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

ДОПУСТИТЬ К ЗАЩИТЕ

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

_____ Л.Б. Соколинский

«___»_____ 2023 г.

Разработка алгоритма для автоматизированной рубрикации статей научного журнала

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2023.308-616.ВКР

Научный руководитель,
профессор кафедры СП, д.ф.-м.н.,
доцент
_____ М.Л. Цымблер

Автор работы,
студент группы КЭ-401
_____ Ю.В. Циммерман

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2023 г.

Челябинск, 2023 г.

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

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
06.02.2023 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студентке группы КЭ-401
Циммерман Юлии Владимировне,
обучающейся по направлению
02.03.02 «Фундаментальная информатика и информационные технологии»

1. **Тема работы** (утверждена приказом ректора от 25.04.2023 г. № 753-13/12)
Разработка алгоритма для автоматизированной рубрикации статей научного журнала.
2. **Срок сдачи студентом законченной работы:** 05.06.2023 г.
3. **Исходные данные к работе**

3.1. Большакова Е.И., Воронцов К.В., Ефремова Н.Э., Клышинский Э.С., Лукашевич Н.В., Сапин А.С. Автоматическая обработка текстов на естественном языке и анализ данных: Учебное пособие – М.: Изд-во НИУ ВШЭ, 2017. – 269 с.

3.2. Сорокин А.Б., Железняк Л.М. Технологии обучения: кластеризация и классификация: Учебное пособие – М.: РТУ МИРЭА, 2021. – 49 с.

3.3. Панфилов К.С. Создание веб-сайта от замысла до реализации: Учебное пособие – М.: ДМК Пресс, 2009. – 440 с.

4. **Перечень подлежащих разработке вопросов**

4.1. Выполнить анализ предметной области и провести обзор существующих решений.

4.2. Выполнить разработку алгоритма рубрикации научных статей на основе кластерного анализа.

4.3. Выполнить разработку приложения для администратора научного журнала, позволяющего выполнять подготовку, рубрикацию и визуализацию данных.
4.4. Разработать тестовые наборы и провести тестирование алгоритма.
5. Дата выдачи задания: 06.02.2023 г.

Научный руководитель,
профессор кафедры СП, д.ф.-м.н., доцент М.Л. Цымблер

Задание принял к исполнению Ю.В. Циммерман

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

В настоящее время в сети Интернет хранится огромное количество различной информации, отражающей любую запрашиваемую тему. Все больше научных статей становятся доступны в электронном формате, что позволяет пользователям из любой точки мира их прочесть, а соответственно, приводит к популяризации науки и научной деятельности.

Многие научные журналы имеют собственные веб-сайты и предоставляют доступ к опубликованным статьям, для чего используют открытые издательские платформы для публикации и рецензирования статей в сети Интернет. Данные платформы предоставляют возможность для создания сайтов, разработки концепции их дизайна и размещения на них статей. Однако, частым явлением является отсутствие рубрикации статей на сайтах журналов, то есть распределения статей по тематическим областям, что вызывает необходимость самостоятельного изучения, какая предметная область рассматривается в статье. В связи с этим возникает сложность в поиске необходимой информации. Данную проблему могла бы решить автоматизированная рубрикация данных, реализующая распределение статей по тематическим группам, которая могла бы повысить иллюстративность контента на сайте.

Основная проблема в настоящий момент заключается в отсутствии систематизированного подхода к выполнению рубрикации текстовых данных. Существующие на данный момент способы подразумевают наличие заранее подготовленных данных, поэтапное выполнение анализа этих данных, а также самостоятельное формирование способа отображения результатов рубрикации.

Постановка задачи

Целью выпускной квалификационной работы является разработка алгоритма для автоматизированной рубрикации статей научного журнала. Для достижения поставленной цели необходимо решить следующие задачи.

1. Выполнить анализ предметной области и провести обзор существующих решений.

2. Выполнить разработку алгоритма рубрикации научных статей на основе кластерного анализа.

3. Выполнить разработку приложения для администратора научного журнала, позволяющего выполнять подготовку, рубрикацию и визуализацию данных.

4. Разработать тестовые наборы и провести тестирование алгоритма.

Структура и содержание работы

Работа состоит из введения, четырех глав, заключения и списка литературы. Объем работы составляет 56 страниц, объем списка литературы – 33 источника.

В первой главе описывается предметная область, а также проводится анализ существующих аналогов и методов, решающих поставленные задачи.

Во второй главе описываются функциональные и нефункциональные требования к разрабатываемому приложению. Приведено проектирование алгоритма рубрикации, архитектуры приложения, его составляющих компонентов и пользовательского интерфейса.

В третьей главе описаны программные средства, используемые в процессе реализации системы, описаны процессы реализации компонентов системы, а также представлен пользовательский интерфейс.

В четвертой главе представлены функциональное тестирование приложения, а также тестирование алгоритма на различных сформированных наборах данных.

В заключении представлены основные результаты выполненной работы и направления, в которых возможны дальнейшие исследования.

В приложениях содержатся спецификация диаграммы вариантов использования системы, интерфейсы и листинги основных модулей, а также тестирование алгоритма на сформированных тестовых наборах.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Описание предметной области

Задача рубрикации данных подразумевает распределение множества объектов на подмножества. Данная группировка объектов осуществляется на основе какого-либо критерия. Внутри каждой полученной группы должны оказаться наиболее похожие между собой объекты и наиболее отличные от объектов, оказавшихся в других группах.

В большинстве обзоров задача распределения объектов по группам решается в несколько этапов. Так, до начала распределения объектов необходимо привести данные к единому формату: провести предобработку, а также преобразование текстовых данных в числовой формат. Для выполнения группировок объектов часто используется кластерный анализ, основанный на вычислении расстояния между векторами. Таким образом, к полученным объектам можно применить один из методов кластеризации, который осуществит группировку объектов на основе похожести.

«Supercomputing Frontiers and Innovations», сокращенно «Superfri», является международным научно-исследовательским журналом, предоставляющим свободный доступ к статьям и обзорам на своем веб-сайте [1]. Тематические области охватывают инновационные суперкомпьютерные технологии, перспективные архитектуры, масштабируемые и параллельные алгоритмы, языки, анализ данных, вопросы, связанные с совместным вычислительным проектированием, сквозные вопросы высокопроизводительных вычислений, а также документы по суперкомпьютерному образованию и приложениям массовых параллельных вычислений в науке и промышленности.

Данный журнал реализован на платформе Open Journal Systems – открытом решении для создания научных журналов [2]. Данная платформа предоставляет возможность создать веб-сайт, разработать его дизайн, а также создать базу данных. Она позволяет использовать различные функции для работы с сайтом и организовывает весь издательский процесс:

прием, рецензирование и распределение статей по каталогам, а также позволяет разделять возможности между пользователями в зависимости от их роли на сайте.

На сайте журнала «Superfri» предоставляется доступ к архиву всех опубликованных выпусков (рисунок 1). В каждом выпуске журнала содержится некоторое число статей. Выпуски научного журнала не отражают общую тематику, опубликованных в них статей. Соответственно, на сайте отсутствует какая-либо тематическая рубрикация, поэтому разработка алгоритма рубрикации будет проводиться в первую очередь для данного журнала.

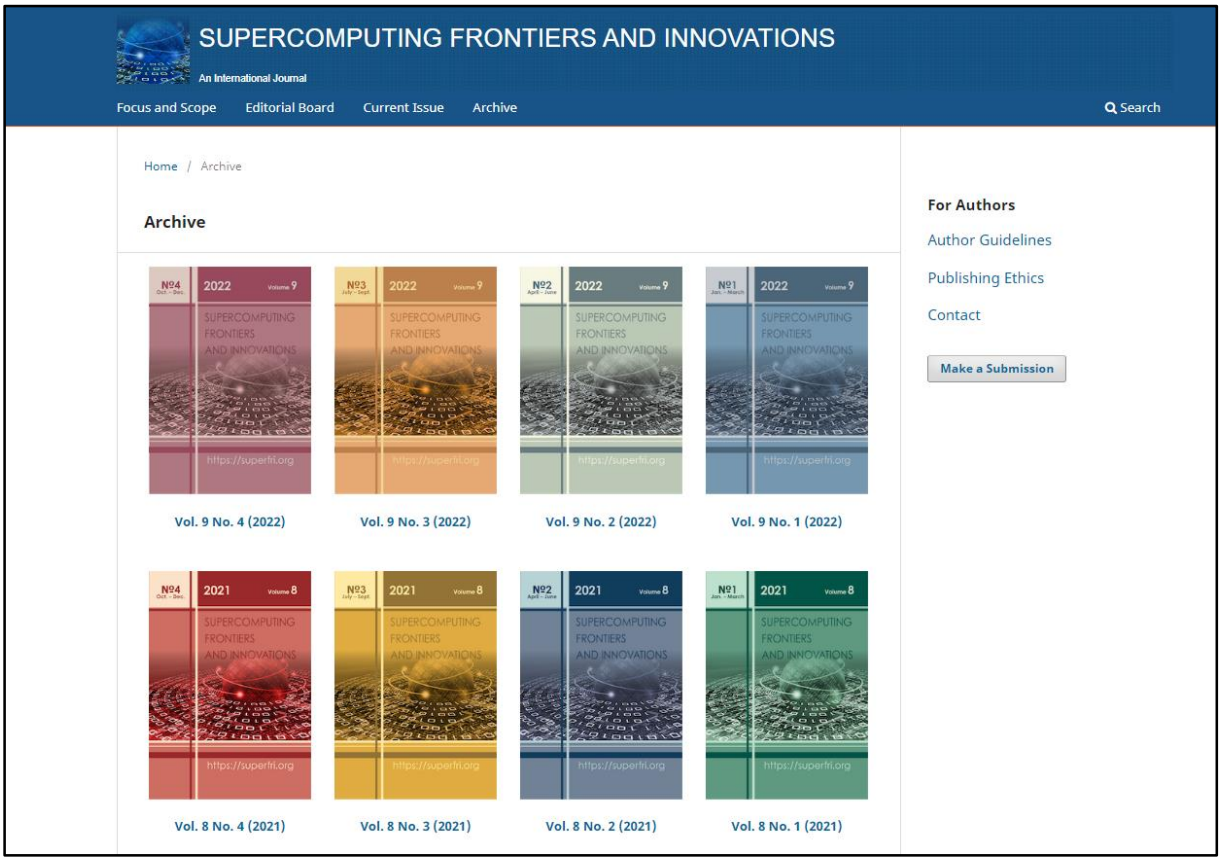

Рисунок 1 – Архив научного журнала «Supercomputing Frontiers and Innovations»

Все опубликованные в журнале статьи имеют уникальный идентификатор (DOI), с помощью которого можно перейти на их веб-страницы. На веб-странице каждой статьи представлена вся основная информация. Так,

можно найти данные об авторах, прочесть аннотацию, а также увидеть набор ключевых слов каждой статьи – текстовые метки, по которым можно найти статью при поиске и определить основные моменты, отраженные в тексте статьи (рисунок 2).

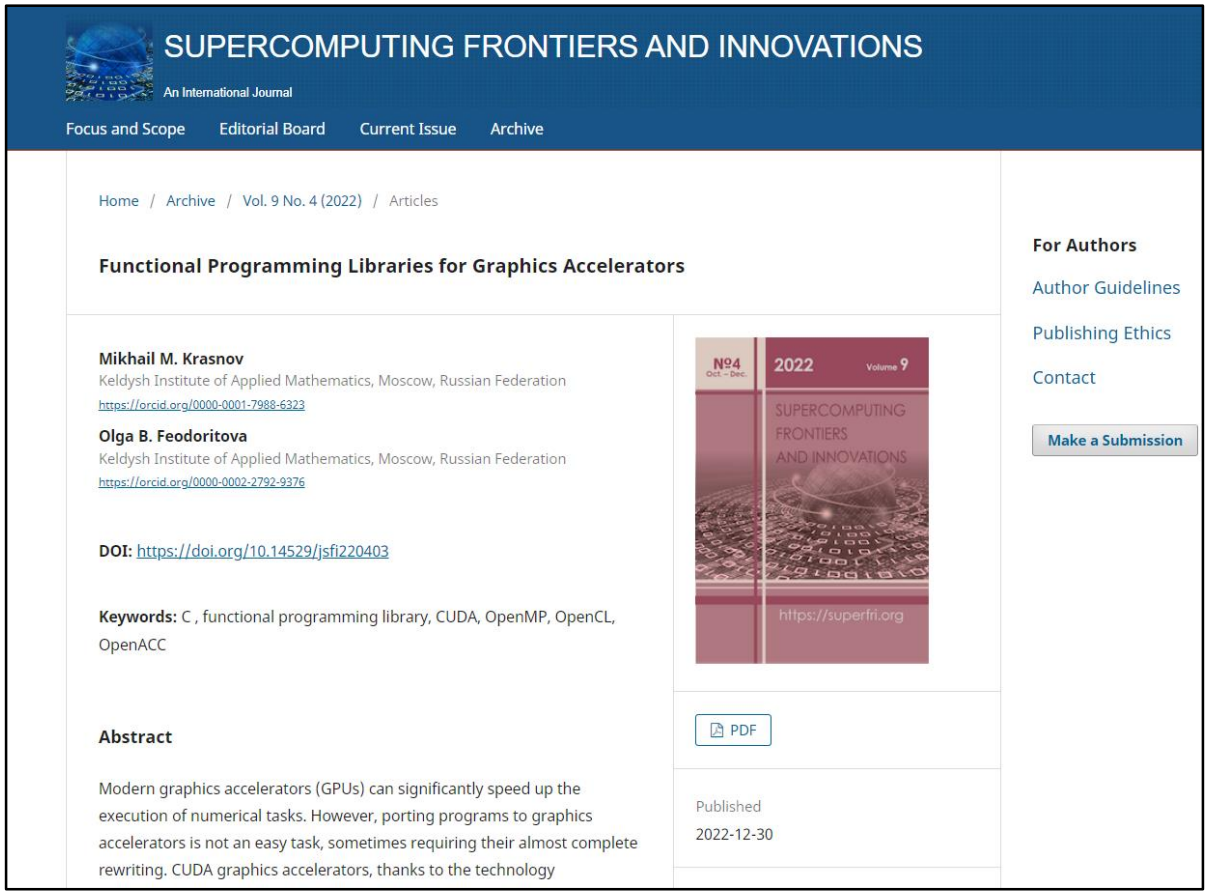

Рисунок 2 – Веб-страница статьи научного журнала

Таким образом, разрабатываемый алгоритм будет тестироваться на данных научного журнала «Supercomputing Frontiers and Innovations». Для создания алгоритма будет необходимо реализовать выполнение следующих этапов работы с данными: возможность извлечения необходимых данных о статьях, выполнить анализ полученных данных, в результате которого будут сформированы группы, отражающие общую тематику статей, а также реализовать отображение полученных групп. Для анализа данных, использующихся при рубрикации статей, понадобятся такие данные как: название, аннотация и ключевые слова статьи.

1.2. Обзор аналогов и существующих решений

Наиболее близким аналогом данной работы является библиотека кластеризации с открытым исходным кодом Carrot^2 [3]. Она выполняет автоматическую кластеризацию найденных ссылок и документов согласно поисковому запросу пользователя, а также визуализацию сгруппированных данных по тематикам, предоставляющую удобный и наглядный интерфейс для пользователя (рисунок 3). Данная библиотека осуществляет кластеризацию веб-страниц и загруженных документов различных форматов. Carrot^2 предлагает несколько алгоритмов кластеризации документов: Lingo и STC, создающие плоские кластеры, и один из наиболее широко используемых алгоритмов – k-means, создающий кластеры в стиле мешка слов.

