе была представлена реализация алгоритмов классического умножения матриц, алгоритма Винограда, оптимизированного алгоритма Винограда. Тестирование показало, что алгоритмы реализованы правильно и работают корректно.

Actual image



к данным – так называемый мьютекс (англ. *mutex* – *mutual exclusion*). Он может быть захвачен для работы в монопольном режиме или освобожден мьютекс. Так, если 2 потока одновременно пытаются захватить мьютекс, успевает только один, а другой будет ждать освобождения.

Набор инструкций, выполняемых между захватом и освобождением мьютекса, называется критической секцией. Поскольку в то время, пока мьютекс захвачен, остальные потоки, требующие выполнения критической секции для доступа к одним и тем же данным, ждут освобождения мьютекса для его последующего захвата, требуется разрабатывать программное обеспечение таким образом, чтобы критическая секция была минимальной по объему.

1.2 Термовая частота

Классификация текстов является одной из основных задач компьютерной лингвистики, поскольку к ней сводится ряд других задач: определение тематической принадлежности текстов, автора текста, эмоциональной окраски высказываний и др. Для обеспечения информационной и общественной безопасности большое значение имеет анализ в телекоммуникационных сетях контента, содержащего противоправную информацию. [3]

Под термом документов будем понимать все одиночные слова, встреченные в тексте хотя бы одного документа коллекции, за исключением стоп-слов, то есть распространенных слов, не характеризующих документы по смыслу, например, предлогов, союзов и т. п. Вдобавок, каждой встреченной форме слова, например, в разных падежах и числах, будет соответствовать один и тот же терм, например, данное слово в начальной форме.

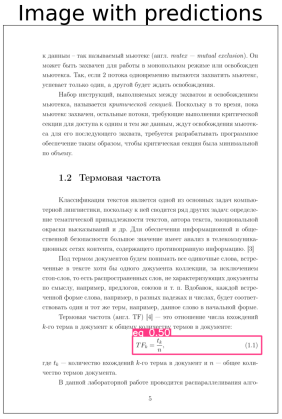
Термовая частота (англ. TF) [4] — это отношение числа вхождений k-го терма в документ к общему количеству термов в документе:

$$TF_k = \frac{t_k}{n}, \tag{1.1}$$

где t_k — количество вхождений k-го терма в документ и n — общее количество термов документа.

В данной лабораторной работе проводится распараллеливание алго-

Image with predictions



к данным – так называемый мьютекс (англ. *mutex* – *mutual exclusion*). Он может быть захвачен для работы в монопольном режиме или освобожден мьютекса. Так, если 2 потока одновременно пытаются захватить мьютекс, успевает только один, а другой будет ждать освобождения.

Набор инструкций, выполняемых между захватом и освобождением мьютекса, называется критической секцией. Поскольку в то время, пока мьютекс захвачен, остальные потоки, требующие выполнения критической секции для доступа к одним и тем же данным, ждут освобождения мьютекса для его последующего захвата, требуется разрабатывать программное обеспечение таким образом, чтобы критическая секция была минимальной по объему.

1.2 Термовая частота

Классификация текстов является одной из основных задач компьютерной лингвистики, поскольку к ней сводится ряд других задач: определение тематической принадлежности текстов, автора текста, эмоциональной окраски высказываний и др. Для обеспечения информационной и общественной безопасности большое значение имеет анализ в телекоммуникационных сетях контента, содержащего противоправную информацию. [3]

Под термом документов будем понимать все одиночные слова, встреченные в тексте хотя бы одного документа коллекции, за исключением стоп-слов, то есть распространенных слов, не характеризующих документы по смыслу, например, предлогов, союзов и т. п. Вдобавок, каждой встреченной форме слова, например, в разных падежах и числах, будет соответствовать один и тот же терм, например, данное слово в начальной форме.

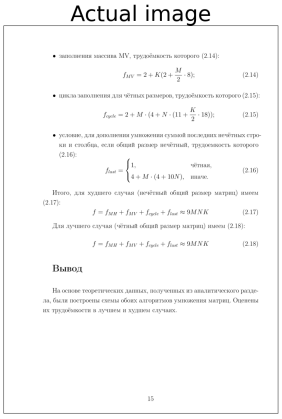
Термовая частота (англ. TF) [4] — это отношение числа вхождений k-го терма в документ к общему количеству термов в документе:

$$TF_k = \frac{t_k}{n}, \tag{1.1}$$

где t_k — количество вхождений k-го терма в документ и n — общее количество термов документа.

В данной лабораторной работе проводится распараллеливание алго-

Actual image



- заполнения массива MV, трудоёмкость которого (2.14):

$$f_{MV} = 2 + K(2 + \frac{M}{2} \cdot 8); \tag{2.14}$$

- цикла заполнения для чётных размеров, трудоёмкость которого (2.15):

$$f_{opiz} = 2 + M \cdot (4 + N \cdot (11 + \frac{K}{2} \cdot 18)); \tag{2.15}$$

- условие, для дополнения умножения суммой последних нечётных строк и столбца, если общий размер нечётный, трудоемкость которого (2.16):

$$f_{nat} = \begin{cases} 1, & \text{чётная,} \\ 4 + M \cdot (4 + 10N), & \text{иначе.} \end{cases} \tag{2.16}$$

Итого, для худшего случая (нечётный общий размер матрицы) имеем (2.17):

$$f = f_{MM} + f_{MV} + f_{opiz} + f_{nat} \approx 9MNK \tag{2.17}$$

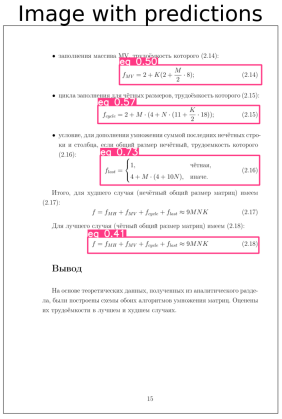
Для лучшего случая (чётный общий размер матрицы) имеем (2.18):

$$f = f_{MM} + f_{MV} + f_{opiz} + f_{nat} \approx 9MNK \tag{2.18}$$

Вывод

На основе теоретических данных, полученных из аналитического раздела, были построены схемы обоих алгоритмов умножения матриц. Оценены их трудоёмкости в лучшем и худшем случаях.

