Введение

В ходе предыдущей исследовательской работы было выявлено несколько ключевых проблем автоматизированной системы нормоконтроля документов.

Среди них мной были выбраны по моему мнению наиболее актуальные и срочные для решения проблемы, такие как:

– создание базы данных приложения, включающее в себя схему хранения правил соответствия форматирования документов различным ГОСТам;

– оптимизация работы процесса обмена информацией между клиентской частью и сервером;

– оптимизация и адаптация существующей модели парсинга текстовых документов к работе с форматами ODT, DOCX, PDF;

– исследование возможности улучшения и оптимизации существующего классификатора параграфов документов.

Цель работы заключается в решение поставленных на предыдущем этапе ключевых проблем сервиса нормоконтроля документов.

Задачи работы:

– развернуть серверную составляющую сервиса;

– спроектировать и реализовать модель база данных;

– оптимизировать процесс взаимодействия клиента и сервера;

– оптимизировать структуру модуля парсинга данных;

– оптимизировать модель классификатора параграфов.

1 Подковка аппаратных и программных средств, необходимых для разработки, тестирования и непосредственной работы информационной системы.

В ходе предыдущих исследований было решено разделить работу сервиса на две части: клиентскую и серверную. Клиентская часть встраивалась в программное обеспечение MS Word в качестве дополнительной надстройки. Однако результат показал, что такое решение было неэффективным.

Основной причиной этого стало то, что большинство пользователей MS Word никогда не пользовались функцией надстроек, а некоторые даже не знают, что это такое. Также оказалось, что разработанный интерфейс для пользователей не очень удобен и местами непонятен. Большим недостатком оказалась сложность установки сервиса на клиентский компьютер пользователями не имевших такой опыт.

Поэтому было решено переделать клиентскую часть в виде веб-сайта, к которому у людей будет постоянный доступ. Это позволит внести множество дополнительного функционала, который было невозможно внедрить предыдущим способом, например оплата сервиса, или возможность поделиться результатами с другими людьми.

Для начала полноценной работы был развернут VPS сервер со следующими характеристиками:

- операционная система: Windows Server 2019;
- процессор: Intel Xeon, 2x2.2ГГц;
- оперативная память: 2 Гб;
- жесткий диск: HDD 40 Гб в RAID 1.

Серверу был выделен IPv4-адрес 195.133.197.161, который служит для постоянного подключения пользователей к сайту. Физически сервер расположен в Королёве, Россия.

Для удобства будущих пользователей был куплен домен normcontrol.ru и normcontrol.online. В записи зоны домена был внесены изменения,

необходимые для его привязки к ip-адресу сервера. Добавленные записи показаны на рисунке 1.

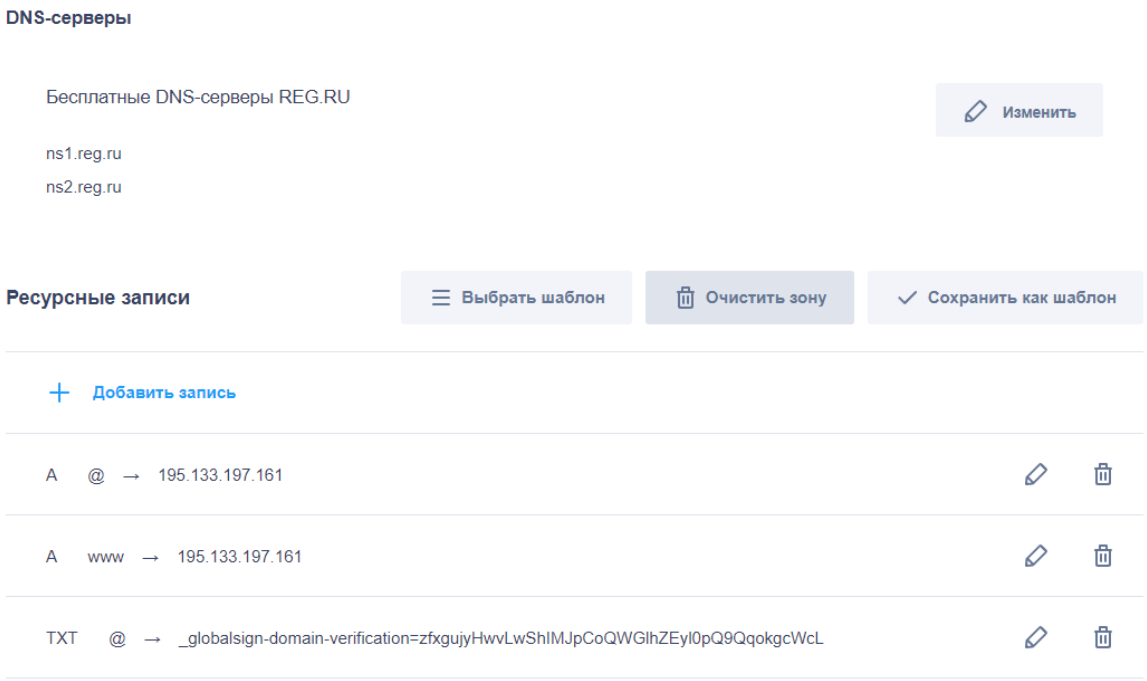


Рисунок 1 – Записи домена normcontrol.ru

Для запуска сайта на сервере была развёрнута портативная программная среда Open Server Panel которая содержит все необходимые модули и программы для разработки веб сайтов и не только [1].

Основные модули, которые необходимы для базовой работы сайта следующие:

– Apache: 2.2.31, 2.4.38, 2.4.41, 2.4.53 + auth_ntlm, fcgid, maxminddb, xsendfile;
– Bind: 9.16.28;
– FTP FileZilla: 0.9.60;
– Ghostscript: 9.56.1;
– Nginx: 1.21.6 + ssl_preread, image_filter, geoip, geoip2, brotli;
– NNCron Lite: 1.17;
– Sendmail: 32;
– PHPMyAdmin: 5.1.3;
– MySQL: 5.1.73, 5.5.62, 5.6.51, 5.7.38, 8.0.29.

Основные настройки портов сервера, занимающиеся приложениями представлены на рисунке 2.

