Оглавление

[Введение 2](#_Toc59649851)

[Глава 1 Аналитическая часть 7](#_Toc59649852)

[Обзор и сравнительный анализ существующих информационных систем (компонентов информационных систем) 7](#_Toc59649853)

[Обоснование направления разработки и требования, предъявляемые к системе 13](#_Toc59649854)

[Функциональные требования. 13](#_Toc59649855)

[Бизнес требования. 14](#_Toc59649856)

[Нефункциональные требования. 15](#_Toc59649857)

[Анализ исходных данных и выбор варианта построения системы 16](#_Toc59649858)

[Выводы по аналитической части 19](#_Toc59649859)

[СПИСОК ЛИТЕРАТУРЫ. 20](#_Toc59649860)

# 

**API** (application programming interface)

Описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой.

**BERT**

Нейронная сеть от Google.

**NLP** (Natural language processing)

Обработка естественного языка — общее направление искусственного интеллекта и математической лингвистики.

# Введение

Тематика рекомендательных систем и машинного обучения, применяемого в них, набирает популярность. С каждым днём мы можем наблюдать, как появляются всё новые и новые алгоритмы и системы, построенные на базе нейронных сетей с использованием машинного обучения. А также всё больше и больше инструментов, позволяющих создавать подобные решения.

Рекомендательные системы — это большой класс моделей, цель которых – формирование рекомендаций пользователю исходя из его предпочтений. Это инструменты и алгоритмы, направленные на предсказание того, что будет интересно пользователю – основываясь на его прошлых действиях и данных об этом пользователе. Эти данные могут быть различными и включать в себя информацию от роста пользователя, до его текущей геопозиции.

Любая рекомендательная система, нуждается в обучении нейронной сети. Существует множество алгоритмов обучения и наиболее подходящим - для данной информационной системы является алгоритм обратного распространения ошибки.

Это алгоритм обучения нейронных сетей прямого распространения. Относится к методам обучения с учителем, поэтому требует, чтобы в обучающих примерах были заданы целевые значения.

В основе идеи алгоритма лежит использование выходной ошибки нейронной сети.

Алгоритм является итеративным и использует принцип обучения «по шагам», когда веса нейронов в сети корректируются после подачи на ее вход одного обучающего примера.

Углубляясь в историю возникновения рекомендательных систем, первое устройство пытающееся предсказать предпочтения пользователей появилось в конце 1990-х. Это был цифровой видеомагнитофон, пытающийся учесть предпочтения пользователей, чтобы автоматически записывать понравившиеся им передачи.

В те времена, не было хорошо структурированных описаний для телепередач, равно как и не было мощности, для обработки столь больших массивов информации. Именно поэтому за основу был взял алгоритм коллаборативной фильтрации.

Это алгоритм, основывающийся на поведенческом сходстве пользователей. Он не использовал описаний телепередач, фильмов и других текстовых представлений. Суть заключалась в том, что если на 1...N устройствах в пределах различной местности – зрители выбирали одну и ту же программу, а затем кто-то из пользователей начинал активно смотреть одну из других программ, то первую – предлагали второму пользователю. Также осуществлялся ежесуточный сбор информации – которая каждую ночь отправлялась на центральный сервер. Эта информация несла в себе массив данных – о последовательностях кликов пользователей. Основываясь на этих данных – алгоритмы на сервере вычисляли необходимые рекомендации для пользователей, в последствии – выгружая их на все устройства.

Однако в середине 2000-х годов обеспечиваемое коллаборативным алгоритмом качество – не достигло желаемых результатов. На протяжении многих лет команды разработчиков трудились над улучшением точности рекомендаций и улучшили этот показатель на 10%. Но из-за больших операционных затрат – идея использования алгоритма коллаборативной фильтрации прекратила своё существование.

В дальнейшем, в процессе развития новых подходов, широкую популярность приобрела – байесовская классификация, позволяющая провести оценку предпочтений пользователя. Например, почему пользователю нравится треки одной группы, но он не станет слушать другую? В ответ на этот вопрос – система предположит, что пользователь больше отдаёт предпочтение жанру первой группы, нежели второй.

Далее началось активное развитие различных классификаторов. Многие компании, занимающиеся семантическим типом рекомендаций – отдали предпочтение классификации контента основывающимся на метаданных присутствующих в общем доступе. Это сильный импульс развитию компаний и рекомендательных систем. Такая компания как Netflix увеличила свой каталог жанров с 560 вариантов до 93116, по данным на Июль 2014-го года.

В наше время на существующем этапе развития рекомендательных систем основную роль стал играть контекст. Он включает в себя интересы пользователя в реальном времени и может зависеть от интересов и окружения пользователей. В наше время им важна комбинация рекомендаций, предоставляемых рекомендательными системами, на основе их реальных предпочтений. Они не испытывают желания самостоятельно, гибко настраивать систему, чтобы та выдавала им нужный результат. Важен эффект взаимодействия системы с пользователем. Если была сформирована не верная рекомендация, пользователь должен иметь возможность дать системе понять – что эта рекомендация ему не подходит. В таких случаях система сразу должна выдать новый, отличный или близкий по схожести результат.

Сам термин “нейронная сеть” впервые появился в середине ХХ века.

Первые работы, были проделаны Мак-Каллоком и Питтсом. В 1943 году ими была разработана компьютерная модель нейронной сети в основу которой легли математические алгоритмы и теории деятельности головного мозга. Они предполагали, что нейроны можно упрощённо рассматривать как устройства, оперирующие двоичными числами, и назвали эту модель «пороговой логикой».

Подобно своему биологическому прототипу нейроны Мак-Каллока–Питтса были способны обучаться путём подстройки параметров, описывающих синаптическую проводимость. Исследователи предложили конструкцию сети, состоящую из электронных нейронов продемонстрировав, её способность выполнять любые логические и числовые операции. Мак-Каллок и Питтс предположили, что разработанная ими сеть в состоянии также обучаться, распознавать образы и обобщать, т. е. обладает всеми чертами интеллекта.

Первым в 1949 году, канадский психолог и физиолог Хебб – высказал идеи о характеристиках соединения нейронов головного мозга и их взаимодействии предположив, что обучение заключается в первую очередь в изменениях силы синаптических связей.

Теория Хебба считается отличным примером самообучения, при котором система самостоятельно обучается выполнять поставленную ей задачу, при этом – не используя помощь со стороны экспериментатора.

В 1954 году в Массачусетском технологическом институте с использованием компьютеров Фарли и Кларк разработали имитацию сети Хебба.

В 1957 году Розенблаттом были разработаны математическая и компьютерная модели восприятия информации мозгом. Они основывались на двухслойной обучающейся нейронной сети. Розенблатт описал также схему логического сложения. В 1958 году он предложил модель электронного устройства, имитирующего процессы человеческого мышления, а два года спустя была продемонстрирована первая действующая машина, способная к обучению на распознавание некоторых букв, написанных на карточках, подносимым к его «глазам», напоминающим кинокамеры.

Исследования нейронных сетей были прекращены до достижения компьютерами больших вычислительных мощностей. Одним из важных шагов, стимулировавших дальнейшие исследования, стала разработка 1975 года Вербосом. Её сутью стал метод обратного распространения ошибки, позволивший более эффективно обучать многослойные нейронные сети.

Данная тематика в наши дни, постепенно начинает затрагивать многие сферы деятельности в нашей жизни. Одна из таких сфер – сфера образовательных услуг.

В каждом высшем учебном заведении, существует большой поток публикаций, в связи с чем, может возникнуть необходимость правильной оценки публикуемых статей. Одним из критериев такой оценки является – актуальность выбранной темы, на данный момент времени. В связи с этим, в рамках данной работы, будет рассмотрено решение, которое представляет собой, рекомендательную систему, способную не только определить актуальность публикуемой статьи, но и дать рекомендации по увеличению этой актуальности.

# Глава 1 Аналитическая часть

## **Обзор и сравнительный анализ существующих информационных систем (компонентов информационных систем)**

В связи с высоким ростом развития информационных технологий и искусственного интеллекта, на данный момент времени существует большое количество различных информационных и рекомендательных систем ­– направленных на формирование рекомендаций пользователям – в различных областях нашей жизни.

В качестве одного из примеров рекомендательной системы в современном представлении можно привести movielens.org[6,7], предлагающий пользователям фильмы на основе их предпочтений.

Этот сервис предоставляет пользователям обширный набор фильмов и их рейтингов, поставленных им пользователями.

Так же, на данный момент существуют рекомендательные системы для большинства социальных сетей, видео-хостингов и сайтов. Так как очень многие компании, заинтересованы в том, чтобы привлекать пользователей и выдавать им, только то, что будет им интересно.

Внедрение таких рекомендательных систем в массовый сегмент, может оказать положительное влияние на получение прибыли – в последующем направляемую на улучшение сервисов, модулей и всей системы в целом, а также – способную поднять востребованность и привлечь новые каналы продаж и инвестиций.

Как и во многих других сферах, в данной сфере так же активно ведутся разработки, направленные на улучшение образовательного процесса за счёт применения искусственного интеллекта и инструментов для работы с большими данными.

Так-как подготовка специалистов является основополагающим аспектом образовательной сферы, возникает необходимость в правильной и удобной для обучающихся – подготовке образовательной программы.

Уже существуют системы – способные помогать студентам получать информацию из записанных преподавателями лекций в видео формате – в текстовом виде. Это достигается за счёт использования технологий и инструментов распознавания речи – способных анализировать аудиоматериалы, распознавать в них человеческую речь и переводить её в текстовый формат представления данных.

В качестве одного из примеров таких систем может выступать CoursEra[-1] Это платформа онлайн обучения, основанная на рекомендательных системах. Она советует пользователям нужные и интересные для них курсы, способные привлечь их внимание.