Image with predictions



- заполнения массива MV, трудоёмкость которого (2.14):

$$f_{MV} = 2 + K(2 + \frac{M}{2} \cdot 8); \tag{2.14}$$

- цикла заполнения для чётных размеров, трудоёмкость которого (2.15):

$$f_{opiz} = 2 + M \cdot (4 + N \cdot (11 + \frac{K}{2} \cdot 18)); \tag{2.15}$$

- условие, для дополнения умножения суммой последних нечётных строк и столбца, если общий размер нечётный, трудоемкость которого (2.16):

$$f_{nat} = \begin{cases} 1, & \text{чётная,} \\ 4 + M \cdot (4 + 10N), & \text{иначе.} \end{cases} \tag{2.16}$$

Итого, для худшего случая (нечётный общий размер матрицы) имеем (2.17):

$$f = f_{MM} + f_{MV} + f_{opiz} + f_{nat} \approx 9MNK \tag{2.17}$$

Для лучшего случая (чётный общий размер матрицы) имеем (2.18):

$$f = f_{MM} + f_{MV} + f_{opiz} + f_{nat} \approx 9MNK \tag{2.18}$$

Вывод

На основе теоретических данных, полученных из аналитического раздела, были построены схемы обоих алгоритмов умножения матриц. Оценены их трудоёмкости в лучшем и худшем случаях.

