Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский
Академический университет Российской академии наук»
Центр высшего образования

Кафедра математических и информационных технологий

Поляков Семен Григорьевич

Выявление сообществ в Stackoverflow

Магистерская диссертация

Допущена к защите.
Зав. кафедрой:
д. ф.-м. н., профессор Омельченко А. В.

Научный руководитель:
Калашников Д. А.

Рецензент:
к. т. н., Тимошенко Д. М.

Санкт-Петербург
2017

Оглавление

Введение

Software Repository Mining – одна из современных сфер научных исследований. Она включает в себя анализ данных, собранных из систем контроля версий, систем отслеживания ошибок, архивов систем почтовой рассылки, и других. Основная цель подобных исследований - извлечь информацию о структуре интересуемых проектов, ее изменениях во времени, основных свойства и элементах [6].
Stackoverflow – система вопросов и ответов о программировании, и также является одним из объектов исследования в Software Repository Mining. В нашем случае, основным источником информации является коллекция **постов** Stackoverflow (рис. 6).

Определение 1. Пост Stackoverflow – четверка (U, $(T_1, \ldots, T_{Tc})$, Text, P), где U – автор поста, T_k – k-й тег поста, Tc – количество тегов в посте, Text – текст поста, P – набор метрик популярности поста.

Определение 2. Тег Stackoverflow – метка, для обозначения информационной технологии, либо термина, относящегося к той или иной информационной технологии.

Рис. 1: Примеры поста и тега из Stackoverflow

№	Кластер
1	android, android-wear, andengine, apk, ..
2	java, java-ee, junit, aop, guice, ..
3	python, pickle, virtualenv, pip, ..
4	...

Таблица 1: Пример кластеризации тегов

Оценив, сколько пользователей Stackoverflow задают и отвечают на вопросы по той или иной технической теме, можно оценить популярность этой темы в целом, за соответствующий период времени. Подобная статистическая информация, в совокупности с другими данными, применяется в IT компаниях для анализа рынка ПО и предсказания уровня продаж [5].

Однако поскольку нет известного однозначного соответствия между постами Stackoverflow и затрагиваемыми в них техническими темами, встает задача составления приближенного списка таких тем, с целью выявить наиболее трендовые из них. При этом в нашем случае сами темы будут описываться используемыми тегами, а следовательно и соответствующими им информационными технологиями. Этого можно добиться, сгруппировав теги по некотрой мере схожести (табл. 1).

Таким образом появляется задача выявления групп похожих элементов, из некоторого общего набора, на основе некоторой меры схожести. В данном случае – набора всех тегов Stackoverflow. Такая задача носит название кластеризации, а если более конкретно – кластеризации тегов.

В главе 1 рассматриваются основные особенности поставленной задачи, а также дается краткое описание существующих исследований по данной теме. Во главе 2 описывается работа по предварительной обработке данных Stackoverflow. В третьей главе описаны основные подходы к кластеризации тегов, их математический аппарат, а также дано описание нового, специфичного для данной задачи регуляризатора, и выписаны метрики полученных результатов. В последней главе описан не сложный алгоритм кластеризации пользователей, работающий на основе результатов, полученных в третьей главе.

1. Постановка и анализ задачи

1.1. Анализ предметной области

На данный момент уже существует пример научной работы, авторы которой исследовали кластеризацию тегов Stackoverflow. Однако существуенным ее недостатком не текущий момент является вычислительно сложный подход, который основан на отображении каждого тега в вектор, размерность которого равна числу всех постов на Stackoverflow. И если на момент написания исследования, такая размерность не превышала нескольких сотен тысяч, то сейчас подобный подход требует использования векторов размерностью равной примерно 35 млн [4]. В виду этого, будет использованы другие подходы.

Одним из наиболее известных примеров кластеризации тегов является кластеризация тегов социальной сети Twitter [7]. Для решения этой задачи, исследователи применяли несколько подходов, а именно кластеризацию графа тегов, тематическое моделирование и кластеризацию векторного представления тегов (через TF-IDF) [1]. Во всех этих случаях, требуется значительная работа по подготовке данных, а именно фильтрации и преобразованию исходных тегов. В первую очередь, это позволяет объединить семантически похожие теги (в качестве примера для Stackoverflow – версии одной и той же технологии). Также это увеличивает производительность программной системы, решающей задачу кластеризации тегов, за счет уменьшения размерности задачи. Также отличительной особенностью Stackoverflow по сравнению с Twtiiter, является то, что посты Stackoverflow содержат не только тексты на естественном языке, но и довольно много программного кода и трассировок ошибок. В рамках текущей работы это накладывает соответствующее ограничение на подходы к кластеризации – они не должны быть основаны на векторизации и анализе текста.

