DOI 10.37882/2223-2966.2023.11.25

РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА КЛАССИФИКАЦИИ ТЕКСТОВ С УЧЕТОМ МОДЕЛИ ВЕКТОРНОЙ СЕМАНТИКИ

DEVELOPMENT TEXT CLASSIFICATION SOFTWARE TOOL BASED ON VECTOR SEMANTICS MODEL

A. Rusakov
D. Malevich
N. Savranskij

Summary. This article presents a study on the development of a software tool for text classification taking into account the vector semantics model. The article provides an overview of modern text classification software. The relevance of the proposed research is substantiated, the object, subject of research, scope and limitations of the software are determined. The main problems solved by the software are formulated, and various mathematical methods, algorithms and software tools that can be used for software development are defined. The article concludes that a hybrid approach for text classification taking into account vector semantics models is promising and very popular in both scientific and practical terms.

Keywords: text classification, vector semantics, information security.

Русаков Алексей Михайлович
Старший преподаватель,
МИРЭА — Российский технологический университет
rusal@bk.ru
Малевич Данила Денисович
МИРЭА — Российский технологический университет
9714075@mail.ru
Савранский Никита Сергеевич
МИРЭА — Российский технологический университет
fayray@mail.ru

Аннотация. В данной статье представлено исследование по разработке программного инструмента классификации текстов с учетом модели векторной семантики. В статье представлен обзор современных программных средств классификации текстов. Обосновывается актуальность предлагаемого исследования, определяются объект, предмет исследования, область применения и ограничения программного обеспечения. Сформулированы основные задачи, решаемые программным обеспечением, и определены различные математические методы, алгоритмы и программные средства, которые могут быть использованы для разработки программного обеспечения. В статье делается вывод о том, что гибридный подход для классификации текстов с учетом моделей векторной семантики является перспективным и весьма востребованным как в научном, так и в практическом плане.

Ключевые слова: классификация текстов, векторная семантика, информационная безопасность.

В современном мире каждый сталкивается с проблемой быстрого поиска нужной информации. То есть, информации, соответствующей запросу пользователя и необходимая ему. Люди тратят много времени на поиск информации, источников, где ее можно найти. Это связано с тем, что им необходимо проанализировать определенный предмет для выполнения определенного задания или цели. Самым популярным средством массовой информации сегодня является глобальная сеть Интернет, где люди со всего мира могут найти нужную им информацию.

Поисковые системы, такие как Яндекс и Google, например, предоставляют все источники, отсортированные в соответствии с ключевыми словами в запросе пользователя. Однако описанный выше поиск информации не всегда дает ожидаемые результаты, так как он основан на словах, которые люди указывают в строке поиска. Проблема возникает тогда, когда автор статьи называет слова, указанные в запросе, в другой формулировке, что приводит к потере необходимой информации. Для решения проблемы быстрого поиска релевантной информации в настоящее время широко используются полнотекстовые поисковые системы, которые описывают документы на основе нескольких слов, введенных пользователем. Наиболее известные современные методы классификации основаны на методах машинного обучения [1, 8]. Эти поисковые системы особенно хорошо подходят для анализа текстов с незаконной информацией о терроризме, беспорядках и протестах, торговле наркотиками и т.д. Благодаря таким системам эксперты могут отфильтровать источники, которые не подходят для конкретной темы, без необходимости читать представленный в них материал, экономя важный ресурс — время. Исходя из этого, была инициирована разработка программного инструмента для классификации текстов.

Необходимость извлечения терминов из текста возникает в различных задачах: машинный перевод; информационный поиск; извлечение информации [2, 3]. Объем и динамизм информации, обрабатываемой в этой области сегодня, делает задачу автоматического извлечения терминов и ключевых слов очень актуальной. Извлеченные термины можно использовать для повышения эффективности обработки документов, такой как индексирование, извлечение и классификация, а также для создания и расширения глоссариев [4, 8].

Постановка задачи

Цель работы является повышение безопасности информационных систем за счет разработки специальных алгоритмов и программного решения для интеллектуального анализа текстов с целью классификации текстов с учетом векторной модели.