Рисунок 4.8 – Результаты использования модели после обучения на 73 эпохах

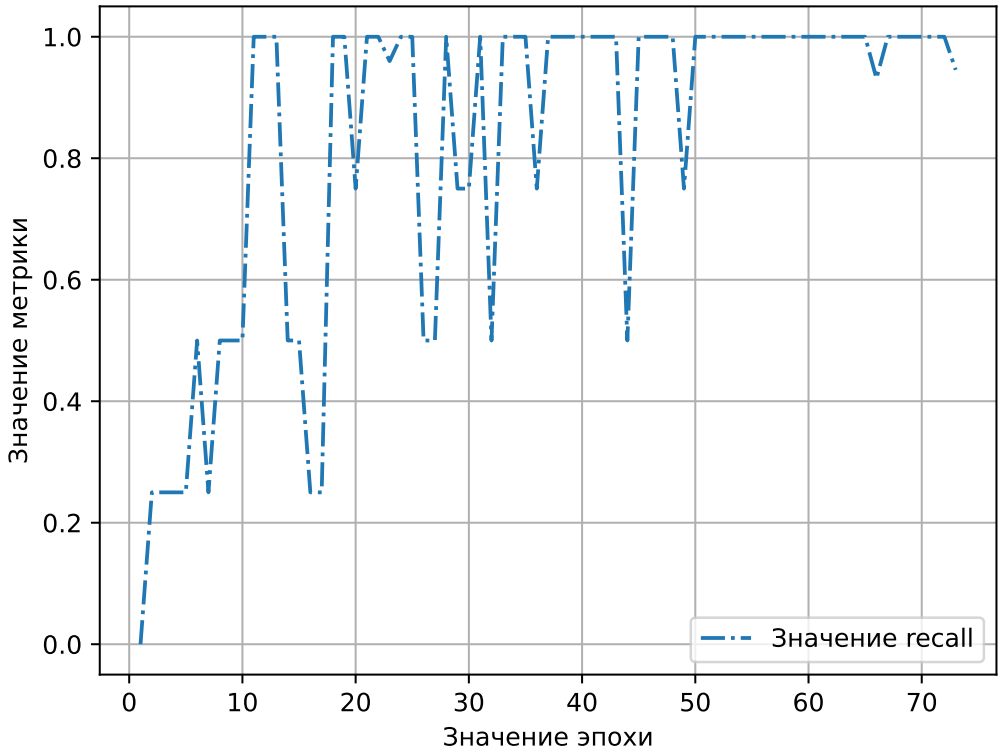


Рисунок 4.9 – Значение метрики recall при обучении на 73 эпохах

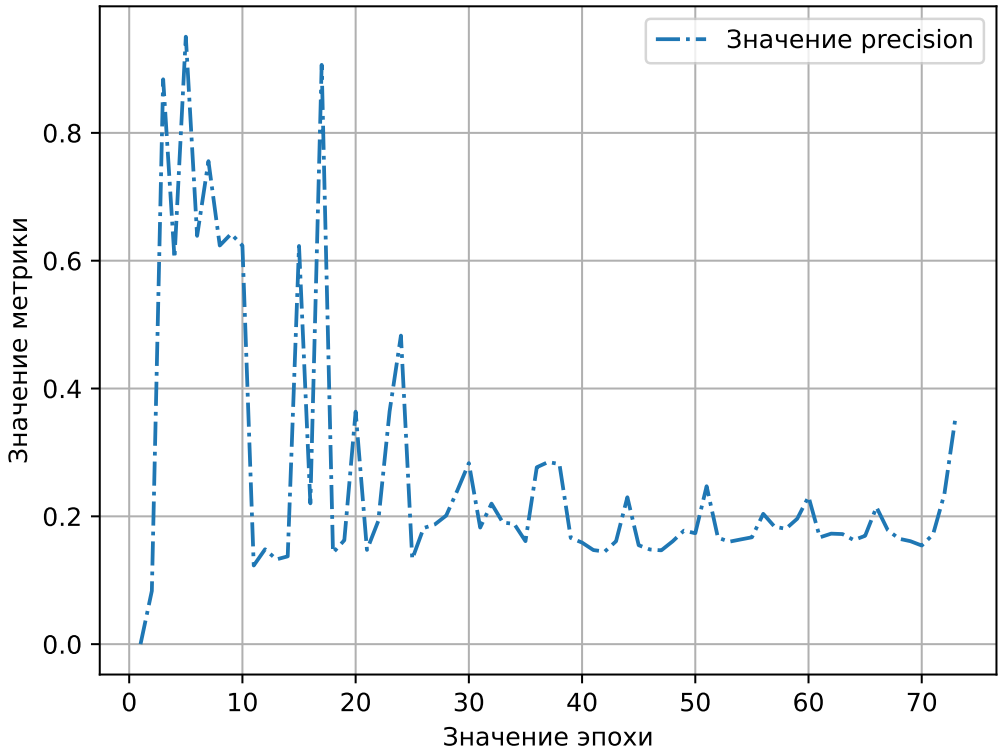


Рисунок 4.10 – Значение метрики precision при обучении на 73 эпохах

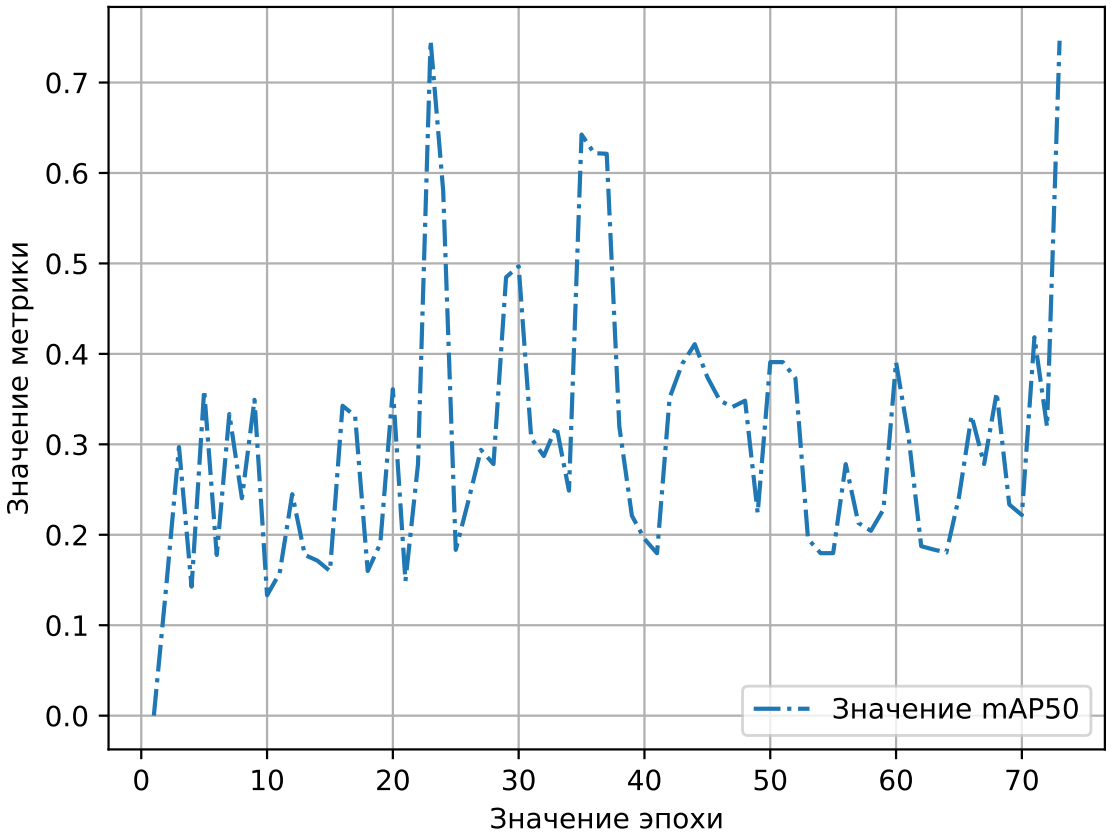


Рисунок 4.11 – Значение метрики mAP50 при обучении на 73 эпохах

Значение метрики *precision* меньше других значений рассматриваемых метрик, модель чаще ошибается при классификации объектов, чем при их выделении (метрика mAP). Для получения более точных результатов классификации необходимо сбалансировать классы (рассматривать одинаковое количество объектов каждого класса) и увеличить их количество.

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы были выполнены следующие задачи:

— проанализированы существующие виды PDF-документов и связанные с ними ограничения;

— классифицированы типовые требования и ошибки при оформлении отчётов: текста, рисунков, графиков, схем алгоритмов, таблиц и списка источников;

— проанализированы существующие решения выделения составных частей (элементов) отчёта, представленного в формате PDF, в соответствии с ГОСТ 7.32 (фрагменты текста, рисунки, графики, схемы алгоритмов, источники, таблицы и пр.) для дальнейшего анализа с использованием средств компьютерного зрения и автоматического анализа текста;

— реализовано программное обеспечение, позволяющее выделить составные части (элементы) отчета, представленного в формате PDF для дальнейшего анализа на соответствие ГОСТ, работа которого возможна на операционной системе «Astra Linux».

В ходе обучение модели YOLOv8 было использовано 297 изображений (267 изображений для обучения и 30 для проверки корректности работы). Для оценки качества работы модели были выбраны следующие метрики:

— precision;

— recall;

— mAP.

Наилучшие результаты были получены при обучении на 23 эпохах, при наблюдении метрик 4.1.2, значение метрики $precision = 0.364$, меньше значений других метрик, можно сделать вывод, что для более точной детекции изображений необходимо увеличить размер валидационной и тренировочной выборок.

