АНАЛИЗ СХОЖЕСТИ ДЕФЕКТОВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КАК ПОДЗАДАЧА РАНЖИРОВАНИЯ СПИСКА ДЕФЕКТОВ ПРИ ФИНАЛЬНОМ ПРИЁМОЧНОМ ТЕСТИРОВАНИИ

ANALYSIS OF THE SIMILARITY OF SOFTWARE DEFECTS AS A SUBTASK OF RANKING OF DEFECTS IN THE FINAL ACCEPTANCE TESTING

I. Pankov

Summary. this article addresses the background of the problem of finding similar defects in the working process of the Department of business quality control for IT companies, specifies whether to search for similar defects in the framework of the task of ranking defects by severity. Given an overview of the standard methods of parameterization of the texts, the metric of the similarity of the two defects and a set of heuristics, allowing improving used method of finding duplicates. Standard metrics for finding the quality of the list of similar defects was defined. Given the main directions for improving the current methodology.

Keywords: Analysis of the similarity of texts, processing of technical documentation, analysis of software defects, text-mining, information retrieval.

Панков Иван Данилович
Аспирант, Российская академия народного хозяйства и государственной службы при Президенте Российской Федерации, г. Москва, pankov.i.d@gmail.com

Аннотация. в данной статье затрагиваются предпосылки возникновения задачи поиска схожих дефектов в рабочем процессе отдела бизнес-контроля качества IT компаний, обосновывается необходимость поиска похожих дефектов в рамках задачи ранжирования дефектов по важности. Приведен обзор стандартных методов параметризации текстов, затронуты метрики схожести описания двух дефектов и набор эвристик, позволяющих увеличить значение метрик схожести заведомо похожих дефектов. Определены стандартные метрики выявления качества списка похожих дефектов и предложен новый. Намечены основные направления по улучшению существующей методики.

Ключевые слова: Анализ схожести текстов, обработка технической документации, анализ дефектов, информационный поиск, интеллектуальный анализ текста.

Задача, с которой часто сталкивается отдел бизнес-контроля качества — финальная приемка перед выпуском новой версии продукта на рынок. Задача нетривиальная, ведь каждый новый продукт технически сложен.

В рамках этой задачи анализируется набор дефектов D, в котором содержатся два подмножества — исправленных в рамках релиза дефектов Dr (Defects resolved) и неисправленных Da (Defects active). Исходя из различных критериев приемки, которые приняты на разных предприятиях, бывают ситуации, когда продукт не разрешается к публикации, если среди дефектов Da присутствуют высокоприоритетные (High) или критичные (Critical). Такие требования понятны, но иногда приводят к ситуации, когда среди Da есть баги с заниженными приоритетами. В случае, если поиск высокоприоритетных дефектов среди подмножества Da оказывается верно организован и такие дефекты будут найдены в кротчайшие сроки, то у разработчиков остается время на исправление дефектов. Такое удачное стечение обстоятельств поможет продукту выйти на рынок в срок. В противном случае компания может столкнуться с откладыванием выхода продукта до исправления выявленных дефектов, что повлечет за собой экономические потери и репутационные риски. Риски связанные с репутацией особенно актуальны в случае проведения активной маркетинговой компании приуроченной к выпуску продукта.

Таким образом, перед экспертами отдела бизнес-контроля качества стоит задача оценки всех дефектов группы Da за срок, отведенный для анализа проекта. Если на поздних этапах приемки темпы проверки отстают от запланированных, то может возникнуть ситуация, при которой эксперты вынуждены работать сверхурочно и с пониженной производительностью.

Данную проблему можно решить двумя способами — автоматическое ранжирование набора Da в порядке убывания потенциальной критичности, основанной на внутренней информации и поиск исторической информации о критичности похожих дефектах в более ранних версиях или параллельных продуктах. Если первая задача теоретически еще может быть решена без применения экспертной валидации, а только посредством проведения кросс-валидации и определения

размера ошибки на тестовой выборке, то вторая задача не может быть решена без экспертной оценки. Вторая задача является приоритетной еще по одной причине. Если для нового дефекта удается найти похожий дефект с наличием Инцидентов у пользователей, то это может резко увеличить важность исходного дефекта.