Основные задачи, решаемые в работе:
- Исследование предметной области;
- Обзор и анализ существующих программных средств для классификации текстов с учетом векторной модели;
- Обзор современных математических методов для интеллектуального анализа текстов в информационных системах;
- Разработать алгоритмы для классификации текстов с учетом векторной модели;
- Создать программный продукт на основе проведенных исследований.

Объект исследования — оцифрованные данные текстовой информации которые можно обработать на компьютере с целью извлечения фактов и получения новой информации.

Предмет исследования — математические методы, модели, методики и алгоритмы, позволяющие классифицировать тексты по их содержанию в автоматическом режиме с помощью современных методов интеллектуального анализа текстов.

Разработка системы классификации текстов на естественном языке

Для решения проблемы быстрого поиска релевантной информации в настоящее время широко используются полнотекстовые поисковые системы, которые описывают документы на основе нескольких слов, введенных пользователем.

Первый этап включает в себя процедуру подготовки текста. Текст отправляется в преамбулу и выполняется предобработка. После завершения этапа предварительной обработки у вас есть готовое текстовое поле, которое вы можете использовать для обучения модели.

Второй этап описывает процедуру получения списка текстов, ранжированных по количеству входных слов. На вход подается подготовленный в начале текстовый корпус. Далее идет классификация текста, связанного с набором слов. Результатом является список текста.

Выбор алгоритма предобработки текстов

Алгоритмы и подходки предварительной подготовки текстовой информации [4]

1. Токенизация — процесс разбиения текста на слова, предложения, абзацы, параграфы.
2. Исключение стоп-слова — часто, в тексте одних слов больше, чем других и они не несут смысла. В таких словах есть помехи для дальнейшего анализа.
3. Стемминг — сложность русского языка заключается в том, что одно и то же слово может быть написано по-разному [6]. Например, слова плохой и плохая имеют одинаковый смысл, но разные формы. Перед началом машинного обучения нужно привести слова в одну форму, чтобы уменьшить размерность. Целью стемминга является усечение концов слов. Поскольку этот процесс является ресурсоемким, этот шаг не используется при внедрении программных продуктов. Вместо этого используется лемматизация.