1.2. Цели и задачи

Целью данной работы является разработка программного решения для выявления сообществ пользователей Stackoverflow для основных кластеров тегов, среди представленных на Stackoverflow.

Для достижения поставленной цели были поставлены следующие задачи:

- Подготовить данные Stackoverflow для кластеризации
- Сравнить качество различных методов кластеризации на данных Stackoverflow
- Оценить количество пользователей в каждом из получившихся кластеров

В дальнейшем, планируется использование полученных результатов в отделе маркетинга JetBrains в качестве одного из источников данных для анализа рынка ПО.

2. Обработка данных

2.1. Отбор популярных тегов

На текущий момент на Stackoverflow используется более 48 тысяч тегов. При этом большая их часть используется достаточно редко (рис. 2)

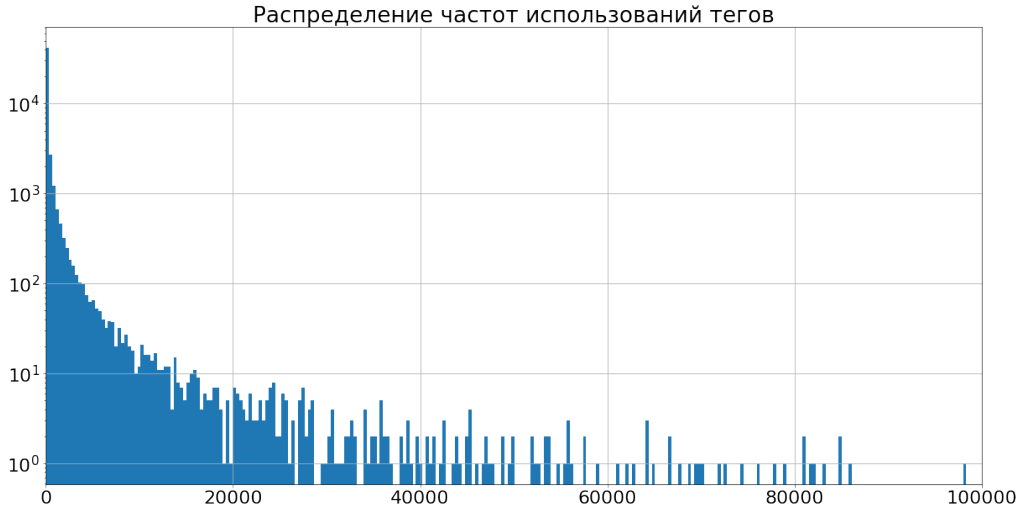

Рис. 2: Распределение частот тегов на Stackoverflow

Для уменьшения размерности задачи кластеризации, а также повышения ее качества, было решено заменить редко используемые теги (непопулярные), популярными, которые чаще всего идут с ними в одном вопросе в паре.

В качестве меры популярности тега было выбрано число вопросов, в которых этот тег используется. Теги с числом вопросов меньше 100 назовем непопулярными. Такой подход подтверждается тем фактом, что доля таких тегов составляет более 70%, однако доля пользователей, которые пользуются хотя бы одним непопулярным тегом – меньше половины. В ходе данной работы было показано, что непопулярные теги очень редко используются без популярных в одном посте. А именно, не имеет значения, есть ли в посте только популярный тег, либо непопулярный с его более общим, популярным аналогом. Чтобы доказать это, была построена модель логистической регрессии, которая предсказывает будет ли количество просмотров поста превышать среднее значение (среди всех постов). Причем до обучения модели были отобраны посты, с числом просмотров не превышающих 95 перцентиль, для отсечения возможного шума. Также были сгенерированы и отобраны входные признаки, которые дали лучшие значения точности и f1-метрики (рис. 3), с проверкой кросс-валидацией на 10 частях.