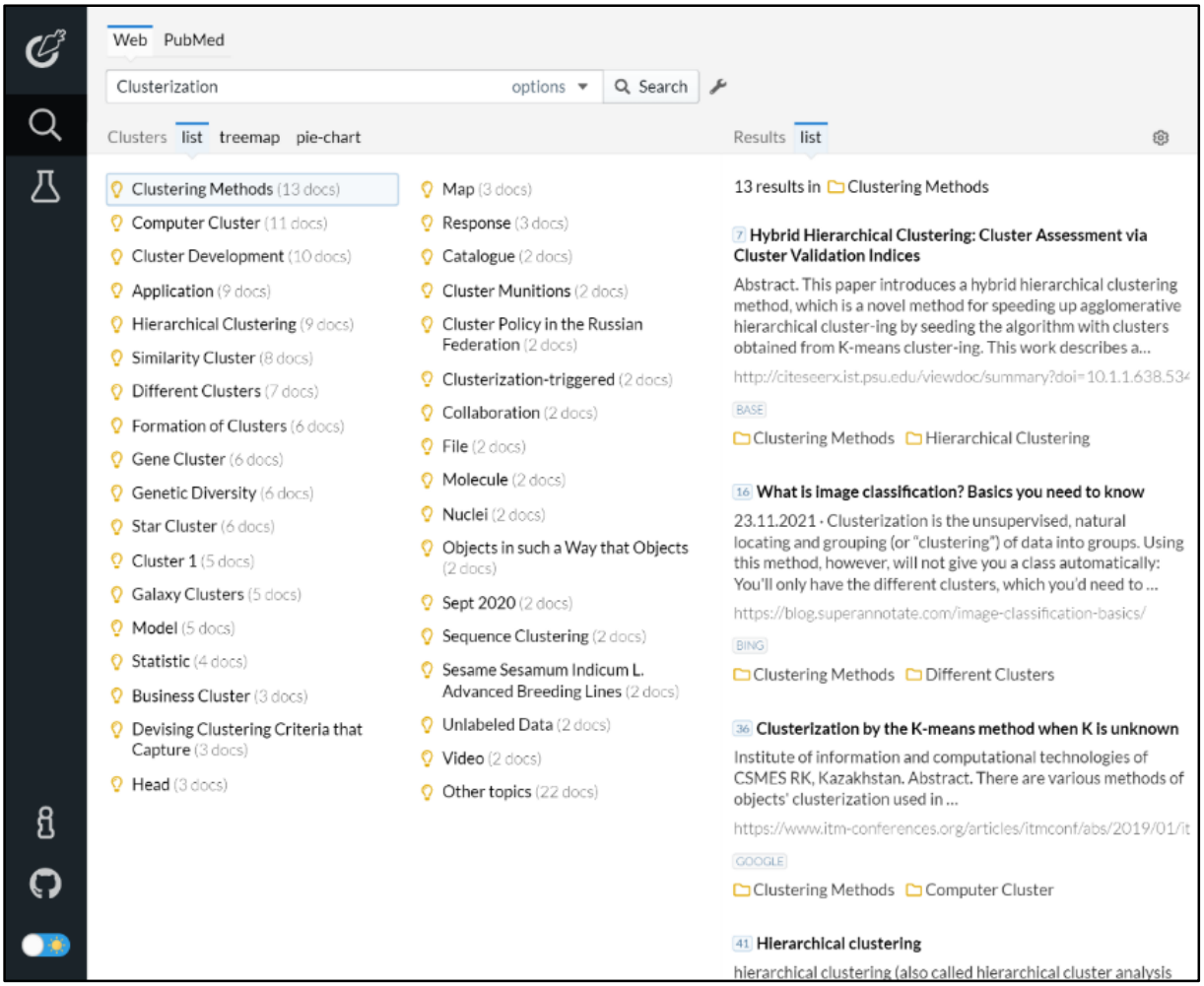

Рисунок 3 – Интерфейс веб-приложения Carrot^2

Так, при введении любого запроса в поисковую строку будет осуществлена группировка ссылок согласно вычисленным темам, к тому же для полученных групп формируются обобщающие названия. Список ссылок и документов, содержащихся в каждой группе можно просмотреть при нажатии на ее название, а также перейти по ссылке к любой выбранной документации.

В большинстве обзоров задача группировки множества объектов на подмножества решается в несколько этапов. Так, до начала распределения объектов необходимо привести данные к единому формату: провести предобработку, а также преобразование текстовых данных к векторной форме. После чего к полученным данным можно применить один из методов кластеризации, основанный на вычислении расстояния между векторами.

Предобработка текстовой информации проводится для наибольшей точности выполнения группировки текстов. В результате предобработки текстовые данные обобщаются путем очищения и приведения слов к начальной форме. Данная процедура включает в себя следующие этапы морфологической обработки: разбиение текста на токены, нормализация, включающая в себя удаление шумовых слов, знаков препинания и приведение буквенных символов к единому регистру, а также лексическая обработка, приводящая слова к инфинитивной форме. Для решения проблемы существуют библиотеки, позволяющие выполнять предобработку данных для различных языков: NLTK, py-morphy2, spacy [4].

Для подготовки текстов к вычислениям выполняется конвертация текстовых данных в числовые, что позволяет использовать полученные данные в вычислительных алгоритмах. Одним из способов конвертации является векторизация слов, приводящая преобразованные слова в векторные представления для их дальнейшего использования при вычислении расстояния между векторами с использованием алгоритмов кластеризации. Существует большое количество методов, выполняющих векторизацию слов. Наиболее

простым способом является прямое кодирование (one-hot encoding), выполняющееся следующим образом: каждый токен представляет собой бинарный вектор, единица ставится тому элементу, который соответствует номеру токена в словаре. Метод мешок слов (bag of words) имеет похожий способ векторизации. Так, вектору выделяется весь документ, и каждый элемент кодируется единицей по порядку следования слов в словаре [5].

Основным минусом данных подходов является то, что они не учитывают важность слов в документах, а лишь отображают частотность появления слов. Для решения данной проблемы был разработан метод TF-IDF, в ходе которого вычисляются частотность слова в документе и инверсия частоты документа [6]. В результате вычислений наибольшую значимость имеют слова, часто встречающиеся в одном документе, при этом встречающиеся редко в остальных. С помощью данного метода все слова получают свой вес, отражающий их важность в текстах. Данный подход также позволяет определить наиболее характерные слова для каждого текста из входного набора.

Все вышеупомянутые методы имеют ряд недостатков, так как не учитывают контекст слов, их порядок в предложении, а также обладают высокой размерностью словарей, что может вызвать снижение эффективности анализа данных [5]. Для решения данной проблемы была разработана модель под названием «Семантическая сеть», которая является информационной моделью предметной области. Она имеет вид ориентированного графа, вершины которого соответствуют объектам предметной области, а дуги (ребра) задают отношения между ними [7].

Семантические сети обучаются на корпусе текстов, прошедших предобработку, и отображают слова в векторном пространстве таким образом, что наименьшее расстояние между словами соответствует наибольшей смысловой близости [8]. Соответственно, слова, появляющиеся в схожих контекстах, отображаются в близкие вектора. Для вычисления смысловой

близости слов вычисляется косинусное сходство, вычисляющее расстояние между векторами данных слов.

Наиболее распространенными реализациями векторного представления слов являются модели Word2vec и Glove [9]. Отличие данных моделей заключается в способе их обучения. Модель Word2vec учитывает совместное использование слов в локальном контексте (соседние слова). Таким образом, в таких моделях, как Word2Vec, ближайшими соседями слов являются синонимы, партонимы и морфологические варианты лексической единицы [10]. Модель Glove улучшает Word2vec тем, что учитывает использование слов в общем контексте, и, соответственно, лучше отражает семантическую похожесть слов. Однако польза обеих моделей зависит от поставленной задачи [11].

Для обучения моделей сформированы специальные корпуса текстов. Это наборы данных, которые содержат большое количество структурированной текстовой информации с различных веб-сервисов: интернет-энциклопедий, социальных сетей, литературных сайтов.

В задачах анализа данных часто используются методы стандартизации данных, которые позволяют привести данные к единому масштабу, что приводит к возможности корректного сравнения данных. Кроме того, для данных с большим количеством признаков часто применяются методы уменьшения размерности. Они приводят к упрощению анализа данных и оптимизации его выполнения, позволяют избавиться от шумов, а также визуализировать данные. Наиболее известными методами снижения размерности являются анализ главных компонент (PCA) и алгоритм t-SNE [12]. Данные методы имеют свои плюсы и минусы на различных данных. Метод PCA базируется на сохранении больших попарных расстояний и максимизации дисперсии. Для алгоритма t-SNE характерно сохранять только небольшие попарные расстояния для лучшей визуализации.

Для группировки числовых данных чаще всего используются кластеризация. На сегодняшний день существует большое множество различных

алгоритмов кластеризации, осуществляющих распределение данных по группам. Методы кластерного анализа можно разделить на две группы: иерархические и неиерархические. В основе первого типа лежат разбиение больших кластеров на меньшие (агломеративный метод), либо объединение меньших кластеров в большие (дивизимный метод) [13]. Неиерархические методы кластеризации заключаются в распределении данных на кластеры до тех пор, пока не будет выполнено правило остановки. Наиболее популярными в использовании считаются следующие алгоритмы кластеризации: k-means, DBSCAN и иерархическая кластеризация [14, 15].

Одним из самых известных методов кластеризации является метод k-means или k-средних, который разбивает множество элементов на заданное число кластеров. Его задача минимизировать среднеквадратичное отклонение на точках каждого кластера. Начальные точки выбираются случайным образом, после чего вычисляется центр масс каждого кластера. В случае, когда изменений в кластерах не происходит, алгоритм завершает свою работу.

Иерархическая кластеризация выполняет рекурсивное разбиение объектов агломеративным или дивизимным методом. Алгомеративный метод подразумевает, что каждый объект является отдельным кластером, после чего происходит объединение похожих объектов до тех пор, пока все кластеры не объединятся в один или не будет найдено необходимое число кластеров. При дивизимном методе все объекты изначально принадлежат одному кластеру, после чего он разбивается на более мелкие до тех пор, пока все объекты не попадут в отдельные кластера или не будет найдено заданное число кластеров. На каждом шаге вычисляется расстояние между кластерами и пересчитывается расстояние между новыми кластерами. Расчет расстояний можно проводить, используя различные метрики и критерии связи, определяющие способ вычисления расстояния и связи между объектами [16].

Для отображения результатов кластеризации используется дендрограмма – дерево, построенное по матрице мер близости между кластерами. Объединение узлов соответствует слиянию двух кластеров. При этом длина ребра соответствует расстоянию между кластерами.

Большинство алгоритмов кластеризации не поддерживает автоматическое определение оптимального числа кластеров, поэтому необходимо проводить оценку качества кластеризации для разного числа кластеров. Оптимальное число кластеров принято брать в промежутке от двух до девяти, согласно числу Миллера. Данный принцип позволяет сохранить основные общие характеристики объектов, не потеряв при этом информацию.

Для оценки качества кластеризации используются различные меры эффективности. Наиболее распространенными в использовании являются силуэтный коэффициент, метод локтя, а также индекс Calinski–Harabasz [17, 18]. В основе оценки эффективности кластеризации лежит вычисление расстояния между элементами внутри кластера, либо между кластерами, однако каждый метод использует разный способ определения данных расстояний. Так, метод локтя измеряет сумму квадратов внутрикластерных расстояний объектов. Оптимальным числом кластеров в данном методе является то, при котором данное расстояние перестает существенно уменьшаться. Силуэтный коэффициент показывает, на сколько среднее расстояние до объектов текущего кластера отличается от среднего расстояния до объектов ближайшего кластера Наибольшее значение силуэтного коэффициента соответствует наиболее оптимальному числу кластеров.

Выводы по первой главе

В этой главе была изучена предметная область работы, а также были рассмотрены существующие аналоги, выполняющие рубрикацию текстовых данных, а также проанализированы возможные методы реализации алгоритма рубрикации статей.

2. ПРОЕКТИРОВАНИЕ

2.1. Определение требований

В ходе проектирования приложения были определены следующие функциональные и нефункциональные требования.

Функциональные требования

Функциональные требования определяют функциональность программного обеспечения, то есть описывают, какое поведение должна предоставлять разрабатываемое приложение.

1. Приложение должно осуществлять извлечение данных о статьях с помощью PDF-парсинга и парсинга веб-страниц.
2. Приложение должно выполнять рубрикацию статей, осуществляющую их распределение по тематическим группам.
3. Приложение должно визуализировать результаты на веб-странице в виде списка и точечной диаграммы с возможностью просмотра статей в группах и перехода к веб-странице статьи на сайте журнала по ее DOI-ссылке.

Нефункциональные требования

К нефункциональным требованиям системы относятся свойства, которыми она должна обладать. Например, удобство использования, безопасность, расширяемость и т.д.

1. Приложение должно иметь интуитивно понятный в использовании пользовательский интерфейс.
2. Основная программная реализация осуществляется с использованием Python.
3. Отображение результата рубрикации должно быть реализовано с использованием HTML, CSS и JavaScript.

2.2. Варианты использования системы

Для проектирования приложения был использован язык графического описания для объектного моделирования UML. Была построена модель взаимодействия внешнего актера с приложением в виде диаграммы вариантов использования. В ходе анализа разрабатываемого приложения были выявлены основные варианты использования (рисунок 4).

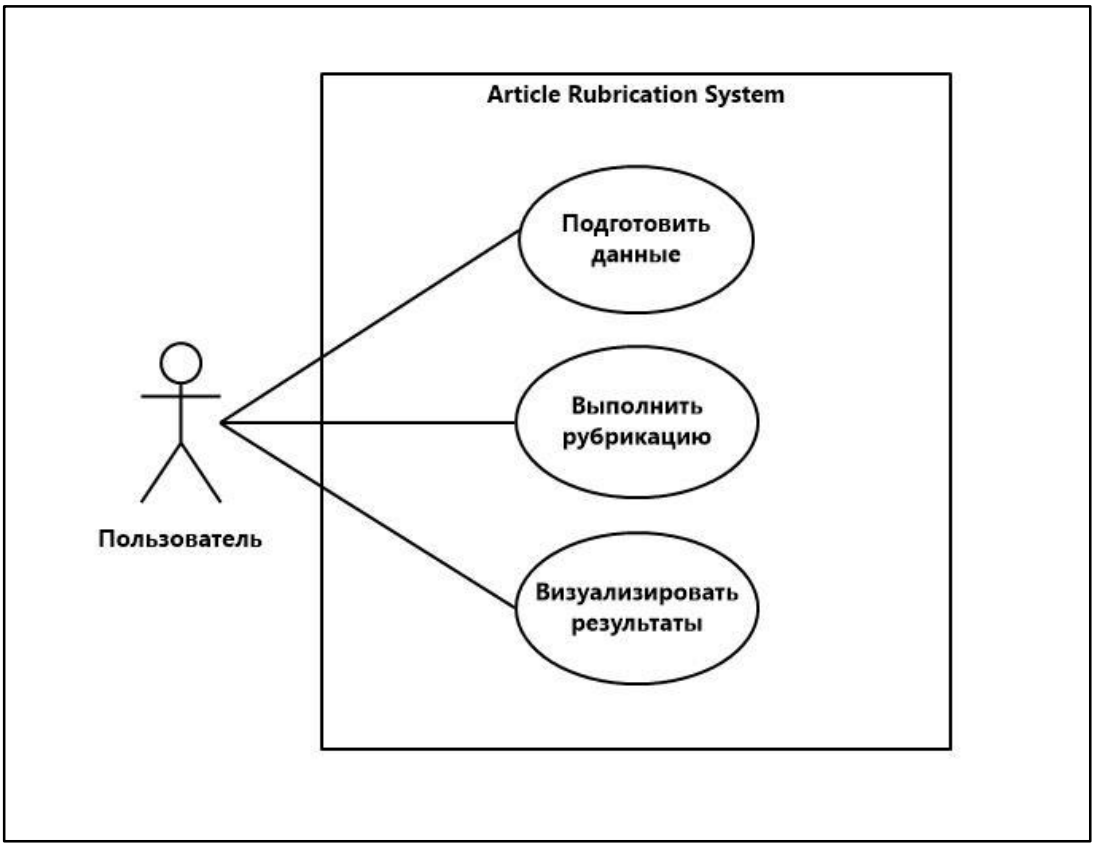

Рисунок 4 – Диаграмма вариантов использования

Для данной диаграммы определен актер – пользователь, который взаимодействует с приложением. Пользователю доступны следующие варианты использования.

1. Вариант использования «Подготовить данные». Пользователь может запустить процесс извлечения данных о статьях, в результате которого будет сформирован файл, содержащий информацию о статьях.

2. Вариант использования «Выполнить рубрикацию». Пользователь может запустить процесс, выполняющий распределение статей по тематическим группам согласно их содержанию.

3. Вариант использования «Визуализировать результаты». Пользователь может запустить процесс визуализации, выполняющий создание файлов, отображающих результаты рубрикации в виде веб-страницы.

Детальная спецификация вариантов использования приложения представлена в приложении А.