Также в качестве примера рекомендательной системы в сфере образовательных услуг можно описать MathGarden [-1] – работающим с конкретными задачами для обучающихся. Он представляет собой тренажёр по арифметике для учеников начальных школ. Рекомендательная система этого ресурса – способна предложить ученику задачи – оптимально подобранные по текущему уровню освоения учеником учебной программы.

Рассматриваемая в данной работе рекомендательная система, призвана определить актуальность публикуемых статей, и дать рекомендации по увеличению этой актуальности.

В отличии от множества других рекомендательных систем описанных выше – данная рекомендательная система не направлена напрямую на улучшение образовательного процесса, но призвана помочь преподавателям и студентам получать информацию по их статьям, способную поднять университет в мировых рейтингах на несколько строчек выше – за счёт рейтинга цитируемости университетов QS.

QS – это общемировой рейтинг университетов мира основанный на количестве цитирования статей. При составлении данного рейтинга процент учёта цитируемости статей составляет 40%, что является весомым фактором, позволяющим утверждать, что описываемая система – позволит увеличивать индекс цитируемости за счёт рекомендаций по увеличению актуальности научных статей и продвигать университет в мировых строчках рейтингов. Также это позволит ВУЗ-у привлекать новых специалистов и учёных, что улучшит образовательный процесс и его качество.

Преподаватели и студенты, использующие данную систему, смогут получать дополнительные возможности повышения своей квалификации, а также получать гранты и награды, за свои научные работы.

Хотелось бы отметить также, немаловажный фактор – сохранения времени, которое без использования данной системы – научные сотрудники тратят на поиск информации о источниках статей, анализу цитируемости и актуальности, а также на время, уходящее на редактирование статей в целях поднятия её описываемых характеристик.

В наше время уже существует большое количество инструментов, позволяющих обрабатывать и анализировать текст применяя методы машинного обучения и статистического анализа текста. В рамках описываемой в данной работе – системы, наиболее подходящим является такой инструмент как: Yet Another Keyword Extractor [1]. Представляющий собой – легкий, автоматический метод извлечения ключевых слов, который основан на статистических характеристиках текста, извлеченных из отдельных документов, для выбора наиболее важных ключевых слов текста.

За счёт того, что данный инструмент использует уже предобученную нейронную сеть – этот метод, можно охарактеризовать как неконтролируемый и автоматический.

Так же, существует такой инструмент как: BERT [2, 3, 4, 5]. Являющийся одним из методов обработки естественного языка, основанный на использовании нейронных сетей. Каждое слово кодируется отдельным числом. Это делается для того, чтобы нейросеть могла работать с поданным на вход закодированным текстом. На практике чаще применяют для одного слова не одно число - а вектор, содержащий в себе более одного числа. Для того, чтобы задать нейросети "ассоциации" со словом, рядом с ним по числовому значению, ставят близкое по смыслу слово или набор слов так, чтобы разность между числовыми значениями этих слов была минимальна.

Оба этих инструмента используют обработку естественного языка (Natural language processing, NLP). Данная обработка относится к области искусственного интеллекта, которая занимается лингвистикой, чтобы дать компьютерам возможность понять, естественный человеческий язык.

Но также, необходимо упомянуть о наличии других всевозможных инструментов для обработки естественного языка, как например Natasha и Word2vec.

Natasha – представляет собой набор открытых инструментов для обработки естественного языка, работающий только с русскоязычными наборами текстов. Он способен осуществлять сегментацию на токены и предложения, морфологический и синтаксический анализ, лемматизацию и извлечение именованных сущностей.

В проекте 9 репозиториев. Natasha объединяет их под одним интерфейсом.

Эти репозитории включают:

1. Natasha — набор качественных открытых инструментов для обработки естественного русского языка. Интерфейс для низкоуровневых библиотек проекта;
2. Razdel — сегментация русскоязычного текста на токены и предложения;
3. Slovnet — deep learning моделирование для обработки естественного русского языка;
4. Navec — компактные эмбеддинги для русского языка;
5. Nerus — большой синтетический датасет с разметкой морфологии, синтаксиса и именованных сущностей;
6. Corus — коллекция ссылок на публичные русскоязычные датасеты + функции для загрузки;
7. Naeval — количественное сравнение систем для русскоязычного NLP;
8. Yargy-парсер — структурированное извлечение информации из текстов на русском языке с помощью грамматик и словарей;
9. Ipymarkup — визуализация разметки именованных сущностей и синтаксических связей.

Word2vec – Представляет собой метод эффективного создания вложений способный работать со словами и имеющий эффективные концепции в разработке рекомендательных механизмов, способных придать смысл данным в любых задачах. Использует векторные представления слов на естественном языке.

Как пример рассмотрим векторное представление слов “Король”,” Мужчина”,” Женщина”. Заместо числового представления будем использовать графическое представление в виде тепловой карты.



Рис. 2 Пример соответствия цвета и числа.

В результате применения Word2vec получаем следующий результат:

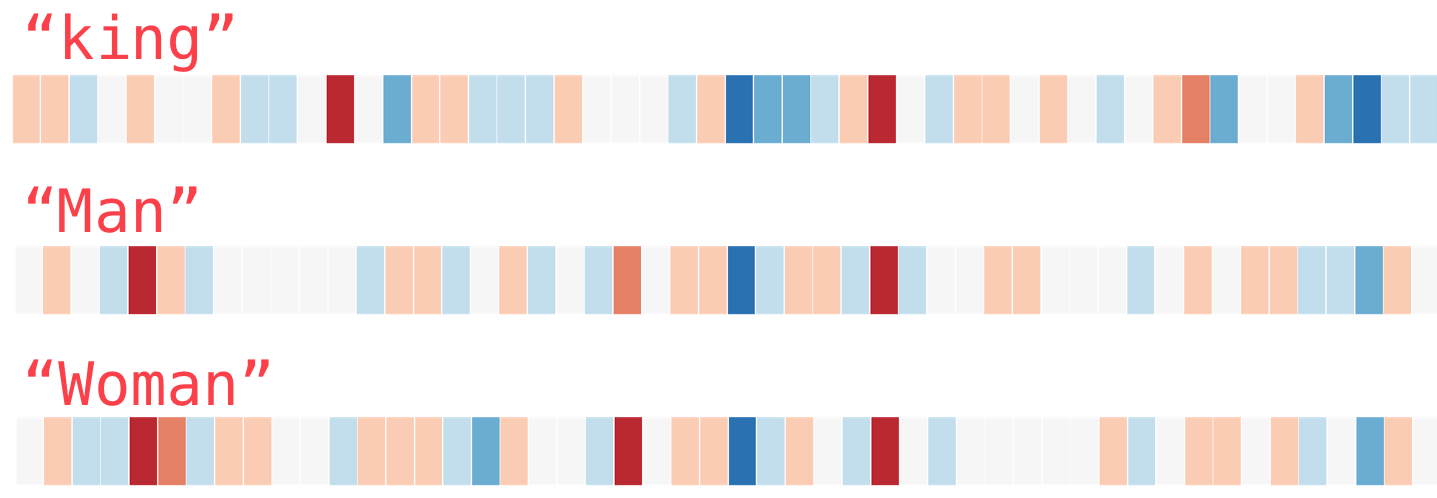


Рис. 2 Пример представления понятий в векторном представлении с использованием тепловой карты.

Из примера на Рис.1 видно, что мужчина и женщина имеют гораздо больше сходств между собой и стоят ближе, чем к королю.

## **Обоснование направления разработки и требования, предъявляемые к системе**

Как отмечалось ранее, описываемая рекомендательная система призвана помочь научным сотрудникам и студентам определять актуальность их научных публикаций. Для этого необходимо применение нейронных сетей и инструментов обработки естественного языка, которые смогли бы помочь нам анализировать текст научных публикаций, и формировать рекомендации пользователям. А также, переобучать нейронные сети, с помощью алгоритма обратного распространения ошибки, для получения более точных и достоверных результатов формирования рекомендаций.

### **Функциональные требования.**

Призваны сформировать состав функций – которая должна обеспечивать информационная система. В результате анализа сформированы следующие требования:

1. Авторизация в систему;
2. Определение прав доступа;
3. Загрузка статьи на анализ;
4. Анализ статьи;
5. Отклонение рекомендаций;
6. Принятие рекомендаций;
7. Отклонение предложенных для изменения результатов;
8. Принятие предложенных для изменения результатов;
9. Редактирование параметров YAKE (для администраторов системы);
10. Редактирования списка пар ключевых слов (для администраторов системы);

### **Бизнес требования.**

Данные требования необходимы для определения целей, которые должно достичь предприятие ­– за счёт внедрения данной информационной системы.

Сформулированы следующие бизнес-задачи:

1. Сократить время на анализ индекса цитируемости;
2. Сократить время на поиск источников;
3. Помочь пользователям редактировать статьи в целях повышения актуальности.

### **Нефункциональные требования.**

Состав не функциональных требований:

1. Отказоустойчивость;
2. Обработка исключительных ситуаций;
3. Система должна быть гибка и легка в использовании;
4. Быстрота работы и скорость выполнения запросов;
5. Полная отчётность о проведённом анализе;
6. Высокий уровень оптимизации и низкий уровень нагрузки на серверное оборудование;
7. Интуитивно понятный интерфейс;
8. Нахождение нужной информации не более чем в 3 клика.

## **Анализ исходных данных и выбор варианта построения системы**

Основная цель разрабатываемой информационной системы состоит в том, чтобы, получив от пользователя публикацию, на основе её текста, определить и дать рекомендации по изменению содержания статьи с целью повышения её актуальности.

Важно отметить, что система, может осуществлять взаимодействие с пользователем, через веб - интерфейс. Данный интерфейс - может быть размещён как на учебном портале университета, так и иметь собственную, отдельную площадку. Такая возможность, существует благодаря разработке RESTful API в описываемой системе. Данный подход гарантирует дальнейшую масштабируемость взаимодействия компонентов информационной системы за счёт их независимого внедрения.



