Фильтрация поисковой выдачи дубликатов кода в IntelliJ IDEA Ultimate

Шамрай М. Б.
группа 343
Научный руководитель: к. т. н. Брыксин Т. А.

Мотивация

- Дубликаты кода — фрагменты кода, которые частично или полностью повторяют друг друга или представляют одну функциональность.
- От клонов кода можно и нужно избавляться — уменьшает сложность кода, издержки на сопровождение и т.д.
- Инструмент, позволяющий эффективно искать дубликаты кода в объемной кодовой базе, а после от них избавляться, был бы полезен для крупных проектов.

IntelliJ IDEA Ultimate

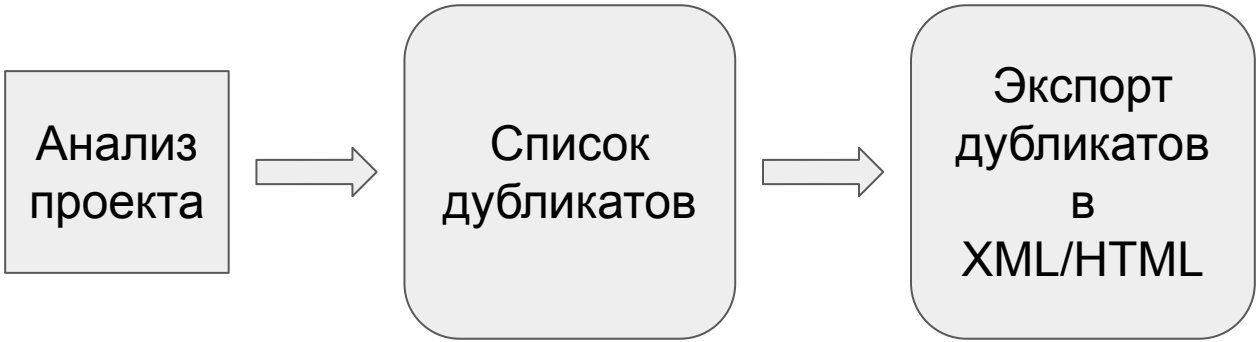

Рис. 1: Функциональные возможности IntelliJ IDEA Ultimate по детектированию дубликатов кода

Мотивация

- Было бы здорово заранее знать, какие из дубликатов можно выделить в метод, и отсортировать их по этому признаку, показывая первыми пользователю.
- Определение, выделяются ли дубликаты в метод непосредственно с помощью механизма выделения, слишком долго.

Постановка задачи

Цель:

Разработать инструмент фильтрации клонов кода в Intellij IDEA Ultimate по возможности выделения их в метод.

Задачи:

- Реализовать анализатор XML-файла
- Векторизовать дубликаты кода
- Обучить модели бинарной классификации на векторизованном коде
- Проанализировать результаты

Анализ XML файла

```
<duplicate cost=''26'' hash=''0''>
    <fragment
        file=''file://$PROJECT_DIR$/src/java/org/
            lwjgl/test/opengles/MappedIndexedVBOTest.java''
        line=''238'' start=''7532'' end=''7789''/>
    <fragment
        file=''file://$PROJECT_DIR$/src/java/org/
            lwjgl/test/opengles/FullScreenWindowedTest.java''
        line=''248'' start=''6723'' end=''6984''/>
</duplicate>
```

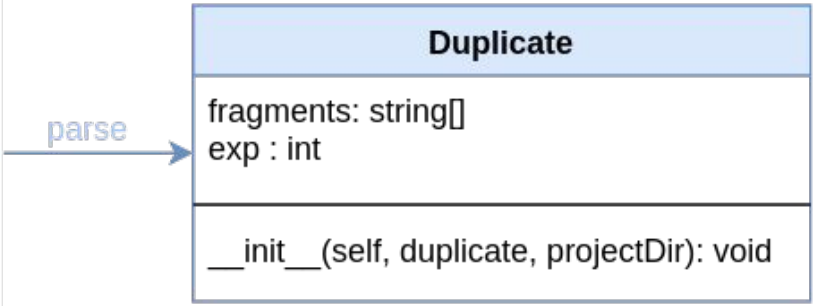

Рис. 2: Схема анализа XML файла дубликатов, fragments — дублирующиеся фрагменты кода, exp — флаг выделяемости в метод

Векторизация

Векторизация кода – это важный этап, который позволяет нам перейти от кода к методам машинного обучения.