В условиях, когда экспертное время слишком дорого для ручной оценки, настройка модели должна быть проведена автоматически без привлечения экспертов на первых этапах. Для понимания того, что же может быть показателем похожести дефектов, хорошо подходят данные о схожести, которые уже имеются в системе. Такими данными является информация о дублировании отчета о дефекте в системе. Данные о дубликатах дефектов имеются как в закрытых коммерческих системах ведения разработки, так и в открытых источниках вроде bugzilla.org. Очевидно, что если экстраполировать значение абстрактной меры похожести двух дефектов до максимума (1 при использовании коэффициента Жаккара и 0 при косинусной мере), то таким теоретическим максимумом должны обладать пары дефектов — Мастер-Дубликат, полностью друг друга копирующие «слово-в-слово». В исследуемой системе информация о дубликатах обозначалось явной связкой $\{D_m, D_d, 1\}$ (Master Defect — Duplicate Defect). К этим данным можно было добавить случайных шумовых пар $\{D_m, D_m, 0\}$ и выявить закономерности — что может являться признаком похожести дефектов.

Процесс обработки текстов

Для целей анализа схожести дефектов необходимо сравнить тексты, описывающие их — текстовые поля заголовка (Title) и описания (Description). В отдельных случаях можно дополнить информацию о дефекте данными из обсуждений дефекта (Comments), но они имеют несравнимо большую размерность и должны отдельно обрабатываться и разбиваться на несколько модулей (выделение разных частей — ссылки, машинный код, общение и обмен комментариями, автоматически генерируемые комментарии системы). Сложность так же вызывает тот факт, что в случае с обсуждением имеет значение очередность.

В [2] достаточно хорошо разобран ряд вопросов предварительной обработки (препроцессинга) текстов дефектов: как учитывать токены (текстовые единицы, в простейшей постановке вопроса — отдельные слова) и какая именно обработка нужна, как разделять тексты на токены, какие токены лишние, как учитывать токены, которые совместно создают неделимые смысловые единицы.

Стандартный подход к препроцессингу является относительно простой последовательностью применения методов: восприятие текстов как простой совокупности слов, лемитизация, выделение коллокаций и квазиколлокаций (учет отрицания при токенах), стемминг отдельных токенов, применение tf-idf меры параметризации, нормализация. [2]

Далее будут рассмотрены некоторые из этих методов.

Препроцессинг методом Bag of words

Для анализа и сопоставления текстов двух дефектов их необходимо привести к набору параметров в векторном пространстве. Стандартным подходом при анализе схожести двух документов является подход оценки текста как совокупности всех слов (bag of words), при котором не оценивается семантика. [3] Такой подход позволяет абстрагироваться от сложных методов анализа смысла, содержащегося в дефекте. Негативной стороной данного подходя является потеря некоторой части информации, которая могла бы улучшить поиск похожих дефектов.

При подходе Bag of words чаще всего используются несколько стандартных способов измерения частот слов *(Term Frequency, Term Occurrences, Binary Term Occurrences*, *TF-IDF)* и их различные комбинации. Использование Binary Term Occurrences в начальной постановке задачи о поиске похожих дефектов может облегчить интерпретируемость результатов уберет необходимости в нормировке. Мера TF-IDF хорошо подходит для «информационно богатых» документов, но в задаче с небольшими описаниями дефектов лучше себя показывает мера IDF. [4] В описании дефекта редко встречаются повторы терминов, но некоторые термины (t) встречаются гораздо чаще других, что заставляет корректировать ситуацию с помощью меры IDF:

$$\mathrm{idf}(t, D) = \log_2 \frac{|D|}{|(d_i \supset t_i|} \qquad (1)$$

где:

$|D|$ — количество документов в корпусе

$|(d_i \supset t_i|$ — количество документов, в которых встречается t_i

Использование меры TF-IDF может слишком сильно увеличить вес слов, встречающихся чаще 1 раза за счет части TF (term frequency). Анализ частотностей токенов рассмотрен в последней части статьи.

Стоп слова, лемматизация, стемминг, выявление коллокаций

Подход «bag of words» сам по себе не гарантирует, что дефекты, для которых половина терминов будет схожа,

окажутся более близки, чем два других дефекта с полностью различными наборами слов. Связанно это с тем, что половина слов в этих дефектах может быть просто высокочастотны, а слова, действительно важные для понимания схожести, будут использоваться в различных формах и сочетаниях.