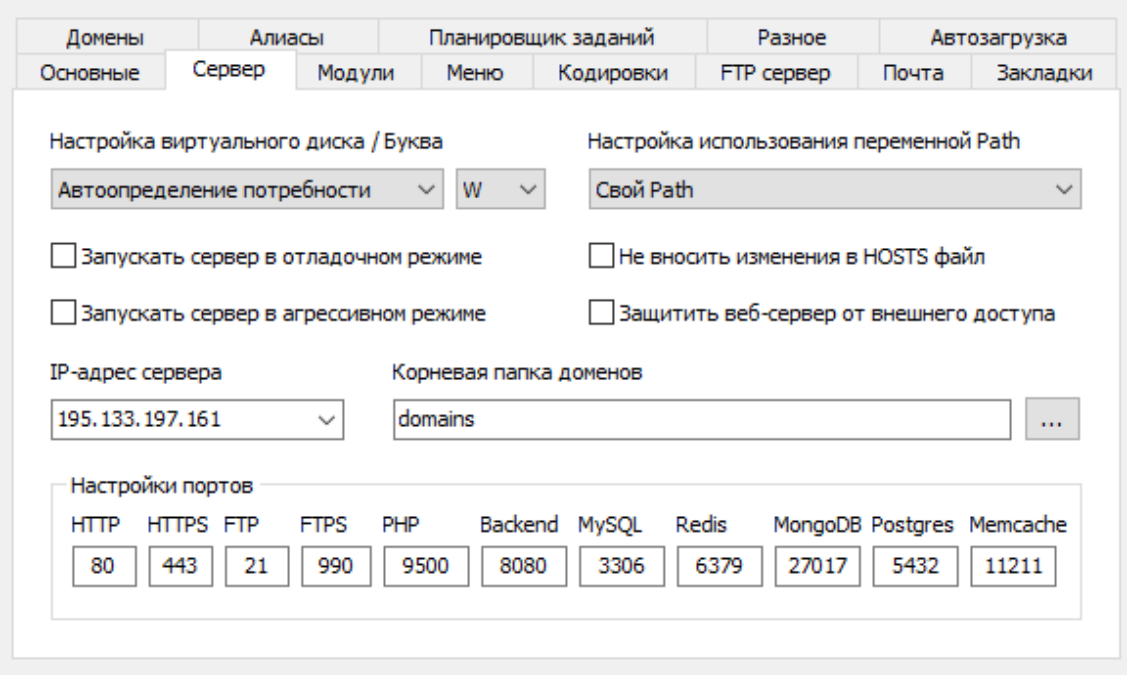


Рисунок 2 – Настройки портов веб-сервера

В связи с ограниченными ресурсами данный VPS сервер также было решено использовать в качестве терминального сервера для полноценной работы на нём и тестирования разработанных модулей. В дальнейшем при полноценном запуске сервиса данный сервер будет выполнять только функцию поддержки веб-сайта.

Для запуска отдельных модулей, уже собранных в докер контейнер был развёрнут другой сервер. В связи с тем, что на развёртываемом сервере будут выполняться процессы уже разработанных приложений, к нему были предъявлены намного меньшие требования.

Характеристики сервера, следующие:

– операционная система: Ubuntu Server 20.04 64 bit;

– процессор: Intel Xeon, 2x2.2ГГц;

– оперативная память: 2048 Мб;

– жесткий диск: HDD 20 Гб.

Так как большинство все разрабатываемые модули сервиса работают и обмениваются информацией посредством API и JSON запроса соответственно,

поэтому серверу был выделен отдельный IPv4-адрес: 193.233.206.162, а также IPv6-адрес: 2605:e440::3:3b

2 Разработка и реализация модели базы данных

Для полноценной работы сервиса ему необходимо хранить различную информацию. Сервис должен иметь к ней постоянный доступ и возможность внесения, чтения и удаления данных. Далее будет определяться информация, подлежащая хранению в базе данных

2.1 Определение информации, подлежащей хранению

В ходе исследования была выделена информация, которая должна храниться в базе данных.

Вся информация была разделена на следующие 3 категории:

– обезличенные данные для обучения классификатора;

– данные ГОСТов;

– правила оформления элементов текста;

– типы элементов текста;

– Данные, необходимые для авторизации и работы веб-сайта.

Данные, необходимые для работы классификатора представлены в таблице 1.

Таблица 1 – Признаки необходимые классификатору

Название признака	Тип	Свойство
text	String	Текст
PrevElementMark	String	Класс предыдущего элемента
CurElementMark	String	Класс текущего элемента
NextElementMark	String	Класс следующего элемента
bold	Boolean	Жирный
italics	Boolean	Курсив
keep_together	Boolean	Не разрывать абзац
keep_with_next	Boolean	Не отрывать от следующего
windows\orphans	Double	Висячие строки

Данные необходимые для проверки нормоконтроля документов представлены в таблице 2.

Таблица 2 – Данные, необходимые для проведения проверки документа

Название свойства	Тип	Свойство
fontName	String	fontName
underlining	Boolean	underlining
subText	Boolean	subText
superText	Boolean	superText
textSize	Integer	textSize
bold	Boolean	Жирный
italics	Boolean	Курсив
keep_together	Boolean	Не разрывать абзац
keep_with_next	Boolean	Не отрывать от следующего
alignment	String	Выравнивание
alignment	String	Выравнивание
indent	Double	Красная строка
mrgrg	Double	Отступ справа
mrglf	Double	Отступ слева
lineSpacing	Double	Межстрочный интервал
mrgtop	Double	Отступ сверху
mrgbtm	Double	Отступ снизу

2.2 Модель базы данных

Для построения модели базы данных и её реализации использовалась СУБД MySQL по причине её доступности и открытости распространения. В СУБД первоначально была установлена база данных wordpress [2]. База данных wordpress уже содержит таблицы для хранения информации о клиенте. Поэтому были добавлены только следующие таблицы:

– sw_gost – таблица содержащая информацию о ГОСТах;

– sw_property – таблица, содержащая информацию о всех необходимых свойствах;

– sw_rule – таблица, содержащая информацию о правилах оформления согласно определённому ГОСТу;

– sv_statistics – таблица, содержащая статистику ошибок каждого пользователя;

– cbt_dataset – таблица для хранения и дополнения датасета.

Сформированная модель со связями между таблицами представлена на рисунке 3.