В итоговой модели использовались следующие признаки:

- Число ответов на вопрос

- Рейтинг автора вопроса
- Популярных тегов в вопросе больше, чем непопулярных
- В вопросе используются только непопулярные теги
- В вопросе используются только популярные теги

```
Call:
glm(formula = MoreThanMeanViews ~ Answers + AuthorReputation +
    MorePopular + OnlyUnpop + OnlyPop, family = binomial, data = trainData,
    control = glm.control(epsilon = 1e-12, maxit = 100, trace = TRUE))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.5430  -0.7808  -0.6399   0.9993   2.4948

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.218005   0.027262 -44.678  < 2e-16 ***
Answers           0.635631   0.008650  73.487  < 2e-16 ***
AuthorReputation  0.237291   0.007684  30.882  < 2e-16 ***
MorePopularTRUE   0.339389   0.029691  11.431  < 2e-16 ***
OnlyUnpopTRUE    -0.748449   0.044597 -16.783  < 2e-16 ***
OnlyPopTRUE      -0.134651   0.016742  -8.043 8.78e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 110161  on 93661  degrees of freedom
Residual deviance: 100312  on 93656  degrees of freedom
AIC: 100324

Number of Fisher Scoring iterations: 5
```

Рис. 3: Модель логистической регрессии числа просмотров поста

Как результат, из 48 тысяч тегов были отобраны 14 тысяч популярных, а также построено отображение из непопулярных тегов в парные и соответствующие им популярные.

2.2. Отбор активных пользователей

На момент написания данной работы, на Stackoverflow зарегистрированно более 7 млн. пользователей. При этом анализ их случайной подвыборки (в 50 тысяч человек) показал, что значительная часть этих пользователей (более 30%) не является релевантными в плане активности при ответах на те или иные технические темы [2]. А именно, у таких пользователей либо все, либо почти все выбранные метрики отрицательные. Исходя из анализа предыдущих исследований [8] были выбраны следующие метрики:

- z-score числа ответов и вопросов ($\frac{q-a}{sqrt(a+q)}$), где q – число вопросов пользователя, a – число ответов пользователя)

- Соотношение числа ответов к вопросам пользователя
- Средний рейтинг ответа пользователя
- Среднее соотношение рейтинга ответов пользователя к суммарному рейтингу ответов на те же посты ($\frac{1}{A}\sum A_{i=0}\frac{r_i}{R_i}$, где A – число ответов пользователя, r_i – рейтинг i-го ответа пользователя (на пост P), R_i – суммарный рейтинг ответов на пост P)
- Средняя разница между длиной ответа пользователя (в словах), и общей средней длиной ответа на Stackoverflow