С проблемами стоп слов в некоторой мере помогает справиться мера idf, но на коллекциях документов с короткими текстами не всегда справляется с задачей отброса слишком частотных слов. Для этого используют списки стоп-слов, получаемые из национальных корпусов или же полученные через отношение частот в национальном корпусе и в исследуемом. Если исследуется узкоспециализированная тематика (набор дефектов программного продукта), то становится необходим дополнительный список стоп слов, который формируется экспертами.

Лемматизация: Даже если удалить все незначащие слова, может оказаться, что некоторые важные слова встречаются в разных формах. Решить эту задачу помогает приведение к нормальной форме — лемматизация. После этого размерность обычно уменьшается незначительно, но степень связанности текстов повышается резко за счет этих слов. [2] Иногда лемматизацию разумно провести до отброса стоп слов, особенно в случаях, когда отбрасывается топ самых частотных токенов.

Стемминг: В рамках задач обработки текстов не на естественном языке или смеси языков иногда встречаются слова, которые не представлены в словарях, но образованы при помощи спряжения транслитерации английских терминов («дампа» вместо «dump»). Привести к нормальной форме такие слова не удается, в связи с чем приходится применять стемминг — усечение слов до основы с отбрасыванием окончаний и в некоторых случаях суффиксов.

Поиск коллокаций: Некоторые термины являются общеупотребимыми, но в паре с другими создают узкий термин, который не имеет ничего общего с изначальным, называемый коллокацией. Примером такой коллокации служит термин «Железная дорога». Если коллокация устойчивая и достаточно частотна (относительно других терминов), то её можно выявлять с помощью метрики *MI* (Mutual Information) с отсечением по порогу. [2] В работе с текстами дефектов хорошо себя зарекомендовал порог $MI \geq N$ для N-грамм с минимальной частотой встречаемости коллокации $v = 5$.

В практике возникали случаи, когда поиск коллокаций итерационно позволял выявить длинные фразы, генерируемые автоматически системой. Такие последовательности токенов в действительности должны восприниматься как единый сигнал и токены, формирующие их, не должны учитываться отдельно.

Выявление коллокаций разумнее делать именно после этапа леммизации, но до стеминга токенов. В противном случае можно исказить статистику встречаемости отдельных составных частей коллокации. Недостатком данной меры является завышение веса редких слов. Эту проблему успешно решают более высокие пороги MI и v.

В отличие от авторов [2] решено было не использовать t-score в виду сильной корреляции результатов на нашей небольшой коллекции документов.

Сравнение текстов двух дефектов

Хорошо разработанные подходы не всегда могут оказаться применимы на практике в виду специфики области применения. Далее будет продемонстрированы некоторые сложности стандартных подходов.

Близость двух дефектов по описаниям и заголовкам предлагается измерять в нескольких размерностях: сравнение только заголовков, сравнение только описаний, сравнение по всем имеющимся текстовым данным. [4] Кроме того, предлагается учитывать при анализе не только односложные токены (униграммы), но и биграммы — последовательности из двух токенов. [2] Однако подход, основанный на униграммах за счет снижения точности позволяет справиться сростом разрежённости параметрического пространства дефектов.

В рамках изучения коллекции дефектов предприятия X возникла ситуация сильной разрежённости векторного пространства признаков. На небольшой коллекции в 10000 документов число токенов, встретившихся более чем в двух различных документах, составляет нескольким более 15000 единиц, что означает большое число размерностей. Распределение частотностей токенов представлено на рисунке 1.

Однако количество словоформ (без фильтрации стоп-слов) в каждом дефекте на порядок меньше. На рисунках 2 и 3 представлено распределение длины заголовков и описаний соответственно.

Препроцессинг заголовков и описания дефектов снижает число признаков с 15000 лишь до 13000, что связанно с огромным числом несловарных токенов, в том числе профессиональных жаргонизмов, наличие частей программного кода в описании дефекта. Одно из решений данной проблемы заключается в исключении всех несловарных токенов, но это может повлечь за собой потерю всей специфической информации и поиску стилистических особенностей дефекта. В конечном итоге