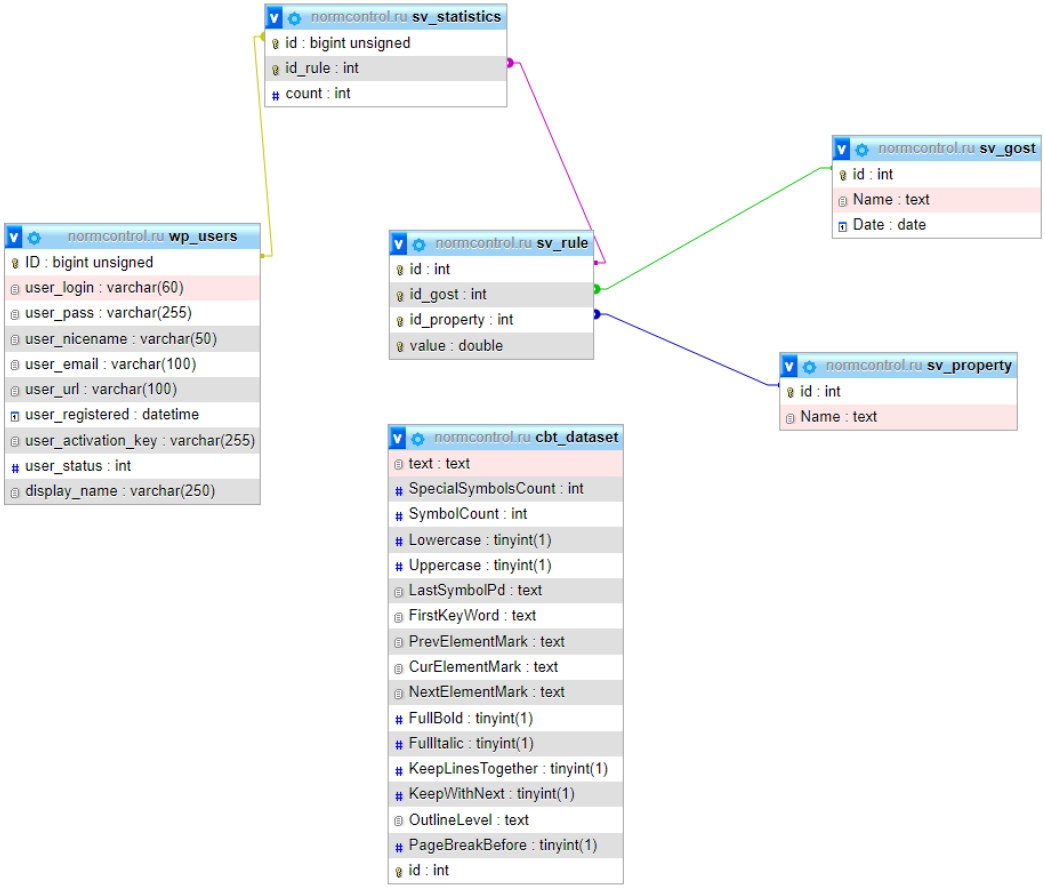


Рисунок 3 – Фрагмент модели базы данных

3 Модернизация сервиса автоматизированного нормоконтроля документов

Самой трудоёмким и важным этапом в ходе научно-исследовательской работы была оптимизация работы существующих модулей системы и естественно добавление новых функций и возможностей, решающих ключевые найденные проблемы сервиса.

Оптимизации и модернизации подверглись такие модули как:

– передача информации между клиентом и сервером;

– классификатор параграфов документа;

– структура модуля парсинга данных из документов.

3.1 Оптимизация работы процесса обмена информацией между клиентской частью и сервером.

3.1.1 Исследование существующей модели взаимодействия клиента и сервера и их компонентов

В настоящий момент клиентская часть информационной системы реализована при помощи технологии Microsoft Add-ins и создана при помощи языка программирования JavaScript. На стороне клиента происходит взаимодействие с клиентом посредством различных интерфейсов. Пользователь настраивает работу приложения под свои нужды. Например, выбирает необходимый ГОСТ, режим работы и отображения ошибок и многое другое.

На стороне сервера, как видно из рисунка 4, размещаются такие компоненты как DocxCorrector, классификатор параграфов и модуль NLP, отвечающий за лингвистический анализ текста [3,4].

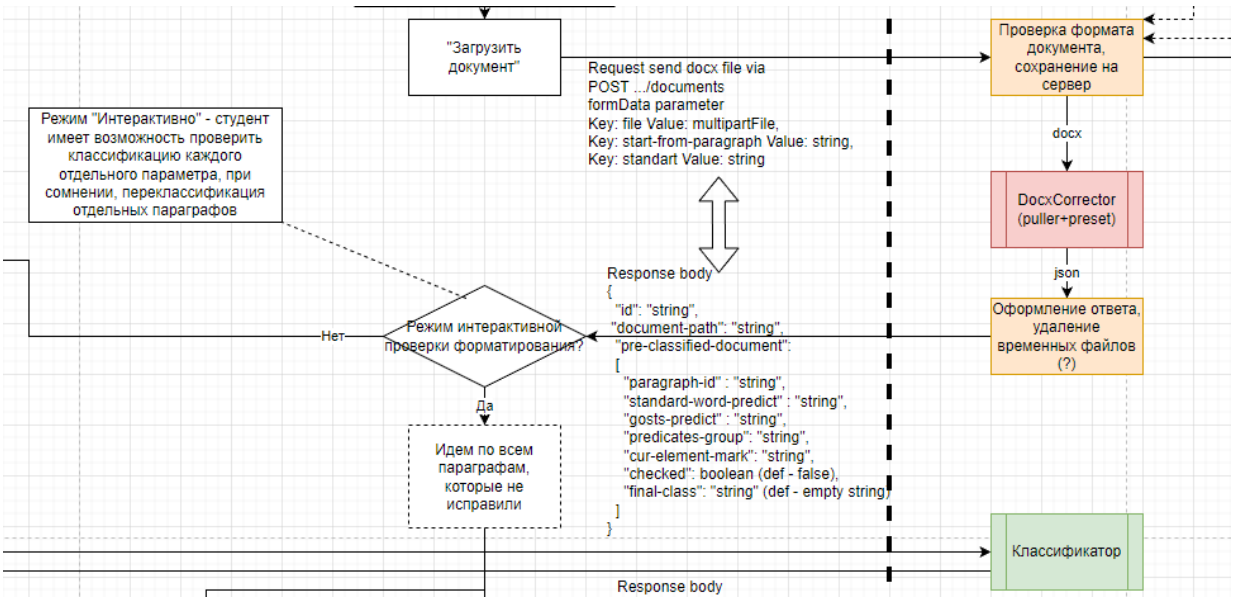


Рисунок 4 – Разделение модулей сервиса на клиенте и сервере

Одной из важнейших компонент системы является согласование правильности классификации параграфов с пользователем. Данное взаимодействие происходит посредством API запросов в JSON формате, как показано на рисунке 5. Аналогичным способом происходит авторизация клиента для продолжения работы в системе и взаимодействия большинства компонентов сервера [3,4].