Следующим шагом в разработке автоматической проверки отчетов является разработка отдельных алгоритмов проверки на соответствие ГОСТ каждого из выделенных элементов отчета.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Наиболее распространенные форматы электронных документов [Электронный ресурс]. — Режим доступа: https://beorg.ru/blog/formaty-elektronnyh-dokumentov/?ysclid=lq6ke2mpko484311954 (дата обращения: 19.10.2023).
2. ISO 32000-1:2008 [Электронный ресурс]. — Режим доступа: https://www.iso.org/standard/51502.html (дата обращения: 19.10.2023).
3. ISO 32000-1:2008 [Электронный ресурс]. — Режим доступа: https://pdfa.org/resource/iso-24517-pdfe/ (дата обращения: 19.10.2023).
4. ISO 19005 (PDF/A) [Электронный ресурс]. — Режим доступа: https://pdfa.org/resource/iso-19005-pdfa/ (дата обращения: 19.10.2023).
5. What if PDF file? [Электронный ресурс]. — Режим доступа: https://docs.aspose.com/page/net/what-is-pdf-file/ (дата обращения: 26.10.2023).
6. Дружим с PDF [Электронный ресурс]. — Режим доступа: https://alexeykalina.github.io/technologies/pdf.html (дата обращения: 26.10.2023).
7. Chapter 4. Document Structure [Электронный ресурс]. — Режим доступа: https://www.oreilly.com/library/view/pdf-explained/9781449321581/ch04.html (дата обращения: 19.10.2023).
8. Стандарт PDF/A [Электронный ресурс]. — Режим доступа: https://yamadharma.github.io/ru/post/2021/07/30/pdf-a-standard/ (дата обращения: 26.10.2023).
9. PDF/A-2 Overview [Электронный ресурс]. — Режим доступа: https://pdfa.org/wp-content/uploads/2011/10/Flyer-PDFA2-Overview-EN.pdf (дата обращения: 19.10.2023).
10. Приказ ФНС России от 24.03.2022 [Электронный ресурс]. — Режим доступа: https://www.nalog.gov.ru/rn77/about_fts/docs/12181055/ (дата обращения: 19.10.2023).

11. https://blog.avepdf.com/what-is-pdfa4/ [Электронный ресурс]. — Режим доступа: https://blog.avepdf.com/what-is-pdfa4/ (дата обращения: 19.10.2023).

12. PDF File types - specialist formats and what they're used for [Электронный ресурс]. — Режим доступа: https://www.adobe.com/uk/acrobat/resources/document-files/pdf-types.html (дата обращения: 19.10.2023).

13. ГОСТ 7.32—2017 Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления. — М.: Стандартинформ, 2017. — 35 с.

14. About OpenCV [Электронный ресурс]. — Режим доступа: https://opencv.org/about/ (дата обращения: 10.11.2023).

15. Pattern matching [Электронный ресурс]. — Режим доступа: https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html (дата обращения: 10.11.2023).

16. Deep Learning with OpenCV [Электронный ресурс]. — Режим доступа: https://learnopencv.com/deep-learning-with-opencvs-dnn-module-a-definitive-guide/ (дата обращения: 10.11.2023).

17. How to run custom OCR model [Электронный ресурс]. — Режим доступа: https://docs.opencv.org/4.x/d9/d1e/tutorial_dnn_OCR.html/ (дата обращения: 10.11.2023).

18. Ultralytics YOLOv8 Docs [Электронный ресурс]. — Режим доступа: https://docs.ultralytics.com/ (дата обращения: 10.11.2023).

19. PP-YOLOv2: A Practical Object Detector / X. Huang [и др.] // CoRR. — 2021. — Т. abs/2104.10419. — arXiv: 2104.10419. — URL: https://arxiv.org/abs/2104.10419.

20. *Redmon J.*, *Farhadi A.* YOLOv3: An Incremental Improvement // CoRR. — 2018. — Т. abs/1804.02767. — arXiv: 1804.02767. — URL: http://arxiv.org/abs/1804.02767.

21. *Bochkovskiy A.*, *Wang C.*, *Liao H. M.* YOLOv4: Optimal Speed and Accuracy of Object Detection // CoRR. — 2020. — Т. abs/2004.10934. — arXiv: 2004.10934. — URL: https://arxiv.org/abs/2004.10934.

22. Improved YOLOv5 network for real-time multi-scale traffic sign detection / J. Wang [и др.] // CoRR. — 2021. — Т. abs/2112.08782. — arXiv: 2112.08782. — URL: https://arxiv.org/abs/2112.08782.

23. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications / C. Li [и др.]. — 2022. — arXiv: 2209.02976 [cs.CV].

24. *Wang C.-Y.*, *Bochkovskiy A.*, *Liao H.-Y. M.* YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. — 2022. — arXiv: 2207.02696 [cs.CV].

25. Top Object Detection Models [Электронный ресурс]. — Режим доступа: https://roboflow.com/models/object-detection (дата обращения: 27.11.2023).

26. Accuracy, precision, and recall in multi-class classification [Электронный ресурс]. — Режим доступа: https://www.evidentlyai.com/classification-metrics/multi-class-metrics (дата обращения: 27.11.2023).

27. What is Mean Average Precision (MAP) and how does it work [Электронный ресурс]. — Режим доступа: https://xailient.com/blog/what-is-mean-average-precision-and-how-does-it-work/ (дата обращения: 27.11.2023).

28. Machine Learning Model Training: What It Is and Why It's Important [Электронный ресурс]. — Режим доступа: https://domino.ai/blog/what-is-machine-learning-model-training (дата обращения: 27.11.2023).

29. *Komatsuzaki A.* One Epoch Is All You Need. — 2019. — arXiv: 1906.06669 [cs.LG].

