Параценко В.А., аспірант
Берест О.Б., к.т.н.
Сумський державний університет

АВТОМАТИЗОВАНА СИСТЕМА ВЕРИФІКАЦІЇ ЯКОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Забезпечення якості програмного коду є комплексною задачею, що вимагає застосування цілого спектру заходів. Створення універсального підходу є неможливим через динамічний характер розробки програмного забезпечення та суб'єктивність критеріїв якості. Для ефективного управління якістю коду необхідно поєднання автоматизованих інструментів та ручного аналізу. В цілях автоматизації процесу контролю доцільно розглянути впровадження систем підтримки прийняття рішень, щодо якості програмного коду.

В поточному дослідженні запропонована система, в основу якої покладена текстова модель [1], побудована на основі архітектури трансформерів[2], яка покликана представити вхідний програмний код у вигляді вектору ознак. Отриманий вектор ознак використовується в моделі кластеризації на основі архітектури Affinity Propagation, яка може виділяти різні компоненти програмного коду. Отримані результати застосовуються для побудови вирішальних правил в системі підтримки прийняття рішень щодо оцінки якості програмного коду.

Для апробації отриманих результатів запропонована модель системи була протестована на відкритому репозиторії програмного коду[3], що містить веб-додаток на мові Java з використання фреймворку Spring. Система спроможна знаходити програмні компоненти, з яких складається веб-додаток. В режимі ручного аналізу було визначено п’ять базових компонент додатку, а саме: контролери, репозиторії, класи-сутності, сервіси і класи конфігурацій. Отримані результати розпізнавання представлені у таблиці 1.

Таблиця 1- результати розпізнавання

Назва компонента	**Кількість, визначена вручну**	**Кількість визначена автоматично**	**Відсоток розпізнавання, %**
Контролери	6	3	50%
Репозиторії	2	2	100%
Класи-сутності	9	5	55%
Сервіси	2	1	50%
Класи конфігурацій	4	3	67%
		Середнє значення:	64%

Поточна реалізація запропонованої системи змогла розпізнати компоненти з точністю в 64%. За результатами кластерізації можуть бути побудовані вирішальні правила, які зручно використовувати в якості джерела інформації щодо прийняття рішень про можливість внесення змін в кодову базу. Наприклад, нові контролери за структурою повинні бути схожі на існуючі, тобто вектори ознак двох таких компонентів повинні мати мінімальну відстань між собою. Контроль якості проведення рефакторингу існуючої кодової бази може також бути спрощений завдяки впровадженню запропонованої технології.

Для покращення якості роботи представленої системи необхідне додаткове донавчання моделі на значній кількості відібраних програмних репозиторіїв для покращення якості векторів ознак та оптимізація моделі кластеризації. Також необхідна база даних типових якісних компонентів програмного коду, які попередньо проаналізовані досвідченими розробниками, які будуть використані в якості еталонних векторів реалізацій.

Список використаних джерел

1. M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation / J. Chen та ін. Findings of the Association for Computational Linguistics ACL 2024, м. Bangkok, Thailand and virtual meeting. Stroudsburg, PA, USA, 2024. С. 2318–2335. DOI: https://doi.org/10.18653/v1/2024.findings-acl.137.

2. Transformers: State-of-the-Art Natural Language Processing / T. Wolf та ін. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, м. Online. Stroudsburg, PA, USA, 2020. DOI: https://doi.org/10.18653/v1/2020.emnlp-demos.6.

3. GitHub - spring-projects/spring-petclinic: A sample Spring-based application. GitHub. URL: https://github.com/spring-projects/spring-petclinic (дата звернення: 17.11.2024).