Схема 1. RESTful API сервис.

При загрузке научной статьи или публикации в систему, последняя в свою очередь – после обращения к RESTful API, сможет подсказать пользователю:

* Какие ключевые слова можно было бы добавить или удалить;
* К какой тематике относится его статья;
* Что можно было бы добавить или исправить в тексте научной публикации.

При несоответствии предложенного рекомендательной системой результата – с мнением пользователя – последний в свою очередь сможет внести коррективы в предложенный результат. Так же, планируется сбор информации об отклонении или принятии пользователем предложенных рекомендаций.



Рисунок 2. Диаграмма BPMN.

**Статистика Российского образования.**

**Высшее профессиональное образование.**

Система **высшего образования** объединяет 607 государственных и 358 негосударственных вузов, в которых обучается 4,7 млн. человек. После относительного спада в середине 90-х годов число студентов ежегодно растет. В 2000 г. число студентов на 10 тысяч населения составило 327 человек, что значительно выше, чем соответствующий показатель 1995 г. - 189 студентов на 10 тысяч населения. Вузы России готовят специалистов по более чем 350 специальностям.  
Преподавательский состав государственных ВУЗов включает 265 тысяч человек, из которых 153 тысячи человек - специалисты высшей квалификации (кандидаты и доктора наук). В негосударственных ВУЗах занято более 42 тысяч преподавателей.

**Аспирантура и докторантура.**

Система **послевузовского образования** готовит специалистов высшей квалификации. Сегодня в системе обучается 118 тысяч человек аспирантов и 4, 2 тысячи докторантов, ежегодный выпуск из аспирантуры составляет почти 25 тысяч человек, из докторантуры - 1,3 тысячи человек. 30 процентов аспирантов и около 40 процентов докторантов по окончании данного образования защищают диссертацию.  
Растет привлекательность данных видов образования, что выражается ежегодным приростом числа аспирантов и докторантов. Так в 1992 году в аспирантуре обучалось всего 52 тысячи человек, в докторантуре -1,6 тысячи человек.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Партнеры**   * Технологический Университет. * Спец книжные магазины; * Спец порталы; | **Решение** | **Ценностное предложение**  Наше приложение, помогает ученым, которые публикуют статью, повысить индекс цитируемости этой статьи, делая их более популярнее и за счет этого повышать шансы на получение грантов. Для этого используется ИИ) | **Скрытое преимущество**   * Скрытые пути получения информации; * Непосредственная близость с аудиторией потребителей; | **Потребители**   * Ученые |
| **Ключевые метрики**   * Процент нагрузки; * Кол-во пользователей; * Точность рекомендаций; * Кол-во соглашений с рекомендациями; * Кол-во правок рекомендаций;   **Ключевые ресурсы**   * Технология * Патент * Прог. код * Человеческие ресурсы * Гранты | **Канал продаж**   * Социальные сети; * Прямые контакты через ректора; * Прямые рекомендации научных сообществ ВУЗов * Реклама через тематические сайты; |
| **Расходы**   * Затраты на хостинг; * Затраты на выделение кадров; * Затраты на вспомогательное ПО; | | | **Потоки выручки**   * Подписка * Freemium * Рекламная модель | |

**Гипотезы:**

1. При уникальности ниже 70% пользователь начинает сам редактировать публикации (30 страниц в среднем в статье, 10 минуту пользователь тратит на 1 страницу. Пользователь тратит 5 ч.)
2. Наш пользователь самостоятельно анализирует различные статьи чтобы определить популярность своей статьи. (10 мин 1 страницы, 30 страниц в среднем, статью необходимо прочитать 2 раза мин. для анализа 5 статей потребуется 25 часов)
3. Наш пользователь самостоятельно редактирует свою статью для повышения ее актуальности среди прочих статей.
4. Пользователь вынужден самостоятельно искать источники информации для своей статьи

## **Выводы по аналитической части**

Была изучена предметная область, выявлены уже существующие рекомендательные системы и определена актуальность тематики рекомендательных систем – во многих областях, в том числе в сфере образовательных услуг. Исходя из этих данных и результатов проделанной на данный момент времени работы – можно сказать, что у описанной системы – не существует аналогов, так-как тематика рекомендательных систем в сфере образовательных услуг – находится на ранних стадиях развития.

**Проектная часть**

- Разработка и расчет структурной схемы системы (компонента системы) 10-12;

**Средства разработки:** IntelliJ IDEA, SUBLIME TEXT 3, WORK Bench mysql, NotePad++, Adobe Photoshop, PyCharm, Jupiter Notebook, WebStorm, Docker, Git.

IntelliJ IDEA - интегрированная среда разработки программного обеспечения на многих языках программирования, в частности Java, Go, JavaScript, Python, Logtalk разработанная компанией JetBrains.

***Поддерживаемые языки***

* Java
* JavaScript
* CoffeeScript
* HTML/XHTML/HAML
* CSS/SASS/LESS
* XML/XSL/XPath
* YAML
* ActionScript/MXML
* Python
* Ruby
* Haxe
* Groovy
* Scala
* SQL
* PHP
* Kotlin
* Clojure
* C
* C++

Разработанную систему планируется запускать на серверной виртуальной машине, под управлением Ubuntu 18.04, 4GB RAM, Intel Xeon с выделенными 4-мя ядрами. В процессе опытной эксплуатации, доказано, что данные характеристики полностью удовлетворяют ресурсо-затратам описываемой информационной системы и подходят в качестве рекомендованных системных требований.

Минимальные системные требования:

* Intel core i3;
* 4GB RAM DDR3;
* 35МБ свободного места на носителе.

Структура компонентов системы:



Рисунок 4. Диаграмма развёртывания.

# C:\Users\Manus\Desktop\dich\SpringMVCController.jpg

Рисунок 1. Основой принцип работы приложения.











Требования к ПО:

* ОС: WindowsXP и выше;
* Браузер: Google Chrome / Mozilla Firefox / Opera/ Internet Explorer (версии 10 и выше)

**Подготовка к работе.**

Порядок загрузки данных и программ.

* Перед началом работы необходимо проверить наличие подключения к интернету на ПК.
  + 1. **Входные данные.**
* Статьи в формате электронных документов;
  + 1. **Выходные данные.**
* Сформированные веб-страницы, сертификаты подтверждения.

**Описание операций.**

Выполняемые функции и задачи.

* Сайт выполняет следующие функции:

1. Предоставление пользователям информации по актуальности их статей;
2. Рекомендации по увеличению этой актуальности;
3. Выгрузка именных сертификатов;
4. Мониторинг трендов актуальностей классов;
5. Администрирование системы в целом.
   * 1. **Описание операций технологического процесса.**

Запуск. При запуске системы появляется главная страница портала.

Навигация по сайту. Для навигации по системе используется главное меню, расположенное в верхней части сайта, а также боковое меню с функционалом конкретного пользователя.

Завершение работы. Для завершения работы с системой достаточно закрыть окно браузера.

**Аварийные ситуации.**

* Большая нагрузка, после чего следует перезагрузка сервера;
* Обработка Http ошибок.

**Рекомендации к освоению.**

Для успешной работы требуется иметь опыт работы с любым приложением Windows и внимательно изучить данное руководство.

## **Результаты решения задачи**

Результаты решения задачи представлены в разделе «Приложение»

**Описание структурных единиц результатной информации.**

Веб-страницы:

* Главная страница – экран загрузки статьи;
* Мониторинг – Мониторинг трендов актуальностей по классам;
* Настройки – Yet Another Keyword Extractor;
* О системе – краткое сопроводительное письмо описывающее назначение системы;
* Результаты фильтрации – содержит информацию о результатах фильтрации ключевых наборов словосочетаний, результаты анализа Yet Another Keyword Extractor-ом, список классов и их актуальности;
* История загрузок – история загруженных пользователем публикаций и сгенерированных сертификатов.

- Расчеты устройств, разработка алгоритмов и программ 10-12;

1. На вход поступает публикация с выделенными в ней ключевыми словами.

Важно учитывать, что статьи должны включать разделы:

* Ключевые слова;
* Основной текст.

1. После загрузки статьи через интерфейс, происходит подготовка её текста к отправке в API – Yet Another Keyword Extractor, развёрнутому в Docker Container-е. Данная процедура - поможет нам определить предполагаемые ключевые слова, на основе текста статьи. На сервере, данная обработка начинается с этапа фильтрации текста за счёт использования инструмента языка Java для программной обработки документов Microsoft, носящего название – Apache POI [8]. Также, если на вход поступает публикация в формате файла PDF, будет использоваться такой инструмент, как IText [11], позволяющий получить текст публикации из файла формата PDF.

После выгрузки текста из файла, он проходит несколько этапов обработки:

* 1. Замена всех переносов параграфов в тексте на «.» для лучшего распознавания текста Yet Another Keyword Extractor-ом;
  2. Удаление англоязычных блоков текста;
  3. Осуществление поиска по всему тексту публикации, с целью выделения нужных для работы, его составных частей таких как:
     1. Блок текста содержащий название статьи;
     2. Блок текста содержащий ключевые слова;
     3. Блок текста содержащий ссылки на источники;
     4. Блок текста содержащий основной текст публикации;
  4. Из найденных частей текста, составляется готовый к анализу текст научной публикации, который будет содержать только необходимые нам блоки текста, загруженной научной публикации, что в свою очередь позволит нам, используя Yet Another Keyword Extractor — более точно определить ключевые слова анализируемой публикации.

1. На основании ключевых слов определяется принадлежность публикации к классам.
2. Выборка сохранённых значений актуальности по классам
3. Формирование рекомендации на основании полученных значений актуальности.

Так — же, разработаны вспомогательные алгоритмы:

1. Определения классов, по ключевым словам;
2. Определения актуальности по классам;
3. Формирования рекомендации по актуальности.