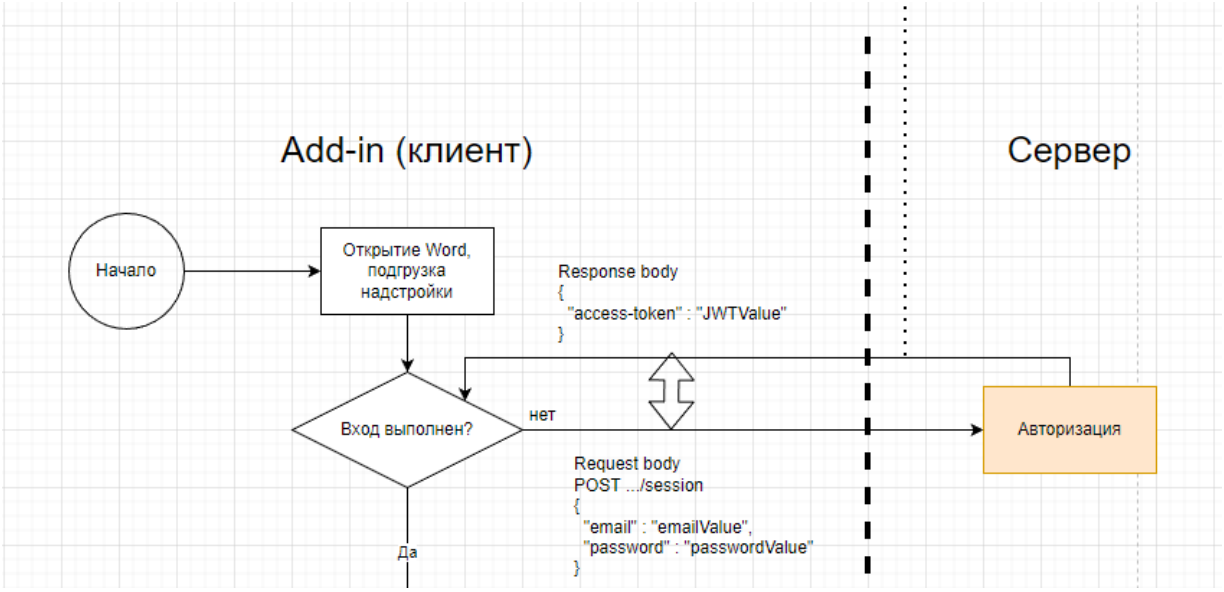


Рисунок 5 – Взаимодействие клиента и сервера посредством API

Однако после парсинга текстового документа, результаты его работы сохраняются в формате csv на сервере для последующей работы классификатора параграфов. Таким образом время работы классификатора увеличивается, что может быть существенной проблемой при увеличении нагрузки и количества клиентов. Также, за счёт постоянного сохранения и удаления файлов, ПЗУ сервера, такие как HDD, сильно изнашиваются и быстро приходят в негодность, что влечёт повышенные траты на содержание оборудования необходимого для работы системы.

3.1.2 Создание алгоритма создания JSON запросов, с передачей информации посредством модели ключ-значение

Алгоритм работы создания JSON работы следующий:

– создание тела запроса;

– добавление метаинформации текста посредством выделения её из класса документа;

– добавление необходимой информации о параграфах текста посредством прохождения циклом по всем объектам списка параграфов;

– отправка запроса;

– ожидание ответа от API.

Для лучшего понимания и визуализации алгоритма представлена блок-схема на рисунке 6.

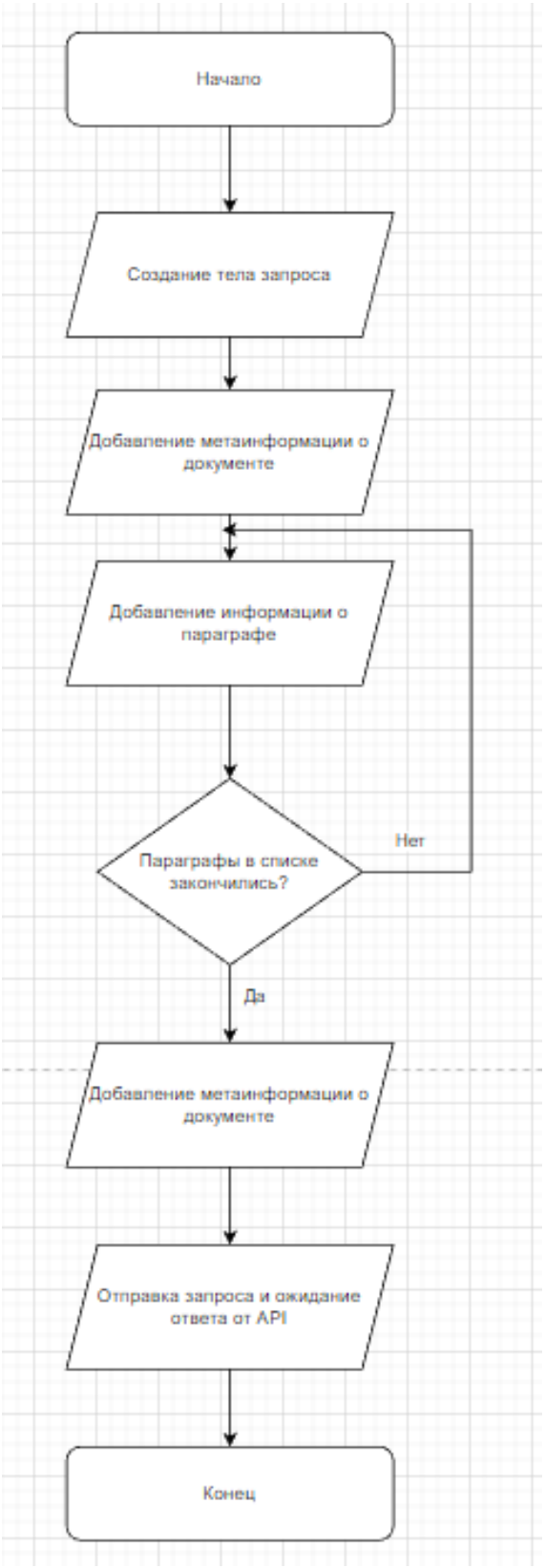


Рисунок 6 – Алгоритм работы процесса создания JSON запроса

3.2 Оптимизация работы существующей модели парсинга документов

Прежде чем решать проблему узконаправленности работы информационной системы, в виде поддержки только формата DOCX текстовых документов, необходимо создать новую структуру работы модуля парсинга документов.

В результате новая структура содержит в себе 4 подмодуля:

– pdf парсер;

– docx парсер;

– odt парсер;

– унифицированный класс для передачи выделенных парсерами данных документа.

Фрагмент новой структуры показан на рисунке 7.