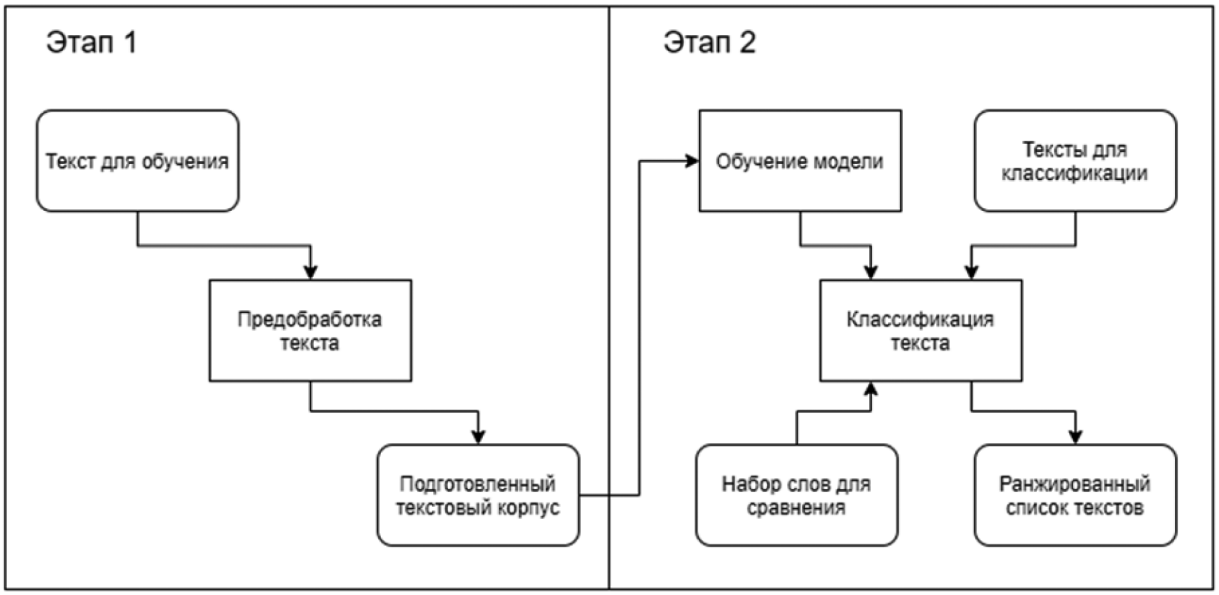

Рис. 1. Схема классификатора

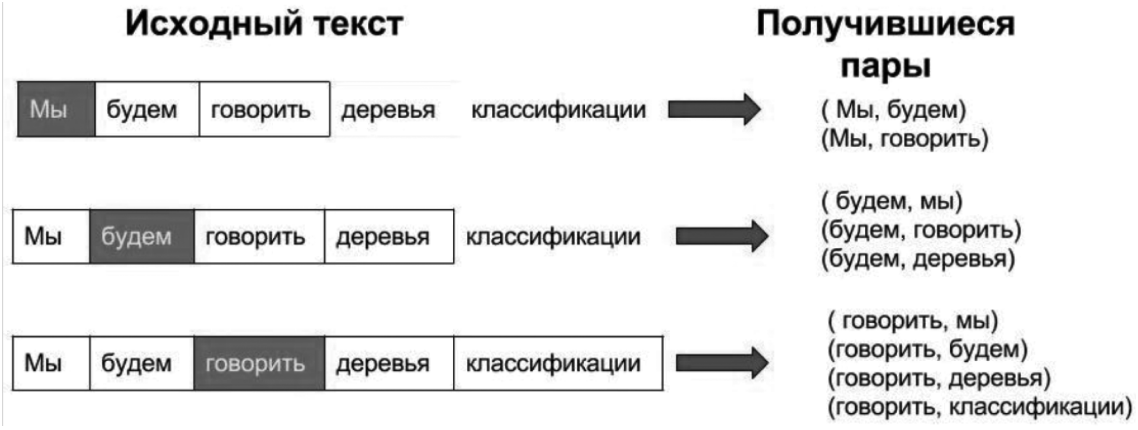

Рис. 2. Пример работы с «векторным» окном размером 5 слов

4. Приведение к первой форме с помощью лемматизациеи — стемминг не всегда дает желаемый результат [5]. Чтобы получить желаемый результат, нам нужно определить первую форму слова.
5. Преобразование текста в нижний регистр — важный шаг, необходимый для правильного сравнения слов в процессе обработки.

Итак, получим алгоритм предобработки текста:

1. Токенизация:

Есть много библиотек, которые могут справиться с этой задачей.

2. Лемматизация:

Для этого вы можете использовать библиотеку pymorphy2, которая делает то, что вы хотите.

Методы и алгоритмы обработки текста для его классификации

Это метод, направленный на сопоставление слов из заданного словаря векторов малой размерности.

Этот алгоритм реализован в библиотеке word2vec и использует метод построения сжатого пространства векторов слов.

Существует два типа моделей Word2Wec: модель CBOW и модель Skip-Gram.

One-hot кодирование применяется пословно. Слово представлено в виде вектора, размеры которого равны значениям словаря, один в месте, соответствующем положению слова в словаре. На диаграмме показана используемая архитектура нейронной сети (см. рис. 3).

Скрытый слой — это весовая матрица, количество строк которой равно размеру словаря, а количество столбцов — размеру нового пространства. После обучения модели эта матрица представляет собой представление слов в векторном пространстве.

$$\partial(Z)_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}},$$

где z — число, получившееся после умножения входного вектора на весовой вектор, соответствующий i-тому слову. Каждый выходной нейрон имеет значение больше нуля, которое в сумме равно единице. То есть каждый i-ый нейрон дает вероятность того, что i-ое слово находится рядом с входным словом. Если два слова часто встречаются вместе, то нейронная сеть подберет веса, дающие сходные распределения на выходе. Это означает, что такие слова располагаются рядом в нашем векторном пространстве.