В графическом представлении весь алгоритм можно представить следующим образом:

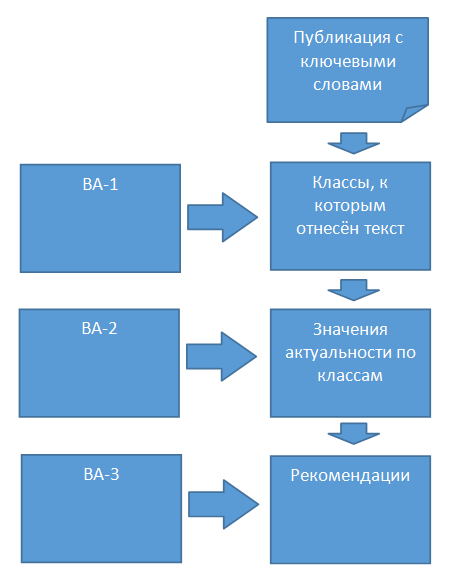


Схема 2. Основной алгоритм.

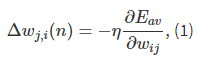
### Алгоритм обратного распространения ошибки

Популярный алгоритм обучения плоскослоистых нейронных сетей прямого распространения (многослойных персептронов). Относится к методам обучения с учителем, поэтому требует, чтобы в обучающих примерах были заданы целевые значения. Также является одним из наиболее известных алгоритмов машинного обучения.

В основе идеи алгоритма лежит использование выходной ошибки нейронной сети

Для вычисления величин коррекции весов нейронов в ее скрытых слоях, где k — число выходных нейронов сети, y — целевое значение, y′ — фактическое выходное значение. Алгоритм является итеративным и использует принцип обучения «по шагам» (обучение в режиме on-line), когда веса нейронов сети корректируются после подачи на ее вход одного обучающего примера.

На каждой итерации происходит два прохода сети — прямой и обратный. На прямом входной вектор распространяется от входов сети к ее выходам и формирует некоторый выходной вектор, соответствующий текущему (фактическому) состоянию весов. Затем вычисляется ошибка нейронной сети как разность между фактическим и целевым значениями. На обратном проходе эта ошибка распространяется от выхода сети к ее входам, и производится коррекция весов нейронов в соответствии с правилом:



где Δwj,i — вес i-й связи j-го нейрона, η — параметр скорости обучения, который позволяет дополнительно управлять величиной шага коррекции Δwj,i с целью более точной настройки на минимум ошибки и подбирается экспериментально в процессе обучения (изменяется в интервале от 0 до 1).

Учитывая, что выходная сумма j-го нейрона равна:



можно показать, что:



Из последнего выражения следует, что дифференциал ∂Sj активационной функции нейронов сети f(s) должен существовать и не быть равным нулю в любой точке, т.е. активационная функция должна быть дефференцируема на всей числовой оси.

Поэтому для применения метода обратного распространения используют сигмоидальные активационные функции, например, логистическую или гиперболический тангенс.

Таким образом, алгоритм использует так называемый стохастический градиентный спуск, «продвигаясь» в многомерном пространстве весов в направлении антиградиента с целью достичь минимума функции ошибки.

### **2.2 АЛГОРИТМ МЕТОДА ОПРЕДЕЛЕНИЯ КЛАССОВ.**

Входные параметры: список (массив) ключевых слов.

Выходные параметры: список (массив) классов.

Логика работы:

Производится преобразование входного списка в выходной, при использовании вспомогательной таблицы №1. Столбцы таблицы – классы, строки таблицы – ключевые слова, значения не пересечении столбцов и строк – веса соответствия, по умолчанию равные нулю.

1. Формируется пустой список для классов.
2. Для каждого ключевого слова во входном списке ищется это ключевое слово в таблице.
   1. Если ключевые слова закончились, возвращается результирующий список классов.
   2. Если ключевого слова нет в таблице – добавляется новая строка, все значения в этой строке заполняются нулями. Осуществляется переход к следующему ключевому слову.
   3. Если ключевое слово есть, то находятся классы, для которых значения в соответствующей строке не равны нулю.
3. Для каждого найденного класса проверяется, есть ли он уже в списке классов.
   1. Если класса нет в списке, он сохраняется в базу данных, со значение из таблицы.
   2. Если класс есть в списке, увеличивается значение на значение из таблицы.

Для первоначального определения классов и их значений на пересечении с ключевыми словами используется алгоритм кластеризации *k*-средних.

Пример работы алгоритма:

Вспомогательная таблица №1:

Таблица 1.

|  |  |  |
| --- | --- | --- |
|  | Класс1 | Класс2 |
| Физика | 1 | 0 |
| Математика | 1 | 1 |

Входной список ключевых слов: {математика, химия}.

Выходной список классов после работы алгоритма: {Класс1 : 1, Класс2 : 1}.

Вспомогательная таблица №1 после работы алгоритма:

Таблица 2.

|  |  |  |
| --- | --- | --- |
|  | Класс1 | Класс2 |
| Физика | 1 | 0 |
| Математика | 1 | 1 |
| Химия | 0 | 0 |

### **2.3 АЛГОРИТМ МЕТОДА ОПРЕДЕЛНИЯ АКТУАЛЬНОСТИ.**

Входные параметры: список (массив) классов.

Выходные параметры: список (массив) пар «класс-актуальность».

Логика работы:

Используется вспомогательная таблица №2, со столбцами «Класс» и «Актуальность».

1. Для каждого класса во входном списке в выходной добавляется - список пар «класс-актуальность».
2. Для формирования таблицы №2, используется алгоритм опроса поискового API Yandex.xml и Google Search API, которое по имени класса возвращает количество результатов поиска. Данное значение будет сохранено во вспомогательную таблицу №2 как значение актуальности класса. Данный алгоритм опроса реализован в виде независимой прослойки, что позволяет легко изменять принцип его работы и внедрять его в другие модули приложения.

Все вспомогательные таблицы, описанные в данных алгоритмах – расположены в СУБД MySQL. Доступ к ней осуществляется благодаря использованию Spring JDBC.

Для определения списка используемых в таблице тем *t* и значений *r* в ячейках таблицы используется алгоритм кластеризации *k*-средних на множестве цифровых представления ключевых словосочетаний *w*, полученных с помощью одного из вышеописанных источников [5,7]. Центр каждого найденного кластера соотносится с темой, название которой определяется с использованием экспертного мнения. Соответствие *r* между ключевым словосочетанием *w* и темой *t* определяется как величина, обратная расстоянию между их цифровыми представлениями.

*Четвёртый метод* – определение актуальности темы по её названию. Актуальность темы *p* трактуется как величина, прямо пропорциональная количеству документов, хранимых в сети Интернет, в которых упоминается данная тема.

Инструментом, реализующим четвёртый метод, является написанный на языке программирования Java с использованием базы данных MS SQL алгоритм, который по названию темы *t* получает количество документов в сети Интернет, связанных с данной темой, и сохраняет это количество в качестве значения актуальности *p* с привязкой к теме *t* в базу данных.

Чтобы использовать алгоритм кластеризации *k*-средних для формирования списка тем, требуется список ключевых слов размеров минимум в несколько сотен, причём ключевые слова должны быть взяты из научных статей различной тематики. Безусловно, список ключевых слов большего размера приведет к более надёжным результатам. Значение величины *k* выбирается не более 10% от размера списка ключевых слов, при этом максимальное значение, используемое в текущей версии информационной системы, равно 200.

Общий алгоритм системы

Пользователь с помощью графического интерфейса загружает текст научной статьи *A* на русском языке в информационную систему, которая, используя первый описанный в разделе 3 метод – статистический анализ текста, извлекает 10 фраз длинной не более 5 слов, которые трактуются как кандидаты в ключевые слова данной статьи.

Данный список подергается корректировке с применением второго из описанных в разделе 3 методов – искусственной нейронной сети, в результате чего для каждого из кандидатов в ключевые слова определяется его класс. Те пункты списка, которые отнесены к классу «отвергнуто пользователем», исключаются из списка.

Полученный список кандидатов в ключевые слова выводится в графический интерфейс, после чего пользователь может внести следующие изменения в список: удалить те пункты, которые, на его взгляд, не соответствуют тексту статьи или являются слишком общими, или добавить дополнительные ключевые словосочетания длиной не более 5 слов. Закончив редактирование списка пользователь подтверждает его, после чего текст статьи и финальный список ключевых словосочетаний сохраняются в базу данных, в том числе, для дальнейшего дообучения искусственной нейронной сети.

### **Алгоритм фильтрации**

## Описание алгоритма

1. Из имеющегося набора научных статей извлекаются тексты и списки ключевых словосочетаний и записываются все в один массив. Для этого можно использовать любые подходящие библиотеки Python. Изначально необходимо уметь извлекать ключевые словосочетания из файлов doc/docx, в дальнейшем – также и из pdf.
2. Для каждого текста и каждого ключевого сочетания в получившемся массиве вычисляются их цифровые представления. Первая версия: цифровое представление словосочетания равно среднему арифметическому цифровых представлений отдельных слов; цифровая версия текста – вектор из цифровых представлений отдельных предложений. Для вычисления цифровых представлений можно использовать один из инструментов: word2vec, BERT, Natasha и др.

Важно учесть, что цифровое представление словосочетания и предложения могут быть как отдельными числами, так и числовыми векторами.

1. Формируется выборка из пар (текст, ключевое словосочетание, класс) таким образом, чтобы в неё попали все возможные сочетания текстов и ключевых словосочетаний.
2. Первоначально необходимо иметь следующие выборки:
   1. Валидационную;
   2. Тестовую.

В соотношении 60% - 20% - 20% случайным образом.