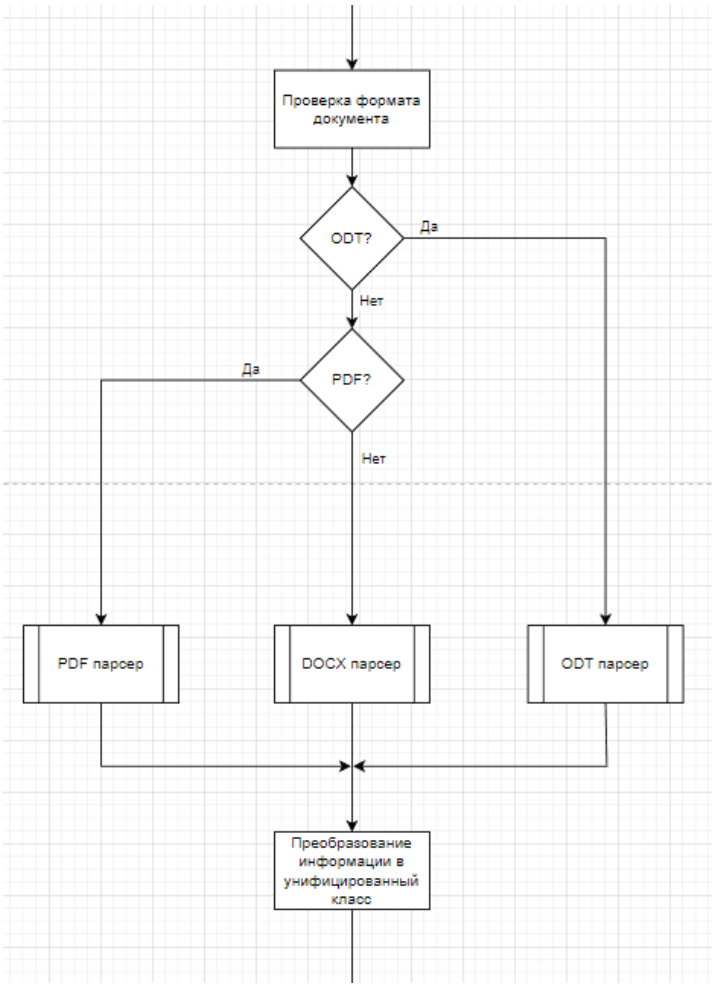


Рисунок 7 – Фрагмент новой структуры парсинга данных

PDF парсер служит для выделения всех возможных и необходимых для работы классификатора и модуля поиска и исправления ошибок.

ODT парсер работает аналогичным образом, но взаимодействует соответственно только с текстовыми документами формата ODT

Аналогично и DOCX парсер работает только с DOCX документами.

Унифицированных класс необходим для преобразования полученных парсерами данных в подходящие для понимания и работы единицы. Это связано с тем, что в различных документах используются различные способы измерения расстояния, различные наименования шрифтов и т. п. Также разработанный класс должен учитывать метаинформацию документов, такую как имя автора, дата создания, информацию о параграфах документов, включая размер шрифта, отступы от границ, межстрочный интервал и т. д, а также их порядок следования в документах.

Разработанная схема класса представлена на рисунке 8.

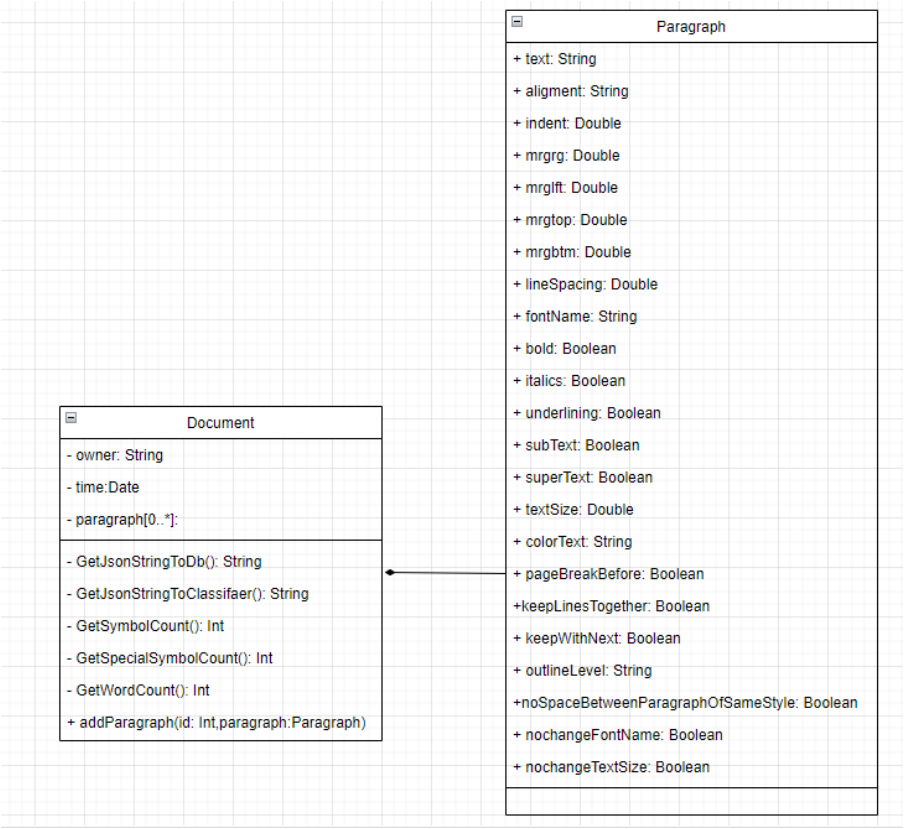


Рисунок 8 – Диаграмма разработанного класса

3.3 Оптимизация работы существующего классификатора параграфов текстовых документов

Для проверки текстовых документов и исправления обнаруженных ошибок необходимо первоначально определить к каким классам принадлежат параграфы. В зависимости от класса параграфа по нынешнему ГОСТу применяются различные правила оформления. Например, в основном весь текст выпускной квалификационной работы должен иметь выравнивание по ширине, однако для заголовка “Введение” и “Выводы” применяется выравнивание по центру. Таким образом для правильной работы сервиса необходимо провести классификацию параграфов документа.

Классификатор применяется сервисов только в трёх случаях:

– когда встречается оформление элемента, которое ещё ни разу не встречалось в текущем документе;

– когда встречается оформление элемента, которое не соответствует ГОСТу;

– когда встречается оформление элемента, которое существенно отличается от стандартного оформления Word.

Классификатор использует алгоритм градиентного бустинга, основаный на открытой технологии CatBoost [5, 6, 7], разработанной компанией «Яндекс». Основные параметры алгоритма были выбраны посредством множественных тестовых запусков. Основными критериями выбора являлись: эффективность классификатора и время его работы.

