АВТОМАТИЗИРОВАННОЕ ИЗВЛЕЧЕНИЕ ДАННЫХ ПРОГРАММНЫХ ПРОЕКТОВ ДЛЯ ИХ ОЦЕНКИ

Телеличко Н. Н.[1]
Научный руководитель: Савельев А. О.[2]
[1]*Томский политехнический университет, ИШИТР, гр. 8ПМ41, e-mail: nnt13@tpu.ru*
[2]*Томский политехнический университет, ИШИТР, к.т.н., доцент ОИТ, e-mail: sava@tpu.ru*

Аннотация

Представлен конвейер автоматизированного сбора данных программных проектов для выявления особенностей мышления разработчиков. Сбор осуществлялся через GitHub API, загрузка – в БД Django. В конвейере преодолены ограничения на загрузку и вывод через GitHub API, а также реализовано сохранение большого объема информации с сохранением связи файлов с их репозиториями.

Ключевые слова: GitHub API, мышление разработчиков, конвейер автоматизированного сбора данных, программные проекты.

Введение

Одним из наиболее популярных ресурсов, на котором располагаются различные программные проекты, является платформа GitHub.

Информация о сборе данных с этой платформы упоминается в ряде статей, которые можно условно сгруппировать на 3 основные группы в зависимости от целей:

1. Определение характеристик разработчиков. В таких статьях оцениваются эмоциональные состояния разработчиков [1, 2], взаимоотношения специалистов [3, 4], лидерские качества [5], или выявляется гендер [6].

2. Выявление параметров проектов. Может применяться как для оценки самих проектов (на популярность [7], анализ событий [8]), так и для их классификации [9], использовании в рекомендательных системах [10].

3. Реализация анализа кода. Осуществляется для выявления закономерностей процесса написания кода [11] и оценки его качества [12, 13].

Часто в исследованиях оцениваются тексты pull request-ов, issues или информация о репозиториях, получаемая с помощью API запросов. В ряде случаев могут проверяться файлы с описаниями проектов [10, 7]. Кроме того, некоторые возможности оценки [14] и сбора статистики [15, 16] по проектам в GitHub-репозиториях реализованы в патентах. А также есть, как минимум, 38 репозиториев с открытым кодом, позволяющих производить анализ репозиториев [17].

Исследования, касающиеся оценки самого кода, проводились Н.А. Махачевым и Н.А. Нажимовой (и А.В. Нажимовым) [13], S. Brooke [6], A. Bernstein (и др.) [11]. В первых двух работах проверка кода осуществлялась в рамках репозиториев преподавателей, т.е., по факту, в этих работах не анализировался сбор данных с большого количества репозиториев. Работа A. Bernstein (и др.) включила в себя сбор данных с 470 репозиториев с Jupyter блокнотами, их маркировку по типу выполняемых действий, связанных с Data Science (предобработка, моделирование и т. д.) выявление паттернов и прогнозирование результатов по ячейкам блокнотов. Исследования S. Brooke оказались наиболее близкими к нашим, в части сбора данных. Работа S. Brooke включала в себя следующие этапы: 1) сбор данных с помощью GitHub API (была собрана информация из 1728 репозиториев); 2) сохранение кода из файлов с расширением «.py» и без «__» в названиях; 3) срабатывание модуля по определению гендера на именах пользователей и почтовых ящиках; 4) оценка файлов с помощью линтеров и 5) проведение исследования на основании данных с линтера и модуля по оценке гендера. При этом сам проект ученого имеет 5 файлов с кодом (по 1 на задачу и дополнительный –

с методами оценки файла линтером и включения ожидания при сборе данных с GitHub), а данные сохраняются и в дальнейшем применяются из «.csv» файлов.

Несмотря на общий прогресс в извлечении данных, результаты их анализа не используются для психологической оценки в целом и мышления в частности, что и обуславливает актуальность и цель данного исследования.

Целью исследования является разработка программного обеспечения, реализующего оценку мышления разработчиков по их коду. Понимание особенностей мышления специалистов может помочь им выстроить более эффективную траекторию развития, а компаниям – подобрать более опытных и совместимых с командой профессионалов.

В рамках данной работы будет описан процесс создания ПО, относящийся к автоматизированному сбору данных с GitHub, на основе которых и будет проводится оценка мышления в дальнейшем исследовании.

Сбор данных (скейпинг)

Сбор данных можно реализовать через интерфейс веб-ресурса или с помощью API-запросов. В первом случае могут применяться библиотеки Selenium и BeautifulSoup, во втором – необходимо знать структуру запросов (URL, доступные методы, передаваемые параметры, получаемые ответы), которая находит отражение в документации к ресурсу.