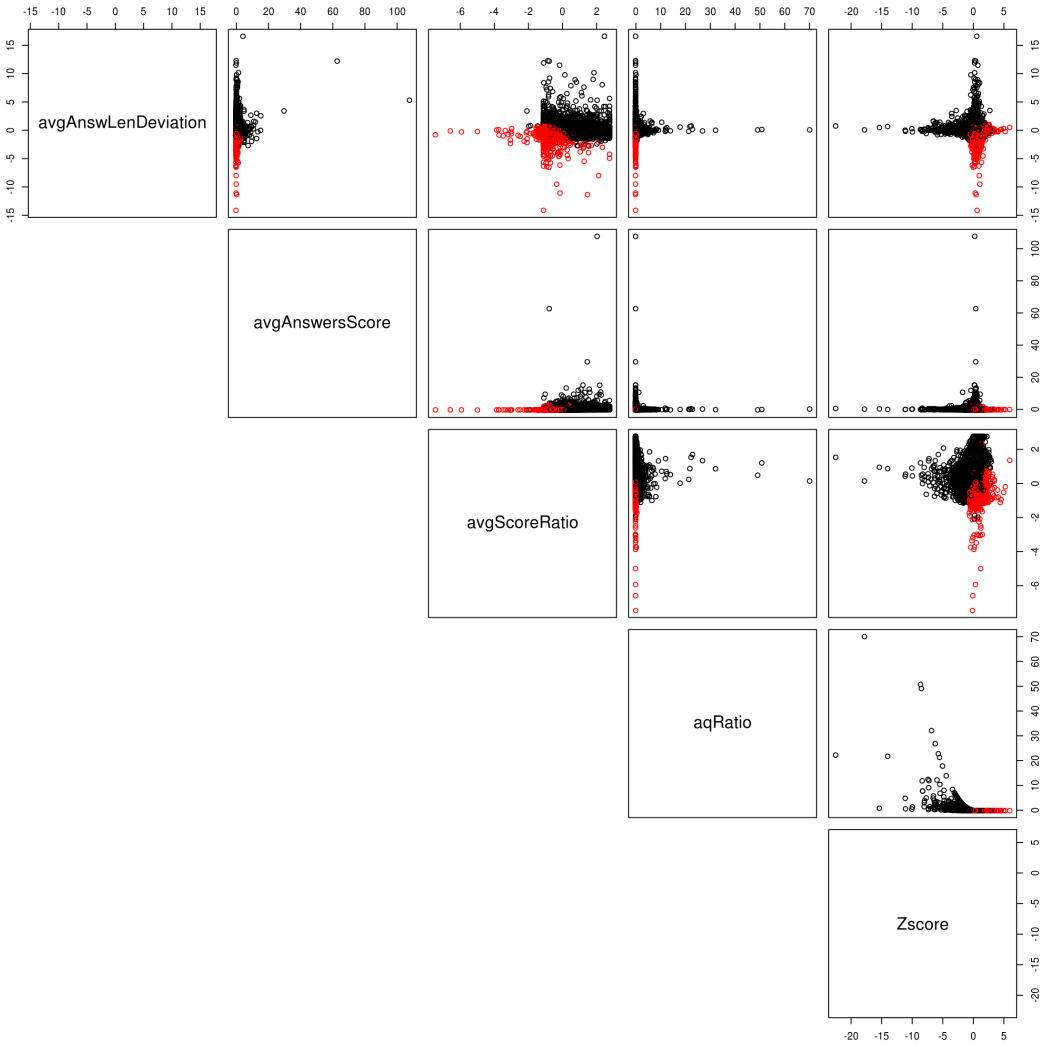

Рис. 4: Результаты иерархической кластеризации для определения активных пользователей

Сопоставив каждому пользователю из подвыборки вектор из значений этих метрик, была получена выборка для кластеризации. После чего были построены несколько моделей, а именно модели K-средних, DBSCAN, иерархическая кластеризация (с

несколькими вариантами подсчета дистанций между кластерами) и EM-алгоритм для модели гауссовых смесей. В результате, почти все модели выделили в отдельные кластТ составила иерархическая кластеризация с алгоритмом подсчета дистанций "ward.D", которая явно разделила пользователей на недостаточно активных – с отрицательными значениями метрик, и остальных (рис. 4). На дендрограмме (рис. 5) видно, что алгоритм кластеризации обнаружил две группы пользователей, с большим расстоянием между центроидами кластеров, в которой левый кластер группирует недостаточно активных пользователей.