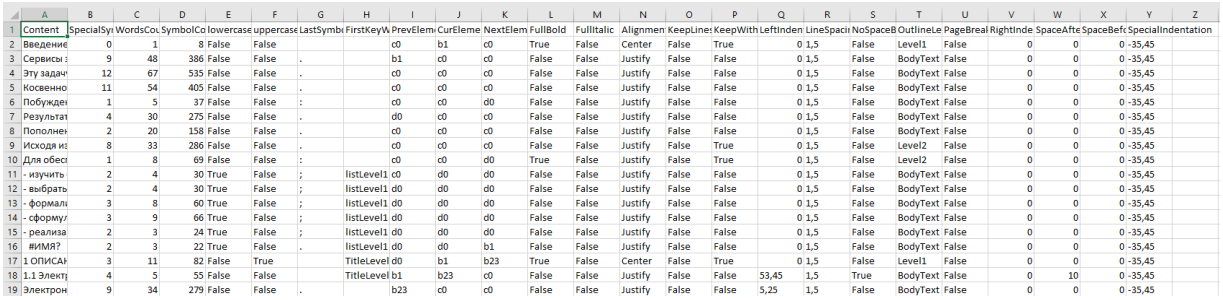
3.3.1 Описание предыдущих оптимизаций классификатора

В ходе исследования была проанализирована работа прежнего алгоритма классификации.

Класс элемента определяется на основе его свойств и параметров, котрые были выделены из документа при помощи вышеупомянутых модулей PDF, DOCX и ODT парсера.

В качестве датасета использовались выборки выпускных работ бакалавров и магистров. Из них были удалены все персональные данные,

такие как имя автора, имя ПК и т. п. В выборке присутствует информация только об основном теле документа, остальные данные такие как список использованных источников, титульный лист, реферат были удалены. Фрагмент датасета представлен на рисунке 9.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Content	SpecialSy	WordsCo	SymbolCo	lowercase	uppercase	LastSymb	FirstKeyW	PrevElem	CurEleme	NextElem	FullBold	FullItalic	Alignmen	KeepLines	KeepWith	LeftInden	LineSpaci	NoSpaceB	OutlineLe	PageBreal	RightInde	SpaceAfte	SpaceBefo	SpecialIn	dentation	
2	Введение	0	1	8	False	False			c0	b1	c0	True	False	Center	False	True	0	1,5	False	Level1	False	0	0	0	-35,45		
3	Сервисы з	9	48	386	False	False	.		b1	c0	c0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
4	Эту задач	12	67	535	False	False	.		c0	c0	c0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
5	Косвенно	11	54	405	False	False	.		c0	c0	c0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
6	Побужде	1	5	37	False	False	:		c0	c0	d0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
7	Результат	4	30	275	False	False	.		d0	c0	c0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
8	Пополнен	2	20	158	False	False	.		c0	c0	c0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
9	Исходя из	8	33	286	False	False	.		c0	c0	c0	False	False	Justify	False	True	0	1,5	False	Level2	False	0	0	0	-35,45		
10	Для обесп	1	8	69	False	False	:		c0	c0	d0	True	False	Justify	False	True	0	1,5	False	Level2	False	0	0	0	-35,45		
11	- изучить	2	4	30	True	False	;	listLevel1	c0	d0	d0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
12	- выбрать	2	4	30	True	False	;	listLevel1	d0	d0	d0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
13	- формали	3	8	60	True	False	;	listLevel1	d0	d0	d0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
14	- сформу	3	9	66	True	False	;	listLevel1	d0	d0	d0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
15	- реализа	2	3	24	True	False	;	listLevel1	d0	d0	d0	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
16	#ИМЯ?	2	3	22	True	False	.	listLevel1	d0	d0	b1	False	False	Justify	False	False	0	1,5	False	BodyText	False	0	0	0	-35,45		
17	1 ОПИСАН	3	11	82	False	True		TitleLevel	d0	b1	b23	True	False	Center	False	True	0	1,5	False	Level1	False	0	0	0	-35,45		
18	1.1 Электр	4	5	55	False	False		TitleLevel	b1	b23	c0	False	False	Justify	False	False	53,45	1,5	True	BodyText	False	0	10	0	-35,45		
19	Электрон	9	34	279	False	False	.		b23	c0	c0	False	False	Justify	False	False	5,25	1,5	False	BodyText	False	0	0	0	-35,45		

Рисунок 9 – Фрагмент используемого датасета

Были определены свойства, определяющие оформление элементов классов, на основе которых была произведена разметка. Разметка была сделана в два этапа: сначала по более общей группе (абзац, список, заголовок) и после по более мелким группам, то есть различным видам абзацев, заголовков, списков. Например, заголовки были поделены на заголовки первого уровня и заголовки уровней 2–3; элементы списка поделены на первый, последний и средний. Подобные разделения предположительно должны были дать улучшение предсказаний модели. В результате первоначальный набор данных для классификатора содержал 14 классов. Полный список классов элементов и описание каждого из них представлены в таблице 1.

Таблица 1 – Классы элементов

Наименование класса	Описание
b1	Заголовок 1 уровня
b2	Заголовок 2-3 уровней
c1	Обычный абзац
c2	Абзац перед списком
d2	Элемент перечисления (списка)
f1	Подпись к таблице
f3	Продолжение таблицы
g1	Рисунок
h1	Подпись к рисунку
i1	Формула
j0	Подпись к формуле

Для наглядности была построена диаграмма, визуализирующая количество параграфов каждого класса на рисунке 10.

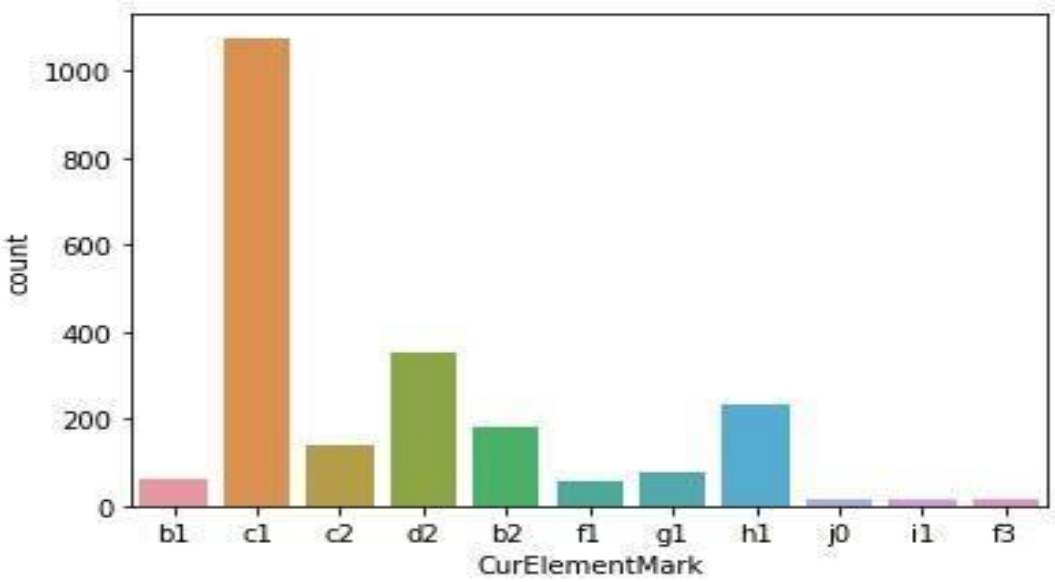


Рисунок 10 – Диаграмма распределения классов датасета

Для обучения алгоритма правильный класс элемента был занесён в столбец CurElementMark. В ходе предыдущих исследований было выявлено что точность предсказания класса элемента сильно зависит от класса предыдущего и следующего параграфа, которые были занесены соответственно в столбцы PrevElementMark и NextElementMark.