2.3. Алгоритм рубрикации

В результате обзора возможных методов выполнения группировки текстовых данных по предметным областям, был спроектирован алгоритм рубрикации статей научного журнала на основе кластерного анализа. Данный алгоритм позволяет выполнять распределение научных статей по тематическим рубрикам, соответствующим их содержанию. В результате работы алгоритма должны быть получены рубрики с набором статей и соответствующих им ключевых слов.

На рисунке 5 представлено поэтапное выполнение работы алгоритмом рубрикации научных статей.

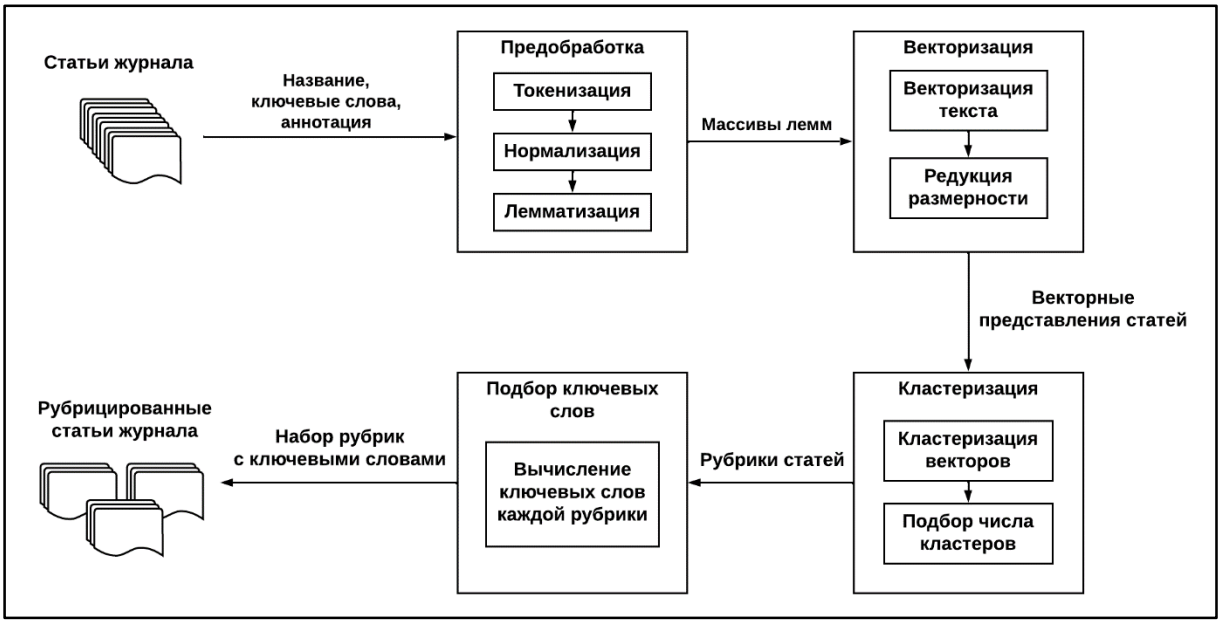

Рисунок 5 – Алгоритм рубрикации научных статей

Рубрикация выполняется на основе следующей информации о статьях: название, ключевые слова и аннотация. Основными этапами работы алгоритма рубрикации статей являются: предобработка текстовых данных, их векторизация, кластеризация и подбор ключевых слов для каждой полученной рубрики.

Предобработка включает в себя выполнение процессов обработки текстовых данных. В начале выполняется токенизация, выполняющая разбиение предложений на более мелкие единицы: слова и знаки препинания. После чего проводится нормализация, вследствие которой из них удаляются слова и символы, которые не несут информативной нагрузки. К ним можно отнести знаки препинания, предлоги и т.п. Завершающим этапом является лемматизация, приводящая оставшиеся слова к инфинитивной форме.

Векторизация представляет собой процесс преобразования текстовых данных в наборы чисел, чтобы использовать их при кластеризации. После преобразования полученные числовые значения необходимо привести к одному формату. Для чего проводится приведение числовых значений к единому масштабу, чтобы сократить различие между значениями и уменьшить потери информации. Затем проводится редукция размерности, выполняющая уменьшение числа признаков в полученных наборах чисел для оптимизации работы алгоритма и дальнейшей визуализации.

Кластеризация выполняет распределение наборов чисел, соответствующих статьям, по группам на основе близости числовых значений. Так, образовываются группы, в которых представлены наиболее похожие между собой объекты. На этапе кластеризации также необходимо определить оптимальное значение кластеров для текущего набора для наиболее корректного распределения данных по группам.

Подбор ключевых слов выполняет определение наиболее характерных слов для каждой рубрики. На данном этапе необходимо вычислить веса важности ключевых слов, входящих в каждую рубрику, и получить набор наиболее соответствующих каждой рубрике терминов.

2.4. Архитектура приложения

В данном разделе рассматривается спроектированная архитектура приложения в виде диаграммы компонентов, которая показывает разбиение системы на структурные компоненты.

Спроектированная архитектура приложения представлена на рисунке 6 в виде диаграммы компонентов.

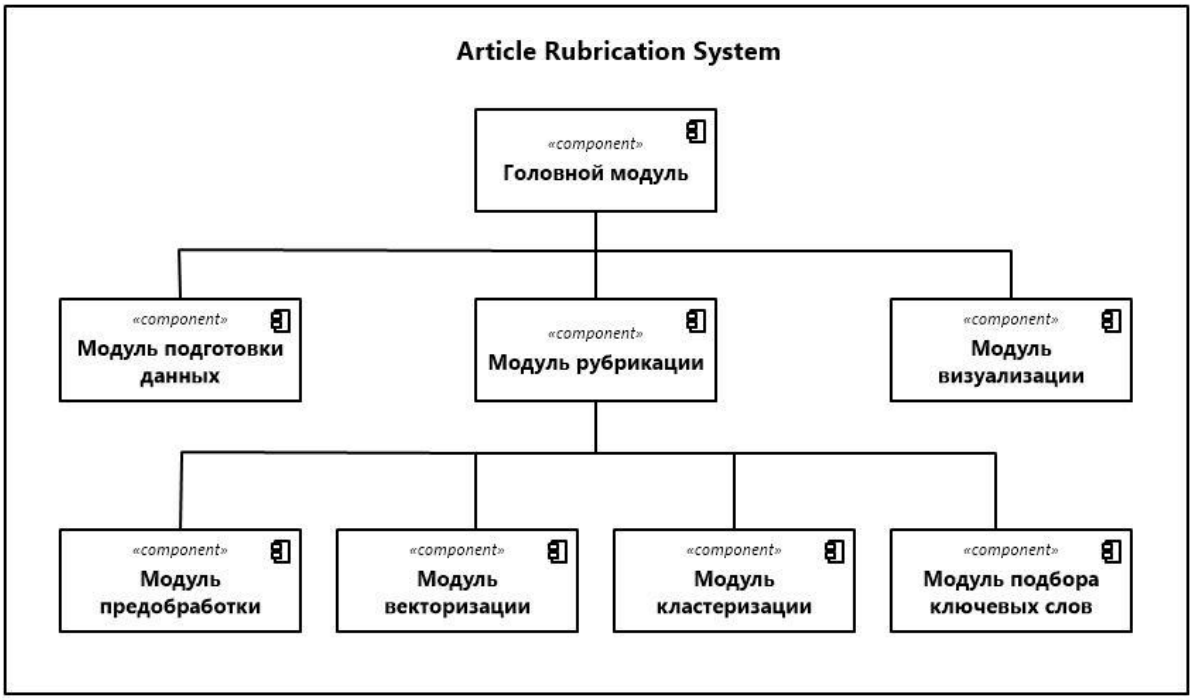

Рисунок 6 – Архитектура приложения

Компонент *головной модуль* выполняет отображение окон приложения, с которыми может взаимодействовать пользователь, и осуществляет запуск подчиненных модулей приложения.

Модуль подготовки данных является компонентом, выполняющим извлечение необходимых данных для рубрикации. Данный этап предполагает осуществление двух способов получения данных о статьях в зависимости от выбора пользователя: из предварительно загруженных файлов, соответствующих электронным версиям выпусков журнала, либо выполнение парсинга веб-страниц архива статей на сайте интернет-журнала. В ходе подготовки

данных необходимо получить следующие данные о каждой статье, представленной на сайте журнала: название, авторы, DOI-ссылка, ключевые слова и аннотация. Результатом работы модуля является многомерный массив с полученной информацией о статьях. Полученные данные записываются в файл для их дальнейшего использования без необходимости повторного запуска данного модуля.

Модуль рубрикации данных запускает выполнение алгоритма рубрикации путем поочередного запуска подчиненных модулей, выполняющих этапы предобработки, векторизации, кластеризации и подбора ключевых слов. Данный компонент также отвечает за работу с внешними файлами для их чтения и записи результата работы, который в дальнейшем используется при визуализации.

Модуль предобработки данных осуществляет предварительную обработку текстовых данных на естественном языке для их приведения к стандартному виду. В результате все текстовые данные проходят морфологическую обработку и очистку, вследствие чего приводятся к формату, соответствующему требованиям для корректного выполнения последующих этапов, выполняющих анализ данных.

Модуль векторизации осуществляет процесс конвертации, выполняющий приведение текстовых данных, полученных в результате предварительной обработки, к числовому виду, чтобы в дальнейшем использовать преобразованные данные в вычислительных алгоритмах. Результатом векторизации является массив векторов, в котором каждый набор числовых значений соответствует одной из статей.

Модуль кластеризации выполняет распределение данных по группам с помощью метода кластеризации. Так, выполняется кластеризация числовых данных, полученных в результате работы модуля векторизации, одним из предложенных пользователю алгоритмов. Данный компонент также выполняет определение оптимального числа кластеров, которое и будет браться в качестве итогового числа групп.

Модуль подбора ключевых слов выполняет вычисление наиболее характерных слов для каждой получившейся группы статей, которые будут использоваться в качестве их описания.

Модуль визуализации создает файлы, с помощью которых происходит отображение результатов, полученных в ходе выполнения вышеперечисленных этапов. Результатом работы данного компонента является веб-страница, представляющая полученные группы статей в соответствии с запросом пользователя. Так, на выбор пользователю предложена визуализация в виде списков и точечной диаграммы.

Интерфейсы описанных модулей представлены в приложении Б.

2.5. Проектирование пользовательского интерфейса

В данном разделе будут представлены спроектированные макеты пользовательского интерфейса приложения. Данные макеты являются примерным представлением итогового продукта и содержат в себе основные необходимые функции.

Для разрабатываемого приложения было решено создать простой и интуитивно понятный интерфейс для удобства пользователя.

При запуске приложения отображается окно, соответствующее главному меню. Оно содержит три кнопки и три выпадающих списка, каждый из которых соответствует возможным параметрам, с которыми можно запустить функцию, привязанную к соответствующей кнопке. На рисунке 7 представлен макет главного меню приложения.

Выбор элемента из выпадающего списка позволяет пользователю определить желаемые параметры запуска соответствующей функции. При нажатии на кнопку, отвечающую за процесс извлечения данных, открывается новое окно для ввода пути к необходимой директории или окно ввода ссылки на веб-страницу архива научного журнала в зависимости от выбранного пользователем параметра.

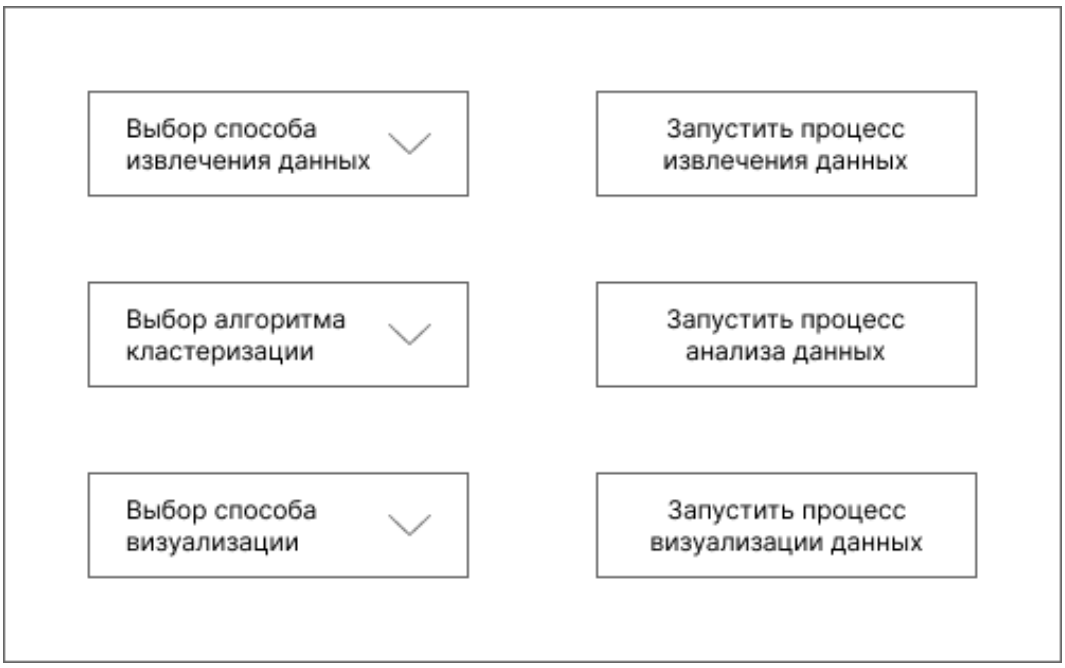

Рисунок 7 – Макет главного меню приложения

Окно ввода директории содержит следующие элементы: текстовое поле, автоматически отображающее путь к выбранной папке, кнопку, открывающую диалоговое окно выбора папки, а также кнопки возврата в главное меню и запуска процесса извлечения данных. На рисунке 8 представлен макет окна выбора директории.

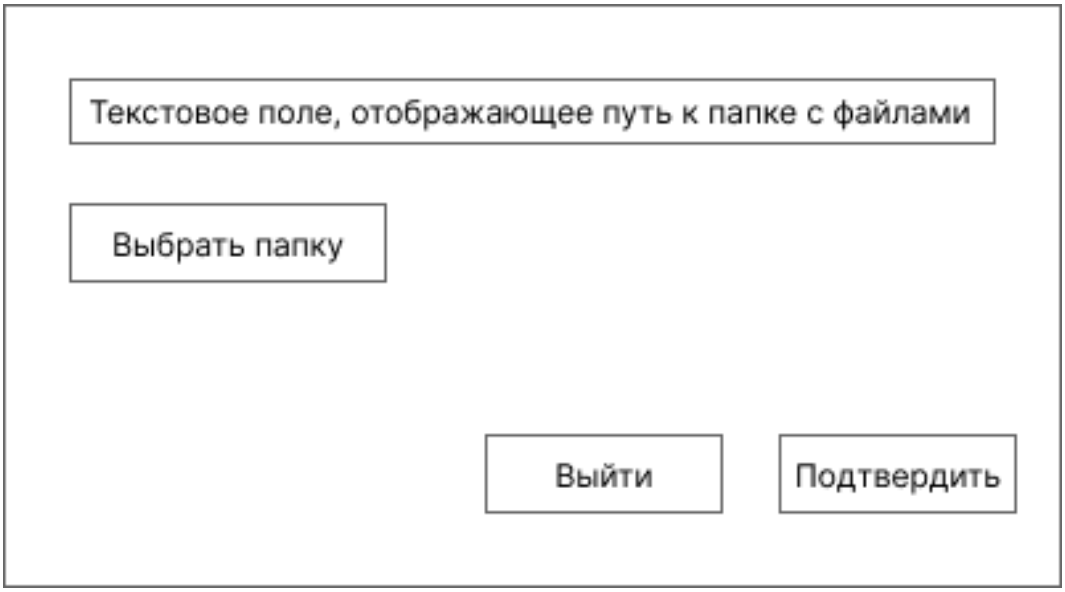

Рисунок 8 – Макет окна ввода директории

Окно ввода ссылки отличается от окна ввода директории отсутствием кнопки выбора папки, так как подразумевает ручной ввод ссылки в текстовое поле.

При запуске какой-либо задачи пользователь получает уведомления о результатах проделанной работы. При вводе некорректной информации или отсутствии необходимых файлов на экран выводится всплывающее окно с соответствующей ошибкой. В случае успешного выполнения запроса на экран выводится соответствующее информационное сообщение. Макет экрана уведомления пользователя представлен на рисунке 9.