Скрейпинг в обоих случаях имеет свои недостатки и ограничения. Сбор данных через интерфейс выполняется медленнее и, в случае сбора информации о репозиториях GitHub, необходимо предусмотреть 3 отдельных способа скрепинга: по репозиториям, пользователям и pull request-ам (для сбора данных о датах создания и обновления репозиториев). Кроме того, возможна блокировка IP машины, с которой осуществляется сбор данных, при большом количестве запросов к страницам сайта. При использовании API также возможны блокировки, но в случае GitHub API вместо блокировки применяются ограничения по количеству запросов и выводимой информации. GitHub допускает до 900 запросов в минуту [18], при превышении которого появляется ошибка 403 Forbidden с текстом «You have exceeded a secondary rate limit…», также возможна ошибка 422 Unprocessable Entity, которая, в частности, может возникать за счет ограничения количества выводимых репозиториев в 1000 единиц (текст ошибки: «Only the first 1000 search results are available»). Однако есть и преимущества: при сборе данных через API не требуется создавать 3 отдельных метода – одним запросом собирается информация со 103 признаками, характеризующими репозитории, пользователей, а также содержащими даты.

Конвейер сбора данных с GitHub

В целях исследования особенностей мышления разработчиков необходимо было реализовать конвейер сбора данных (кода и информации о репозиториях и авторах) с GitHub. Основные этапы разработки конвейера представлены на рисунке 1.

Процесс разработки сопровождался некоторыми сложностями. Первая проблема была связана с описанными выше ограничениями по количеству запросов и выводимой информацией при сборе данных через GitHub API. Проблема была преодолена при выполнении Процесса 1 за счет добавления ожиданий при возникновении ошибок 403 и 422, а также уменьшения количества выводимых за раз данных за счет конкретизации фильтра с помощью дат. Даты вводились в фильтр с помощью параметра «oauth created», который со значениями имеет следующий вид:

```
q=oauth%20created:{year_num}-{month_num}-01..{year_num}-{month_num}-{month_days_count}
```

При таком подходе итоговый URL запрос за март 2025 будет представлен в виде:

```
https://api.github.com/search/repositories?q=language:python&sort=stars&order=abs&per_page=100q=oauth%20created:2025-03-01..2025-03-31&page=1
```

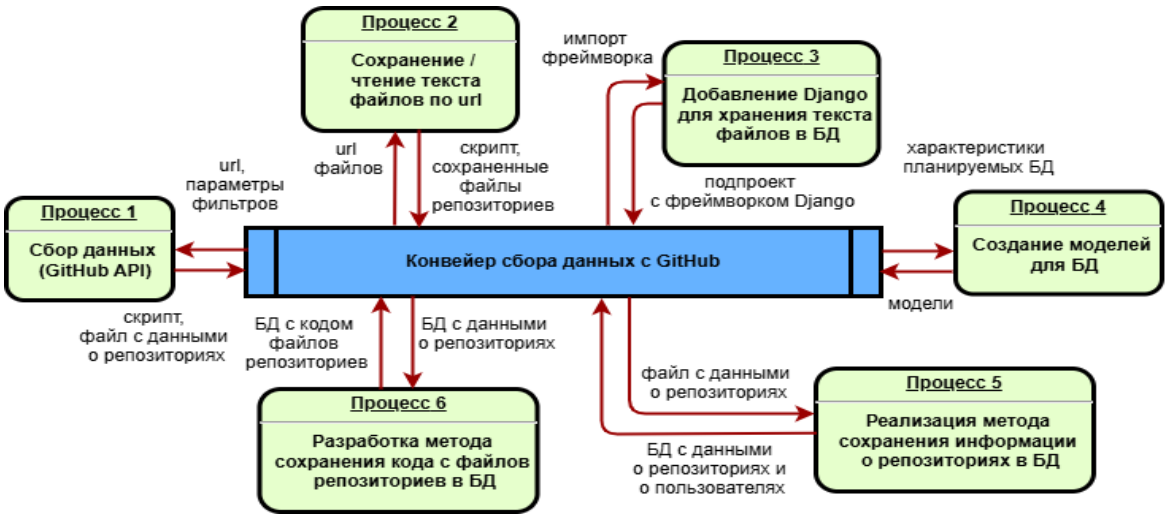

Рис. 1. Конвейер сбора данных с GitHub

Следующей задачей (Процесс 2) стало сохранение файлов с кодом проектов или их содержимого. Код (рисунок 2) включил в себя, как и в предыдущем процессе, использование запросов, которые отличались передаваемыми URL и отсутствием явных ограничений.

```
def github_read_file_content(self, username, repository_name, file_name, github_token=None):
    headers = {}
    if github_token:
        headers['Authorization'] = f"token {github_token}"

    url = f'https://api.github.com/repos/{username}/{repository_name}/contents/{file_name}'

    r = requests.get(url, headers=headers)
    r.raise_for_status()
    data = r.json()
    file_content = data['content']
    file_content_encoding = data.get('encoding')
    if file_content_encoding == 'base64':
        file_content = base64.b64decode(file_content).decode()

    return file_content
```