Остальные свойства были собраны при помощи модуля DocxCorrector посредством парсинга порядка 10 текстовых документов магистерских и бакалаврских выпускных квалификационных работ. Все необходимые свойства представлены в таблице 2.

Таблица 2 – Признаки, участвующие в обучении классификатора

Название признака датасета	Описание
SpecialSymbolsCount	Количество знаков препинания
WordsCount	Количество слов элемента
SymbolCount	Количество символов
Lowercase	Весь текст нижним регистром
SymbolCount	Количество символов
Lowercase	Весь текст нижним регистром
Uppercase	Весь текст верхним регистром

Продолжение таблицы 2

Название признака датасета	Описание
LastSymbolPd	Символ в конце элемента
FirstKeyWord	Ключевое слово класса
PrevElementMark	Класс предыдущего элемента
CurElementMark	Класс текущего элемента
NextElementMark	Класс следующего элемента
FullBold	Жирный шрифт элемента
FullItalic	Курсив
Alignment	Расположение элемента
KeepLinesTogether	Между строками нет разрывов или таблиц
KeepWithNext	Не отрывать элемент от следующего
LeftIndentation	Отступ слева
LineSpacing	Междустрочный интервал
NoSpaceBetweenParagraphsOfSameStyle	Расстояние между параграфами одного стиля
OutlineLevel	Уровень заголовка
PageBreakBefore	Разрыв страницы перед Документом
RightIndentation	Отступ справа
SpaceAfter	Отступ после
SpaceBefore	Отступ перед
SpecialIndentation	Отступ первой строки

Для оценки оптимизации модели классификатора были определены следующие метрики качества [8]:

– accuracy Score;
– precision;
– recall;
– F1-Score;
– macro Average;
– weighted Average;
– confusion Matrix.

Accuracy Score показывает сколько классов были определены правильно относительно общего количества прогнозов. Для определения остальных

метрик использовалась библиотека Sklearn – classification_report [9]. Она позволяет определить не только такие метрики как, Macro F1 и Weighted F1, но и выделить метрики Precision, Recall, F1-score и Support относительно каждого класса.

Как видно из рисунка диаграммы распределения классов, количество классов в датасете не сбалансировано, поэтому был применён алгоритм RandomOverSampler, который в зависимости от настроек дублирует определённые строки набора данных [10, 11].

При обучении и тестировании алгоритма получились следующие данные, представленные в таблице 3.

Таблица 3 – Результаты тестирования актуальной версии классификатора

Номер класса	Precision	Recall	F1-score	Support
0	1.00	0.62	0.77	58
1	0.86	0.92	0.88	155
2	0.92	0.94	0.93	908
3	0.65	0.63	0.64	120
4	0.93	0.91	0.92	297
5	1.00	1.00	1.00	42
6	1.00	1.00	1.00	12
7	0.98	0.98	0.98	66
8	0.99	1.00	1.00	190
9	1.00	1.00	1.00	923
10	1.00	1.00	1.00	922
Accuracy			0.96	3693
Macro avg	0.94	0.91	0.92	3693
Weighted avg	0.96	0.96	0.96	3693

3.3.2 Оптимизация классификатора

В ходе исследования работы классификатора действующего классификатора мной было выделено несколько ключевых аспектов и проблем.

Первой среди выделенных проблем была слишком большая размерность пространства признаков. После предыдущей оптимизации из первоначальных

24 признаков осталось только 21. Однако при исследовании было выявлено что это количество может быть уменьшено. Также за счёт большего количества параметров проходящий трафик между клиентом и сервером может быть достаточно большим.

Второй и по моему мнению основной проблемой является неправильно поставленная на предыдущих этапах перед классификатором задача. При подготовке датасета были использованы уже сданные и прошедшие нормоконтроль отчёты. Таким образом полученные данные были пригодны для использования классификации правильно оформленных параграфов, возможно с небольшими ошибками. Однако главной целью классификатора по моему мнению является классификация параграфов независимо от его оформления в документе. Также стоит учитывать, что количество вариантов ошибок в оформлении элементов может стремиться к бесконечности, что делает представленную задачу очень сложной.

В качестве возможных решений можно было использовать два варианта:

– подготовка нового датасета из ещё не прошедших нормоконтроль отчётов;

– изменить имеющийся датасет таким образом, чтобы он был минимально зависим от оформления элементов.

Как уже упоминалось ранее количество ошибок оформления может быть очень велико, что делает первый вариант наименее подходящим. Также следует учитывать, что собрать такой датасет может быть сложной задачей, т. к. такие документы являются интеллектуальной собственностью человека, а до официального оформления, в т. ч. нормоконтроля авторы будут негативно относиться к таким процессам.

В результате мной был выбран второй вариант для решения выделенной проблемы. В ходе исследований и тестов были удалены следующие признаки элементов:

– alignment;

– leftIndentation;

– lineSpacing;

– rightIndentation;

– spaceAfter;

– spaceBefore.