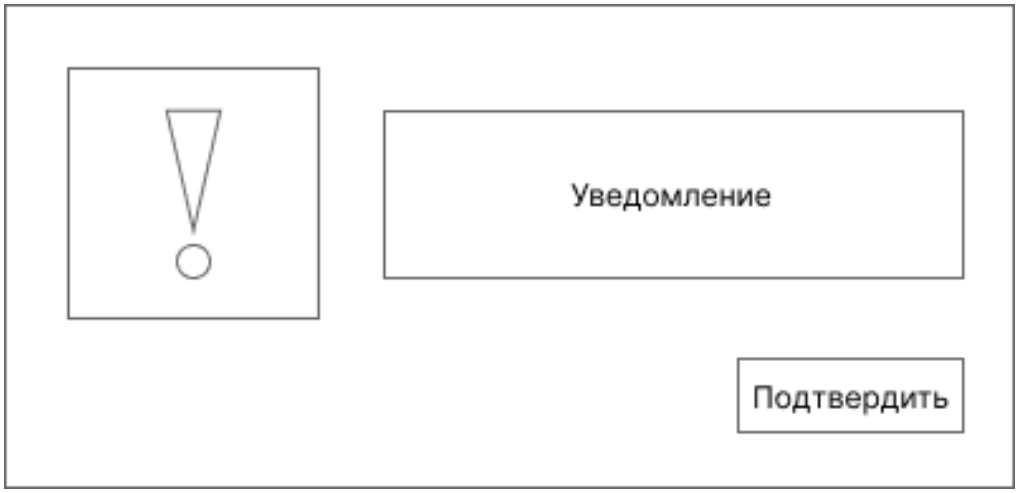

Рисунок 9 – Макет экрана уведомления пользователя

Выводы по второй главе

В этой главе были описаны этапы проектирования разрабатываемого приложения. Так, были определены функциональные и нефункциональные требования для системы, и была представлена диаграмма вариантов использования приложения. В данной главе также описан алгоритм рубрикации статей и представлено проектирование всех составляющих приложения, отображены архитектура приложения в виде диаграммы компонентов и макеты пользовательского интерфейса.

3. РЕАЛИЗАЦИЯ

3.1. Программные средства реализации

Для разработки программной части приложения был выбран высокоуровневый язык программирования Python версии 3.9.1. Разработка велась в редакторе кода PyCharm Community 2020.3.3 [19].

3.2. Реализация компонентов приложения

В данном разделе приводится описание реализации отдельных компонентов приложения. Код программной части представлен в репозитории на github [19].

Реализация модуля подготовки данных

Модуль подготовки данных выполняет поиск и извлечение необходимых для рубрикации данных. К ним относятся следующая информация о статьях: DOI-ссылка, название, аннотация, ключевые слова и авторы. В зависимости от запроса пользователя данный этап может выполняться двумя способами: парсингом PDF-файлов, предварительно загруженных в выбранную директорию, и путем парсинга веб-страниц, соответствующих статьям журнала, которые представлены на его сайте.

Для реализации извлечения данных из локальной папки используется библиотека PyPDF2 [21] для работы с файлами формата PDF.

Для парсинга веб-страниц научного журнала используется библиотека requests [22] для выполнения запроса и получения данных с сайта, после чего производится поиск необходимых элементов на веб-странице с помощью библиотеки Beautiful Soup [23].

В результате подготовки данных становятся доступны следующие данные о статьях: название, авторы, DOI-ссылка, ключевые слова и аннотация. Для формирования и записи полученных результатов используется библиотека pandas [24], позволяющая хранить данные и записывать результаты в CSV-файл.

Всего в результаты извлечения данных было получено 225 записей о статьях научного журнала. Пример представления статьи в файле CSV показан в таблице 1.

Таблица 1 – Пример представления статьи в CSV-файле

Title	Authors	Doi	Keywords	Abstract
Functional Programming Libraries for Graphics Accelerators	Mikhail M. Krasnov, Olga B. Feodoritova	https://doi.org/10.14529/jsfi220403	C, functional programming library, CUDA, OpenMP, OpenCL, OpenACC	Modern graphics accelerators (GPUs) can significantly speed up the execution of numerical task. However, porting programs to graphics accelerators is not an easy task, sometimes requiring their almost complete rewriting.

Реализация модуля предобработки данных

Модуль предобработки данных выполняет трансформацию исходных данных к стандартному формату. На данный этап поступает следующая информация о статьях: название, аннотация и ключевые слова. Они проходят морфологическую обработку и очистку, которая включает удаление шумовых символов и слов.

Для реализации данной задачи на вход модуля поступает текстовая строка, которая приводится к нижнему регистру. После чего выполняется токенизация, разбивающая строку на токены (слова, числа и знаки препинания). Затем полученные данные очищаются от чисел, знаков препинания и шумовых слов, для чего используется готовый список стоп-слов. Для оставшихся слов проводится лемматизация, выполняющая приведение слов к инфинитивной форме. После чего списки слов очищаются от дубликатов и передаются в вышестоящий модуль.

Для выполнения вышеперечисленных задач используется функционал библиотеки NLTK [25]: функции токенизации и лемматизации, а также корпус стоп-слов.

В таблице 2 представлены примеры обработки текстовых данных модулем предобработки. Данные прошли этапы токенизации, нормализации, лемматизации, а также было выполнено удаление дубликатов.

Таблица 2 – Пример работы модуля предобработки

Исходные данные	Обработанные данные
Data Compression for the Exascale Computing Era Survey.	data compression exascale compute era survey
The number of cores and the memory bandwidth have increased in a balanced fashion.	number core memory bandwidth increase balance fashion
The model is made more general by differentiating between application and machine models.	model make general differentiate application machine
This method enables us to compute instances up to L (2,21) using both CPU and GPU computational power with load balancing	method enable compute instance cpu gpu power load balance

Реализация модуля векторизации

Модуль векторизации приводит текстовые данные, полученные после предварительной обработки, к числовому формату. Для выполнения данного этапа используется готовая модель, содержащая слова в виде векторов. В данном подходе слова имеют вид векторных представлений, где расстояние между векторами отражает степень смысловой близости соответствующих слов.

В качестве корпуса векторных представлений слов была выбрана модель GloVe – glove-wiki-gigaword-50 [26]. Для ее загрузки используется модуль downloader библиотеки Gensim [27]. Вследствие данного этапа все слова статей, полученные в результате предобработки, преобразовываются в числовой формат в соответствии с их векторными представлениями в загруженном корпусе.

После преобразования слов к векторным представлениям необходимо оптимизировать данные путем стандартизации для приведения данных к единому масштабу и уменьшения размерности в связи с тем, что высокая размерность векторов может привести к некорректному выполнению дальнейшего распределения статей по группам.

Для решения данных задач выполняется стандартизация и нормализация числовых данных с помощью функций StandardScaler и normalize, а также уменьшение размерности данных с помощью функции PCA библиотеки sklearn [28], реализующей анализ и декомпозицию методом главных компонент. Данные записываются в массив с использованием библиотеки pandas [24]. Таким образом, выходным параметром является массив оптимизированных числовых данных, каждая пара которых соответствует одной из статей.

Реализация модуля кластеризации

Модуль кластеризации включает в себя выполнение двух задач: выполнение кластеризации данных, полученных в результате работы модуля векторизации, и нахождения оптимального числа кластеров для данного набора данных.

Модуль кластеризации выполняет четкое распределение объектов, представленных в виде пары чисел, по группам с помощью одного из алгоритмов кластеризации, согласно пользовательскому запросу (иерархическая кластеризация или кластеризация методом k-средних). Для выполнения кластеризации данный модуль использует следующие функции библиотеки sklearn: AgglomerativeClustering [29], выполняющая агломеративную иерархическую кластеризацию, а также KMeans [30], выполняющая кластеризацию методом k-средних.

Для определения оптимального числа кластеров выполняется кластеризация выбранным методом для различного числа кластеров (от двух до девяти). В процессе выполнения кластеризации вычисляется силуэтный коэффициент, позволяющий определить наиболее оптимальное число кластеров. Так, число кластеров, при котором кластеризация показала наибольшее значение силуэтного коэффициента считается оптимальным. Для вычисления силуэтного коэффициента используется метод silhouette_score библиотеки sklearn [31].

Следовательно, в качестве итогового распределения объектов по группам используется результат, полученный вследствие выполнения кластеризации для выбранного оптимального числа кластеров.

Реализация модуля подбора ключевых слов

После выполнения кластеризации необходимо определить ключевые слова, наиболее соответствующие каждой из полученных групп статей. Для выполнения данного этапа осуществляется оценка важности ключевых слов для каждого из полученных наборов. Для данной задачи используется метод TF-IDF, выполняющий вычисление веса каждого слова, где наибольший вес соответствует наибольшей важности данного слова.

Для данного вычисления текстовые данные статей, входящие в один кластер, объединяются, а полученные данные представляют собой отдельные тексты, соответствующие одной из рубрик. Так, для всех слов внутри каждой группы вычисляется значение TF – частота появления слова в текущем документе, и IDF – обратная частота документа [6], после чего полученные значения перемножаются.

На основе вычисленных значений определяются одиннадцать слов с наивысшим весом важности для каждой группы статей, которые будут использоваться в качестве характеристики группы.

Реализация модуля рубрикации

Модуль рубрикации выполняет поочередный запуск следующих компонентов: модуль предобработки, модуль векторизации, модуль кластеризации и модуль вычисления ключевых слов.

До начала выполнения вышеперечисленных этапов необходимо удостовериться в наличии CSV-файла, созданного в результате работы модуля подготовки данных. Для получения данных из CSV-файла используется библиотека pandas [24].

После выполнения работы всеми подчиненными модулями, результаты записываются в JSON-файл, который будет содержать информацию о

статьях и полученных группах. Так, для каждой статьи представлены следующие данные: DOI-ссылка, название, авторы, ключевые слова, векторное представление статьи и номер рубрики.

В JSON-файле также хранится вся необходимая информация о полученных группах статей. Каждая группа содержит свой номер, число входящих в нее статей, ключевые слова, а также соответствующие им веса. В данном файле также содержится информация об общем числе кластеров. На рисунке 10 показан пример представления статьи в JSON-файле.


```
"articles": [
  {
  "doi": "https://doi.org/10.14529/jsfi160402",
  "title": "In Situ Visualization and Production of Extract Databases",
  "authors": "Brad Joseph Whitlock, Earl P. N. Duque",
  "keywords": "In Situ, HPC, Visualization, Extract Database, SENSEI,
Libsim, Workflow",
  "axisX": "-0.2467905496351912",
  "axisY": "0.03694846510957771",
  "class": "0"
  },
]
```


Рисунок 10 – Пример представления статьи в файле JSON

Реализация модуля визуализации

Модуль визуализации осуществляет отображение результатов, полученных в ходе работы модулей системы, выполняющих рубрикацию статей. Результаты могут быть представлены в двух видах: список и точечная диаграмма.

Данный модуль реализует создание HTML-файла с прописанным стилем документа, отвечающим за внешний вид веб-страницы, и файла на языке JavaScript, позволяющего получить данные из JSON-файла, а также отобразить результаты рубрикации в соответствии с запросом пользователя приложения. На выбор пользователя, данные могут быть представлены в виде списка групп с возможностью перехода к веб-странице выбранной статьи, а также в виде точечной диаграммы, где каждая точка соответствует отдельной статье.

На рисунке 11 представлена визуализация в виде списка. Визуализация в виде списка предполагает наличие оглавления с полученными рубриками в начале страницы. Оглавление содержит в себе перечисление рубрик, соответствующий им набор ключевых слов, а также число статей, входящих в них. В качестве названия рубрики принимается ключевое слово, имеющее наибольший вес важности. Для перехода к конкретной рубрике можно воспользоваться данным оглавлением. При нажатии на название рубрики осуществляется переход к соответствующей части веб-страницы.

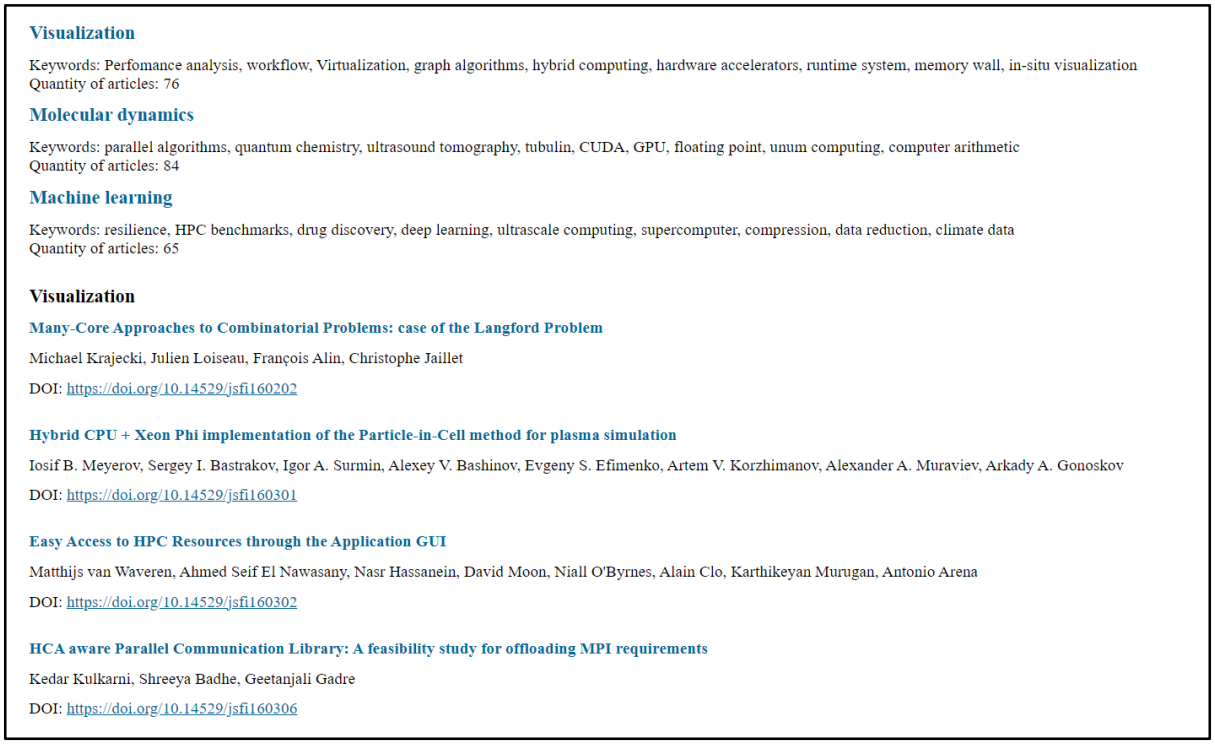

Рисунок 11 – Визуализация в виде списка

После оглавления по порядку отображаются полученные рубрики статей в виде списков с перечислением имеющихся статей. Каждый элемент списка представляет собой набор данных о статье: название и авторы статьи, а также ее DOI-ссылка. У пользователя также есть возможность перехода к персональной веб-странице статьи в научном журнале, которая осуществляется при нажатии на соответствующую ей ссылку.

Визуализация в виде точечной диаграммы предполагает отображение статей в виде точек на плоскости, согласно их числовым значениям, и легенды, представляющей собой набор ключевых слов рубрики. На рисунке 12 представлена визуализация полученных результатов в виде точечной диаграммы.

