И. А. Гончарик А. Б. Гуринович

ОБРАБОТКА РЕЗУЛЬТАТОВ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ

В статье исследуется проблема анализа и обработки больших объёмов данных о результатах автоматизированного тестирования. Производится описание алгоритма, предлагающего конкретные шаги для её решения.

Введение

В области автоматизированного тестирования программного обеспечения существует ряд задач, решение которых требует ручной обработки и анализа специалистом-автоматизатором. В современных условиях, когда объёмы данных для анализа стремительно растут, встаёт вопрос об автоматизации процесса разбора результатов тестирования и их анализа. Использование эффективного алгоритма для автоматической обработки в этой области поможет сократить временные затраты и нивелировать человеческий фактор.

I. Выбор показателя успешности выполнения теста

Процесс разработки алгоритма для анализа результатов тестирования следует начать с поиска оптимального критерия, по которому делается вывод об успешности выполнения теста, или показателя, свидетельствующего об обратном. В разрезе автоматизированного тестирования таким показателем служат записи в журнале работы программы (логи).

Записи в журнале являются оптимальным показателем для оценки успешности выполнения теста по ряду факторов:

- Записи, сообщающие об ошибке, содержат информацию об источнике этой ошибки.
- Журнал хранит записи о выполнении операций, предшествующих «падению» теста.
- Набор лексем в тексте ошибки записи журнала уникально её определяет.
- Специалист-автоматизатор делает вывод о причинах неудачного завершения теста, опираясь именно на записи в журнале работы программы.

II. Принцип работы

Перед обработкой записей в журнале их следует нормализовать, «очистив» от стоп-слов, числовых значений, даты, времени, а также привести к нижнему регистру. Опираясь на принятые ранее пользователем решения, алгоритм характеризует лексемы в записи журнала на основе факторов TF-IDF.

TF-IDF (от англ. TF — term frequency, IDF — inverse document frequency) — статистическая мера, используемая для оценки важности слова в контексте документа, являющегося частью коллекции документов или корпуса. Вес некоторого слова пропорционален частоте употребления этого слова в документе и обратно пропорционален частоте употребления слова во всех документах коллекции.

TF (term frequency — частота слова) — отношение числа вхождений некоторого слова к общему числу слов документа. Таким образом, оценивается важность слова t_i в пределах отдельного документа.

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

где, n_t есть число вхождений слова t в документ, а в знаменателе — общее число слов в данном документе.

IDF (inverse document frequency — обратная частота документа) — инверсия частоты, с которой некоторое слово встречается в документах коллекции. Основоположником данной концепции является Карен Спарк Джонс. Учёт IDF уменьшает вес широкоупотребительных слов. Для каждого уникального слова в пределах конкретной коллекции документов существует только одно значение IDF.

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

где, |D| – число документов в коллекции; $|d_i \in D | t \in d_i|$ — число документов из коллекции D, в которых встречается t (когда $n_t \neq 0$).

Выбор основания логарифма в формуле не имеет значения, поскольку изменение основания приводит к изменению веса каждого слова на постоянный множитель, что не влияет на соотношение весов.

Таким образом, мера TF-IDF является произведением двух сомножителей:

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

Большой вес в TF-IDF получат слова с высокой частотой в пределах конкретного документа и с низкой частотой употреблений в других документах. [1]

Векторы значений TF-IDF, полученные для слов из записей журнала, учитываются при выборе соответствующего типа дефекта для теста, падение которого нужно проанализировать. На основе записи журнала, наиболее близкой к анализируемой, извлекается тип дефекта, выбранный пользователем при ручном анализе, и используется в качестве результата работы алгоритма. Пошагово процесс изображён на рисунке 1. Используя метод k-ближайших соседей, алгоритм выбирает наиболее подходящий тип дефекта для анализируемого теста.

Заключение

Подводя итог стоит отметить, что описанный алгоритм способен значительно ускорить процесс разбора «упавших» тестов, однако для его корректной работы необходимы тренировочные данные и предварительное обучение. В качестве учителя выступает непосредственно специалист тестировщик-автоматизатор, данными для обучения являются записи в журнале работы программы. Оценив уникальность и выбрав ключевые слова в записях журнала, алгоритм способен определить причину «падения» теста, опираясь на ранее полученную информацию.

Список литературы

1. Автоматизированное тестирование программного обеспечения / Э. Дастин, Дж. Рэшка, Д. Пол // Издательство: Лори, 2003. – 580 с.
2. Experiences of Test Automation: Case Studies of Software Test Automation / D. Graham, M. Fewster // Addison-Wesley, 2012. – 607 с.
3. Problems with Test Automation and Modern QA [Electronic resource] / A. Ghahrai. – 2018. – Mode of access: https://www.testingexcellence.com/problems-test-automation-modern-qa/.
4. Динамические библиотечно-поисковые системы / Дж Солтон // — Мир, 1979.
5. Machine Learning / T. Mitchell // — McGraw-Hill Science/Engineering/Math, 1997.

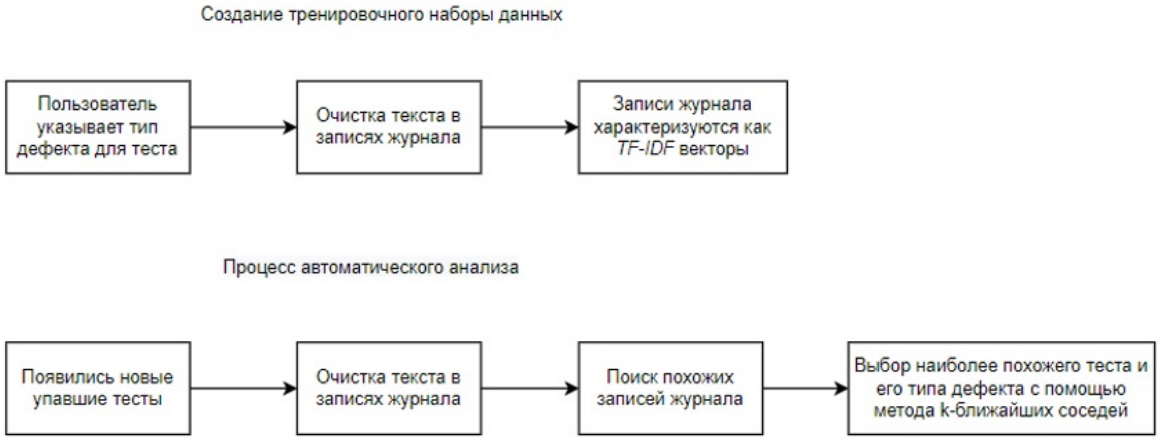

Рис. 1 – Шаги создания тренировочного набора данных и процесса автоматического анализа

Гончарик Илья Александрович, магистрант кафедры информационных технологий автоматизированных систем БГУИР, amsterget@gmail.com.

Научный руководитель: Гуринович Алевтина Борисовна, заместитель декана по научно-методической работе ФИТиУ БГУИР, кандидат физико-математических наук, доцент, gurinovich@bsuir.by.