В результате после обучения алгоритма с теми же параметрами, какие были выбраны на предыдущих этапах оптимизации для визуального понимания была снова построена матрица ошибок Confusion Matrix [12]. Она представлена на рисунке 11.

```
Shrink model to first 2990 iterations.
[[ 40   2   0   0   0   0   0   0   0   0   0]
 [ 15 137   0   0   3   0   0   0   0   0   0]
 [  0   2  36   1   2   0   0   0   0   0   0]
 [  0   0   2 110   7   0   0   0   0   0   0]
 [  0   2   5   1 285   0   0   0   0   0   0]
 [  0   0   0   0   0  42   0   0   0   0   0]
 [  0   1   0   0   0   0 911   0   0   0   0]
 [  0   0   0   0   0   0   0  67   0   0   0]
 [  0   0   0   0   0   0   0   0 198   0   0]
 [  1   3   0   0   3   0   0   1   0 926   0]
 [  0   0   0   4   2   0   0   0   0   0 918]]
```

Рисунок 11 – Матрица ошибок оптимизированного классификатора

Далее на рисунке 12 приведен отчёт по классификации построенный при помощи библиотеки Sklearn и метода classification_report.

	precision	recall	f1-score	support
b1	0.71	0.95	0.82	42
b2	0.93	0.88	0.91	155
c1	0.84	0.88	0.86	41
c2	0.95	0.92	0.94	119
d2	0.94	0.97	0.96	293
f1	1.00	1.00	1.00	42
f3	1.00	1.00	1.00	912
g1	0.99	1.00	0.99	67
h1	1.00	1.00	1.00	198
i1	1.00	0.99	1.00	934
j0	1.00	0.99	1.00	924
accuracy			0.98	3727
macro avg	0.94	0.96	0.95	3727
weighted avg	0.99	0.98	0.99	3727

Рисунок 12 – Результаты тестирования классификатора

Из матрицы ошибок и полученным метрикам после тестирования видно, что классификатор практически не ошибается. Точность классификации составила 98%. За счёт высоких и практически одинаковых показателей метрик показатель F1 Score также высок.

Ниже приведена сводная таблица для более удобного сравнения первоначального и усовершенствованного алгоритма классификации. Для показателей Precision, Recall и F1-Score было взято среднее арифметическое по всем классам. Это представлено в таблице 4.

Таблица 4 – Сравнительная таблица метрик

Метрика	Показатели первоначального классификатора	Показатели классификатора после оптимизации
Accuracy	96%	98%
Macro Average Precision	94%	94%
Macro Average Recall	91%	96%
Macro Average F1-Score	93%	95%
Weighted Average Precision	96%	99%
Weighted Average Precision	96%	98%

Продолжение таблицы 4

Метрика	Показатели первоначального классификатора	Показатели классификатора после оптимизации
Weighted Average F1-Score	96%	99%
Скорость обучения	75 с	57.5 с
Скорость предсказания	0.00535 с	0.00490 с

Из данных таблицы следует что показатели значительно увеличились, с учётом большого изменение признаков. Показатель Macro Average Recall увеличился больше всех с 91% до 96%, что указывает на то, что выросла доля верно предсказанных объектов, которые модель определила к данному классу. Самым весомым событием стало увеличение общей точности классификатора с 96% до 98%. Также следует учитывать увеличение скорости обучения с 75 до 57.5 с. и ускорение скорости предсказания с 0.00535 с. до 0.00490 с что имеет весомое значение при увеличении количества клиентов и соответственно нагрузки.

ЗАКЛЮЧЕНИЕ

В ходе исследования работы была изучен сервис, разрабатывающийся в рамках НИРМ «Сервис автоматизированного нормоконтроля документов» с 2019 по 2021 год в университете ИТМО.

Были решены ключевые проблемы рассмотренного сервиса, такие как:

– создание базы данных приложения, включающее в себя схему хранения правил соответствия форматирования документов различным ГОСТам;

– оптимизация работы процесса обмена информацией между клиентской частью и сервером;

– оптимизация и адаптация существующей модели парсинга текстовых документов к работе с форматами ODT, DOCX, PDF;

– исследование возможности улучшения и оптимизации существующего классификатора параграфов документов.

Для решения поставленных задач были развернуты 2 сервера. Первый сервер используется в качестве терминального сервера и хостинга веб-сайта. На нём была установлена программная среда Open Server Panel которая содержит все необходимые модули и программы для разработки веб сайтов. Были настроены основные параметры сети и порты сервера. Второй сервер используется в качестве Docker сервера, для развёртки, работы и тестирования модулей сервиса.

В результате работы была создана база данных сервиса на основе СУБД MySQL и утилиты phpMyAdmin, построена и реализована модель базы данных сервиса.

Был проведён процесс оптимизации взаимодействия клиента и сервера и модулей внутри сервера.

И главным результатом является уменьшение размерности признаков классификатора, при увеличении точности распознавания и других метрик, а также ускорение работы классификатора.

Список использованных источников

1) Документация Open Server [Электронный ресурс]. – 2022. – URL: https://ospanel.io/docs/ (дата обращения: 25.02.2022).

2) Documentation WordPress[Электронный ресурс]. – 2022. – https://developer.wordpress.com/docs/ (дата обращения: 26.02.2022).

3) Отчёт о научно-исследовательской работе по теме: «Сервис автоматизированного нормоконтроля документов и обучения оформлению документации» [Электронный ресурс]. – 2021. – URL: https://docs.google.com/document/d/1UhLpBaSYeUUQ2FQzBQokHMZEGiYQQi5e/edit#heading=h.gjdgxs (дата обращения: 02.03.2022)

4) Тартынских П.С., Комаров М.С., Насыров Н.Ф. Подходы к автоматизированному анализу оформления электронных документов формата docx//Сборник тезисов докладов конгресса молодых ученых. Электронное издание. – СПб: Университет ИТМО, 2020. Электронное издание. – [2020, электронный ресурс]. – URL: https://kmu.itmo.ru/digests/article/4592, своб.— 2020

5) Nail Nasyrov, Mikhail Komarov, Petr Tartynskikh, Nataliya Gorlushkina. Automated formatting verification technique of paperwork based on the gradient boosting on decision trees // Procedia Computer Science, Volume 178, 2020, pp. 365-374, ISSN 1877-0509

6) Catboost Documentation [Электронный ресурс]. – URL: https://catboost.ai/docs/concepts/parameter-tuning.html (дата обращения: 27.03.2022).

7) Gradient Boosting from scratch [Электронный ресурс].–URL: https://blog.mlreview.com/gradient-boosting-from-scratch-1e317ae4587d(датаобращения: 05.04.2022).

8) Шунина Юлия Сергеевна, Алексеева Венера Арифзяновна, Клячкин Владимир Николаевич Критерии качества работы классификаторов // Вестник УлГТУ[Электронный ресурс]. –2015. No 2 (70). URL:

56https://cyberleninka.ru/article/n/kriterii-kachestva-raboty-klassifikatorov (дата обращения: 15.04.2022).

9) User Guide [Электронный ресурс]. – URL: https://scikit-learn.org/stable/user_guide.html

10) Machine Learning —Multiclass Classification with Imbalanced Dataset [Электронныйресурс]. URL: https://towardsdatascience.com/machine-learning-multiclass-classification-with-imbalanced-data-set-29f6a177c1a(дата обращения: 27.04.2022).

11) Документация библиотеки Imblearn[Электронный ресурс].–URL: https://imbalanced-learn.org/stable/(датаобращения: 01.05.2022).

12) Confusion Matrix for Your Multi-Class Machine Learning Model [Электронныйресурс]. URL: https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826 (дата обращения: 07.05.2022).