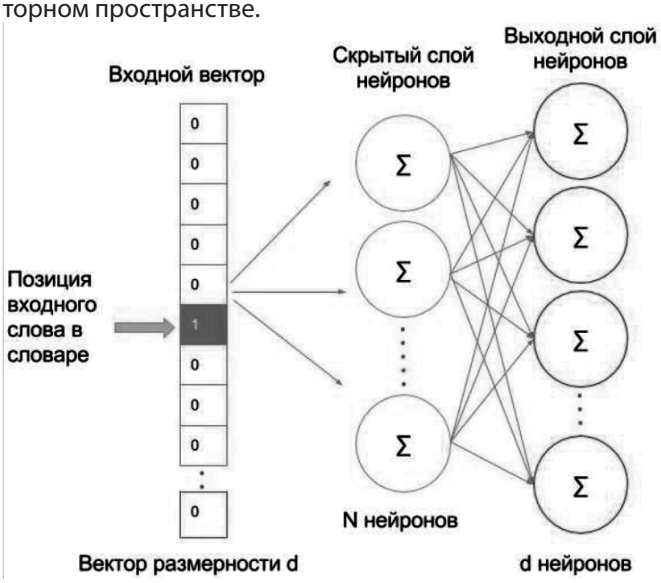

Рис. 3. Проектирование архитектуры нейронной сети

У word2vec имеется свои недостатки. К примеру, с его помощью не могут представляться слова, которых нет в обучающей выборке.

Эта проблема решена с помощью последней библиотеки fastText с помощью N-граммовых символов. Можно

выдавать векторные представления редких слов, так как некоторые слова могут быть перефразированы [7].

Проектирование программных модулей

Поскольку в разработанном веб-приложении используется модульное разделение, информационная система может быть усовершенствована более гибко, и даже если в модуле возникает проблема, работа самой программы не будет нарушена.

Программный комплекс будет состоять из следующих частей:

1. Модуль создания корпуса текстов
2. Модуль создания текста
3. Модуль обучения модели
4. Модуль классификации

На вход модуля подготовки корпуса текстов подается необработанный текст в формате CSV. Выполните действия предварительной обработки, такие как токенизация, удаление стоп-слов и лемматизация. Этот модуль сгенерирует подготовленный текстовый файл для дальнейшей работы в формате CSV с информацией в структуре (см. рис. 5).

Работа модуля создания текста аналогична работе модуля создания корпуса текстов. Разница лишь в том, что готовые файлы хранятся в отдельной директории сервера. На вход модуля классификации текстов подается датасет специально подготовленных текстов и настройки модели (см. рис. 6).