30. Добавление библиотек глубокого обучения для разработки программ на языке Python [Электронный ресурс]. — Режим доступа: https://wiki.astralinux.ru/plugins/servlet/mobile?contentId=68912355#content/view/68912355 (дата обращения: 27.11.2023).

31. Object Detection using KerasCV YOLOv8 [Электронный ресурс]. — Режим доступа: https://learnopencv.com/object-detection-using-kerascv-yolov8/ (дата обращения: 27.11.2023).

32. Label Studio [Электронный ресурс]. — Режим доступа: https://github.com/HumanSignal/labelImg (дата обращения: 10.11.2023).

ПРИЛОЖЕНИЕ А

Листинг А.1 – Пример части тела PDF файла

```
/Type /Page
/Contents 169 0 R
/Resources 167 0 R
/MediaBox [0 0 595.276 841.89]
/Parent 93 0 R
/Annots [ 166 0 R ]
>>
endobj
166 0 obj
<<
/Type /Annot
/Subtype /Link
/Border[0 0 0]/H/I/C[0 1 0]
/Rect [119.772 380.481 128.456 396.796]
/A << /S /GoTo /D (cite.0@pdf_levels_std) >>
>>
endobj
170 0 obj
<<
/D [168 0 R /XYZ 84.039 825.051 null]
>>
endobj
25 0 obj
<<
```

Листинг А.2 – Пример «хвоста» PDF файла

```
trailer
<<
/Size 44
/Root 42 0 R
/Info 43 0 R
/ID [<7298F57CACD45F4041F17029C0BBF710>
   <7298F57CACD45F4041F17029C0BBF710>] >>
startxref
11085
\%\%EOF
```

ПРИЛОЖЕНИЕ Б

Введение

Рисунок Б.1 – Пример ошибочного оформления ненумерованного заголовка

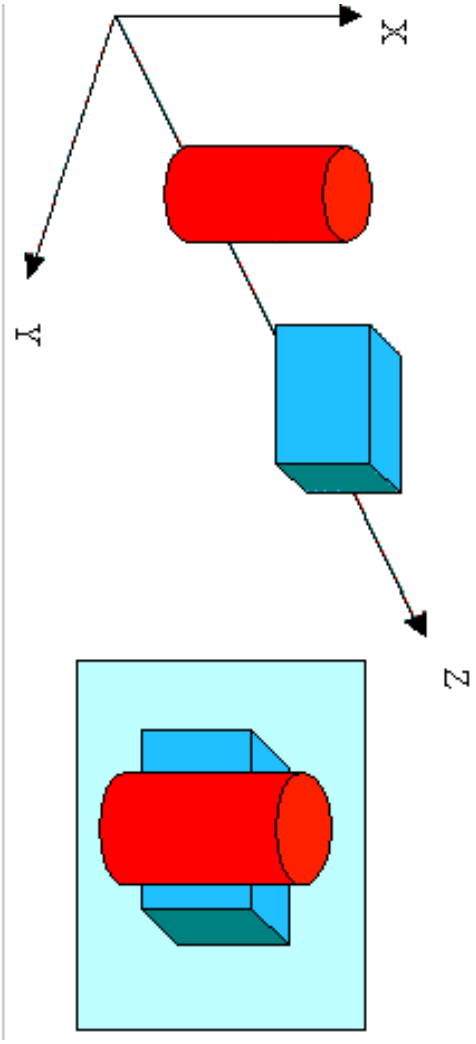


Рисунок 1 - Пример работы Z-буфера

Рисунок Б.2 – Пример ошибочного оформления рисунка — некорректный поворот

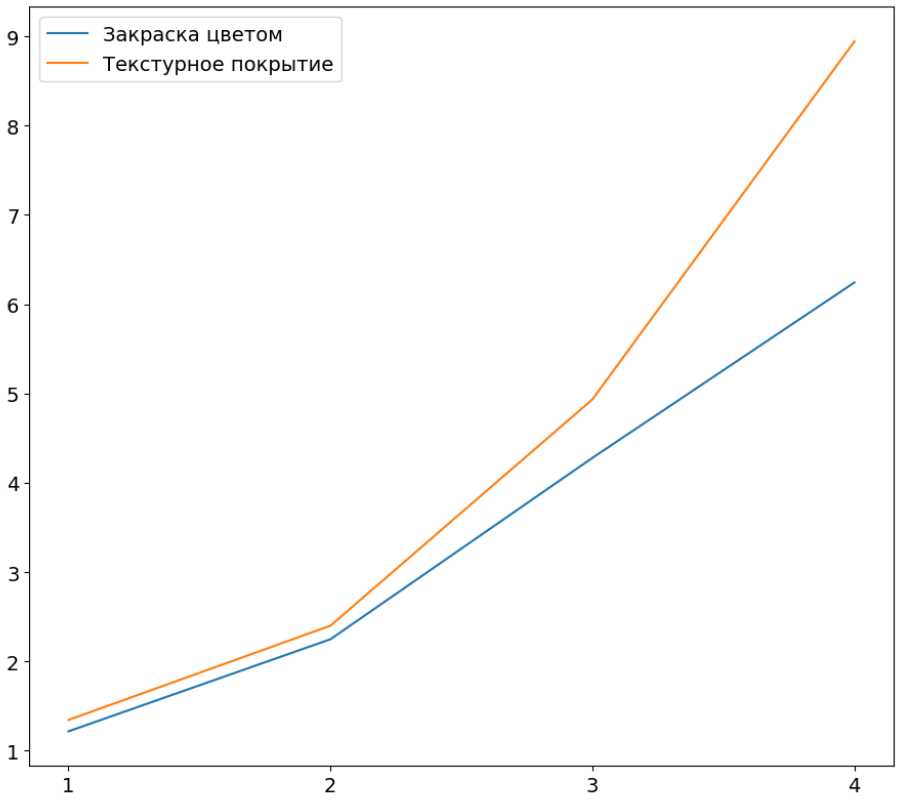


Рисунок Б.3 – Пример ошибочного оформления графика — отсутствуют единицы измерения