1. Формируется структура нейронной сети прямого распространения, исходя из максимально возможного числа предложений в текстах. Например, если максимальное число предложений в текстах равно 100 и в качестве цифрового представления словосочетаний и предложений используются отдельные числа, то на входе сети будет 101 нейрон, а вся структура сети может иметь вид: 101 – 50 – 20 – 1. В качестве активационных функций скрытых слоёв ReLU, выходного слоя – сигмоиду. Количество скрытых слоёв и нейронов в них необходимо установить эксперементальным путём.
2. Обучение нейронной сети на обучающей выборке. С помощью валидационной выборки подбираются параметры нейронной сети, которые приводят к наилучшему значению метрики (используем F1-меру).

### **2.4 АЛГОРИТМ МЕТОДА ФОРМИРОВАНИЯ РЕКОМЕНДАЦИИ.**

Входные параметры: список (массив) пар «класс-актуальность».

Выходные параметры: текст рекомендации.

Логика работы:

1. Необходимо начать с пустого текста рекомендации.
2. Осуществляется сравнение актуальности каждого класса из входного списка с пороговым значением
3. Если пороговое значение превышено, добавляется текст формата «класс – актуальность» к тексту рекомендации. По умолчанию пороговое значение равно 0.

Важно отметить, необходимость сделать редактируемым на этапе обучения — список получаемых из Yet Another Keyword Extractor ключевых слов, с возможностью корректировки как самого ключевого слова, так и значения его «важности».

После внесения всех необходимых изменений, предусмотрено сохранение результатов анализа классов и процесса выделения ключевых слов в базу данных, вместе с привязкой к тексту публикации и параметрами Yet Another Keyword Extractor, использованными для получения ключевых слов. А также сохранение списка пар класс-актуальность и самой сформированной рекомендации.

**Нормировка чисел**

Для нормировки больших чисел и приведения их к интервалу [0..1] используем следующий алгоритм:

Определяем набор классов , где , для каждого из которых путём запроса в поисковй API определена актуальность .

Тогда, для определения актуальности набора ключевых слов , где , используем следующий алгоритм:

1. Нормируем актуальности , используя функцию softmax:
2. Для каждого ключевого слова вычисляем расстояния от него до каждого из классов:
3. Нормируем расстояния для каждого ключевого слова :

1. Актуальность ключевого слова вычисляем следующим образом:

Итого, значения нормированных актуальностей классов принадлежат отрезку [0,1], значения нормированных расстояний до классов для каждого ключевого слова – аналогично.

Так как, суммы нормированных значений равны 1 их можно трактовать как вероятности.

Актуальности ключевых слов также принадлежат отрезку [0,1], они больше, если больше актуальности классов, расстояние до которых меньше.

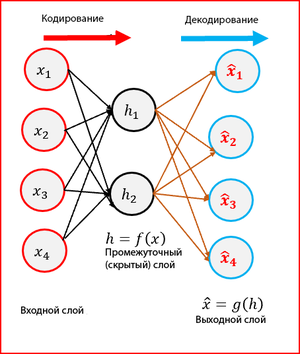
@UtilityClass  
public class MathUtil {  
 public double[] stableSoftmax(final double[] args) {  
 final double[] log = Arrays.*stream*(args)  
 .map(Math::*log*)  
 .toArray();  
  
 final double[] numenator = Arrays.*stream*(Arrays.*stream*(log)  
 .map(operand -> operand - Arrays.*stream*(log)  
 .max()  
 .orElseThrow(IllegalArgumentException::new))  
 .toArray()  
 ).map(Math::*exp*)  
 .toArray();  
  
 return Arrays.*stream*(numenator)  
 .map(operand -> operand / Arrays.*stream*(numenator).sum())  
 .toArray();  
 }  
}

## Нейросетевой фильтр

Также, как и Yet Another Keyword Extractor – является обособленной частью системы и взаимодействует с ней по средством HTTP запросов.

Используется для фильтрации ключевых словосочетаний полученных из Yet Another Keyword Extractor и позволяет понять - какие из ключевых словосочетаний пригодны для дальнейшей обработки.

Построен с использованием архитектуры сети «Автокодировщик» (рис. 2) и обучен на авторских ключевых словосочетаниях.



**Рисунок 2 – Структура сети** «**Автокодировщик**»

Данная архитектура нейросети – позволяет применять обучение без учителя используя метод обратного распространения ошибки.

Также, данная архитектура выбрана благодаря тому, что позволяет эффективно работать с текстовым представлением информации. В результате множества проведённых экспериментов, данная архитектура очень хорошо проявила и зарекомендовала себя.

После этапа обучения формируется динамический порог, который должны будут проходить ключевые фразы во время фильтрации.

- Экспериментальные исследования и тестирование программ 5-10;

В следствии проведённых экспериментов установлено – что оптимальное количество слоёв для качественного обучения и функционирования нейросети должно быть равным 3 слоям. В их состав входят:

1. Входной слой;
2. Скрытый слой;
3. Выходной слой.

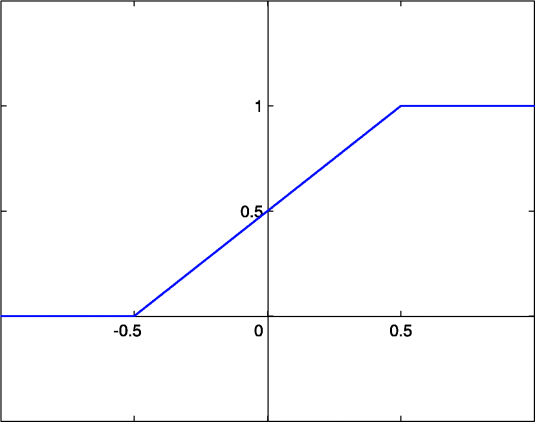
Важно отметить, что, работая с текстом, нам необходимо преобразовывать текст в цифровое представление – эмбеддинги каждый из которых – представляет собой числовое множество в многомерном пространстве.

Количество нейронов входного и выходного слоя – соответствует размеру эмбеддинга ключевых словосочетаний и должно быть равным 300 нейронам.

На скрытом слое было принято решение использовать 100 нейронов. Количество эпох на этапе обучения было выбрано 40, а активационные функции для входного и скрытого слоя – ReLU, и выходного слоя – линейная.

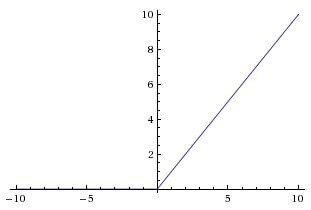
Линейная функция активации:

Представляет собой прямую линию и пропорциональна есть взвешенной сумме нейрона.