Рис. 2. Метод для чтения файлов с кодом

Необходимость сохранения большого количества файлов с кодом потребовала применения базы данных, которая была включена в проект вместе с фреймворком Django (Процесс 3). Данный фреймворк позволит также разработать интерфейс создаваемого ПО. Элементы фреймворка были включены в отдельный подпроект. Применяемая во фреймворке база данных потребовала реализации класса с моделями (Процесс 4), поля которых определялись полученной информацией о репозиториях и файлах. По результатам Процессов 1-4 были реализованы базы данных со структурой и взаимосвязями, представленными на рисунке 3. При этом данные, полученные с помощью GitHub API, были частично отфильтрованы. а признаки переименованы.

Последние 2 процесса (Процесс 5 и Процесс 6) включили в себя разработку методов по добавлению данных в БД.

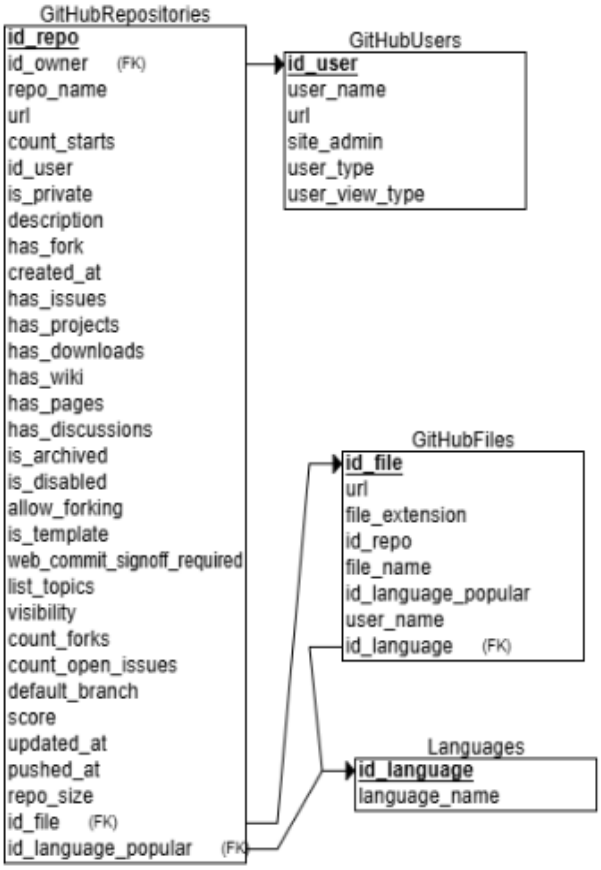

Рис. 3. ER-диаграмма со взаимосвязью GitHub-сущностей

Текущие ограничения конвейера

На данный момент конвейер имеет ряд ограничений:

- сохранение данных о репозиториях в БД реализуется не напрямую из запросов, а через промежуточный файл.
- файлы сохраняются долго, так что требуется добавление многопоточности.
- URL «вшиты» в методы, требуется доработка интерфейса с выводом на него различных параметров, в том числе по URL.

Заключение

Были исследованы возможности сбора данных о программных проектах, представленных в GitHub, и разработан конвейер, реализующий этот сбор. Конвейер учитывает имеющиеся ограничения в API-запросах и сохраняет информацию о репозиториях, пользователях и коде в базы данных.

Подобный конвейер будет использоваться для оценки мышления разработчиков, но также может быть применим для ряда других задач, например, определения различных характеристик программных проектов или их авторов, в том числе с применением методов ИИ.

Список использованных источников

1. Lu, X. Emojis predict dropouts of remote workers: an empirical study of emoji usage on GitHub / X. Lu, W. Ai, Z. Chen, Y. Cao, Q. Mei. // Plos One. – 2022. – Volume 17. – № 1. – P. 1-21. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=54980520 (date of the application: 23.03.2025).

2. Sinha, S. F. Are You Listening?': Sentiment Diffusion in GitHub / S. F. Sinha // Academy of Management Proceedings. – 2022. – Vol. 2022, No. 1. – DOI 10.5465/ambpp.2022.10036abstract. (дата обращения: 23.03.2025).

3. Панов К.А. Методика наблюдения за взаимодействием разработчиков открытого программного обеспечения / К. А. Панов, Ф. Н. Винокуров. // Вестник РГГУ. Серия: Психология. Педагогика. Образование. – 2024. – № 2. – С. 117-130. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=67318762 (дата обращения: 28.03.2025).