Способы векторизации, использованные в работе:

- code2vec*
- Bag-of-words
- TF-IDF

*code2vec: Learning distributed representations of code. Uri Alon, Meital Zilberstein, Omer Levy, Eran Yahav, in POPL'2019

code2vec

- “Мешок” путей AST
- Готовая реализация
- Модель натренирована на 16M примерах

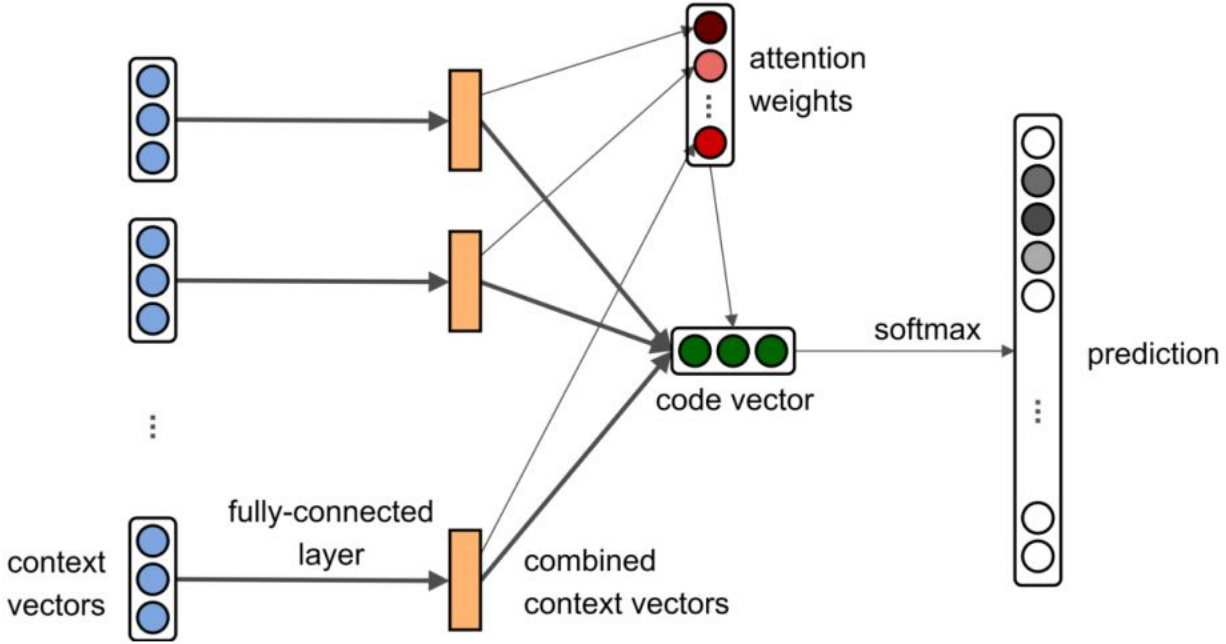

Рис. 3: Архитектура code2vec

Bag-of-words

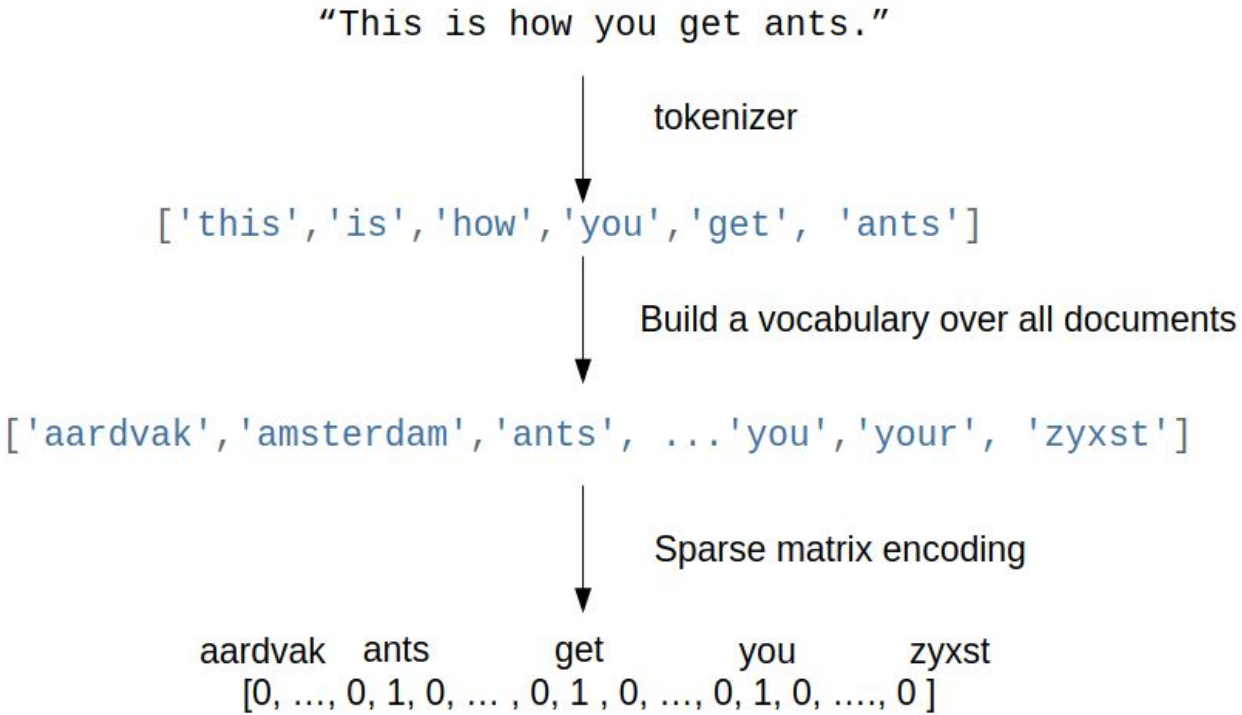

Рис. 4: Принцип векторизации с помощью bag-of-words

TF-IDF

- TF (Term Frequency) – отношение числа вхождений некоторого токена к общему числу токенов документа (1)
- IDF (Inverse Document Frequency) – инверсия частоты, с которой некоторый токен встречается во всех документах коллекции (2)
- Вес токена в документе вычисляется как произведение TF и IDF (3)

$$(1)\quad tf(t,d) = \frac{n_t}{\sum_k n_k}$$

$$(2)\quad idf(t,D) = log\frac{|D|}{|\{d_i \in D : t \in d_i\}|}$$

$$(3)\quad tfidf(t,d,D) = tf(t,d)idf(t,D)$$

Бинарная классификация

- hyperopt-sklearn — библиотека, предоставляющая удобный интерфейс для обучения и валидации моделей классификации.
- fastText* — фреймворк, написанный в Facebook Research, для быстрой классификации текстов.

*Bag of Tricks for Efficient Text Classification. Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, arXiv 2016