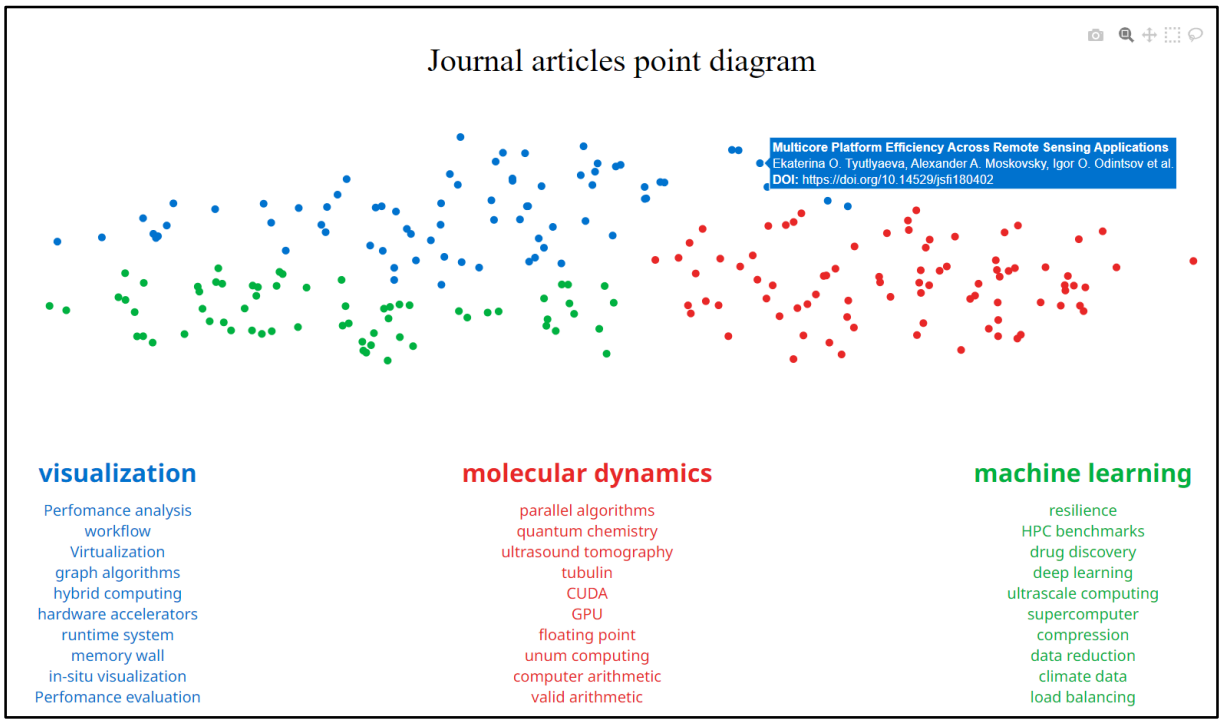

Рисунок 12 – Визуализация в виде точечной диаграммы

Отображение статей в виде точек на плоскости позволяет увидеть распределение статей по рубрикам в соответствии с их числовыми представлениями, полученными в ходе векторизации данных. Так, статьи распределяются на экране как на координатной плоскости, где каждая точка соответствует одной из статей. При наведении на точку отображаются название статьи, ее авторы, а также DOI-ссылка, а при нажатии выполняется открытие веб-страницы статьи на сайте журнала в отдельной вкладке.

Каждая рубрика определяется уникальным цветом, что позволяет определить принадлежность статьи к одной из них.

Для отображения данных в виде точечной диаграммы используется библиотека Plotly для JavaScript [32].

Данный способ отображения может быть полезен как администратору сайта, чтобы оценить качество распределения статей по рубрикам, так и пользователю, так как данный метод визуализации является более интерактивным.

Легенда представляет собой наборы ключевых слов для каждой рубрики. Ключевые слова распределяются в порядке убывания веса важности. В качестве названия рубрики принимается термин с наиболее высоким весом. Цвет текста каждого из наборов ключевых слов соответствует цвету точек данной рубрики.

Легенда также позволяет пользователю просмотреть список статей, содержащих любой термин из списка ключевых слов рубрики. На рисунке 13 представлено отображение списка статей на веб-странице при нажатии на ключевое слово «GPU».

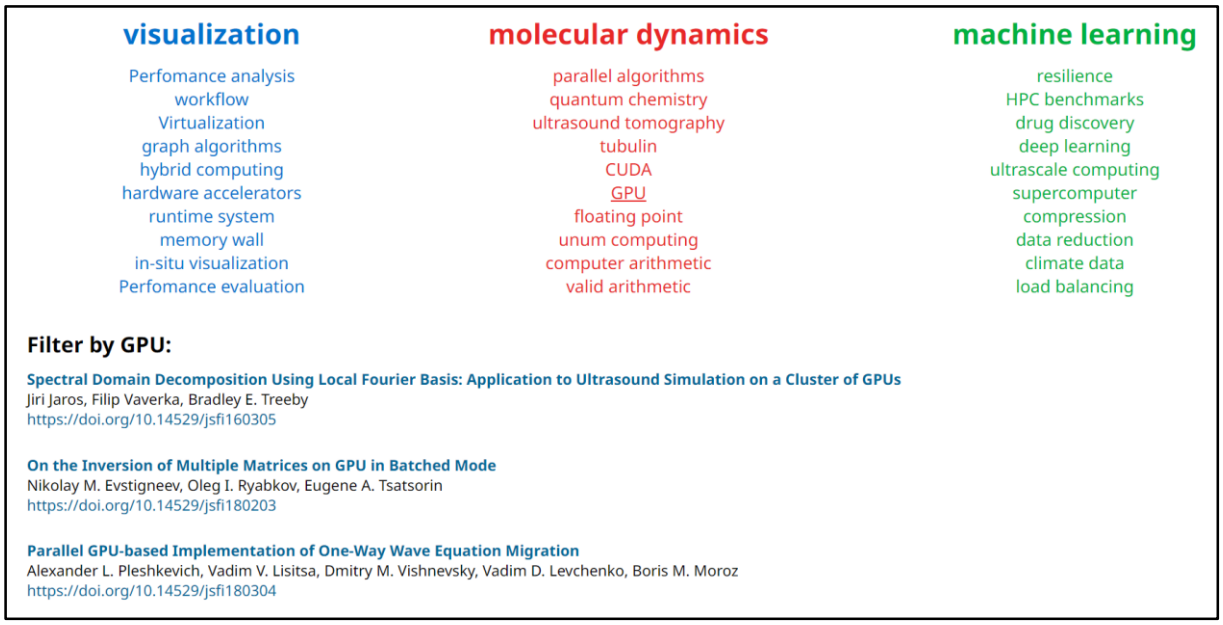

Рисунок 13 – Отображение статей при поиске по ключевому слову

Таким образом, при нажатии на любое из ключевых слов рубрики ниже отобразится дополнительный блок, в котором будет представлен список статей, входящих в данную рубрику и содержащих выбранный термин в своем списке ключевых слов.

3.3. Реализация пользовательского интерфейса

Реализация пользовательского интерфейса велась с учетом разработанных макетов. Реализованы следующие окна: главное меню, окно выбора директории, окно выбора ссылки, а также окно уведомления пользователя. Для реализации окон приложения была использована библиотека tkinter [33].

На рисунке 14 представлено главное меню пользовательского интерфейса разработанного приложения. Главное меню содержит три выпадающих списка и три кнопки.

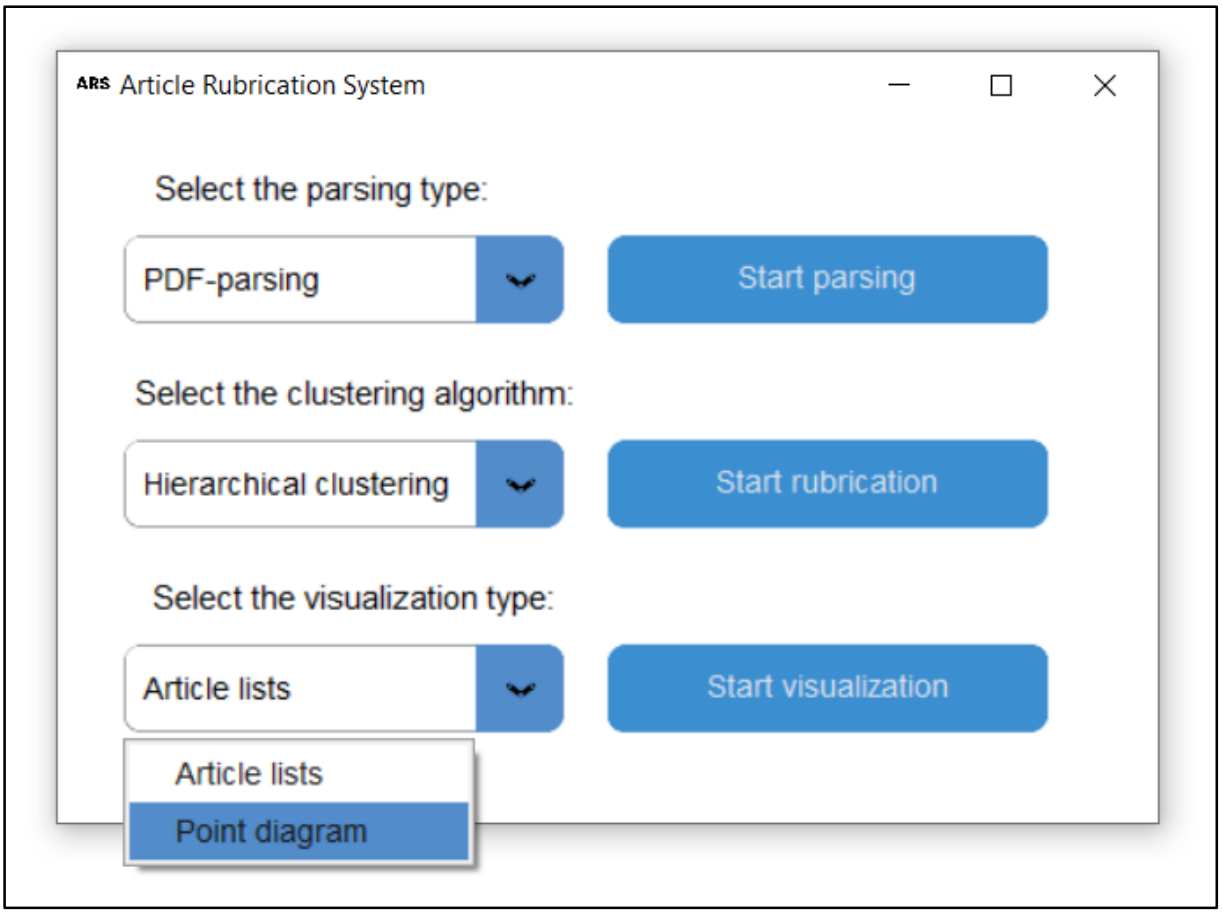

Рисунок 14 – Окно главного меню

Каждый выпадающий список соответствует кнопке напротив и позволяет выбрать один из параметров выполнения данной функции. Так, пользователь может выбрать тип парсинга данных: извлечение данных из PDF-

файлов или парсинг веб-страниц научного журнала, алгоритм кластеризации, который будет использоваться при рубрикации (иерархическая кластеризация или метод k-средних), а также способ визуализации (списки или точечная диаграмма). При нажатии на кнопку запускается соответствующая функция с выбранным параметром в списке.

Для выполнения подготовки данных необходимо указать директорию, в которой находятся файлы выпусков журнала, или ссылку на сайт, парсинг веб-страниц которого будет производиться.

На рисунке 15 представлено окно выбора директории. При нажатии на кнопку «Select Folder» отображается диалоговое окно, с помощью которого можно выбрать желаемую папку, в которой хранятся файлы. Указанный путь отобразится в текстовом поле.

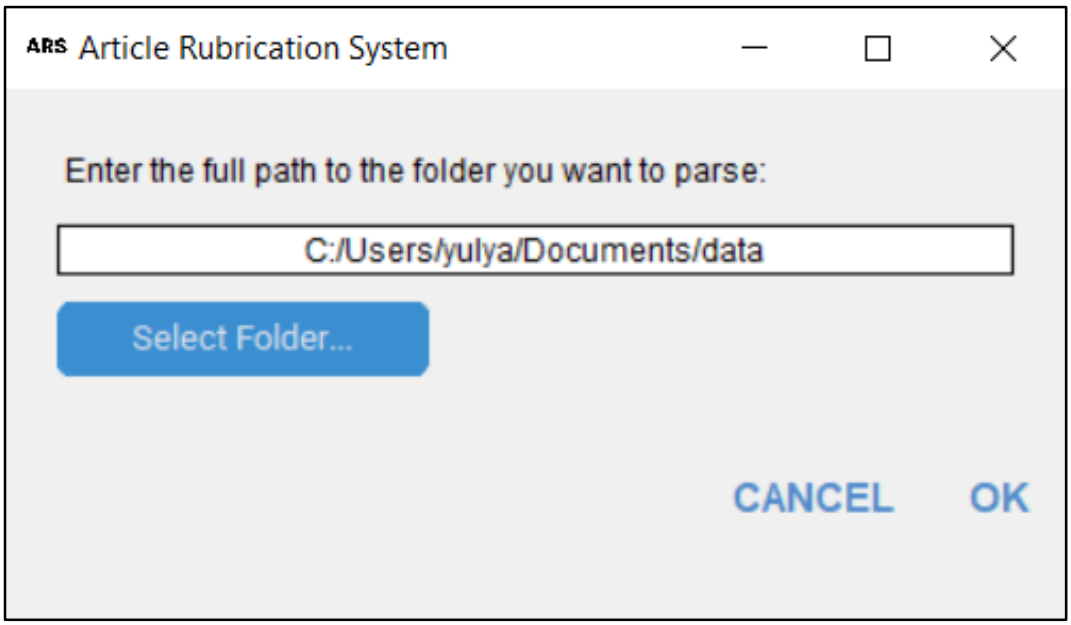

Рисунок 15 – Окно выбора директории

Окно ввода ссылки отличается от окна выбора директории тем, что не содержит кнопку, открывающую диалоговое окно. Для запуска подготовки данных с использованием парсинга веб-страниц пользователю необходимо ввести ссылку на сайт журнала напрямую в текстовое окно.

Процессы рубрикации и визуализации подразумевают наличие соответствующих файлов, созданных после выполнения предшествующих этапов, которые их формируют. В случае их отсутствия на экран выводится предупреждение.

При выполнении какой-либо задачи приложение информирует пользователя о результате работы. При получении некорректных данных появляется всплывающее окно, информирующее о характере ошибки. В случае успешного выполнения запроса на экран выводится соответствующее информационное сообщение.

На рисунке 16 представлена попытка запуска рубрикации до выполнения подготовки данных.

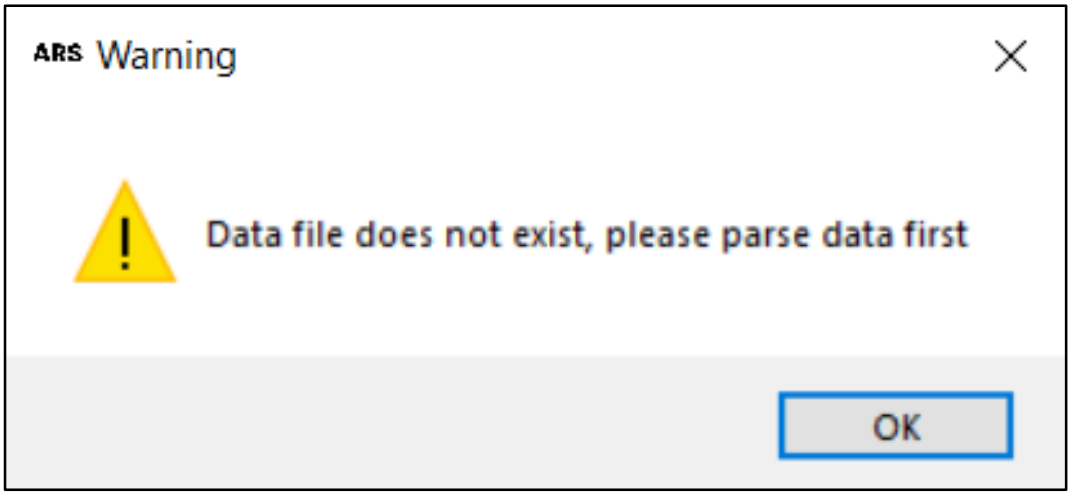

Рисунок 16 – Предупреждение об отсутствии файла

Выводы по третьей главе

В этой главе были рассмотрены программные средства, используемые при реализации приложения, были описаны процессы реализации отдельных компонентов и приложения в целом, а также был представлен разработанный пользовательский интерфейс приложения.

4. ТЕСТИРОВАНИЕ

4.1. Функциональное тестирование

В ходе данного тестирования проверялось соответствие приложения предъявленным функциональным требованиям. Тестирование приложения проводилось вручную. В таблице 3 приведен протокол ручного тестирования основных аспектов работы системы.