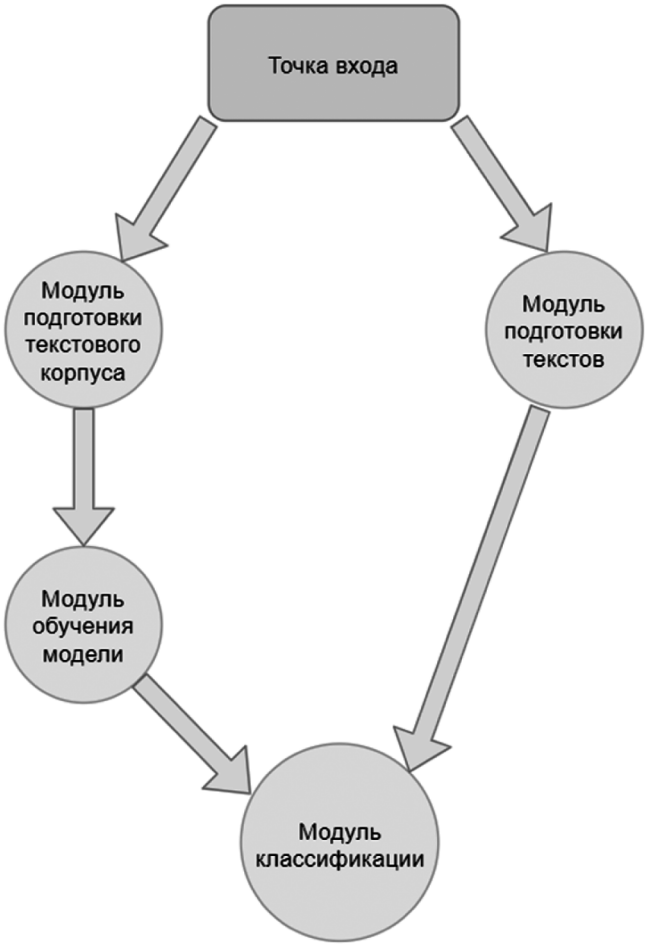

Рис. 4. Схема программного комплекса

	A	B	C	D	E	F	G
1	safeguards_txt						
2	['ответственный', 'должностной', 'лицо', 'полномочие', 'обязанность', 'ответственность'						
3	['разработать', 'локальный', 'акт', 'вопрос', 'обработка', 'персональный', 'данные', 'лицо'						
4	['металлический', 'сейф', 'охранный', 'пожарный', 'сигнализация', 'физический', 'охрана						
5	['разработать', 'документ', 'работа', 'персональный', 'данные', 'сотрудник', 'ознакомить'						
6	['установить', 'сейф', 'хранение', 'личный', 'дело', 'работник', 'персональный', 'данные',						
7	['назначить', 'ответственный', 'организация', 'обработка', 'персональный', 'данные', 'изд						
8	['назначить', 'ответственный', 'организация', 'обработка', 'персональный', 'данные', 'изд						
9	['ограниченный', 'доступ', 'сейф', 'бумажный', 'носитель', 'приказ', 'назначение', 'ответс						
10	['назначить', 'лицо', 'ответственный', 'организация', 'обработка', 'персональный', 'данны						
11	['доступ', 'персональный', 'данные', 'работник', 'предоставить', 'специально', 'уполномо						
12	['оператор', 'обработка', 'персональный', 'данные', 'принимать', 'необходимый', 'правов						
13	['назначение', 'ответственный', 'организация', 'обработка', 'персональный', 'данные', 'ут						
14	['разграничение', 'право', 'доступ', 'сотрудник', 'база', 'персональный', 'данные', 'наличи						
15	['назначение', 'приказ', 'должностной', 'лицо', 'допустить', 'обработка', 'персональный',						

