

Санкт-Петербургский государственный университет

Направление "Большие данные и распределенная цифровая платформа"

**Лабораторная работа по дисциплине
Системное программирование в Linux**

"Распределенная обработка и анализ больших данных."

Выполнил:

Зайнуллин Мансур Альбертович

Группа: 23.Б16-пу

Руководитель:

Киямов Жасур Уткирович

Санкт-Петербург

2024

Оглавление

1	Цель работы	3
2	Описание задачи	4
3	Теоретическая часть	6
3.1	Сбор данных из социальной сети ВКонтакте	6
3.2	Параллельная обработка данных	6
3.3	Выделение тем из постов с помощью модели ChatGPT	6
3.4	Эмбединги текста	7
3.5	Кластеризация эмбедингов	7
3.6	Кэширование данных	7
3.7	Анализ и представление результатов	8
4	Описание программы	9
4.1	Основные модули и их функции	9
4.1.1	main.py	9
4.1.2	vk_topics_analyzer_tools.py	10
4.2	Алгоритм работы программы	11
5	Рекомендации пользователю	14
5.1	Инструкция по установке программы	14
5.1.1	Шаг 1: Установка Python 3.9	14
5.1.2	Шаг 2: Установка Git	14
5.1.3	Шаг 3: Клонирование репозитория	15
5.1.4	Шаг 4: Создание виртуального окружения и установка зависимостей	15
5.1.5	Шаг 5: Настройка конфигурации	15
5.2	Инструкция по эксплуатации программы	16
5.2.1	Шаг 1: Запуск программы	16
5.2.2	Шаг 2: Настройка параметров анализа	17
5.2.3	Шаг 3: Получение VK API ключа	17
5.2.4	Шаг 4: Запуск анализа	17
5.2.5	Шаг 5: Загрузка кэша	18

5.2.6	Шаг 6: Сохранение результатов	18
5.2.7	Шаг 7: Остановка программы	18
6	Контрольный пример	19
6.0.1	Шаг 3: Использование графического интерфейса . . .	20
7	Контрольный пример	21
8	Выводы по работе	27
9	Полезные ссылки	28

1 Цель работы

Разработать программное приложение для сбора, обработки и анализа больших объемов текстовых и визуальных данных из социальной сети ВКонтакте с целью выявления популярных тем и трендов.

2 Описание задачи

Необходимо разработать программное приложение для обработки и анализа больших объемов текстовых и визуальных данных из социальной сети ВКонтакте. Программа должна выполнять следующие задачи:

1. Сбор данных:

- Извлечение текстовых сообщений и изображений из постов указанных групп ВКонтакте.
- Обработка данных из нескольких источников параллельно для повышения эффективности.

2. Выделение тем:

- Автоматическое извлечение ключевых тем (тегов) из содержимого постов с использованием модели искусственного интеллекта.

3. Преобразование в эмбединги:

- Преобразование выделенных тем в числовые векторы (эмбединги) для дальнейшего анализа.

4. Кэширование данных:

- Сохранение собранных данных и эмбедингов в файл кэша для повторного использования и уменьшения времени обработки при последующих запусках программы.

5. Кластеризация эмбедингов:

- Группировка эмбедингов тем с помощью выбранного алгоритма кластеризации для выявления схожих тем и определения популярных трендов.
- При наличии сохраненного кэша данных переход непосредственно к этапу кластеризации для подбора

оптимальных гиперпараметров метода кластеризации и получения наилучших результатов.

6. Анализ и представление результатов:

- Выявление и отображение популярных тем и трендов на основе кластеризации данных.
- Генерация отчётов с результатами анализа для дальнейшего использования.

3 Теоретическая часть

3.1 Сбор данных из социальной сети ВКонтакте

Сбор данных из ВКонтакте осуществляется через официальное API, предоставляющее доступ к различным ресурсам социальной сети, включая посты, комментарии и вложения. Для получения данных необходимо иметь ключ доступа (access token), который позволяет аутентифицироваться и выполнять запросы к API. Запросы к API выполняются с использованием HTTP-запросов, а данные возвращаются в формате JSON для дальнейшей обработки.

3.2 Параллельная обработка данных

Параллельная обработка данных предполагает выполнение нескольких операций одновременно с целью повышения эффективности и сокращения времени обработки. В данном проекте используется многопоточность, где каждый поток отвечает за сбор и обработку данных из отдельной группы ВКонтакте. Это позволяет эффективно использовать ресурсы процессора и ускоряет процесс анализа больших объемов данных.

3.3 Выделение тем из постов с помощью модели ChatGPT

Для автоматического выделения тем (тегов) из постов используется модель ChatGPT. С помощью специально разработанного промпта (prompt.txt) модель анализирует текст и изображения постов, выделяя ключевые темы без детализации. Это позволяет структурировать данные и подготовить их для дальнейшего анализа. Модель обучена распознавать основные темы, обсуждаемые в постах, что облегчает процесс кластеризации и выявления трендов.

3.4 Эмбединги текста

Эмбединги представляют собой числовые векторы фиксированной размерности, которые отражают семантическое содержание текста. Преобразование тем в эмбединги осуществляется с использованием моделей машинного обучения, таких как модели OpenAI. Эмбединги позволяют количественно сравнивать и кластеризовать темы на основе их семантической близости, что является основой для последующего анализа и выявления трендов.