Таблица 3 – Функциональное тестирование приложения

№	Название теста	Действие	Ожидаемый результат	Тест пройден?
1	Проверка работоспособности кнопки извлечения данных.	В главном меню выбрать пункт «Start parsing» с параметром «Web-parsing».	Открывается окно ввода ссылки.	Да
2	Проверка работоспособности модуля рубрикации.	В главном меню выбрать пункт «Start rubrication».	Запускается процесс рубрикации с выбранным параметром. Результат рубрикации записывается в JSON-файл.	Да
3	Проверка работоспособности кнопки визуализации данных.	В главном меню выбрать «Start visualization».	Запускается процесс визуализации с выбранным параметром. Результат записывается в HTML-файл и файл JavaScript.	Да
4	Проверка работоспособности модуля при отсутствии файла с данными.	В главном меню выбрать пункт «Start rubrication», не запустив перед этим процесс извлечения данных.	Отображается окно с предупреждением об отсутствии необходимого файла.	Да
5	Проверка работоспособности окна ввода ссылки.	В окне ввода ссылки ввести ссылку на нужный сайт. После чего нажать «OK».	Запускается процесс извлечения данных с сайта. Полученные данные записываются в CSV-файл.	Да
6	Проверка работоспособности окна ввода директории.	В окне ввода ссылки выбрать нужную папку. После чего нажать «OK».	Запускается процесс извлечения данных из PDF-файлов. Полученные данные записываются в CSV-файл.	Да
7	Проверка ввода некорректных данных.	В окне ввода ссылки ввести случайный набор символов.	Отображается окно с ошибкой о невозможности получения доступа к сайту.	Да

4.2. Вычислительные эксперименты

В данном разделе представлены вычислительные эксперименты для набора данных, полученного после извлечения данных из PDF-файлов, соответствующих выпускам научного журнала «Supercomputing Frontiers and Innovations».

Для представления результатов иерархической кластеризации обычно используется дендрограмма. Данный метод визуализации показывает степень близости отдельных объектов и кластеров, а также графически демонстрирует последовательность их разделения. Высота разбиения отражает степень похожести объектов, так, чем ниже высота звена, тем сильнее похожи данные.

На рисунке 17 представлены результаты иерархической кластеризации для текущего набора данных в виде дендрограммы, из которой можно увидеть, что наиболее оптимальным решением будет разделение данных на три группы.

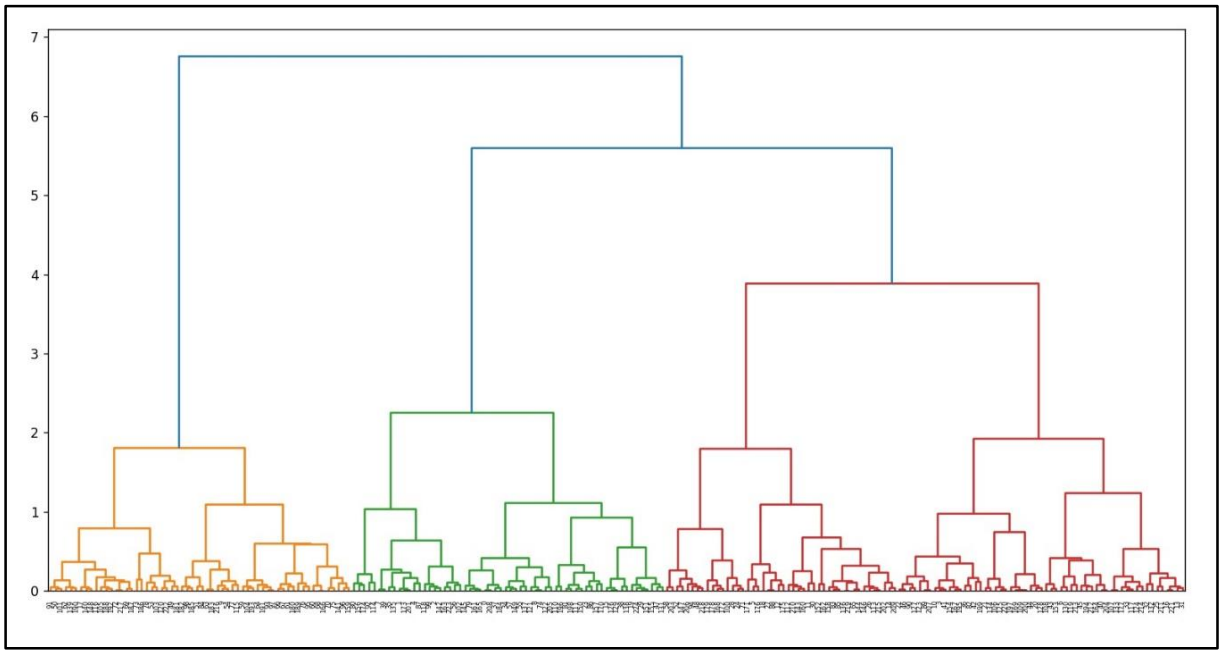

Рисунок 17 – Представление результатов иерархической кластеризации в виде дендрограммы

Для числа кластеров от двух до девяти было проведено вычисление силуэтного коэффициента, согласно которому оптимальным будет распределение данных из текущего набора по трем кластерам, так как ему соответствует наибольшее значение силуэтного коэффициента (рисунок 18).

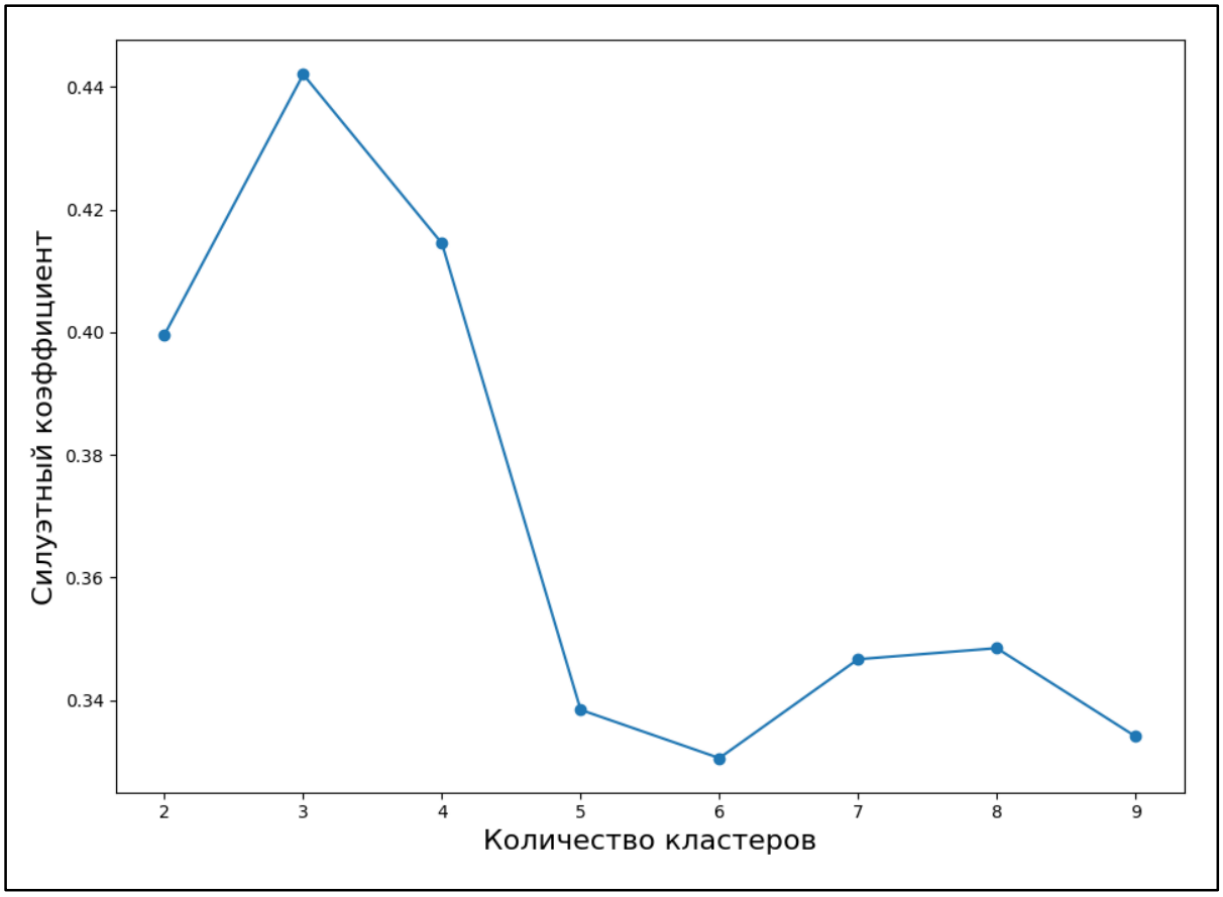

Рисунок 18 – График силуэтного коэффициента

Результаты кластеризации также представлены в виде точек на плоскости, где каждая точка соответствует одному из объектов. На рисунке 19 представлено распределение общего числа статей по трем кластерам согласно оптимальному числу, определенному путем вычисления силуэтного коэффициента. Статьи, объединенные в один кластер, окрашены общим цветом.

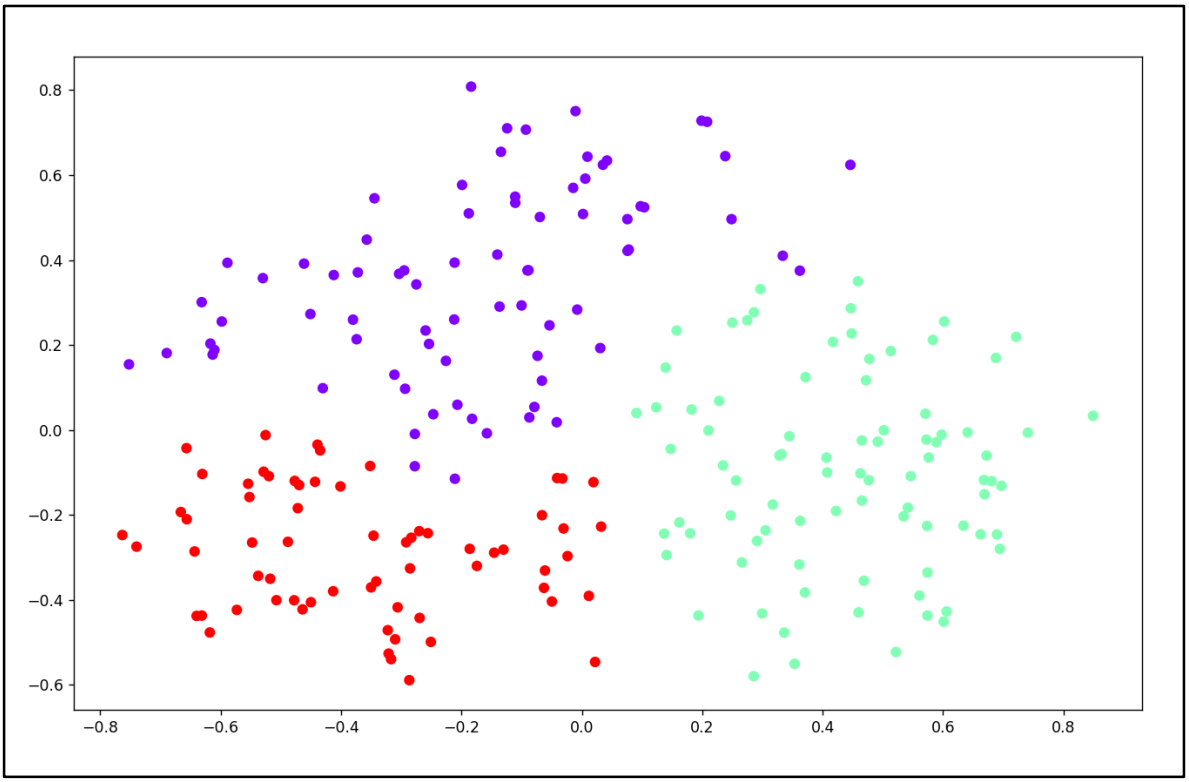

Рисунок 19 – Распределение объектов на плоскости

Для оценки качества кластеризации были сформированы отдельные тестовые наборы, наиболее наглядно отражающие принадлежность к различным тематическим группам статей. Результаты представлены в приложении Г.

Выводы по четвертой главе

В четвертой главе представлены результаты функционального тестирования, подтверждающего работоспособность приложения и его соответствие функциональным требованиям. В данной главе также были представлены вычислительные эксперименты для полученного набора данных статей научного журнала, иллюстрирующие корректность выполнения рубрикации.

ЗАКЛЮЧЕНИЕ

В рамках данной работы было разработано приложение, осуществляющее подготовку данных, их рубрикацию, а также визуализацию рубрик статей для научного журнала «Supercomputing Frontiers and Innovations». При этом были решены следующие задачи.

1. Выполнен анализ предметной области и проведен обзор существующих решений.
2. Выполнена разработка алгоритма рубрикации научных статей на основе кластерного анализа.
3. Выполнена разработка приложения для администратора научного журнала, позволяющее осуществлять подготовку, рубрикацию и визуализацию данных.
4. Проведено тестирование алгоритма на разработанных тестовых наборах.

Дальнейшая работа будет направлена на интеграцию созданного алгоритма в плагин для платформы Open Journal Systems, позволяющего выполнять автоматизированную рубрикацию на сайте научных журналов.

ЛИТЕРАТУРА

1. Supercomputing frontiers and innovations. [Электронный ресурс]. URL: https://superfri.org (дата обращения: 08.02.2023 г.).

2. Open Journal Systems. [Электронный ресурс]. URL: https://open-journalsystems.com/ (дата обращения: 08.02.2023 г.).

3. Carrot2 search results clustering engine. [Электронный ресурс]. URL: https://search.carrot2.org (дата обращения 08.02.2023 г.).

4. Hassani H., Beneki C., Unger S. et al. Text Mining in Big Data Analytics. // Big Data and Cognitive Computing, 2020. – V. 4. – № 1(1). – 34 p.

5. Y.A. Kravchenko, A.M. Mansour, J.H. Mohammad. Text vectorization using data mining methods. // Izvestiya SFedU. Engineering Sciences, 2021. – № 2(219). – 155–165 pp.

6. Осипова Ю.А., Лавров Д.Н. Применение кластерного анализа методом k-средних для классификации текстов научной направленности. // МСиМ, 2017. – № 3(43). – С. 108–121.

7. Kuznetsov S., Poelmans J. Knowledge representation and processing with formal concept analysis. // Wiley interdisciplinary reviews: Data mining and knowledge discover, 2013. – № 3. – 200–215 pp.

8. Naderalvojoud B., Akcapinar E.S. Sentiment aware word embeddings using refinement and senti-contextualized learning approach. // Neurocomputing, 2020. – V. 405. – 149–160 pp.

9. Si Y., Wang J., Xu H., Roberts K. Enhancing clinical concept extraction with contextual embeddings. // Journal of the American Medical Informatics Association, 2019. – V. 26. – № 11. – 1297–1304 pp.

10. Mikolov T., Sutskever I., Chen K. Distributed Representations of Words and Phrases and their Compositionality. // Advances in Neural Information Processing Systems 26 (NIPS 2013), 2013. – 3111–3119 pp.

11. Pennington J., Socher R., Manning C. Glove: Global vectors for word representation. // Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014. – 1532–1543 pp.

12. Maaten L., Postma E., Herik J. Dimensionality Reduction: A Comparative Review. // Journal of Machine Learning Research, 2007. – № 10(1). – 1–41 pp.

13. Егоров А.В., Куприянова Н.И. Особенности методов кластеризации данных. // Известия Южного федерального университета. Технические науки, 2011. – № 11. – С. 174–178.

14. Пархоменко П.А., Григорьев А.А., Астраханцев Н.А. Обзор и экспериментальное сравнение методов кластеризации текстов. // Труды ИСП РАН, 2017. – № 2. – С. 161–200.

15. Hotho A., Nürnberger A., Paab G. A Brief Survey of Text Mining. // LDV Forum: GLDV Journal for Computational Linguistics and Language Technology, 2005. – V. 20. – 19–62 pp.

16. M. Gupta, R. Jain. A Performance Evaluation of SMCA Using Similarity Association and Proximity Coefficient Relation for Hierarchical Clustering. // International Journal of Engineering Trends and Technology, 2014. – V. 15. – № 7. – 354–359 pp.

17. Сивоголовко Е.В. Методы оценки качества четкой кластеризации. // Компьютерные инструменты образования, 2011. – № 4. – С. 14–31.

18. Nevzorova O., Gizatullin B. Analysis of the cluster structure of collections of mathematical papers with different UDC codes. // Kazan Federal University, 2022. – 10 p.