Рис. 5. Представление подготовленного текстового корпуса

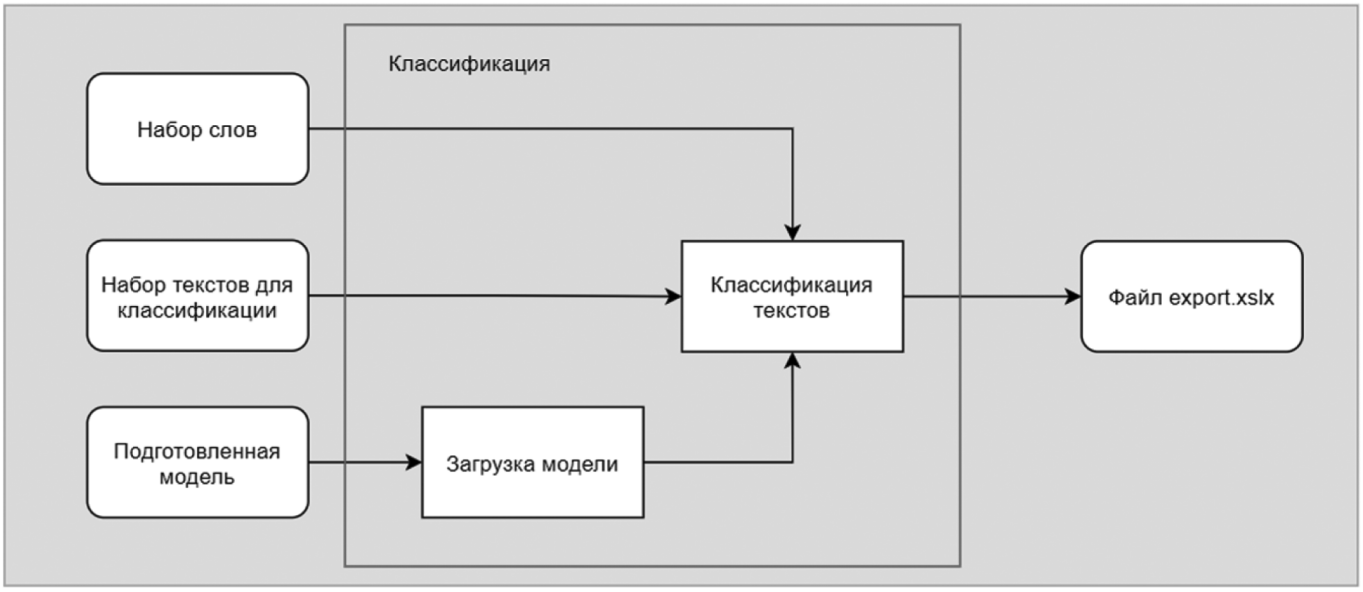

Рис. 6. Блок-схема работы классификатора

Описание алгоритма классификации

Рис. 7. Блок-схема алгоритма классификации

Алгоритм переподготовки текстов

Вместе с файлом отправляется набор заголовков столбцов из файла вместе с данными для обработки (см. рис. 8).

1. Токенизация выполняется стандартным способом библиотеки NLTK.
2. Лемматизация выполняется путем перебора массива слов и применения функции normal_forms из библиотеки pymorphy2 к каждому слову.

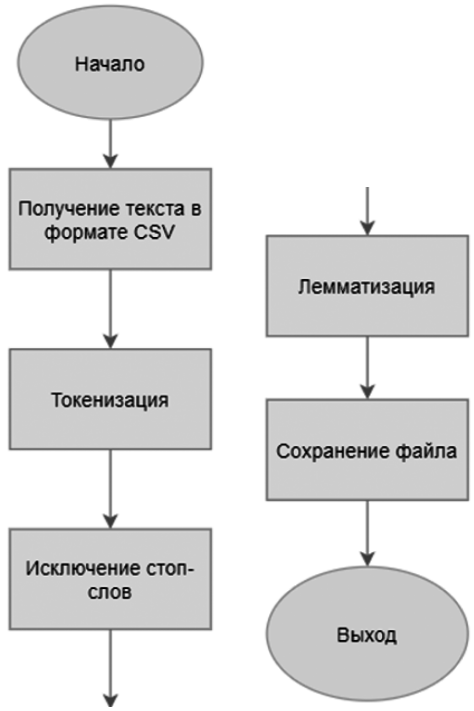

Рис. 8. Блок-схема алгоритма переподготовки текстов

На этапе сохранения файла полученный воссозданный CSV-файл помещается в соответствующую папку, относительно задачи.

Алгоритм подготовки модели

Алгоритм подготовки модели выделен в отдельный шаг. Это связано с тем, что это довольно ресурсоемкая задача и эффективнее загрузить подготовленную модель во время классификации, чем ждать пока подготовленная модель будет обучена (см. рис. 9).

Рис. 9. Блок-схема алгоритма подготовки модели

Этот алгоритм использует сбор данных сайта. Затем создается массив слов, который заполняется в процессе прохождения цикла. После подготовки массива слов создается объект библиотеки FastText.

Алгоритм классификации (см. рис. 10)

Сначала происходит получение необходимых данных для классификации, включая текст, необходимый для классификации, обученную модель и список слов. Если слова совпадают, выходные значения записываются в массив. Последним шагом является сохранение выходного файла на сервер.

Тестирование программного обеспечения

В ходе тестирования программного обеспечения была проведена серия экспериментов, в результате которых был сделан вывод о том, что разработанное программное обеспечение может быть использовано для решения задач классификации текстов (см. рис. 11).

Заключение

Были рассмотрены и проанализированы математические методы и программные средства для классификации текстов произвольной длины. Все методы имеют свои плюсы и минусы в рамках различных задач. Однако можно отметить наиболее эффективный метод классификации — метод, основанный на векторном представлении слов. При правильном использовании этого метода можно получить наиболее точные результаты классификации.