**A = cx**

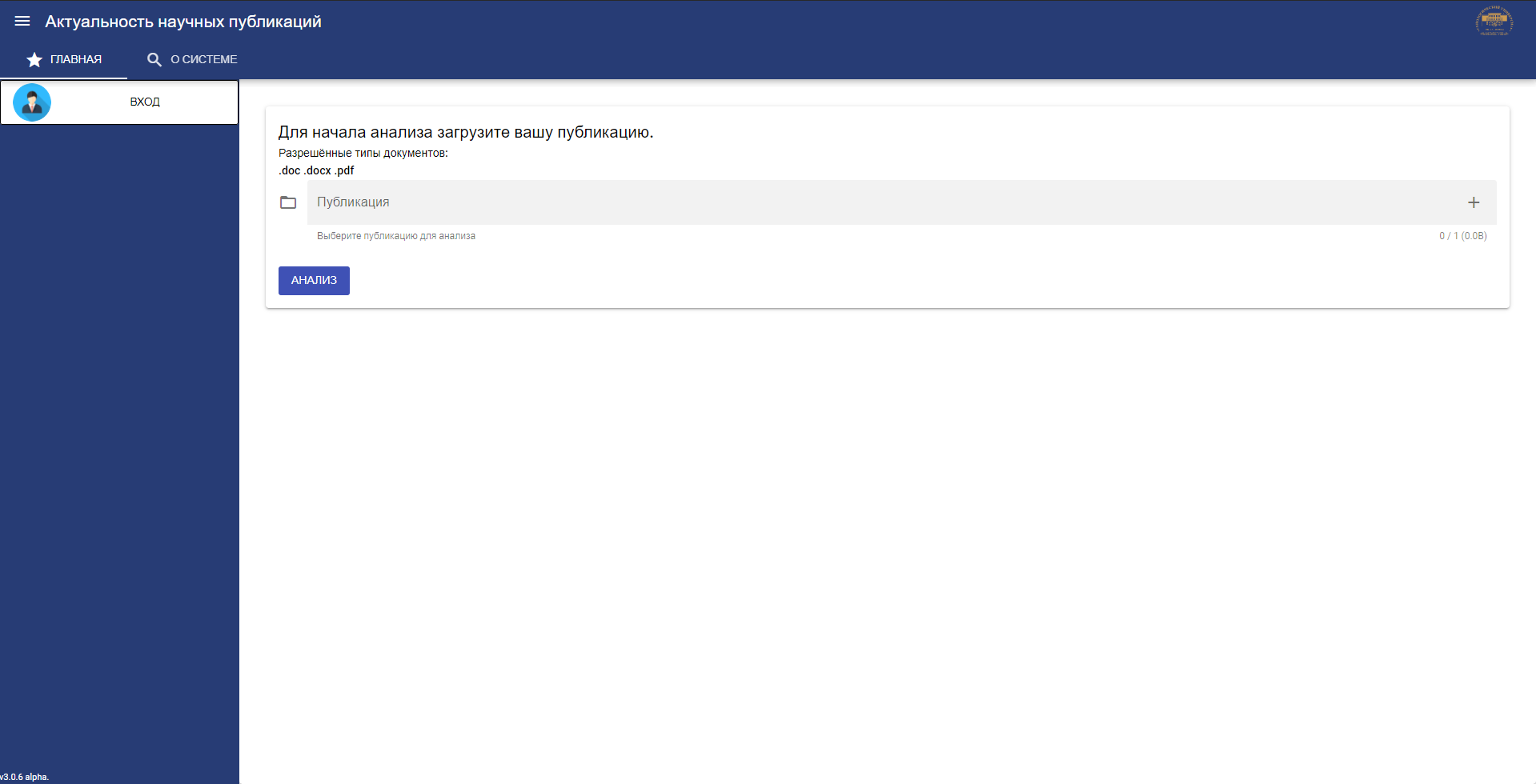
ReLu

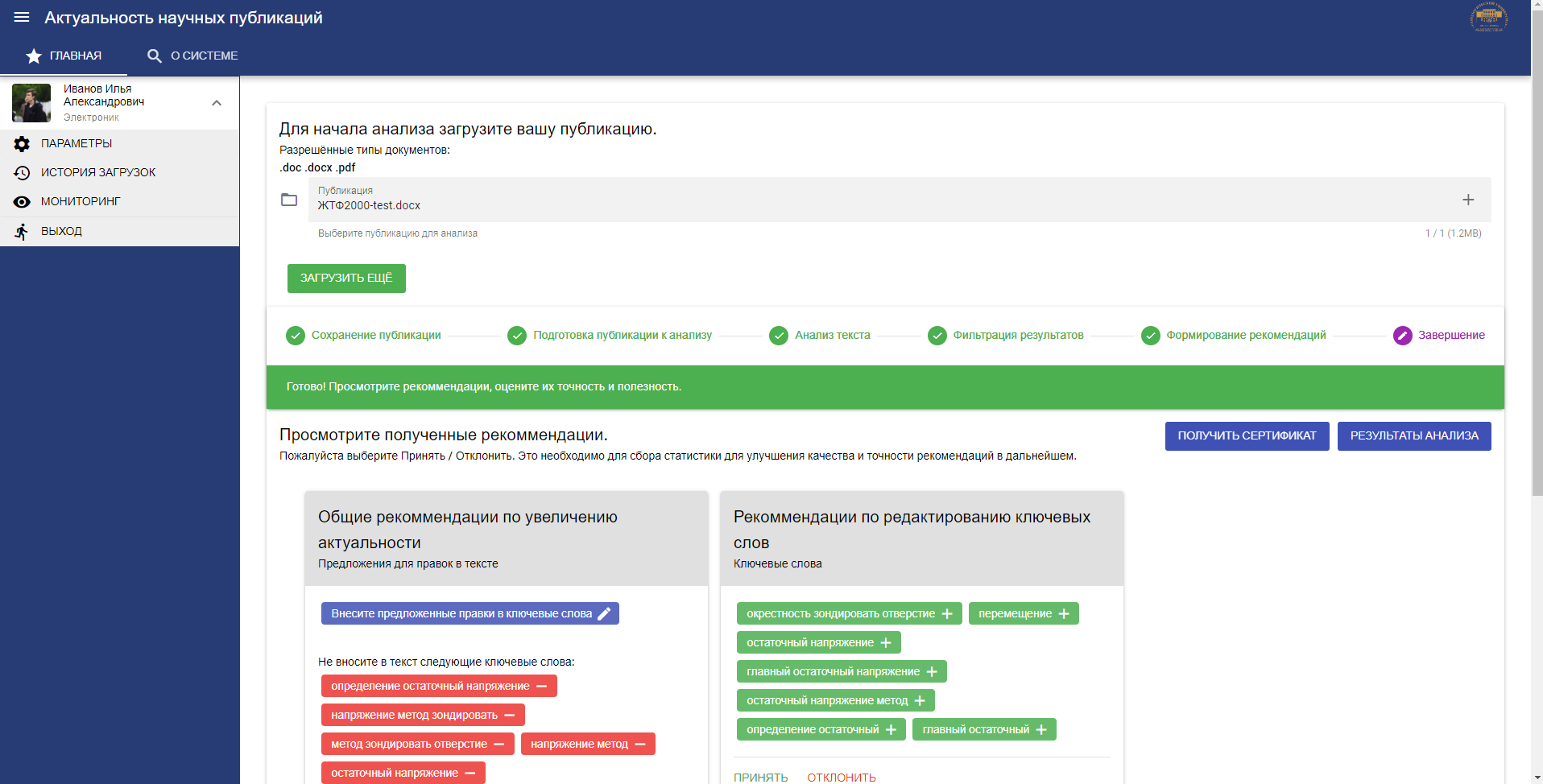


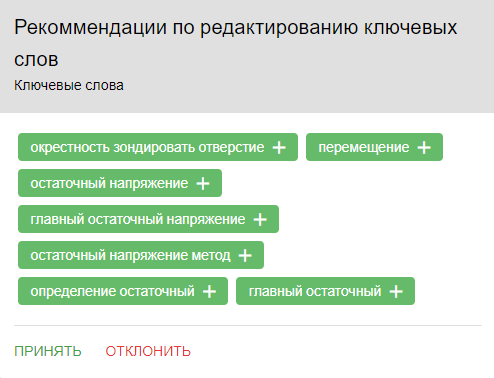
**f (x) = max(0, x**)

Экспериментальным путём установлено, что описанные выше параметры сети, показывают наилучшую эффективность при обучении. Это доказано на основе метрик оценки качества обучения сети. Одна из таких метрик – среднеквадратичная ошибка изменение которой – можно было наблюдать на графике сходимости во время обучения сети.

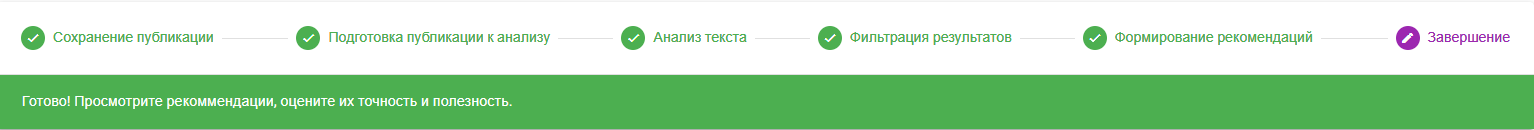
- Особенности конструкции и эргономические характеристики 5-10;