4. Shameer SH. Relationship between diversity of collaborative group members' race and ethnicity and the frequency of their collaborative contributions in GitHub / S. Shameer, G. Rodríguez-Pérez, M. Nagappan. // Empirical Software Engineering. – 2023. – Volume 28. – № 4. – P. 1-14. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=63073773 (дата обращения: 28.03.2025).

5. Haim Faridian P. Leading open innovation: the role of strategic entrepreneurial leadership in orchestration of value creation and capture in github open source communities / P. Haim Faridian. // Technovation. – 2023. – Volume 119. – P. 1-41. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=60190756 (дата обращения: 28.03.2025).

6. Brooke S. Programmed differently? Testing for gender differences in python programming style and quality on GitHub / S. Brooke. // Journal of computer-mediated communication. – 2023. – Volume 29. – № 1. – P. 1-13. – [Электронный ресурс]. – URL: elibrary.ru/item.asp? id=64802525 (дата обращения: 28.03.2025).

7. Wang T. Study the correlation between the readme file of GitHub projects and their popularity / T. Wang, Sh. Wang, Ts. H. P. Chen. // Social Science Research Network. – 2022. – № 4281782 – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=57754968 (дата обращения: 28.03.2025).

8. Воинов Н.В. Система обработки больших данных для анализа событий репозитория GitHub / Н.В. Воинов, К. Родригес Гарсон, И.В. Никифоров, П.Д. Дробинцев. // Международная конференция по мягким вычислениям и измерениям. – 2019. – Том 1. – С. 283-286. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=38309155 (дата обращения: 28.03.2025).

9. Sas C. Gitranking: A ranking of GitHub topics for software classification using active sampling / C. Sas, A. Capiluppi, C. Di Sipio, Ju. Di Rocco, D._Di_Ruscio // Social Science Research Network. – 2022. – № 4182105 – [Электронный ресурс]. – URL: elibrary.ru/item.asp? id=56612993 (дата обращения: 28.03.2025).

10. Sun X. Personalized project recommendation on GitHub / X. Sun, W. Xu, X. Xia, X. Chen, B. Li. // Science China Information Science. – 2018. – Volume 61. – № 5. – P. 1-14. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=52951341 (дата обращения: 23.03.2025).

11. Bernstein A. Workflow analysis of data science code in public GitHub repositories / A. Bernstein, D. Ramasamy, C. Sarasua, A. Bacchelli. // Empirical Software Engineering. – 2023. – Volume 28. – № 1. – P. 1-47. – [Электронный ресурс]. – URL: elibrary.ru/item.asp? id=59826226 (дата обращения: 28.03.2025).

12. Махаличев Н.А. Автоматизированная проверка лабораторных работ студентов с использованием GitHub / Н.А. Махаличев. // Наука настоящего и будущего. – 2023. – Том 3. – С. 220-224. – [Электронный ресурс]. – URL: elibrary.ru/item.asp?id=54614166 (дата обращения: 23.03.2025).

13. Нажимова Н.А. Автоматизированная система проверки кода для подготовки специалистов в сфере информационных технологий. / Н. А. Нажимова, А. В. Нажимов. // Современные наукоемкие технологии. – 2024. – № 6. – С. 37-42. – [Электронный ресурс]. – URL: elibrary.ru/ item.asp?id=67917898 (дата обращения: 28.03.2025).

14. Пат. 2023663351 Российская Федерация, Программа для автоматизации проверки лабораторных работ студентов в репозиториях GitHub. / Махаличев Н. А., Шевская Н. В.; заявитель и патентообладатель Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский университет «ЛЭТИ» им. В.И. Ульянова (Ленина)». – опубл. 22.06.2023, Бюл. № 7. – 1 с.

15. Пат. 2019614382 Российская Федерация, Программа для хранения, обработки и обновления статистики репозиториев GitHub. / Борисовский Д.Ю., Мелихова П.А., Бачинский М.О., Заславский М.М.; Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)» (СПбГЭТУ «ЛЭТИ») (RU). – опубл. 03.04.2019, Бюл. № 4 – 1 с.

16. Пат. 2018617454 Российская Федерация, Построение показателей использования языков программирования пользователями сервиса github. / Курилович М.Е.; заявитель и патентообладатель Курилович Марат Евгениевич. – опубл. 25.06.2018, Бюл. № 7. – 1 с.

17. GH Archive / I. Grigorik // gharchive.org. – [Электронный ресурс]. – URL: gharchive.org/ (дата обращения: 28.03.2025).

18. Rate limits for the REST API // GitHub Docs. – 2022. – [Электронный ресурс]. – URL: docs.github.com/en/rest/using-the-rest-api/rate-limits-for-the-rest-api?apiVersion=2022-11-28 (дата обращения: 28.03.2025).