3.5 Кластеризация эмбедингов

Кластеризация — метод группировки объектов на основе их сходства. В данном проекте используются алгоритмы кластеризации, такие как K-Means, DBSCAN и HDBSCAN, для группировки эмбедингов тем. Выбор алгоритма зависит от характеристик данных и целей анализа. Кластеризация позволяет выявить группы схожих тем, что упрощает анализ популярных трендов. Для оптимизации результатов кластеризации используется подбор гиперпараметров, основанный на ранее сохраненных данных кэша.

3.6 Кэширование данных

Кэширование данных подразумевает сохранение промежуточных результатов обработки в файл для последующего использования. В проекте реализовано сохранение собранных постов, выделенных тем и их эмбедингов в файл кэша. Это позволяет избежать повторного сбора и обработки одних и тех же данных, что значительно сокращает время выполнения программы при повторных запусках. При наличии кэша программа переходит непосредственно к этапу кластеризации, что обеспечивает более быструю настройку и получение результатов.

3.7 Анализ и представление результатов

Анализ результатов кластеризации включает идентификацию наиболее популярных тем и трендов на основе сгруппированных эмбедингов. Результаты анализа представлены в виде отчётов, включающих статистику по кластерам, представительные темы и соответствующие посты. Это обеспечивает наглядное представление данных и облегчает их интерпретацию для пользователей. Генерация отчётов осуществляется автоматически, что упрощает процесс получения информации о текущих трендах в социальной сети ВКонтакте.

4 Описание программы

Программа для анализа трендов в социальной сети ВКонтакте состоит из нескольких модулей, реализующих сбор, обработку и анализ данных. Основные компоненты программы и их функции представлены ниже:

4.1 Основные модули и их функции

4.1.1 `main.py`

- **VKAnalyzerApp:** Класс, реализующий графический интерфейс пользователя (GUI) с использованием библиотеки Tkinter.
 - `create_input_frame()`: Создает секцию для ввода идентификаторов групп ВКонтакте и количества постов для анализа.
 - `create_settings_frame()`: Создает секцию для настройки параметров кластеризации, включая выбор метода кластеризации и ввод гиперпараметров.
 - `create_buttons_frame()`: Создает кнопки для получения VK API ключа, запуска анализа, загрузки кэша и сохранения результатов.
 - `create_result_frame()`: Создает область для отображения результатов анализа.
 - `get_vk_access_token_gui()`: Открывает браузер для авторизации ВКонтакте и получения `access_token`.
 - `load_cache()`: Загружает кэшированные данные из файла и заполняет соответствующие поля ввода.
 - `run_analysis()`: Собирает введенные пользователем данные, выполняет проверку корректности ввода и запускает процесс анализа.
 - `save_result()`: Сохраняет результаты анализа в текстовый файл.

- `update_hyperparameter_fields()`: Обновляет поля ввода гиперпараметров при изменении метода кластеризации.
- `create_hyperparameter_fields()`: Создает поля ввода для гиперпараметров выбранного метода кластеризации.

4.1.2 `vk_topics_analyzer_tools.py`

- **Tag**: Класс для представления темы (тега) с атрибутами текста, эмбединга, принадлежности к кластеру и расстояния до центра кластера.
- **MessengerPost**: Класс для представления поста с текстом, изображениями, выделенными тегами и оценками для кластеров.
- **save_cache(domains, count_posts)**: Сохраняет собранные данные и эмбединги в файл кэша (`vk_topics_analyzer_tools.json`).
- **load_cache()**: Загружает данные из файла кэша, если он существует и не пуст.
- **get_vk_wall_posts(vk_api_key, domain, page_number)**: Получает посты со стены указанной группы ВКонтакте.
- **process_vk_wall_posts(posts)**: Обрабатывает полученные посты, извлекая текст и изображения.
- **get_tags_list_for_post(messenger_post)**: Использует модель ChatGPT для выделения тем (тегов) из поста.
- **get_embedding(text)**: Получает эмбединг текста с использованием модели OpenAI.
- **post_handler(post)**: Обрабатывает отдельный пост, выделяет теги и получает их эмбединги.
- **parallel_hundred_posts_handler(vk_api_key, domain, page_number)**: Параллельно обрабатывает 100 постов из указанной группы.

- **synchronous_hundred_posts_handler(vk_api_key, domain, page_number):** Синхронно обрабатывает 100 постов из указанной группы.
- **parallel_groups_handler(vk_api_key, domains, count_posts, suppress_excessive_parallelization):** Параллельно обрабатывает посты из нескольких групп ВКонтакте.
- **cluster_tags(method, params, num_display_tags):** Выполняет кластеризацию эмбедингов тегов с использованием выбранного метода и гиперпараметров.
- **calculate_scores_for_posts(k1, k2):** Вычисляет оценки для каждого поста на основе кластеров тегов.
- **find_representative_posts(num_display_posts):** Находит представительные посты для каждого кластера.
- **get_vk_access_token():** Открывает браузер для авторизации ВКонтакте и получения access_token.
- **get_groups_id():** Получает от пользователя список идентификаторов групп ВКонтакте для анализа.
- **main():** Основная функция программы, управляющая процессом анализа данных.
- **run_analysis_with_gui(domains, count_posts, vk_api_key, clustering_method, clustering_params, num_display_tags, num_display_posts):** Функция, запускающая анализ с учетом кэширования и настроек гиперпараметров.