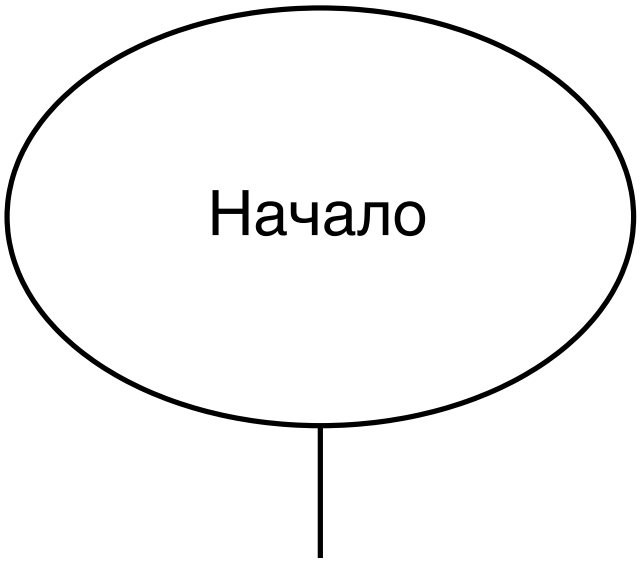


Рисунок Б.4 – Пример ошибочного оформления схемы — некорректный символ начала

Рисунок Б.5 – Пример ошибочного оформления схемы — некорректный символ процесса

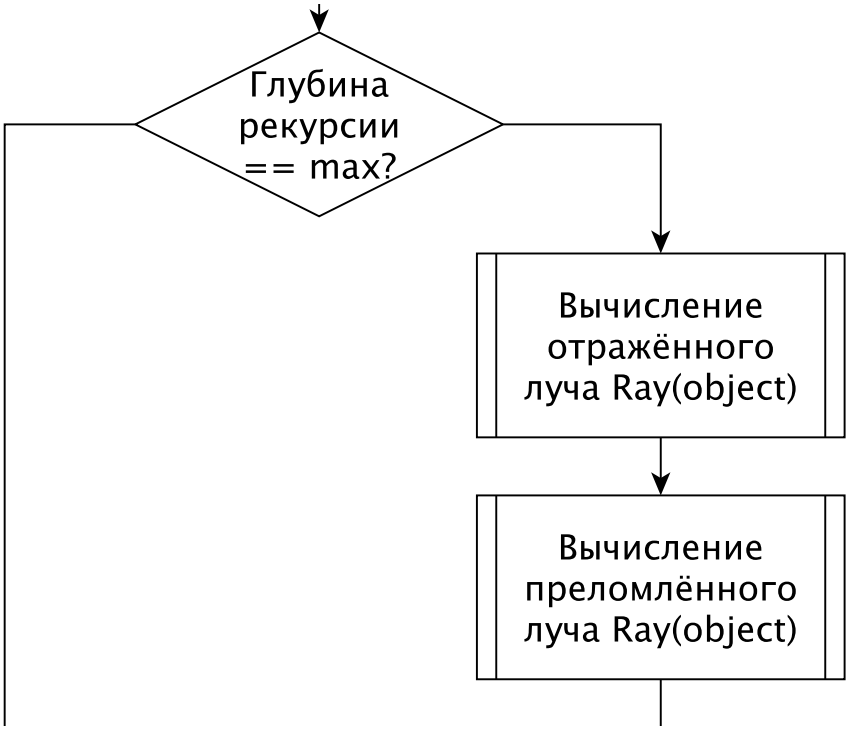


Рисунок Б.6 – Пример ошибочного оформления схемы — не подписана ни одна из веток символа процесса—решение

$$D(i,j) = \begin{cases} 0 & \text{если } i = 0, j = 0 \\ j & \text{если } i = 0, j > 0 \\ i & \text{если } j = 0, i > 0 \\ min(min(D(i,j-1)+1 & \\ \quad D(i-1,j)+1) & \\ \quad D(i-1,j-1) + m(S_1[i], S_2[j]) & \\ \quad \left[\begin{array}{ll} D(i-2,j-2)+1 & \text{если } i > 1, j > 1, \\ & S_1[i-1] == S_2[j-2], \\ & S_1[i-2] == S_2[j-1] \end{array} \right. & \text{иначе} \end{cases} \quad (1)$$

Рисунок Б.7 – Пример ошибочного оформления системы уравнений — отсутствуют знаки препинания после уравнений

- One
- Two
- Three

Рисунок Б.8 – Пример ошибочного оформления ненумерованного списка — некорректный символ перед элементами списка

— Первый,

— Второй,

— Третий.

Рисунок Б.9 – Пример ошибочного оформления ненумерованного списка — некорректный регистр буквы следующего пункта после запятой в предыдущем

1. первый,
2. второй,
3. третий.

Рисунок Б.10 – Пример ошибочного оформления нумерованного списка — некорректный символ после номера элемента списка