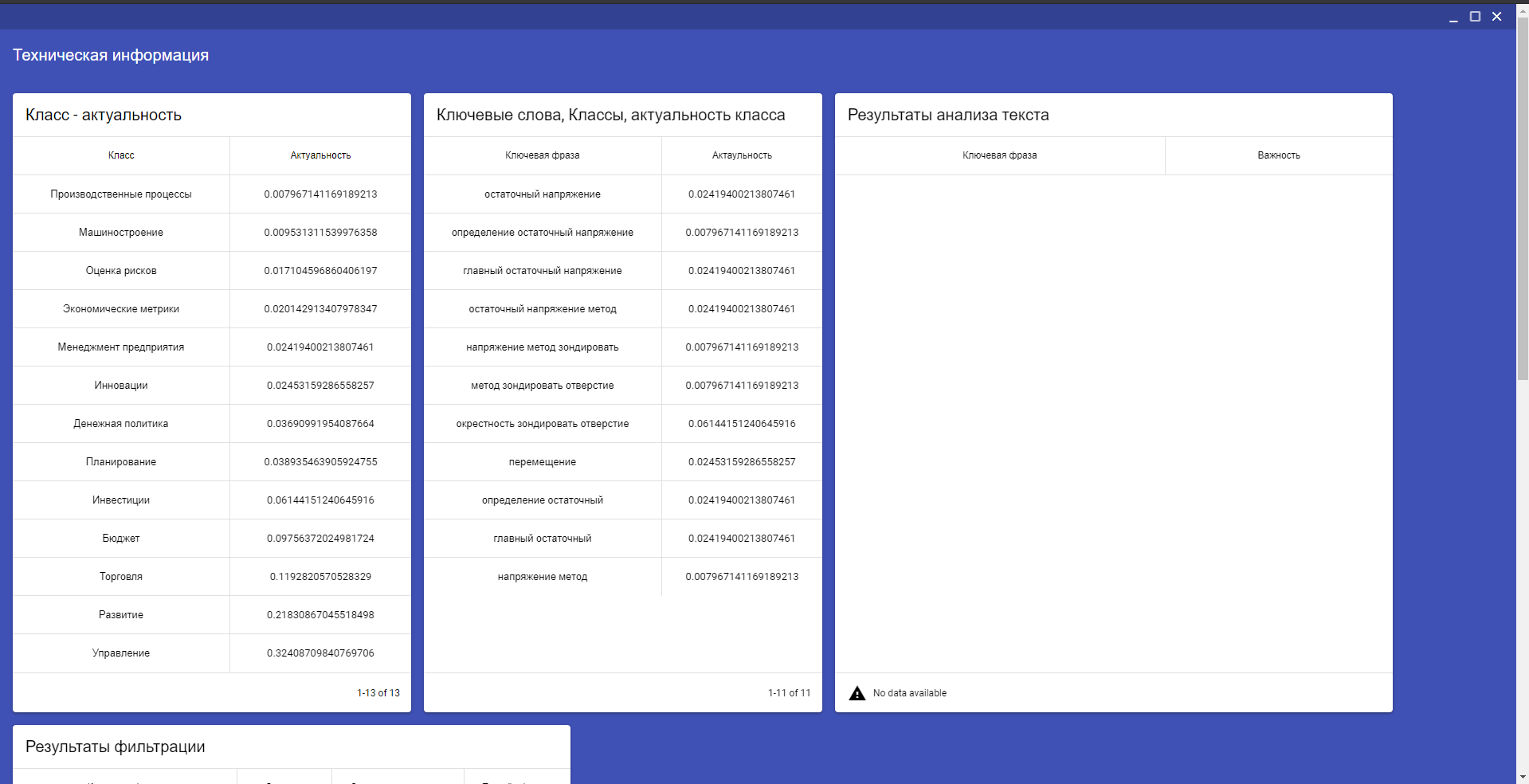


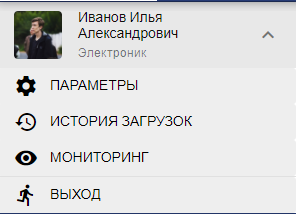


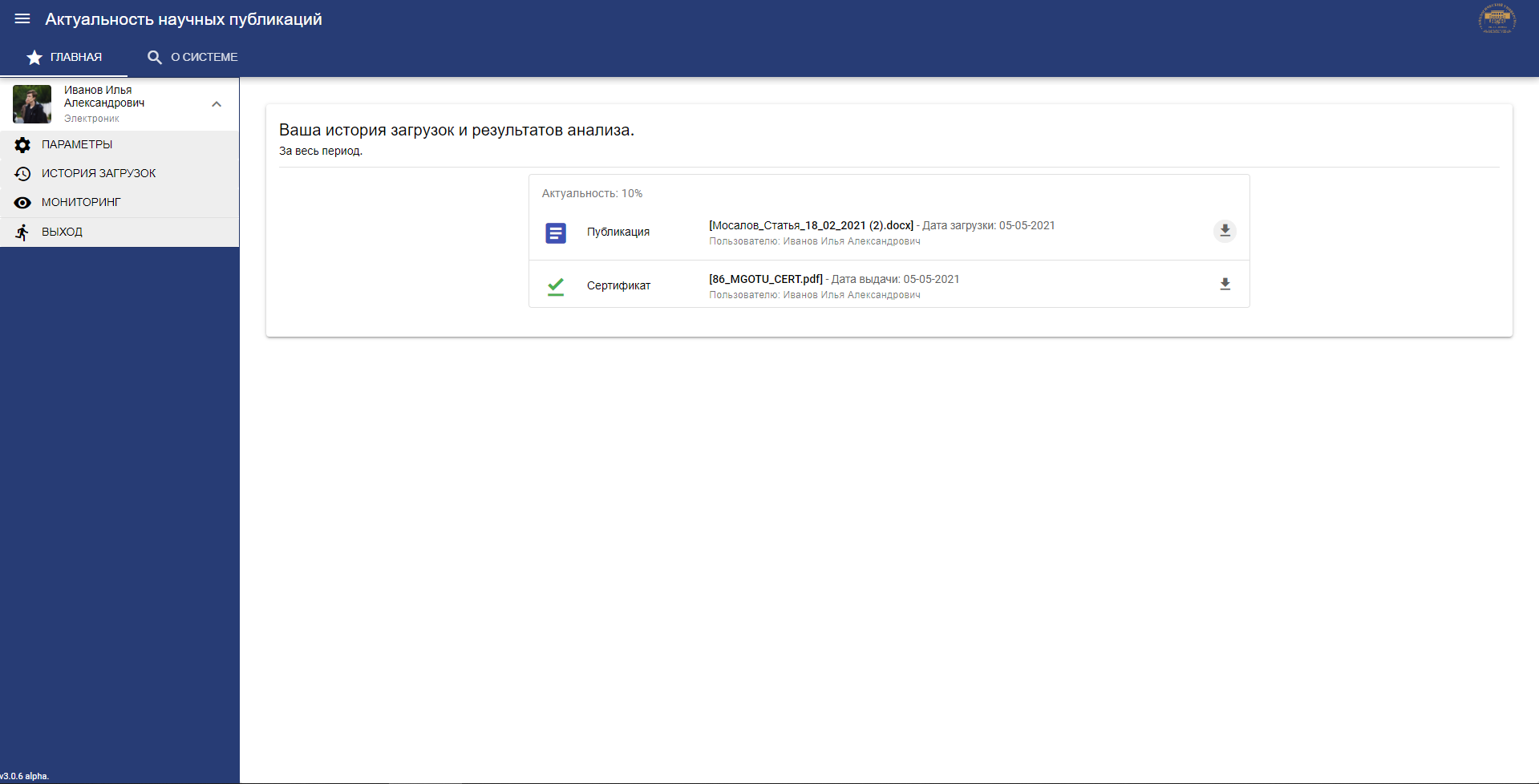
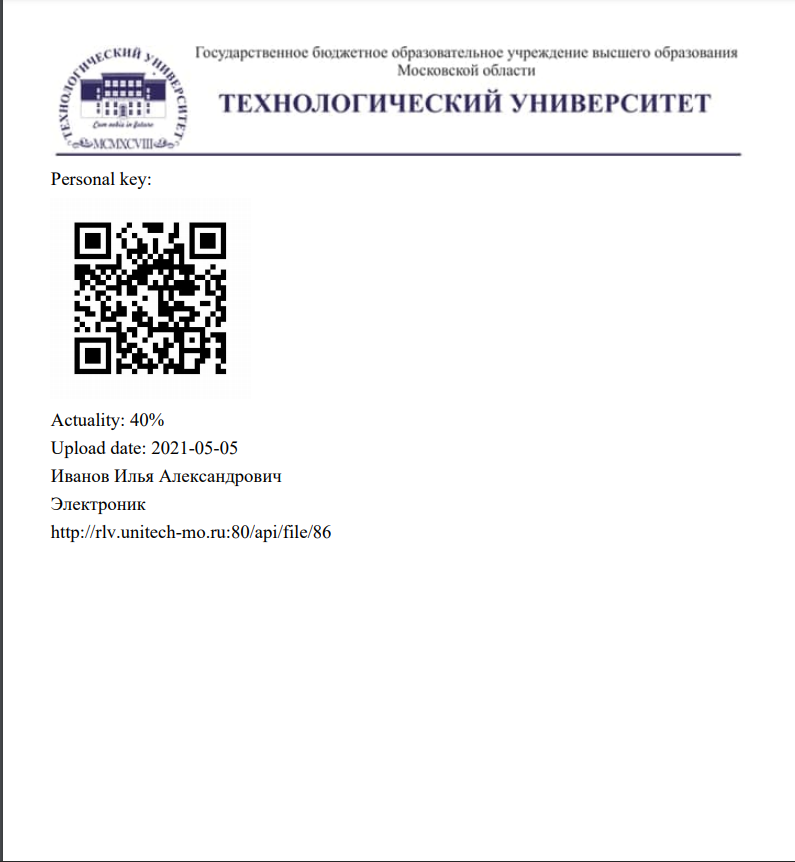


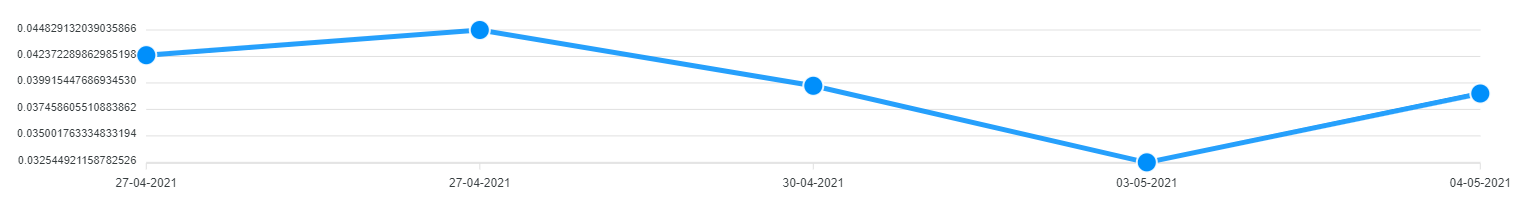
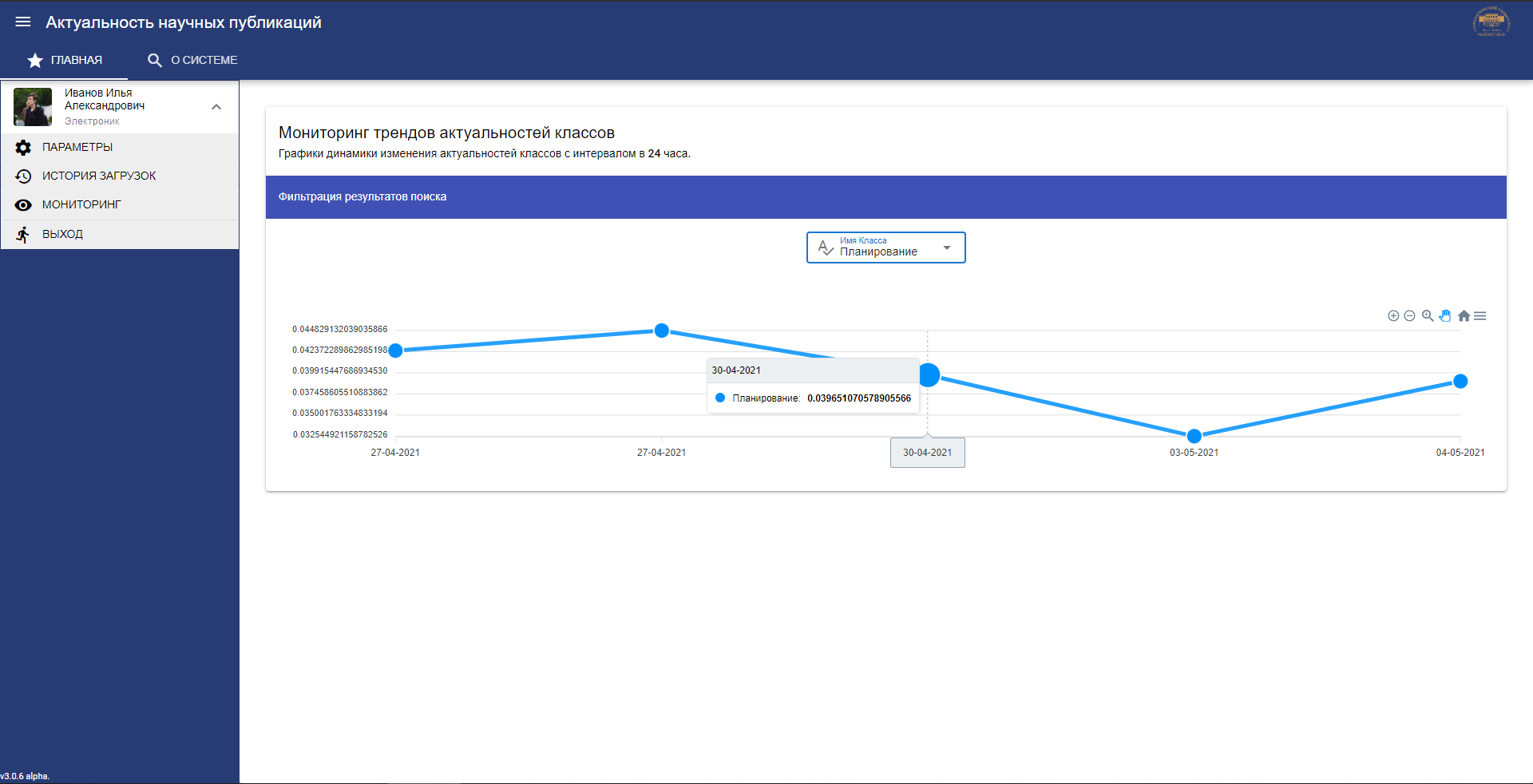
На этапе формирования рекомендации, существует упомянутая выше – возможность принять или отклонить рекомендацию, а также внести коррективы в предложенный результат. Пример – Рисунок N. Это также необходимо для сбора информации и переобучения нейронной сети.