4.2 Алгоритм работы программы

Алгоритм работы программы представлен в следующих шагах:

1. Сбор данных:

- Пользователь вводит идентификаторы групп ВКонтакте и количество постов для анализа через GUI.
- Программа выполняет параллельный сбор постов (текста и изображений) из указанных групп с использованием VK API.

2. Выделение тем (тегов):

- Для каждого поста программа использует модель ChatGPT с заданным промптом (`prompt.txt`) для выделения тем.
- Извлеченные темы сохраняются как теги для дальнейшего анализа.

3. Преобразование тем в эмбединги:

- Каждая выделенная тема преобразуется в эмбединг с использованием модели OpenAI.
- Эмбединги сохраняются вместе с соответствующими тегами.

4. Кэширование данных:

- Все собранные данные, включая посты, теги и их эмбединги, сохраняются в файл кэша (`vk_topics_analyzer_tools.json`).
- При повторном запуске программы, если кэш существует, программа использует его для ускорения процесса.

5. Кластеризация эмбедингов:

- Программа выполняет кластеризацию эмбедингов тегов с использованием выбранного метода (например, HDBSCAN).
- Подбор оптимальных гиперпараметров осуществляется на основе существующего кэша данных.
- Результаты кластеризации сохраняются для дальнейшего анализа.

6. Анализ и представление результатов:

- Выявляются популярные темы и тренды на основе сгруппированных кластеров.

- Результаты отображаются в GUI и могут быть сохранены пользователем.

5 Рекомендации пользователю

5.1 Инструкция по установке программы

Для установки и настройки программы необходимо выполнить следующие шаги:

5.1.1 Шаг 1: Установка Python 3.9

1. Перейдите на официальный сайт Python: <https://www.python.org/downloads/release/python-390/>.
2. Скачайте установочный файл Python 3.9 для вашей операционной системы.
3. Запустите установочный файл и следуйте инструкциям. Во время установки обязательно поставьте галочку на опции "Add Python to PATH".
4. После установки откройте командную строку (Terminal) и проверьте установку:

```
python3.9 --version
```

Вы должны увидеть версию Python 3.9.

5.1.2 Шаг 2: Установка Git

1. Перейдите на официальный сайт Git: <https://git-scm.com/downloads>.
2. Скачайте установочный файл Git для вашей операционной системы.
3. Запустите установочный файл и следуйте инструкциям установки с настройками по умолчанию.
4. После установки откройте командную строку и проверьте установку:

```
git --version
```

Вы должны увидеть версию Git.

5.1.3 Шаг 3: Клонирование репозитория

1. Откройте командную строку.
2. Перейдите в директорию, куда хотите клонировать проект:

```
cd ~
```

3. Клонировать репозиторий проекта:

```
git clone https://github.com/MansurYa/VK-trend-analyzer.git
```

4. Перейдите в директорию проекта:

```
cd VK-trend-analyzer
```

5.1.4 Шаг 4: Создание виртуального окружения и установка зависимостей

1. Создайте виртуальное окружение:

```
python3.9 -m venv venv
```

2. Активируйте виртуальное окружение:

```
source venv/bin/activate
```

3. Установите необходимые пакеты из файла requirements.txt:

```
pip install --upgrade pip  
pip install -r requirements.txt
```

5.1.5 Шаг 5: Настройка конфигурации

1. Откройте файл config.json для редактирования:

```
nano config.json
```

2. Отредактируйте параметры согласно вашим требованиям. Пример конфигурации:

```
{  
    "db_path": "event_log.db",  
    "log_rotation_days": 7,  
    "monitor_paths": ["/var/log", "/etc"],  
    "email_notifications": true,  
    "email_recipients": ["example@gmail.com"],  
    "smtp_server": "smtp.yandex.ru",  
    "smtp_port": 587,  
    "smtp_user": "your_email@yandex.ru",  
    "smtp_password": "your_password"  
}
```

3. Сохраните изменения и закройте редактор (в nano: нажмите Ctrl + O, затем Enter, затем Ctrl + X).

5.2 Инструкция по эксплуатации программы

После установки и настройки программы выполните следующие шаги для её использования:

5.2.1 Шаг 1: Запуск программы

1. Откройте командную строку.
2. Перейдите в директорию проекта, если вы ещё не там:

```
cd ~/VK-trend-analyzer
```

3. Активируйте виртуальное окружение:

```
source venv/bin/activate
```

4. Запустите программу:

```
python main.py
```

5. Откроется окно графического интерфейса программы.