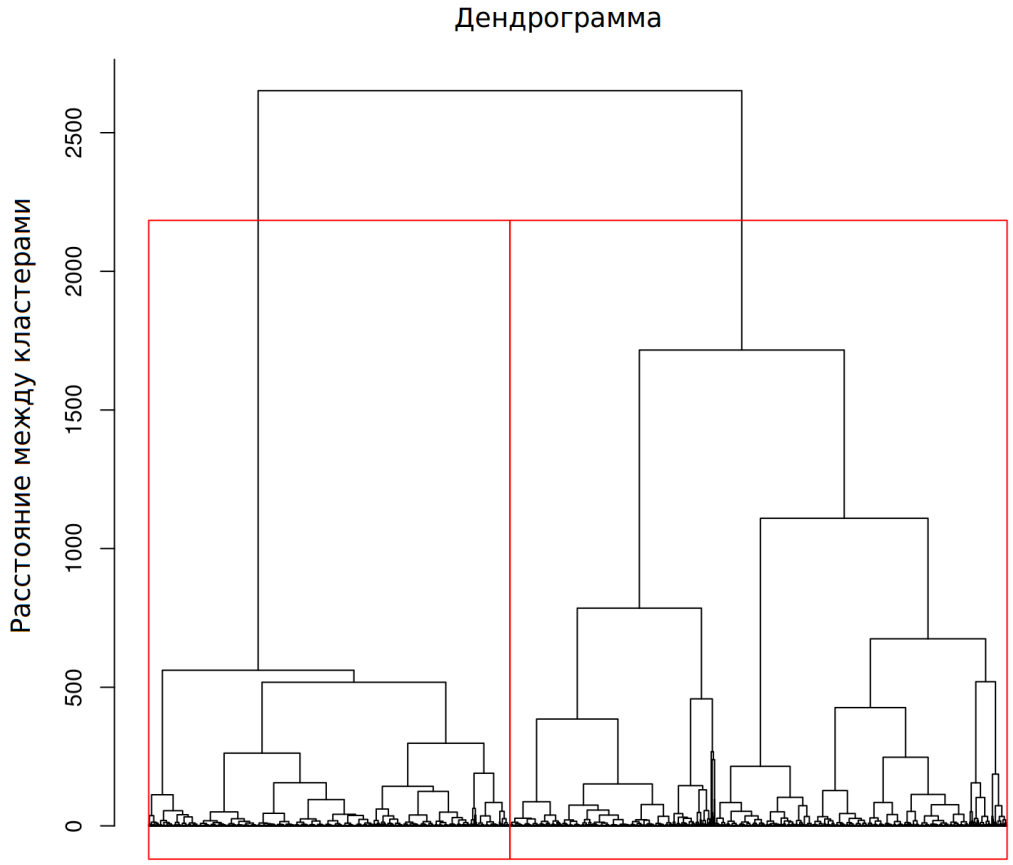

Рис. 5: Результаты иерархической кластеризации для определения активных пользователей

3. Кластеризация тегов

3.1. Введение

Задача кластеризации является одной из частых задач в области разведовательного анализа данных и уществует несколько постановок задачи кластеризации. В данной работе, будем рассматривать постановку задачи жесткой кластеризации, а именно найдем отображение из каждого тега в номер соответствующего ему кластера.

Для решения данной задачи в такой постановке есть несколько подходов.

Во-первых это подход с построением меры схожести тегов и кластеризации графа тегов, ребра которого взвешены значениями этой меры схожести. Этот подход активно исследуется в последние годы, однако для него пока не известно алгоритма, опережающего остальные как по качеству, так и по времени работы для произвольных графов. Поиск такого алгоритма является открытой проблемой.

Во-вторых это менее известный подход с построением тематической модели и преобразованием ее результатов для решения задачи кластеризации.

Существует также подход с кластеризацией набора векторов. Однако в случае тегов известные попытки сопоставить каждому тегу вектор, основанный на их совстречаемости, приводят к построению векторов высокой размерности, кластеризация которых вычислительно сложна, а потому в данной работе такой подход не рассматривается.

3.2. Кластеризация графа

Для решения задачи кластеризации тегов построим неориентированный граф. Вершинами графа будут теги, а ребра проведем между вершинами с ненулевой мерой схожести, определение которой дадим ниже, и также взвесим эти ребра значением данной меры. Построим несколько мер схожести тегов, каждая из которых является коэффициентом Жаккара посчитанном на числе общих:

- Постов, взвешенных по рейтингу
- Просмотров (по общим постам)
- Ответов (по общим постам)
- Пользователей (по общим постам)
- Число общих соседних вершин, полученным по предыдущим метрикам

Далее, нормируем значения этих мер так, чтобы у них было нулевое среднее и дисперсия равна 1, после чего взвесим граф их линейной комбинацией. В дальнейшем, будем оптимизировать линейные коэффициенты для каждого из выбранных алгоритмов кластеризации.

В целях оптимизации производительности, а также регуляризции полученных коэффициентов, вместо итеративных алгоритмов оптимизации, был произведен перебор всех подмножеств описанных мер, с коэффициентом 1.

Первая проблема, которая возникает при поиске кластеров в графах (так же называемых *сообщества*), является точное определение того, что называть кластером, либо как измерить степень похожести подграфа на кластер [3]. Однако общая интуиция состоит в том, что внутри кластера должно быть больше ребер, чем между кластером и остальной частью графа. В некоторых случаях авторы соответствующих исследований дают определения подобным *сообществам* алгоритмически – как результат работы представленных алгоритмов [3].

Перейдем к рассмотрению непосредственно алгоритмов. Для описания их вычислительной сложности введем несколько обозначений:

- Пусть n – число вершин в графе.
- Пусть m – число ребер в графе.

На текущий момент существует много методов кластеризации, рассмотрим основные [3].

- Алгоритмы разбиения графа. Для наших целей они не подходят, так как для них требуется указать размеры всех искомых кластеров.
- Алгоритмы иерархической кластеризации, которые в свою очередь делятся на *объединяющие* и *разделяющие*. Принцип первых состоит в объединении кластеров с наименьшим расстоянием между ними, при этом алгоритм начинает работу с того, что каждый элемент исходного множества принимается за отдельный кластер, время его работы в общем случае $O(n^2 log(n))$. Принцип работы второго состоит в разделении исходного множества, каждый шаг при этом максимизирует расстояние между получившимися кластерами, а время работы $O(2^n)$.
- Разделяющие алгоритмы. Их принцип похож с разделяющими алгоритмами иерархической кластеризации, а отличие заключается в том, что в данном случае удаляются ребра между кластерами, а не те, которые соединяют вершины с наименьшей мерой схожести, и сооветственно в таком случае нет априорного знания о том, что межкластерные ребра будут соединять вершины с малой мерой схожести. Лучшее время $O(\frac{m^4}{n^2})$.