ПРИЛОЖЕНИЕ В

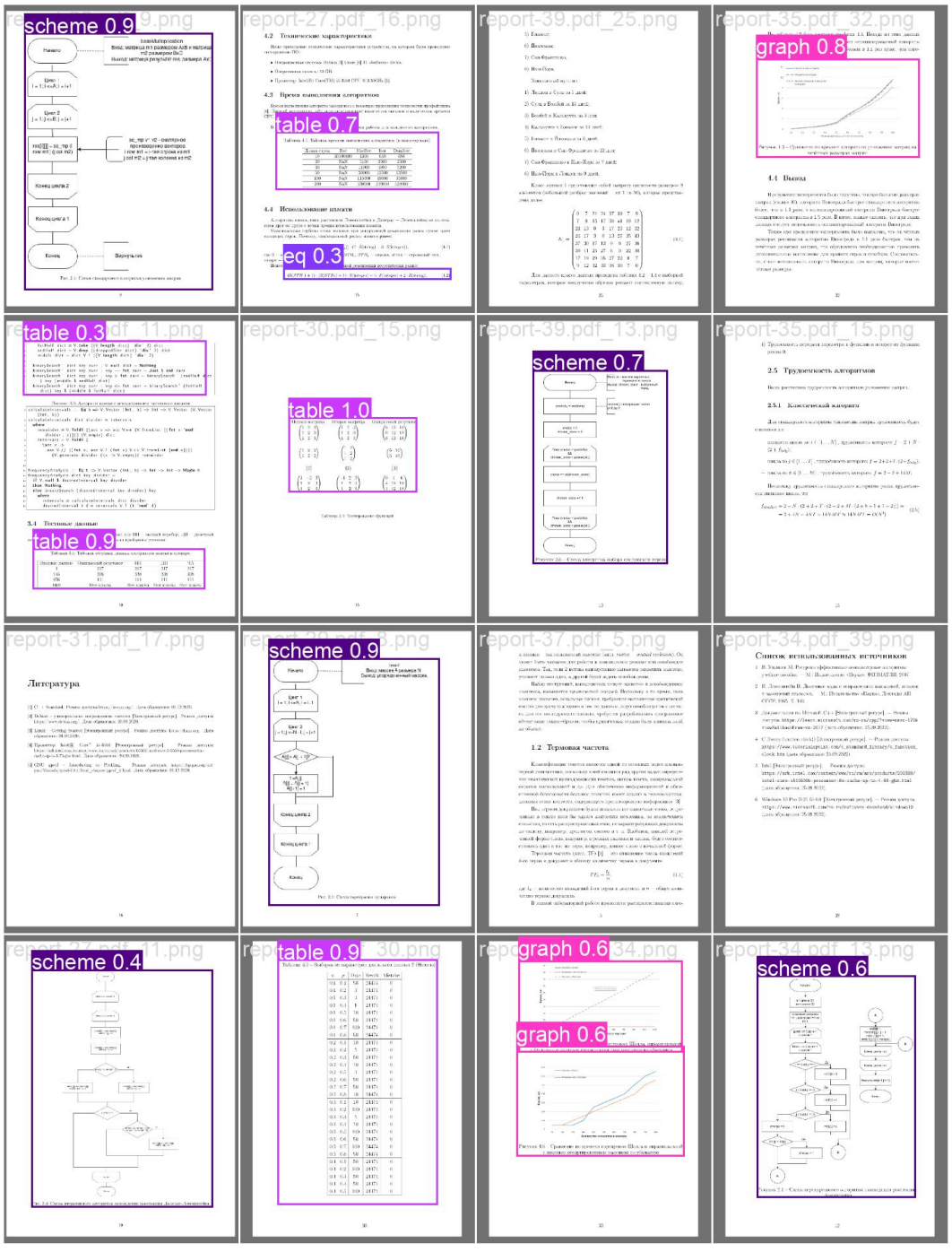


Рисунок В.1 – Предсказания на валидационной выборке после 10 эпох обучения

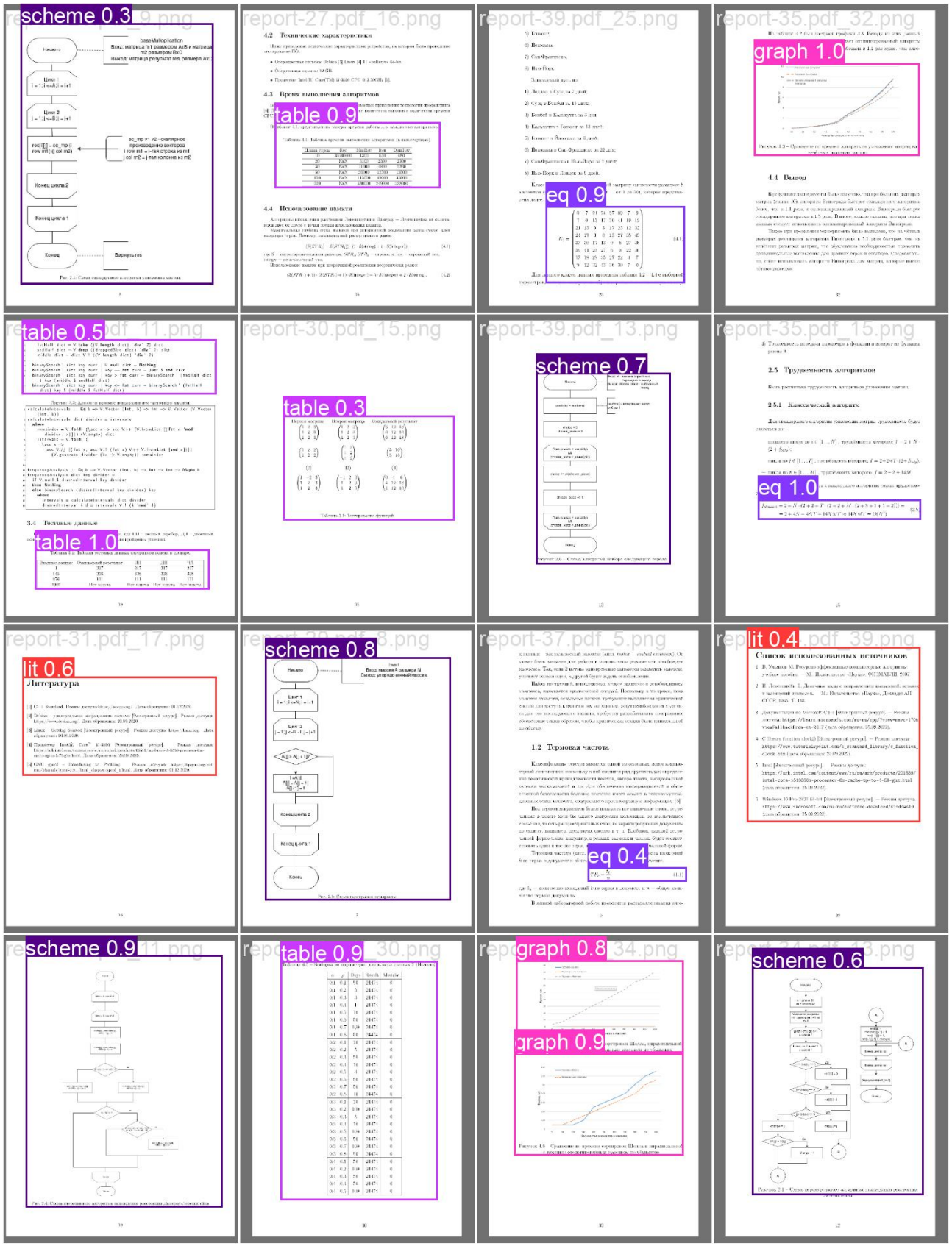


Рисунок В.2 – Предсказания на валидационной выборке после 73 эпох обучения

ПРИЛОЖЕНИЕ Г

Листинг Г.1 – Исходный код обучения YOLOv8

```
import os
import numpy as np
import pandas as pd
import shutil
import cv2
import random
import matplotlib.pyplot as plt
import copy

#!pip install ultralytics необходимо установить ultralitics



#!unzip /datasets/NIRS/annots.zip  необходимо распаковать датасет

#!unzip /datasets/NIRS/images_annoted.zip  необходимо
   распаковать датасет

import numpy as np
import cv2



path_images = '/content/images_annoted/'
path_annots = '/content/annots/'

img_name = 'ReportLab01.pdf_11'

annot_name = 'ReportLab01.pdf_11.png'

image =
   cv2.imread('/content/images_annoted/ReportLab01.pdf_11.png')
height = np.size(image, 0)
width = np.size(image, 1)
cv2.imshow(image)

def draw_bbs(image_name,ext = '.png'):
```

```
  print(path_images + image_name +ext)
  image = cv2.imread(path_images + image_name +ext)
  annots_file = path_annots + image_name + '.txt'
  lines = open(annots_file).readlines()
  print(image.shape)
  image_height = image.shape[0]
  image_width = image.shape[1]
  for line in lines:
    bbox_yolo_format = list(map(float,line.split()))[1:]
    print(bbox_yolo_format)
    x, y, width, height = int(bbox_yolo_format[0] *
       image_width), int(bbox_yolo_format[1] * image_height),
       int(bbox_yolo_format[2] * image_width),
       int(bbox_yolo_format[3] * image_height)
    color =(255, 0, 0)
    thickness = 2
    cv2.rectangle(image, (x - width // 2, y - height // 2), (x +
       width // 2, y + height // 2), color, thickness)
  cv2.imshow(image)


img_name = '/report-31.pdf_5'

draw_bbs(img_name)

from ultralytics import YOLO

model=YOLO('yolov8n.yaml').load('yolov8n.pt')

config = '''
train: '/content/images/train'
val: '/content/images/test'
# Classes
names:
  0: dog
  1: person
  2: cat
  3: tv
  4: car
  5: meatballs
  6: marinara sauce
```

```
  7: tomato soup
  8: chicken noodle soup
  9: french onion soup
  10: chicken breast