5.2.2 Шаг 2: Настройка параметров анализа

1. В разделе "Введите ID групп через запятую" введите идентификаторы групп ВКонтакте, из которых нужно собрать посты. Идентификаторы можно получить из URL групп (например, для группы <https://vk.com/overhearspbsu> идентификатором будет overhearspbsu).
2. В разделе "Введите количество постов (кратно 100)" укажите количество постов для анализа из каждой группы. Значение должно быть кратно 100.
3. В разделе "Настройки кластеризации" выберите метод кластеризации из выпадающего списка (например, HDBSCAN).
4. Введите необходимые гиперпараметры для выбранного метода кластеризации.
5. Укажите количество тегов и постов для отображения на каждом кластере.

5.2.3 Шаг 3: Получение VK API ключа

1. Нажмите кнопку "Получить VK API ключ".
2. В браузере откроется страница авторизации ВКонтакте. Войдите в свою учётную запись и предоставьте доступ приложению.
3. После авторизации скопируйте URL из адресной строки браузера и вставьте его в появившемся диалоговом окне программы.
4. Если ключ успешно получен, появится уведомление об успешной авторизации.

5.2.4 Шаг 4: Запуск анализа

1. После ввода всех необходимых параметров и получения VK API ключа нажмите кнопку "Запустить анализ".

2. Программа начнёт сбор данных, выделение тем, преобразование их в эмбединги и кластеризацию.
3. По завершении анализа результаты отобразятся в текстовом поле "Результат анализа".

5.2.5 Шаг 5: Загрузка кэша

1. Если вы ранее запускали программу и у вас есть сохранённый кэш, нажмите кнопку "Загрузить кэш".
2. Программа загрузит данные из кэша и заполнит соответствующие поля ввода, позволяя сразу перейти к этапу кластеризации.

5.2.6 Шаг 6: Сохранение результатов

1. После выполнения анализа и получения результатов нажмите кнопку "Сохранить результат".
2. Выберите место для сохранения файла и укажите имя файла.
3. Результаты будут сохранены в выбранном файле в текстовом формате.

5.2.7 Шаг 7: Остановка программы

1. Для остановки программы закройте окно графического интерфейса.
2. Также можно остановить программу, нажав `Ctrl + C` в терминале, где была запущена программа.

6 Контрольный пример

- **db_path**: Путь к базе данных SQLite для хранения событий.
 - Пример: `"db_path": "event_log.db"`
- **log_rotation_days**: Количество дней, по истечении которых старые записи будут удалены из базы данных.
 - Пример: `"log_rotation_days": 7`
- **monitor_paths**: Список директорий, которые необходимо мониторить на изменения файловой системы.
 - Пример: `"monitor_paths": ["/var/log", "/etc"]`
- **email_notifications**: Включение (true) или отключение (false) отправки email-уведомлений.
 - Пример: `"email_notifications": true`
- **email_recipients**: Список адресов электронной почты, на которые будут отправляться уведомления.
 - Пример: `"email_recipients": ["example@gmail.com"]`
- **smtp_server**: Адрес SMTP-сервера для отправки электронных писем.
 - Пример: `"smtp_server": "smtp.yandex.ru"`
- **smtp_port**: Порт SMTP-сервера.
 - Пример: `"smtp_port": 587`
- **smtp_user**: Логин (адрес электронной почты) для аутентификации на SMTP-сервере.
 - Пример: `"smtp_user": "your_email@yandex.ru"`
- **smtp_password**: Пароль для аутентификации на SMTP-сервере.
 - Пример: `"smtp_password": "your_password"`

6.0.1 Шаг 3: Использование графического интерфейса

После запуска программы откроется окно графического интерфейса с основными функциями:

- **Просмотр событий:**

- Таблица отображает последние зарегистрированные события, включая время, пользователя, PID, тип события и описание.

- **Фильтрация событий:**

- В верхней части окна находятся поля для фильтрации событий по пользователю, типу события и дате.
- Введите необходимые параметры и нажмите кнопку " " для отображения соответствующих событий.