Валидационная функция потерь

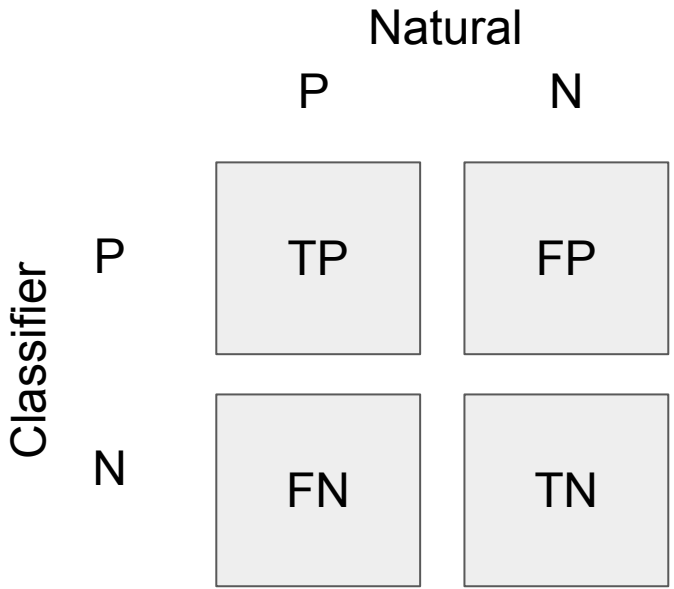

Рис. 5: Confusion matrix

$$FPR = \frac{FP}{n}$$

$$FNR = \frac{FN}{n}$$

$$L = FNR + 10 \times FPR$$

Набор данных

- 6 Java проектов (20969 групп дубликатов)
- Random Undersampling

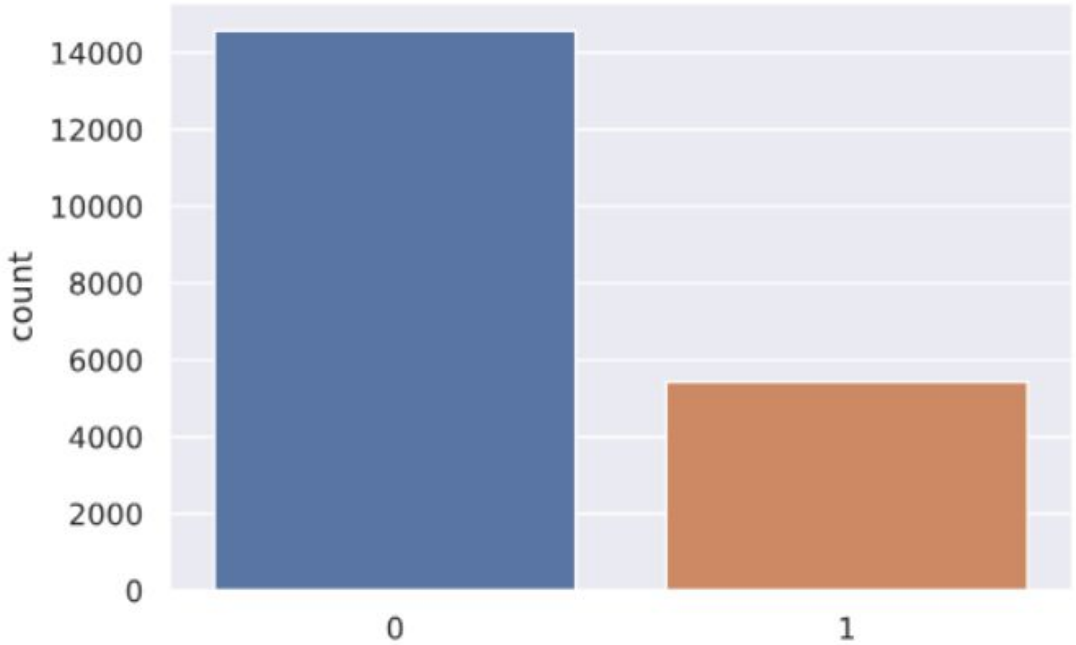

Рис. 6: Распределение данных по возможности выделения, 1 – можно выделить в метод, 0 – нельзя

Оценка результатов (1)

	FNR	FPR	accuracy	precision	recall
TF-IDF	0.19	0.1	0.71	0.76	0.62
Bag-of-words	0.13	0.14	0.73	0.73	0.75
code2vec	0.07	0.2	0.73	0.69	0.85
fastText	0.11	0.14	0.76	0.75	0.79

Таблица 1: Результаты экспериментов на различных метриках с threshold = 0.5

Оценка результатов (2)

	TF-IDF	Bag-of-words	code2vec	fastText
ROC AUC	0.80	0.82	0.83	0.83

Таблица 2: Значения метрики ROC AUC

ROC AUC (ROC = Receiver Operating Characteristic, AUC = Area Under the Curve) – площадь под кривой ошибок, где кривая ошибок – это кривая зависимости TPR от FPR при варьировании порога для бинаризации.

Оценка результатов (3)

	threshold	FNR	FPR	accuracy	precision	recall
TF-IDF	0.5	0.19	0.1	0.71	0.76	0.62
Bag-of-words	0.52	0.18	0.1	0.72	0.76	0.65
code2vec	0.51	0.17	0.1	0.72	0.76	0.66
fastText	0.62	0.15	0.1	0.75	0.78	0.72

Таблица 3: Результаты экспериментов на различных метриках с провалидированным threshold к значению FPR = 0.1

Итоги

- Написан инструмент для разбора XML-файла, который является результатом анализа дубликатов IntelliJ IDEA Ultimate
- Реализована векторизация кода с помощью code2vec, bag-of-words и TF-IDF
- Обучены бинарные классификаторы на векторизованном коде
- Для каждого метода векторизации выбрана лучшая модель бинарной классификации с помощью валидационной функции потерь
- Применен фреймворк fastText
- Проанализированы результаты