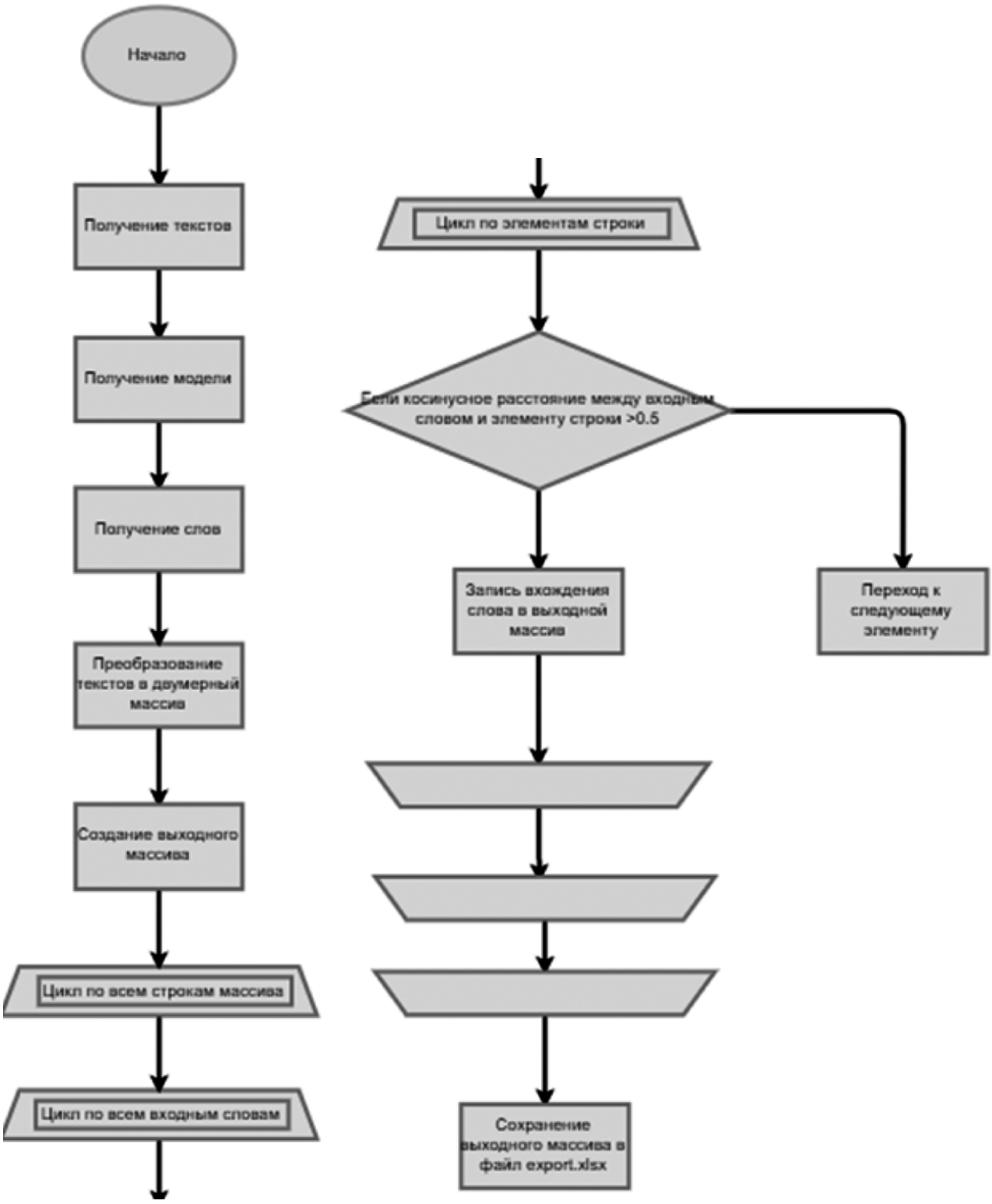

Рис. 10. Блок-схема алгоритма классификации

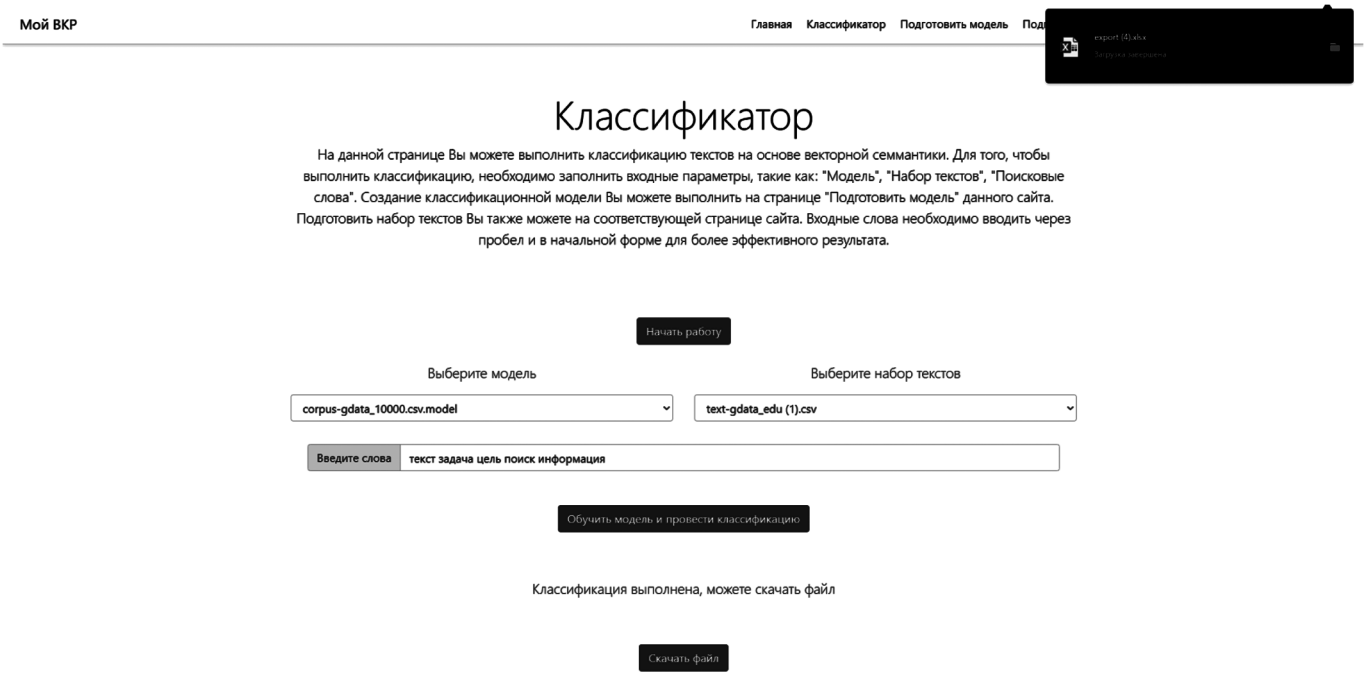

Рис. 11. Пример работы программы

ЛИТЕРАТУРА

1. Авербух К.Я. Общая теория термина. — Иваново: Ивановский государственный университет, 2004.
2. Антонов В.Ю., Ефремова Н.Э. Автоматическое выявление терминологических вариантов в русскоязычных текстах // Ломоносов — 2010: Материалы XVII Международной научной конференции студентов, аспирантов и молодых ученых: секция «Вычислительная математика и кибернетика». Сборник тезисов. — 2010. — С. 80.
3. Бородин А.И., Вейнберг Р.Р., Литвишко О.В. Методы обработки текста при создании чат-ботов //Хуманитарни Балкански изследвания. — 2019. — Т. 3. — №. 3. — С. 108–111.
4. Машечкин И.В. и др. Методы автоматического аннотирования и выделения ключевых слов в задачах обнаружения экстремисткой информации в сети Интернет //Современные информационные технологии и ИТ-образование. — 2016. — Т. 12. — №. 1.
5. Сорокин А.Н., Родионов Д.А. Применение нейросетей и машинного обучения для анализа содержания веб-страниц // Современные информационные технологии и ИТ-образование: Сборник научных трудов. 2018. Т. 14. №. 2. С. 52–57.
6. Татьяна Б. Методы автоматической классификации текстов. — 2020, С. 738–745
7. Allahyari M. et al. A brief survey of text mining: Classification, clustering and extraction techniques //arXiv preprint arXiv:1707.02919. — 2017.
8. Silge J., Robinson D. Text mining with R: A tidy approach. — «O'Reilly Media, Inc.», 2017.

Журнал «Современная наука: актуальные проблемы теории и практики»