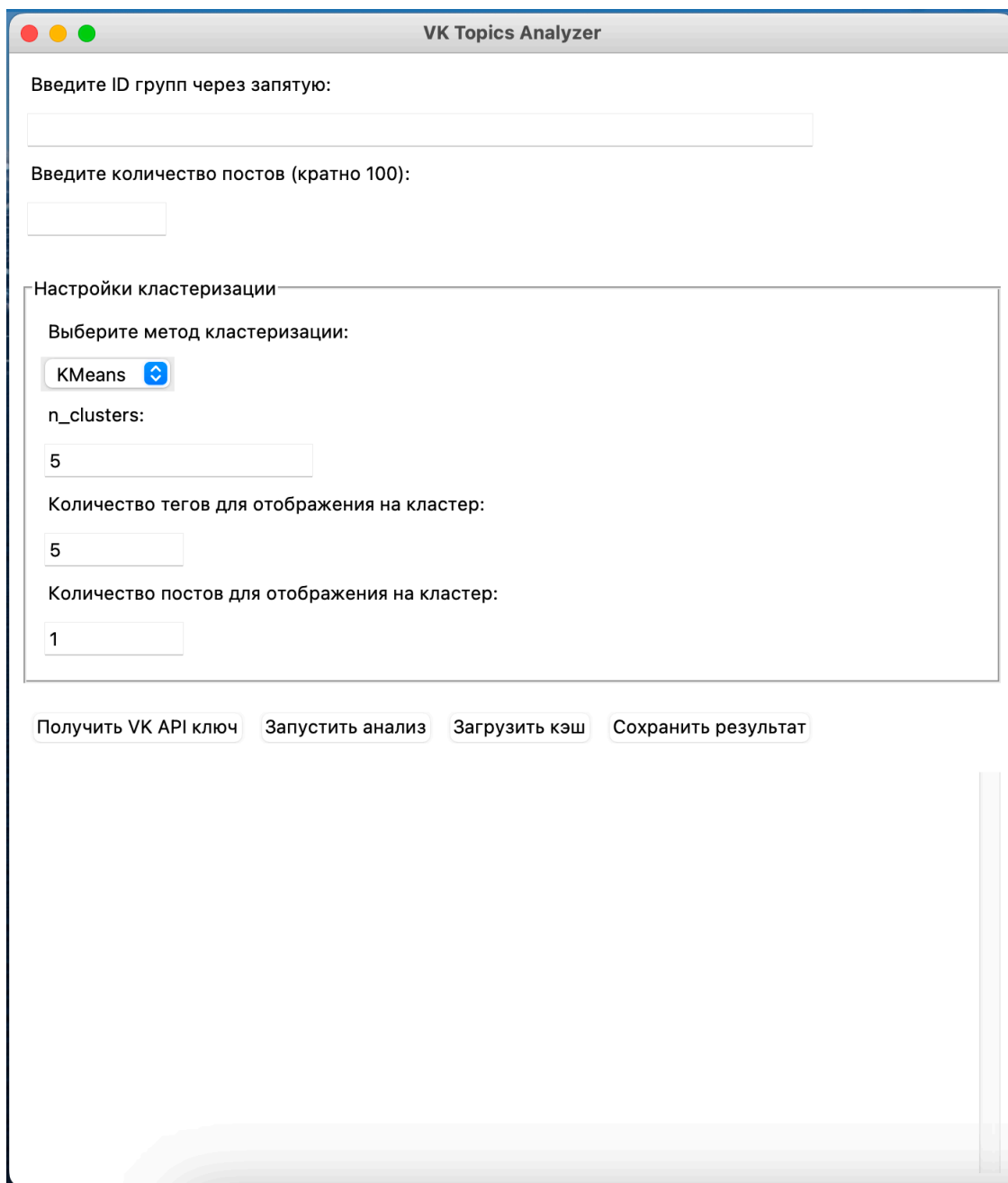
- **Обновление списка событий:**

- Нажмите кнопку "Обновить" для загрузки последних событий из базы данных.

- **Генерация отчётов:**

- Нажмите кнопку "Генерировать отчет" для создания статистического отчёта. Откроется новое окно с графиком, отображающим количество событий по типам.

7 Контрольный пример



The image shows a web application window titled "VK Topics Analyzer". It contains several input fields and a settings section. At the top, there are two input fields: one for "Введите ID групп через запятую:" and another for "Введите количество постов (кратно 100):". Below these is a section titled "Настройки кластеризации" (Clustering Settings). Inside this section, there is a label "Выберите метод кластеризации:" followed by a dropdown menu showing "KMeans". Below the dropdown are three more input fields: "n_clusters:" with the value "5", "Количество тегов для отображения на кластер:" with the value "5", and "Количество постов для отображения на кластер:" with the value "1". At the bottom of the window, there are four buttons: "Получить VK API ключ", "Запустить анализ", "Загрузить кэш", and "Сохранить результат".

VK Topics Analyzer

Введите ID групп через запятую:

Введите количество постов (кратно 100):

Настройки кластеризации

Выберите метод кластеризации:

KMeans

n_clusters:

5

Количество тегов для отображения на кластер:

5

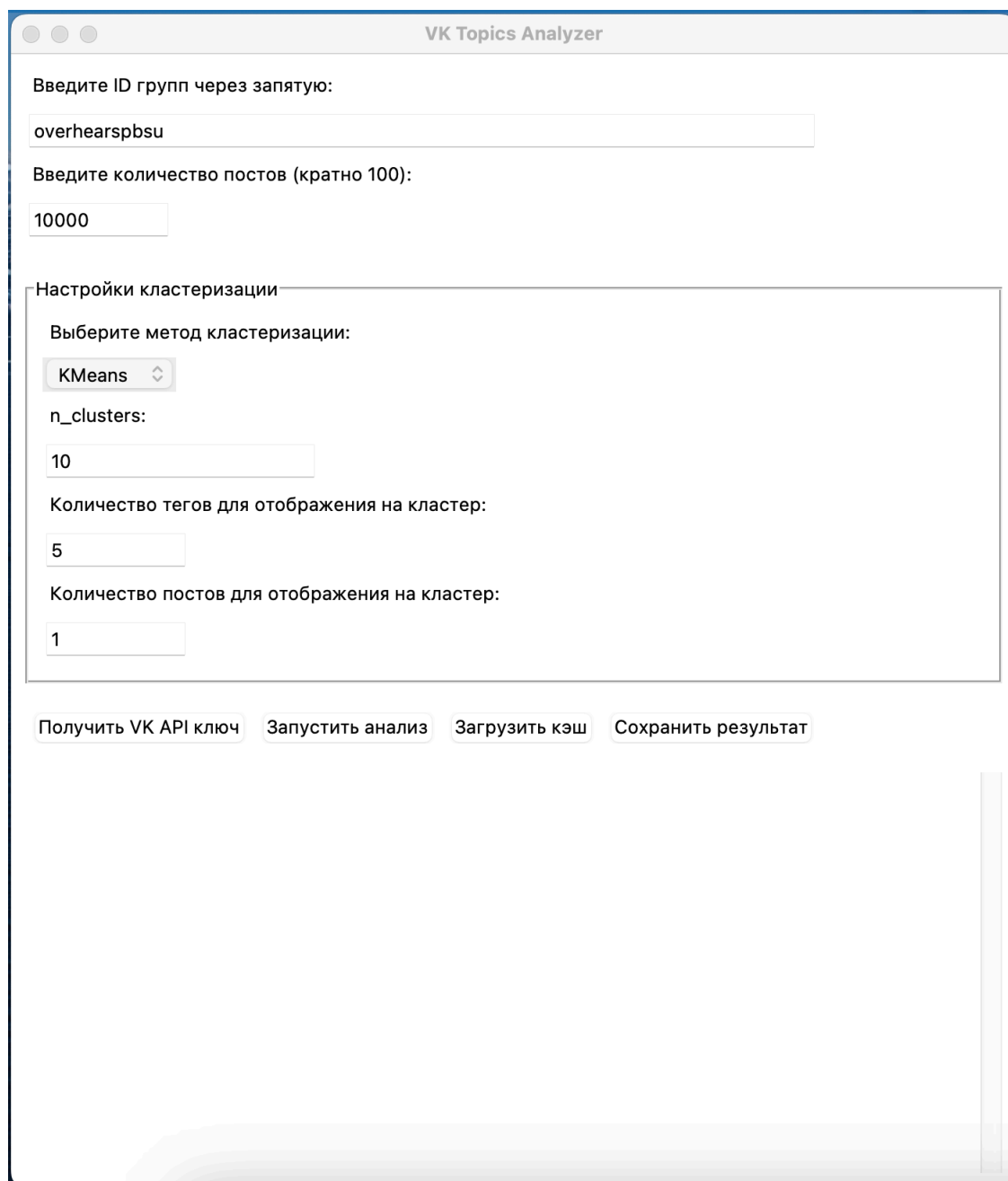
Количество постов для отображения на кластер:

1

Получить VK API ключ Запустить анализ Загрузить кэш Сохранить результат

рис. 1 Интерфейс программы

ID группы можно получить из ссылки на группу. Например, из "https://vk.com/overhearspbsu" ID="overhearspbsu".



The image shows a web application titled "VK Topics Analyzer". It has a light gray header with the title. Below the header, there are several input fields and a settings section. The first input field is labeled "Введите ID групп через запятую:" and contains the text "overhearspbsu". The second input field is labeled "Введите количество постов (кратно 100):" and contains the text "10000". Below these is a section titled "Настройки кластеризации" (Clustering Settings). Inside this section, there is a label "Выберите метод кластеризации:" followed by a dropdown menu showing "KMeans". Below that is a label "n_clusters:" followed by an input field containing "10". Then, there is a label "Количество тегов для отображения на кластер:" followed by an input field containing "5". Finally, there is a label "Количество постов для отображения на кластер:" followed by an input field containing "1". At the bottom of the form, there are four buttons: "Получить VK API ключ", "Запустить анализ", "Загрузить кэш", and "Сохранить результат".

рис. 2 Заполнили поля

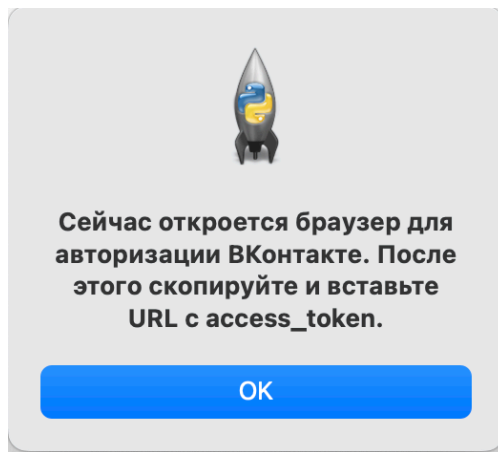


рис. 3 Нажали кнопку "Получить VK API ключ"