19. PyCharm. [Электронный ресурс] URL: https://www.jetbrains.com/pycharm/ (дата обращения: 10.03.2021 г.).

20. Article Rubrication System. [Электронный ресурс]. URL: https://github.com/ytsmm/Article-Rubrication-System (дата обращения: 19.05.2023 г.).

21. PyPDF2. PyPI. [Электронный ресурс] URL: https://pypi.org/project/PyPDF2/ (дата обращения: 14.03.2023 г.).

22. Requests. PyPI. [Электронный ресурс] URL: https://pypi.org/project/requests/ (дата обращения: 14.03.2023 г.).

23. Beautifulsoup4. PyPI. [Электронный ресурс] URL: https://pypi.org/project/beautifulsoup4/ (дата обращения: 14.03.2023 г.).

24. Pandas. [Электронный ресурс] URL: https://pandas.pydata.org/ (дата обращения: 16.03.2023 г.).

25. NLTK: Natural Language Toolkit. [Электронный ресурс] URL: https://www.nltk.org/ (дата обращения: 20.03.2023 г.).

26. GloVe: Global Vectors for Word Representation. [Электронный ресурс]. URL: https://nlp.stanford.edu/projects/glove/ (дата обращения: 26.03.2023 г.).

27. Gensim. PyPI. [Электронный ресурс]. URL: https://pypi.org/project/gensim/ (дата обращения: 26.03.2023 г.).

28. PCA decomposition. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html (дата обращения: 30.03.2023).

29. Agglomerative clustering. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html (дата обращения: 03.04.2023).

30. KMeans. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html (дата обращения: 03.04.2023).

31. Silhouette Score. [Электронный ресурс] URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html (дата обращения: 10.04.2023).

32. Plotly. Graphic Libraries. [Электронный ресурс] URL: https://plotly.com/graphing-libraries (дата обращения: 23.04.2023 г.).

33. Tkinter Python interface. [Электронный ресурс] URL: https://docs.python.org/3/library/tkinter.html (дата обращения: 11.05.2023 г.).

ПРИЛОЖЕНИЯ

Приложение А. Спецификация вариантов использования

Спецификация вариантов использования (ВИ) разработанной системы приведена в таблицах 1–3.

Таблица 1 – Спецификация ВИ «Подготовить данные»

Прецедент: Подготовить данные
ID: 1
Краткое описание: Запуск процесса извлечения данных о статьях.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь нажимает на кнопку «Start parsing». 2. Пользователь вводит путь к данным. 3. Приложение запускает функцию извлечения данных, передавая параметр парсинга. 4. Приложение записывает полученные данные в CSV-файл. 5. Приложение уведомляет пользователя о выполнении работы.
Постусловия: Нет
Альтернативные потоки: Нет

Таблица 2 – Спецификация ВИ «Выполнить рубрикацию»

Прецедент: Выполнить рубрикацию
ID: 2
Краткое описание: Запуск алгоритма рубрикации, выполняющего распределения данных по группам.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: Наличие файла с данными о статьях.
Основной поток: 1. Прецедент начинается, когда пользователь нажимает на кнопку «Start rubrication». 2. Приложение запускает процесс рубрикации, передавая параметр кластеризации. 3. Приложение извлекает данные о статьях из CSV-файла. 4. Приложение запускает выполнение рубрикации. 5. Приложение записывает полученные результаты в JSON-файл. 6. Приложение уведомляет пользователя о выполнении работы.
Постусловия: Нет
Альтернативные потоки: Нет

Таблица 3 – Спецификация ВИ «Визуализировать результаты»

Прецедент: Визуализировать результаты.
ID: 3
Краткое описание: Создание веб-страницы, отображающей результаты рубрикации.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: Наличие файла с результатами рубрикации.
Основной поток: 1. Прецедент начинается, когда пользователь нажимает на кнопку «Start visualization». 2. Приложение запускает модуль визуализации, передавая параметр визуализации. 3. Приложение создает файлы, формирующие веб-страницу согласно запросу пользователя с отображением результатов рубрикации данных. 4. Приложение уведомляет пользователя о выполнении работы.
Постусловия: Нет
Альтернативные потоки: Нет

Приложение Б. Интерфейсы модулей приложения

На этапе проектирования были сформированы интерфейсы основных модулей приложения, содержащие описание функции, а также ее входные и выходные данные.

На листингах 1–7 представлены интерфейсы основных программных модулей приложения.

Листинг 1 – Интерфейс модуля извлечения данных

```
# Функция выполняет извлечение данных о статьях
# parsingType – строковый параметр, содержащий тип парсинга
# link – строковый параметр, содержащий путь к данным
# result – возвращаемое значение: 1 при успешном выполнении или ошибка
def parsing(parsingType, link):
    return result
```

Листинг 2 – Интерфейс модуля рубрикации данных

```
# Функция запускает поочередный вызов модулей, выполняющих рубрикацию
# clusterType – строковый параметр, содержащий тип кластеризации
# result – возвращаемое значение: 1 при успешном выполнении или ошибка
def rubrication(clusterType):
    return result
```

Листинг 3 – Интерфейс модуля предобработки данных

```
# Функция выполняет предобработку текстовых данных
# data - строковый массив объединенных данных о статьях (название, ключевые
слова, аннотация)
# optData – строковый массив обработанных данных
def preprocessing(data):
    return optData
```

Листинг 4 – Интерфейс модуля векторизации данных

```
# Функция выполняет векторизацию текстовых данных
# data - строковый массив предобработанных данных
# vectorData – массив pd.DataFrame() векторных представлений статей
def vectorizing(data):
    return vectorData
```

Листинг 5 – Интерфейс модуля кластеризации данных

```
# Функция выполняет кластеризацию числовых данных
# vectors - массив pd.DataFrame() векторных представлений статей
# clusterType – строковый параметр, содержащий тип кластеризации
# labels – целочисленный массив с номерами кластеров
# optimalScore – целочисленный параметр, соответствующий числу кластеров
# number – целочисленный массив с количеством статей в каждом кластере
def clustering(vectors, clusterType):
    return labels, optimalScore, number
```

Листинг 6 – Интерфейс модуля определения ключевых слов

```
# Функция выполняет определение ключевых слов кластеров
# keywords – строковый массив с ключевыми словами статей
# labels – целочисленный массив с номерами кластеров
# number – целочисленный параметр, соответствующий числу кластеров
# clusterKeywords – строковый массив с ключевыми словами кластеров
# weights – двумерный массив со списками весов ключевых слов кластеров
def getKeywords (keywords, labels, number):
    return clusterKeywords, weights
```

Листинг 7 – Интерфейс модуля визуализации

```
# Функция выполняет визуализацию полученных результатов
# visualType – строковый параметр, содержащий тип визуализации
# result – возвращаемое значение: 1 при успешном выполнении или ошибка
def visualization (visualType):
    return result
```

Приложение В. Код основных модулей приложения

В данном разделе представлены листинги модулей векторизации и кластеризации данных. Программный код остальных модулей представлен в личном репозитории на github [20].

Листинг 8 – Векторизация данных

```
import numpy as np
import pandas as pd
import gensim.downloader as api
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, normalize

corpus = api.load("glove-wiki-gigaword-50")


# Функция выполняет векторизацию текстовых данных
# data - строковый массив предобработанных данных
# vectorData – вещественный массив pd.DataFrame() двумерных векторов
def vectorizing(data):
    features = []

    for tokens in data:
        zero_vector = np.zeros(corpus.vector_size)
        vectors = []

        for token in tokens:
            if token in corpus:
                try:
                    vectors.append(corpus[token])
                except KeyError:
                    continue

        if vectors:
            vectors = np.asarray(vectors)
            avg_vec = vectors.mean(axis=0)
            features.append(avg_vec)

        else:
            features.append(zero_vector)

    vectorData = pcaModule(features)

    return vectorData


# Функция выполняет редукцию векторов
# X - вещественный массив векторных представлений статей
# X_principal – вещественный массив pd.DataFrame() двумерных векторов
def pcaModule(X):
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)
    X_normalized = normalize(X_scaled)
    X_normalized = pd.DataFrame(X_normalized)

    X_principal = PCA(n_components=2).fit_transform(X_normalized)
    X_principal = pd.DataFrame(X_principal)
    X_principal.columns = ['X', 'Y']

    return X_principal
```

Листинг 9 – Кластеризация данных

```
import numpy as np
import scipy.cluster.hierarchy as shc
from sklearn.metrics import silhouette_score
from sklearn.cluster import KMeans, AgglomerativeClustering


# Функция выполняет кластеризацию числовых данных
# vectors – вещественный массив pd.DataFrame() двумерных векторов
# clusterType – строковый параметр, содержащий тип кластеризации
# labels – целочисленный массив с номерами кластеров
# n – целочисленный параметр, соответствующий числу кластеров
# size – целочисленный массив с количеством статей в каждом кластере
def clusterization(vectors, clusterType):
    if clusterType == 'Hierarchical clustering':
        labels, n, size = hierarchial(vectors)
        return labels, n, size
    elif clusterType == 'K-Means clustering':
        labels, n, size = kmeans(vectors)
        return labels, n, size


# Кластеризация k-means
def kmeans(X):
    k = []  # Число кластеров
    clResult = []  # Результаты кластеризации

    for i in range(2, 10):
        clResult.append(KMeans(n_clusters=i))
        k.append(i)

    scores = [silhouette_score(X, clResult[i].fit_predict(X)) for i in
range(8)]
    optimalScore = k[np.argmax(scores)]  # Оптимальное число кластеров
    model = KMeans(n_clusters=optimalScore).fit(X)
    labels = model.labels_  # Список номеров кластеров
    number = [sum([1 for j in labels if j == i]) for i in range(opti-
malScore)]  # Число объектов в кластерах

    return labels, optimalScore, number


# Иерархическая кластеризация
def hierarchial(X):
    k = []  # Число кластеров
    clResult = []  # Результаты кластеризации

    for i in range(2, 10):
        clResult.append(AgglomerativeClustering(n_clusters=i))
        k.append(i)

    scores = [silhouette_score(X, clResult[i].fit_predict(X)) for i in
range(8)]
    optimalScore = k[np.argmax(scores)]  # Оптимальное число кластеров
    model = AgglomerativeClustering(n_clusters=optimalScore).fit(X)
    labels = model.labels_  # Список номеров кластеров
    number = [sum([1 for j in labels if j == i]) for i in range(opti-
malScore)]  # Число объектов в кластерах

    return labels, optimalScore, number
```

Приложение Г. Тестирование работы алгоритма

Для оценки качества выполнения рубрикации были сформированы тестовые наборы, отражающие различные тематические группы статей.

В таблице 4 представлен тестовый набор данных, где каждый объект характеризуется набором ключевых слов. Ожидаемый кластер определен вручную и показывает, какие объекты должны быть объединены в одну группу. Итоговый кластер определяется в результате выполнения рубрикации приложением.

Таблица 4 – Распределение данных тестового набора № 1 по кластерам

№	Ключевые слова	Ожидаемый кластер	Итоговый кластер
1	docking, protein-ligand, global minimum, force field, quantum-chemical method	1	1
2	docking, protein-ligand, global optimization, tensor train, force field, quantum- chemical method	1	1
3	HPC, Visualization, Extract Database, SENSEI, Libsim, Workflow	2	2
4	Scientific Visualization, Data Staging Methods, Data Reductions, HPC	2	2
5	AlgoWiki, parallel structure, algorithms properties, Top500 methodology, problems, methods, algorithms, implementations, computing platforms	3	3
6	scalability, supercomputer, AlgoWiki, parallel structure, problems, methods, algorithms, implementations, computing platforms	3	3
7	numerical weather prediction, global atmosphere model, computational efficiency, I/O optimization	4	4
8	numerical weather prediction, climate modeling, hybrid computing, programming models	4	4
9	workflow, heterogeneous storage, climate, weather	4	4
10	regional climate model, domain size, simulations, computational efficiency	4	4
11	vector computers, NVIDIA GPUs, graph algorithms, graph framework, VGL, CUDA, optimisation	5	5
12	graph algorithms, NEC SX-ACE, vector computing, data-intensive applications	5	5
13	seismic imaging, multi-arrival seismic migration, GPU aiwlib	6	6
14	GPU, nested OMP, MPI, seismic imaging, migration	6	6

На рисунке 1 изображено распределение значений силуэтного коэффициента для числа кластеров в пределах заданных границ, наибольшее значение соответствует шести кластерам, соответственно, статьи будут распределены по шести группам.

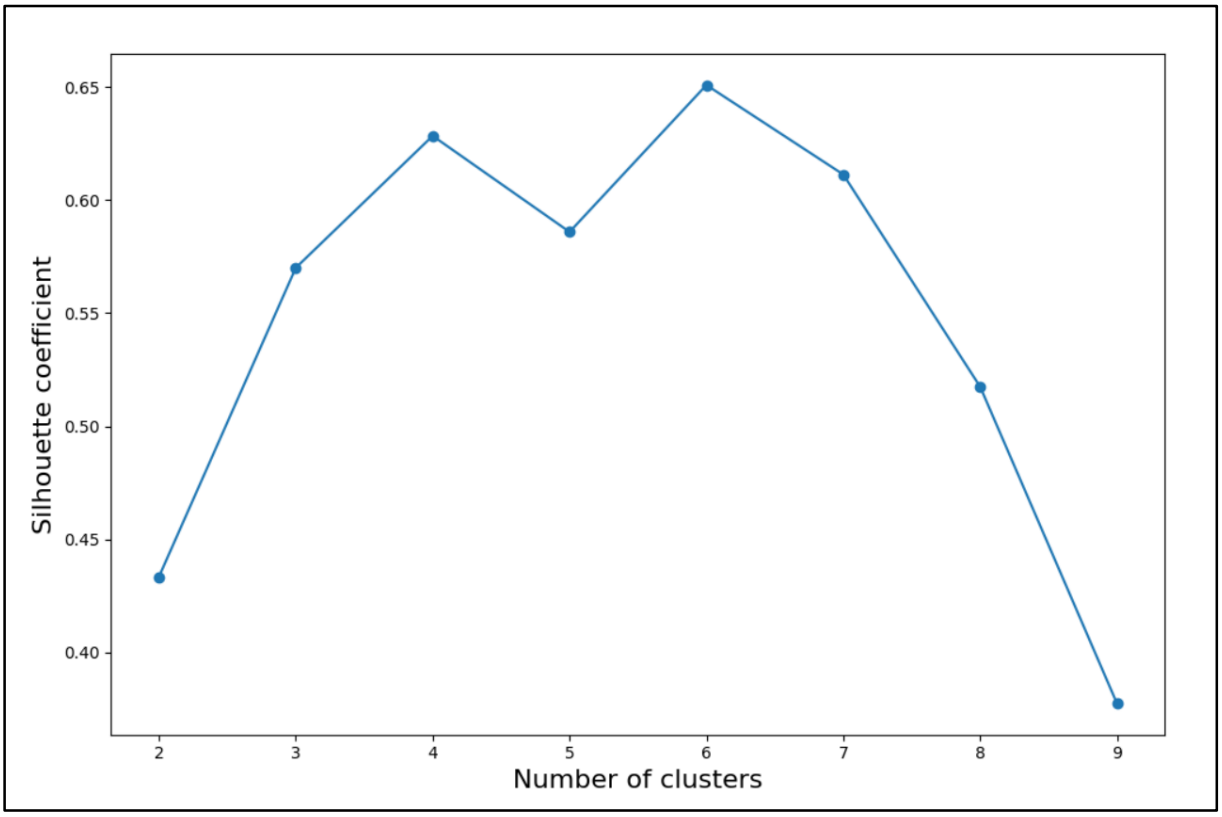

Рисунок 1 – График силуэтного коэффициента

Результаты кластеризации представлены в виде точек на плоскости, где каждая точка соответствует одному из объектов. Координаты точек определяются на этапе векторизации. На рисунке 2 представлено распределение общего числа статей по четырем кластерам. Статьи, объединенные в один кластер, окрашены общим цветом.