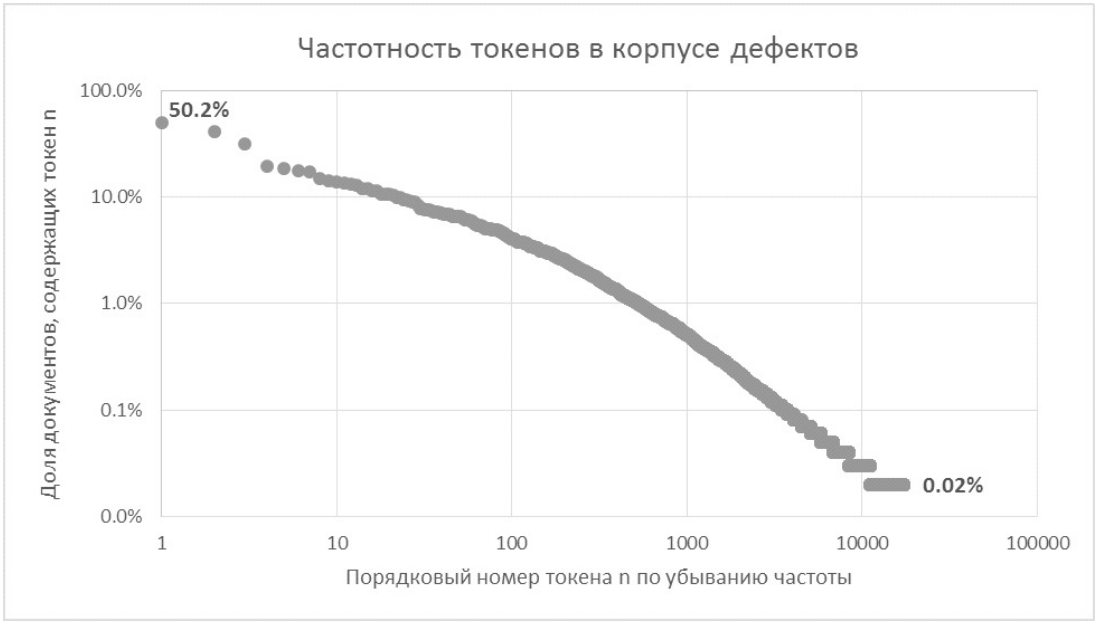

Рисунок 1. Частотность токенов в корпусе дефектов

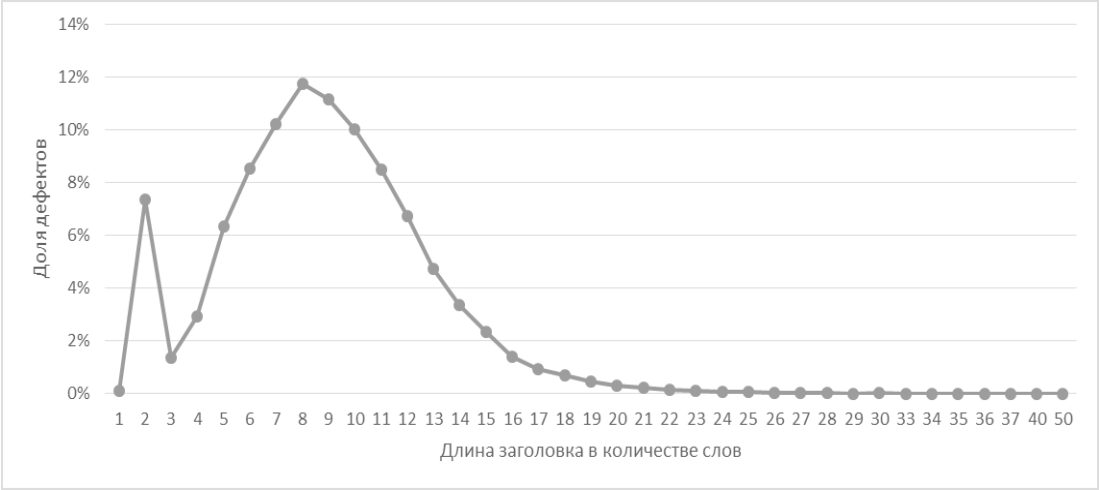

Рисунок 2. Распределение количества токенов в заголовках дефектов

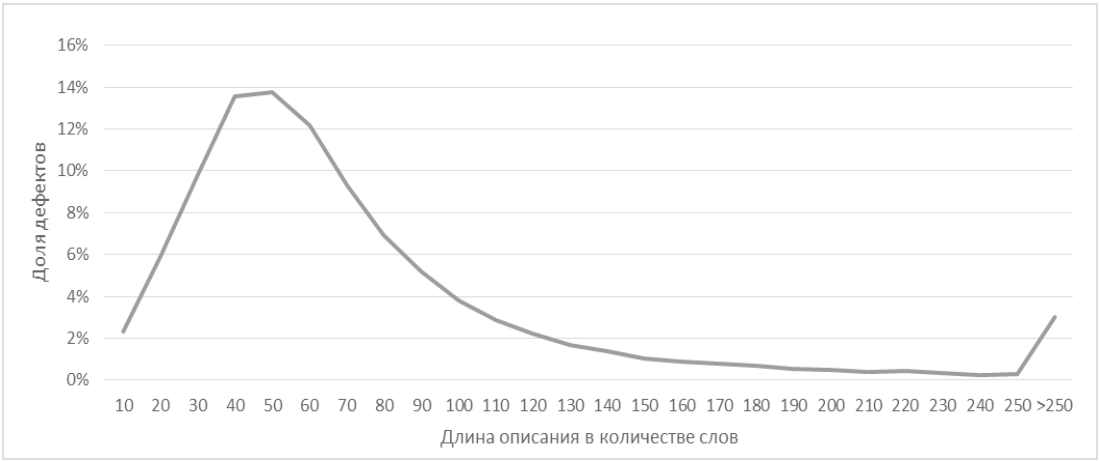

Рисунок 3. Распределение количества токенов в описаниях дефектов

дефекты группируются в кластеры, главной характеристикой которой был специфический стиль автора отчетов о дефекте.

На рисунке 2 можно увидеть скачок в распределении для двух слов. Это связанно с автоматической генерацией большого числа дефектов по событию. Заголовок таких автоматических дефектов всегда состоит из двух токенов — событие и условие. Эта информация важна для поиска схожести двух дефектов, но полное совпадение таких лаконичных заголовков еще не гарантирует, что причины возникновения дефекта идентичны. При такой разрежённости разумно использовать подходы, хорошо работающие с разряженными матрицами.

Анализ схожести

Для анализа схожести дефектов использовалась косинусная мера и метод поиска ближайшего соседа. Для целей проверки точности обученной модели была собрана выборка из пар {Dm, Dd, 1} и {Dm, Dm, 1}. Если у Dm было более 1 дубликата Dm-d, то все Dm-d устанавливались в однозначное соответствие Dm- di — Dm- dj (i>j) по дате заведения. Так удалось решить проблему с кликами неявных дубликатов.

В ходе анализа для каждого дефекта формировалась выборка N ближайших соседей и считалась доля дубликатов среди найденных соседей. При низкой точности модели увеличение числа соседей увеличивает долю дубликатов среди ответов, но нужно помнить, что экспертам будет необходимо проверять эти N соседей, что влечет сложность использования модели. Для устранения этой проблемы в будущем планируется введение регуляризации и настройка порога отсечения по расстоянию для рекомендации ≤N соседей с целью увеличения точности.