  11: ribs
  12: pulled pork
  13: hamburger
  14: cavity
  15: eq
  16: scheme
  17: table
  18: pic
  19: graph
  20: lit'''

config_name = "config.yaml"

with open(config_name,'w') as f:
  f.write(config)


val_size = 0.1
train_size = 0.9

os.rename('images_annoted','images')

os.rename('annots','labels')

path_images = '/content/images/'
path_annots = '/content/labels/'

labels = list(os.listdir('/content/labels'))
train_labels = labels[:int(len(labels) * train_size)]
val_labels = labels[int(len(labels) * train_size):]

images= list(os.listdir('/content/images'))
train_images = images[:int(len(images) * train_size)]
val_images = images[int(len(images) * train_size):]

os.makedirs('train')

```

```
os.makedirs('test')

for image in train_images:
  shutil.move(path_images + image,'train/')



shutil.move('train','images/')




for image in val_images:
  shutil.move(path_images + image,'test/')


shutil.move('test','images/')

os.makedirs('train')

os.makedirs('test')

for label in train_labels:
  shutil.move(path_annots + label,'train/')


shutil.move('train','labels/')

for label in val_labels:
  shutil.move(path_annots + label,'test/')

#!mv test labels/
shutil.move('test','labels/')


results=model.train(data=config_name, epochs=10, resume=True,
   iou=0.5, conf=0.001)



def show_model_perfomance(images):
```

```
154   plt.figure(figsize=(60,60))
155
156   for i in range(1,8,2):
157       test_image=images[i]
158       ax=plt.subplot(4,2,i)
159
160       # Display actual image
161       plt.imshow(cv2.imread(test_image))
162       plt.xticks([])
163       plt.yticks([])
164       plt.title("Actual image", fontsize = 40)
165
166       # Predict
167       res = model(test_image)
168       res_plotted = res[0].plot()
169       ax=plt.subplot(4,2,i+1)
170
171       # Display image with predictions
172       plt.imshow(res_plotted)
173       plt.title("Image with predictions", fontsize = 40)
174       plt.xticks([])
175       plt.yticks([])
176
177
178 val_img_path = '/content/images/test/'
179
180 val_images_paths = list(map(lambda x: val_img_path +
      x,val_images))
181
182
183 show_model_perfomance(val_images_paths)
```