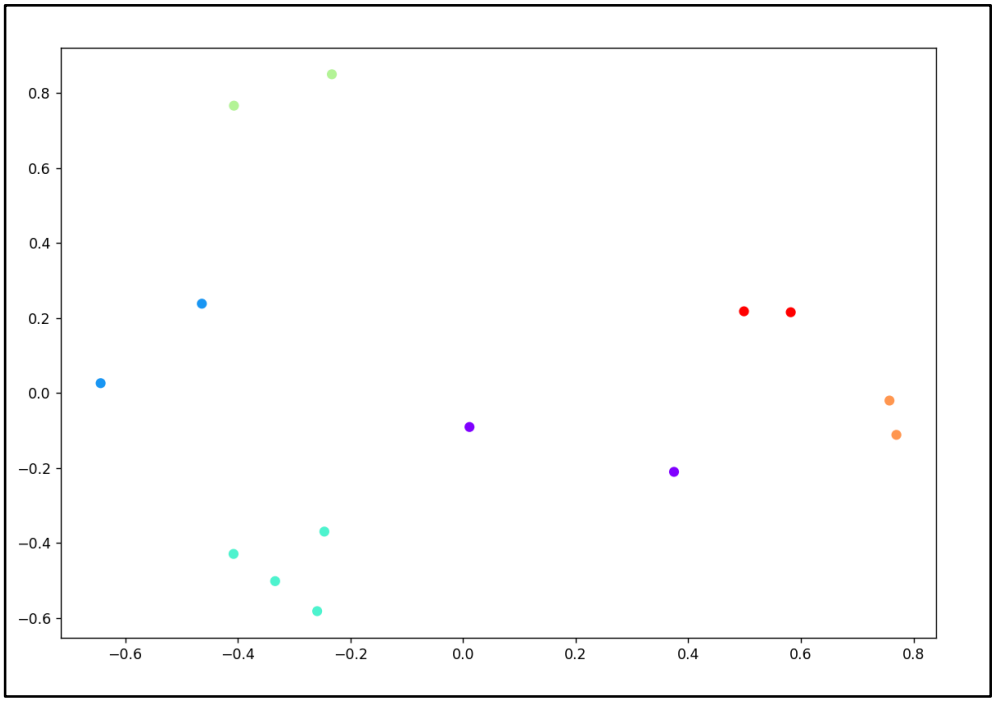

Рисунок 2 – Распределение объектов на плоскости

На рисунке 3 представлен результат иерархической кластеризации для текущего набора данных в виде дендрограммы. Она показывает степень близости отдельных объектов и последовательность их объединения. Оптимальное число кластеров по дендрограмме выбирается визуально.

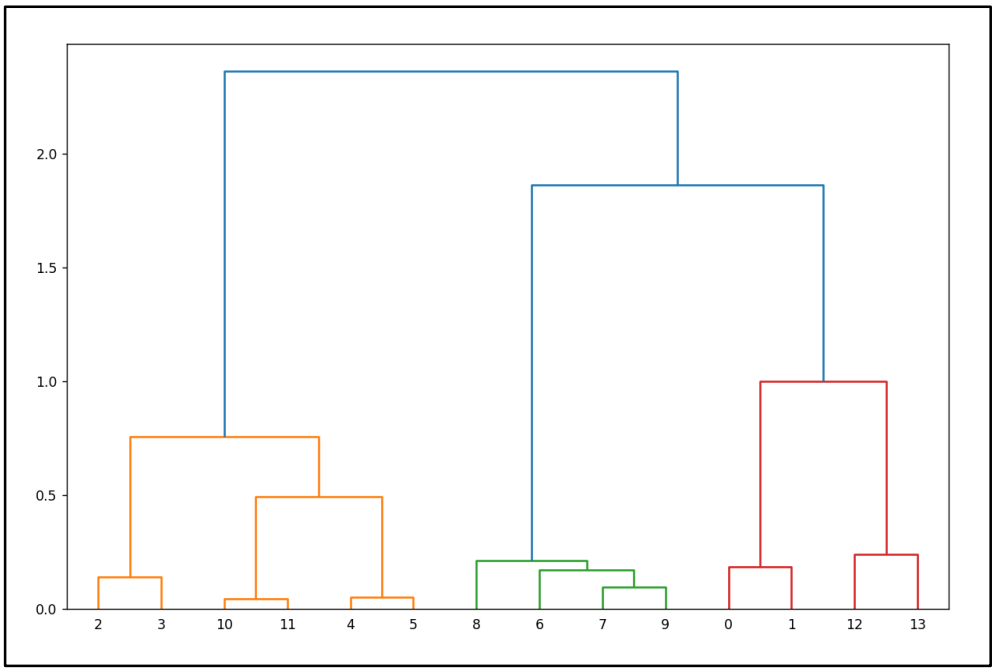

Рисунок 3 – Результат иерархической кластеризации в виде дендрограммы

В таблице 5 представлено распределение данных тестового набора №2 по кластерам.

Таблица 5 – Распределение данных тестового набора №2 по кластерам

№	Ключевые слова	Ожидаемый кластер	Итоговый кластер
1	SARS-CoV2 main protease, QM/MM MD, GPU-accelerated algorithms, substrate activation	1	1
2	SARS-CoV2 main protease, QM/MM MD, molecular dynamics, covalent inhibitor, molecular docking	1	1
3	SX-Aurora TSUBASA, OS Offload, VH Call, VEO, vectorization ratio	2	2
4	SX-Aurora TSUBASA, optimization, vector computing, power efficiency, Himeno benchmark, HPCG	2	2
5	NVIDIA GPU, NEC SX-Aurora TSUBASA, liquid crystals, HPC, co-design, perfomance optimization, Monte Carlo, cubic lattice	2	2
6	molecular dynamics, molecular mechanics, QM/MM MD, GPU-accelerated algorithms, enzyme-substrate complexes, reaction intermediates	1	1
7	metallo-lactamase, boronate inhibitors, MD, QM/MM MD, quantum theory of atoms in molecules (QTAIM), GPU-accelerated algorithms	1	1
8	molecular dynamics, slow-growth thermodynamic integration method, DPS protein, DNA stabilization, DNA-DPS binding free energy	1	1
9	shallow water, supercomputer modeling, heterogeneous computing systems, MPI, OpenMP, CUDA	3	3
10	land surface model, soil, river network, MPI, OpenMP	3	3
11	sea dynamics modeling, variational data assimilation, observations, sea surface temperature	3	3

На рисунках 4–6 представлена визуализация результатов кластеризации для тестового набора данных № 2 в виде дендрограммы и объектов на плоскости, а также график изменения силуэтного коэффициента.

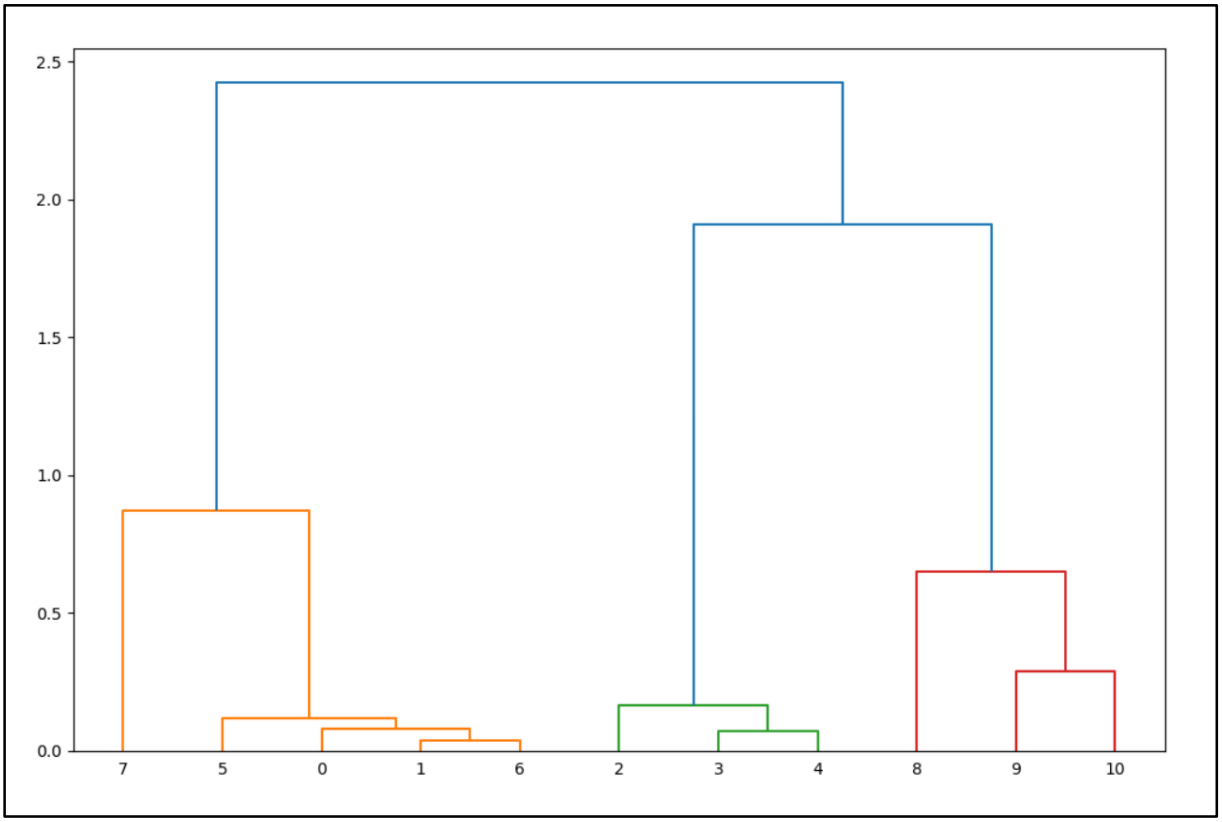

Рисунок 4 – Результат иерархической кластеризации в виде дендрограммы

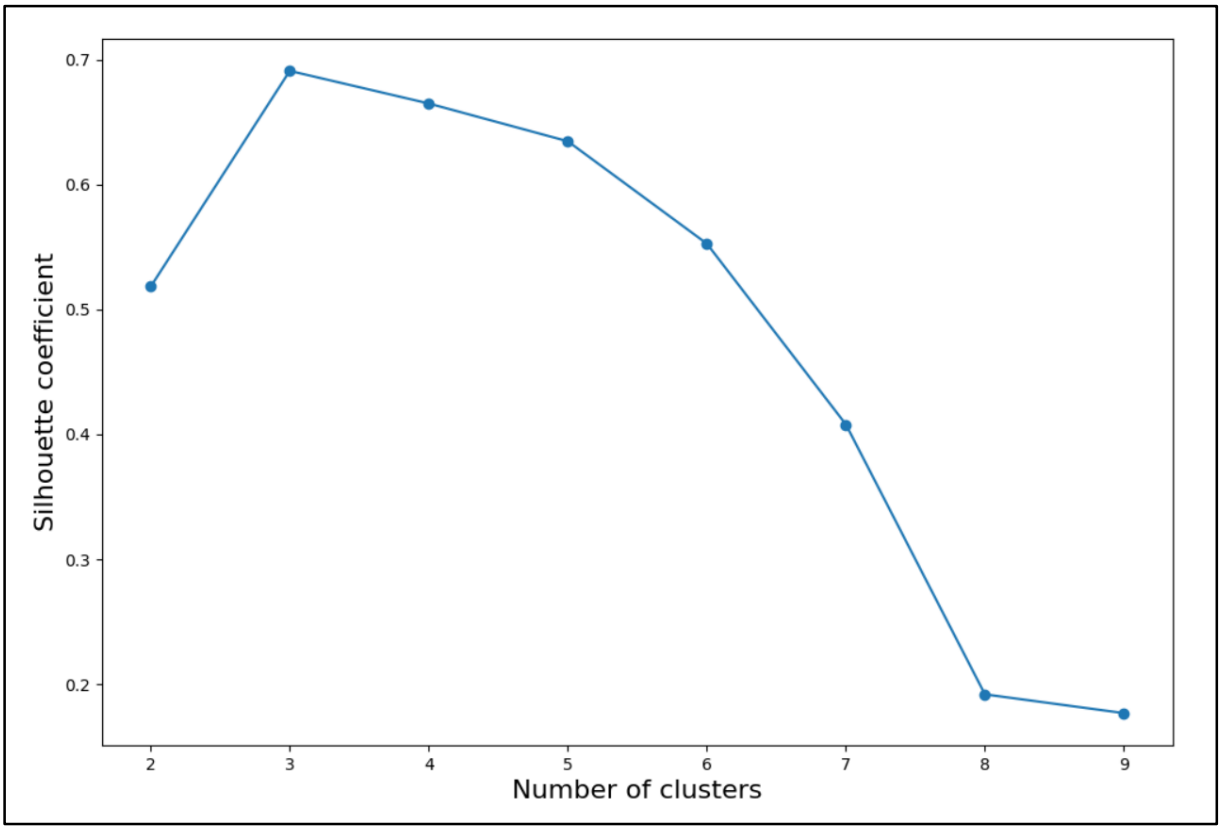

Рисунок 5 – График силуэтного коэффициента

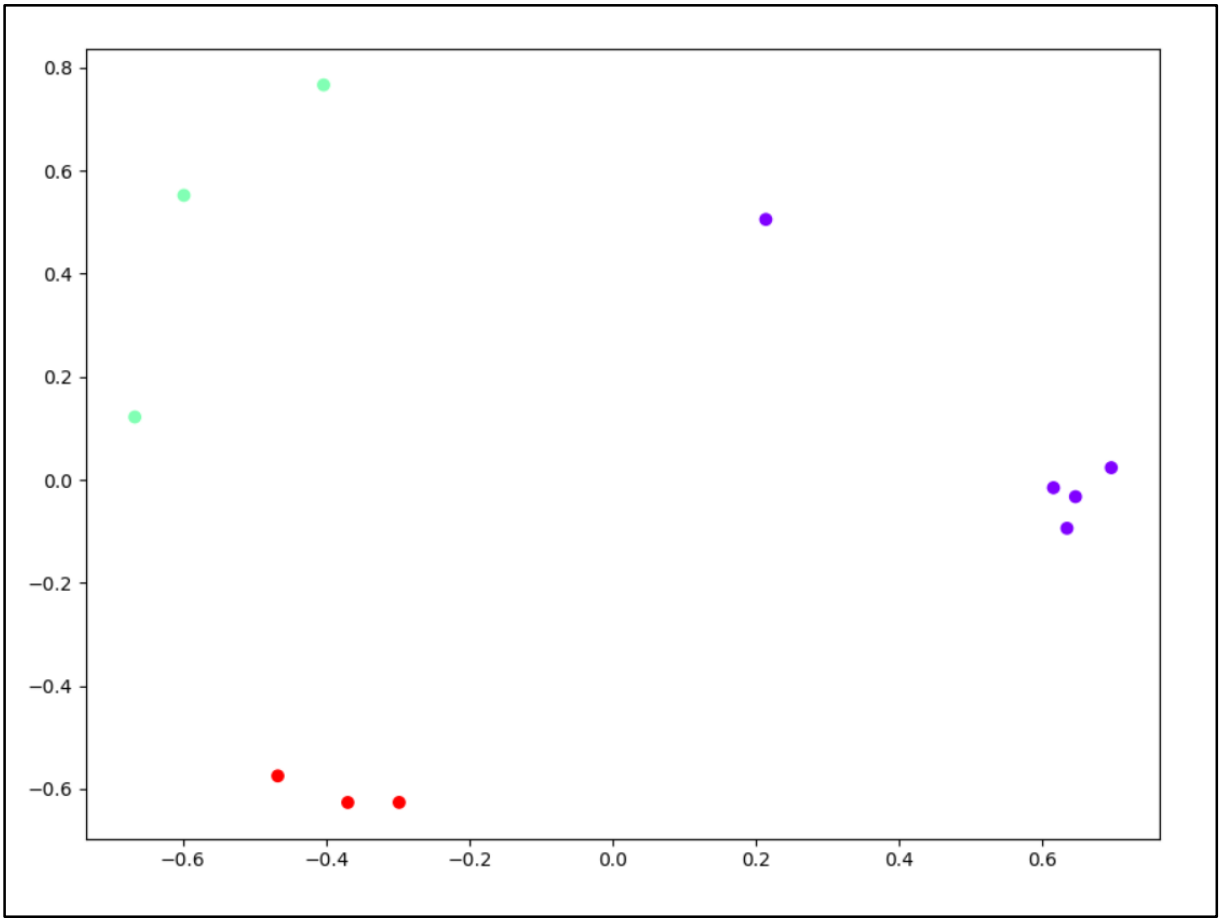

Рисунок 6 – Распределение объектов на плоскости

Полученные на тестовых наборах результаты демонстрируют, что разработанный алгоритм рубрикации позволяет корректно распределить статьи по различным предметным областям.