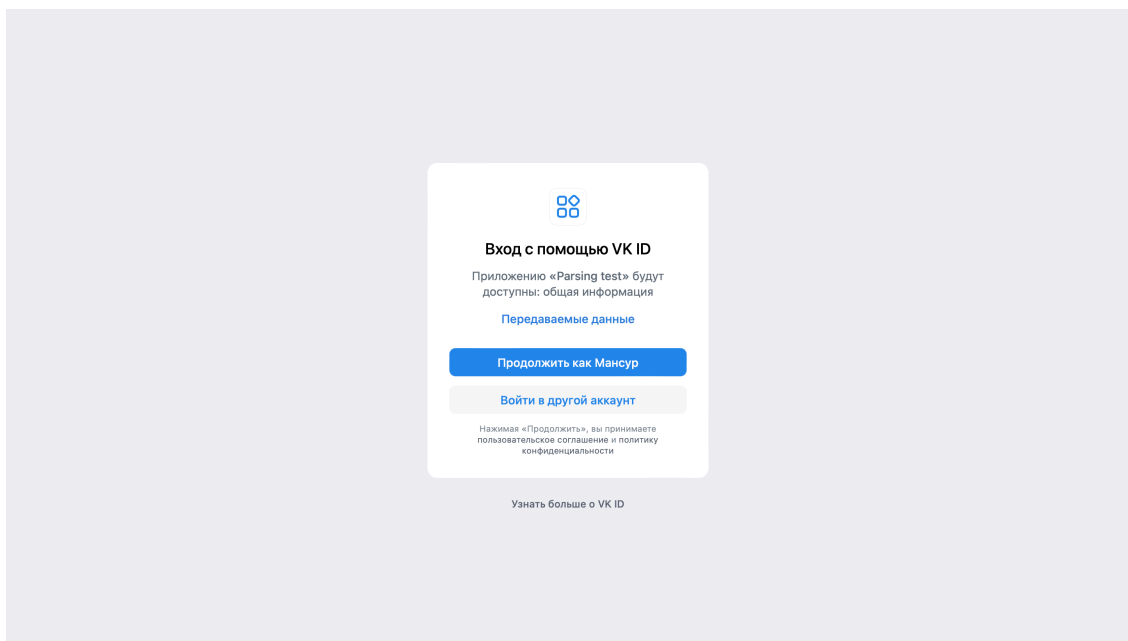


рис. 4 Проходим авторизацию VK

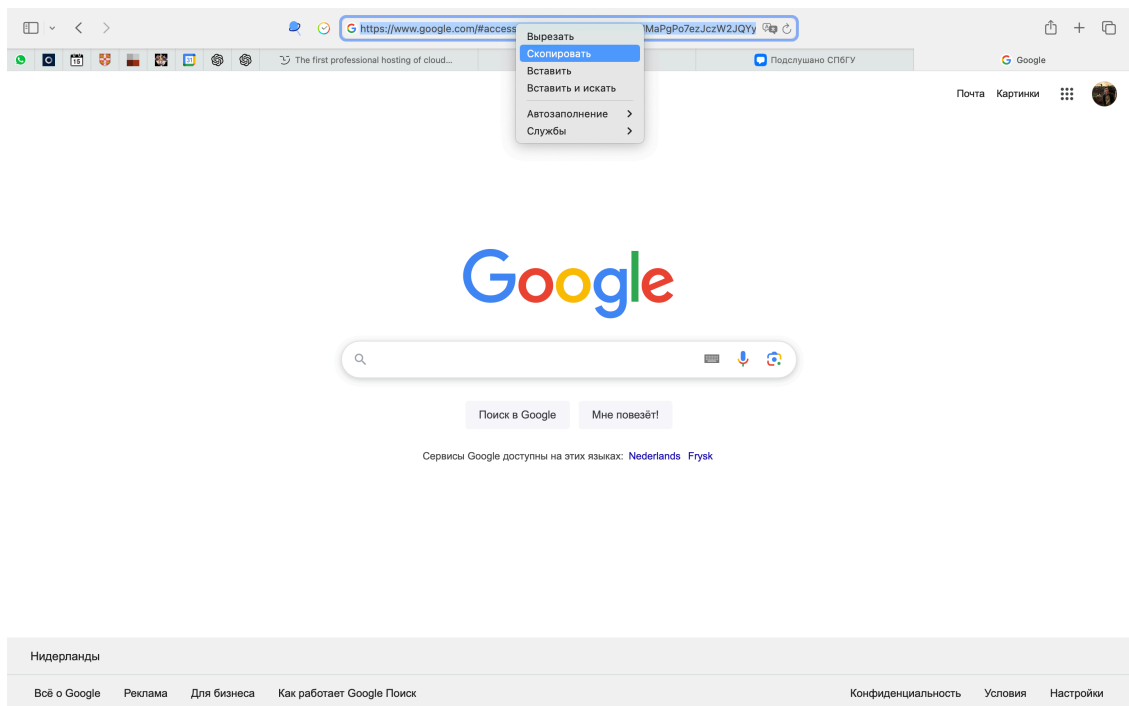


рис. 5 Копируем ключ

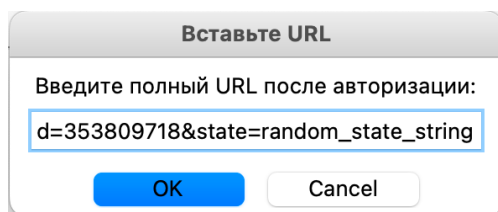


рис. 6 Вставляем ключ

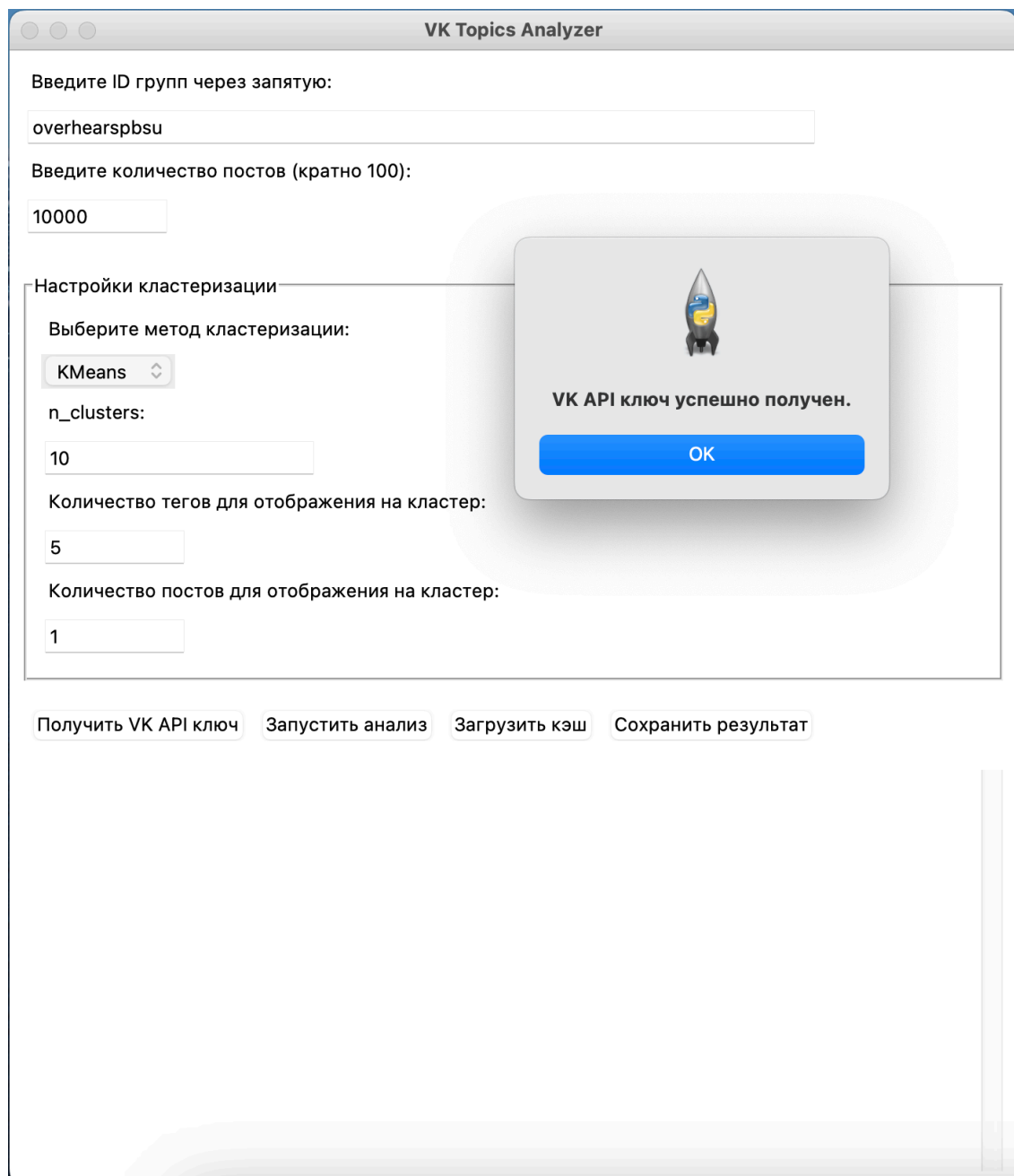


рис. 7 VK API ключ успешно получен

Получить VK API ключ

Запустить анализ

Загрузить кэш

Сохранить результат

```
0.6675663444223681
- Тег: Обучение на факультете межкультурной коммуникации и перевода, Расстояние до центра
кластера: 0.6682416773254357
- Тег: Обучение на физическом факультете, Расстояние до центра кластера: 0.6687487579146207
- Тег: Обучение на психфаке, Расстояние до центра кластера: 0.6691627443897253

Представительные посты этого кластера:
Пост с score (1.5934774717387912):
Текст поста:
...
Как на факультете филологии? Есть зарубежное направление?

Агон
...
Пост с score (1.5828187917074086):
Текст поста:
...
Приветик, расскажите пожалуйста про факультет ЖУРНАЛИСТИКИ.
Что там, как там.
Нравится ли.
Как обучение проходит.
```

рис. 8 Запускаем анализ и получаем через какое-то время результат

8 Выводы по работе

Была разработана программа для сбора, обработки и анализа постов из ВКонтакте. Программа выполняет параллельный сбор данных, выделение тем с помощью ChatGPT, преобразование тем в эмбединги и их кластеризацию. Реализовано кэширование данных для повышения эффективности. Программа позволяет выявлять популярные темы и тренды, обеспечивая удобный интерфейс для настройки и анализа результатов.

9 Полезные ссылки

Ссылка на репозиторий проекта: <https://github.com/MansurYa/VK-trend-analyzer.git>