Итоги

Подход, опробованный на коллекции тестовых документов, является стандартным для работы с большими корпусами. Но в данном случае стало очевидно, что необходим уход к меньшим размерностям без потери смысла в виду малого набора документов в корпусе дефектов. Более сложный подход, который планируется реализовать в ближайшей модификации — использование набора синонимов и замена всех слов на центральный термин в кластере синонимов. Для этого предполагается расширить выборку данными из комментариев, обучить модель Word2Vec или использовать тематическое моделирование на используемом корпусе и выделить кластеры синонимов значимых слов.

ЛИТЕРАТУРА

1. Александров М. А. Разработка общей методологии анализа общественного мнения Интернет-сообщества и ее приложение к заданным темам (власть, экономика, коррупция и пр.) на основе инструментов Data/Text Mining // Москва, 2013. — 204 с.
2. Большакова Е. И., Клышинский Э. С., Ландэ Д. В., Носков А. А., Пескова О. В., Ягунова Е. В. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика: учеб. пособие — М.: МИЭМ, 2011. — 272 с.
3. Маннинг Кристофер Д., Рагхаван Прабхакар, Шютце Хайнрих, Введение в информационный поиск, — М.: Вильямс, 2014. — 528 с.
4. Chengnian Sun1, David Lo2, Xiaoyin Wang3, Jing Jiang2, Siau-Cheng Khoo A Discriminative Model Approach for Accurate Duplicate, Bug Report Retrieval, Cape Town, South Africa, 2010
5. Hooimeijer P., Weimer W. Modeling Bug Report Quality // ASE'07, Atlanta, Georgia, USA, 2007