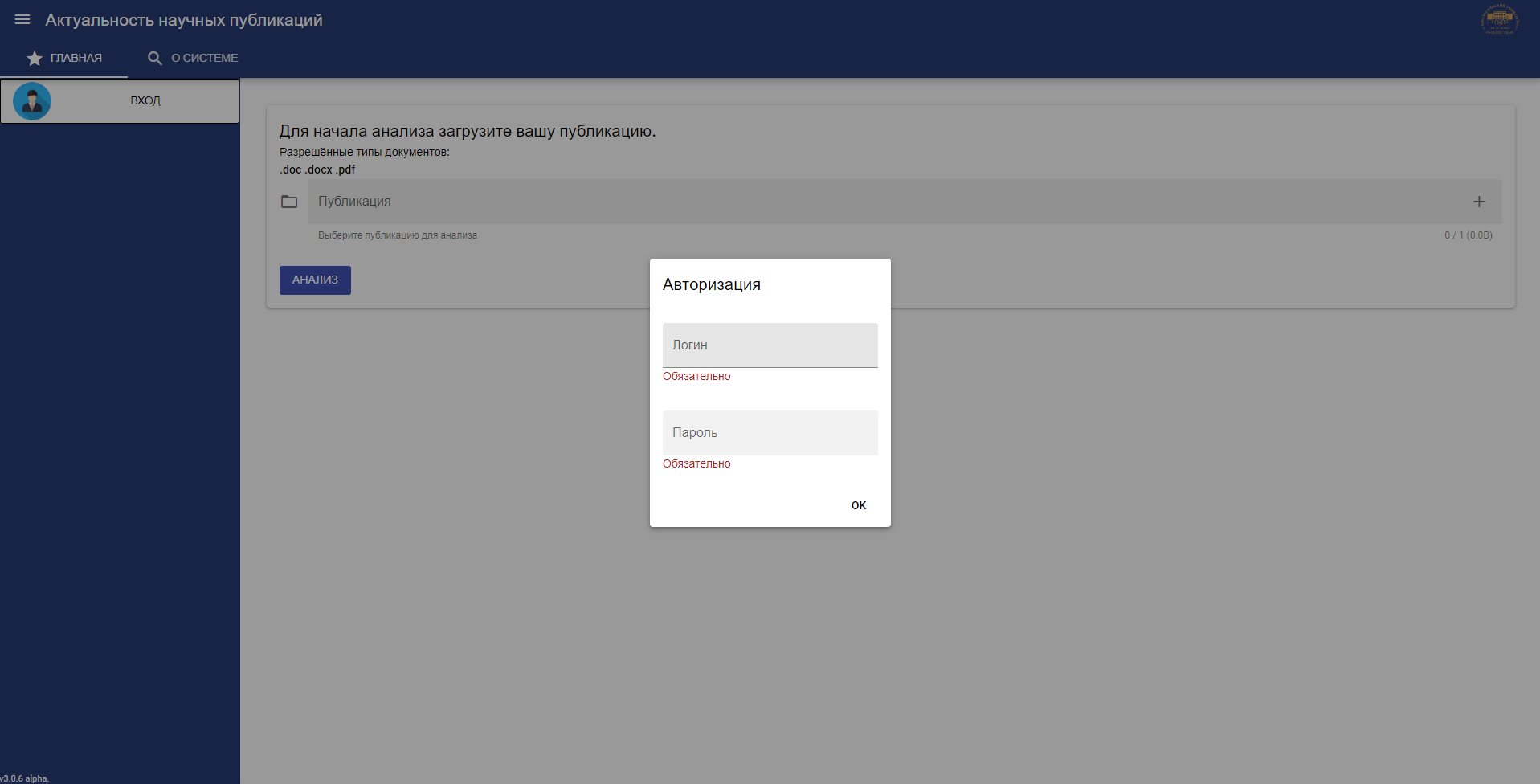


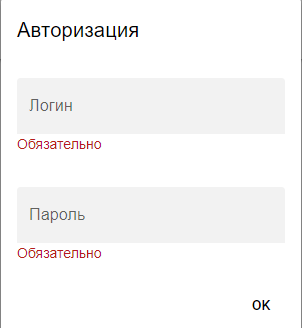


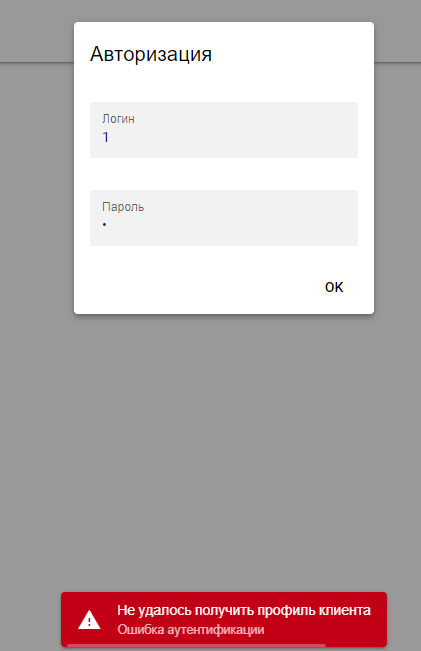










****

**-** Выводы по проектной части 1-2.

За счёт функционала и выбранных инструментов, данная система легко расширяема и обладает достаточной гибкостью при настройке отдельных её модулей, которые могут в дальнейшем быть использованы как микросервисы.

Реализация RESTful API - позволит использовать описываемую систему - с любыми интерфейсами и в любых проектах, начиная с приложений под различные операционные системы и заканчивая веб – приложениями и приложениями под мобильные устройства.

- **Заключени**е

Информационная система, описанная в рамках данной работы, относится к классу рекомендательных систем. Исходя из этого, мы получаем готовый инструмент - для статистического анализа текста публикаций и формирования рекомендаций, помогающих пользователям на их основе, принимать необходимые решения по редактированию содержания их публикаций, что как следствие – поможет увеличить их актуальность.

За счёт функционала и выбранных инструментов, данная система легко расширяема и обладает достаточной гибкостью при настройке отдельных её модулей, которые могут в дальнейшем быть использованы как микросервисы.

Реализация RESTful API - позволит использовать описываемую систему - с любыми интерфейсами и в любых проектах, начиная с приложений под различные операционные системы и заканчивая веб – приложениями и приложениями под мобильные устройства.

В настоящее время - описываемая информационная система, находится на стадии опытной эксплуатации в ГБОУ ВО МО «Технологический университет».

# СПИСОК ЛИТЕРАТУРЫ.

1. Yet Another Keyword Extractor (Yake). Unsupervised Approach for Automatic Keyword Extraction using Text Features. [Электронный ресурс] // URL: <https://github.com/LIAAD/yake> (дата обращения: 27.08.2020).
2. A Visual Guide to Using BERT for the First Time. [Электронный ресурс] // URL: <https://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/> (дата обращения: 14.07.2020).
3. BERT Word Embeddings Tutorial. [Электронный ресурс] // URL: <https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/> (дата обращения: 17.07.2020).
4. Keyphrase Extraction using SciBERT. [Электронный ресурс] // URL: <https://github.com/pranav-ust/BERT-keyphrase-extraction> (дата обращения: 19.08.2020).
5. BERT. [Электронный ресурс] // URL: https://github.com/google-research/bert (дата обращения: 12.07.2020).
6. MovieLens Non-commercial, personalized movie recommendations. [Электронный ресурс] // URL: https://movielens.org/ (дата обращения: 03.08.2020)
7. MovieLens Unplugged: Experiences with an Occasionally Connected Recommender System. [Электронный ресурс] // URL: http://files.grouplens.org/papers/miller-iui03.pdf (дата обращения: 02.09.2020).
8. Apache POI. [Электронный ресурс] // URL: https://poi.apache.org (дата обращения: 09.07.2020).
9. OkHTTP. [Электронный ресурс] // URL: https://square.github.io/okhttp (дата обращения: 11.07.2020).
10. FlyWay. [Электронный ресурс] // URL: https://flywaydb.org/documentation (дата обращения: 19.07.2020).
11. IText. [Электронный ресурс] // URL: https://itextpdf.com/ru (дата обращения: 21.07.2020).