- Спектральный алгоритм кластеризации. Алгоритм заключается в представлении векторов исходной матрицы смежности в базисе собственных векторов (как правило – первых k), и в последующей кластеризации обычным алгоритмом, работющим с векторными представлениями кластеризуемых элементов, чаще всего это алгоритм K-средних. Время работы имеет ограничение сверху $O(n^3)$, однако в случае приближенного вычисления собственных векторов сходимость зависит от разницы между k-м и (k+1)-м собственными значениями матрицы смежности и обычно происходит существенно быстрее верхней оценки.

- Алгоритмы, оптимизацирующие модулярность.

  Определение 3. Модулярность кластеризации графа – метрика качества кластеризации, цель которой в сравнении плотности ребер в кластере с плотностью, которую можно ожидать в произвольном подграфе (так называемая *нулевая модель*), имеющим те же степени вершин.

  В общем виде ее формула имеет вид:

$$Q = \frac{1}{2m} \sum ij(A_{ij} - P_{ij})\delta(C_i, C_j),$$

  где A - матрица смежности, P_{ij} – ожидаемое число ребер между вершинами i и j, и δ функция принимает значение 1, если i и j принадлежат одному кластеру, и 0 в противном случае.
  В данной работе используется вариант вычисления модулярности, в котором $P_{ij} = \frac{s_i s_j}{2W}$, где k_i – *сила* (сумма весов инцидентных ребер) вершины i, а W – сумма весов всех ребер (так как имеется взвешенный неориентированный граф).
  Оптимизация такой метрики является целью многих исследований последних лет, посвященных кластеризации графов, и одним из популярных алгоритмов для этого является алгоритм жадный алгоритм разработанный Ньюманом М. Этот алгоритм похож на объединяющий алгоритм иерархической кластеризации, с тем единственным отличием, что объединение происходит для тех вершин, которое дает наилучшее значение модулярности (жадный шаг). Время работы $O((n + m)n)$
  Более производительным вариантом является жадный алгоритм Блондела В. [9] В нем изначально все вершины графа заносятся в отдельные сообщества, и первый шаг состоит из последовательного обхода всех вершин: для вершины i вычисляется ее вклад в общую модулярность, появляющуюся из объединения

вершин i и ее соседа j в сообщество, с последующим выбором соседней вершины j, дающий наибольший вклад в модулярность. После обхода получается первый уровень разбиения, и получившиеся кластера заменяются вершинами, ребра который являются объединением ребер, входящих в них вершин, при этом веса ребер соединяющих одни и те же вершины суммируются. После чего эти шаги повторяются до прекращения роста значения модулярности. Время работы – $O(m)$.

- Алгоритмы, основанные на случайных блужданиях. Среди них, одним из наиболее популярных является Марковский алгоритм кластеризации (Markov Cluster algorithm, MCL). Этот метод симулирует процесс диффузии потока в графах. В алгоритме используется *матрица переходов*, задающая вероятности перейти из каждой вершины в каждую, и в получающаяся из матрицы смежности добавлением петель и нормировкой весов переходов для каждой вершины. Сам алгоритм состоит из двух шагов. На первом шаге, называемом *расширение*, матрица переходов возводится в степень (обычно равная 2). Значение M_{ij} получившейся матрицы задает вероятность p, что случайное движение по графу, начинающееся в вершине i, достигнет j за p шагов (соответствует диффузии потока). Второй шаг, называемый *инфляция*, который не имеет физического соответствия, состоит в возведении всех элементов матрицы M в некоторую степень α. При этом обычно во время умножения матриц, MCL сохраняет только как макисмум k ненулевых элементов на колонку, где $k \ll n$. Время работы – $O(nk^2)$.

В итоге, поскольку была необходм оптимизация коэффициентов для ребер графа, были выбраны и опробованы одни из быстрых алгоритмов кластеризации, а именно алгоритмы Спектральной кластеризации, Fast Greedy и Марковский алгоритм кластеризации.

Для оценки качества работы выбранных алгоритмов были выбраны популярные метрики современных исследователей [3]:

- Модулярность (см. формулу 3.2), ее главное ограничение - определена только для графовых моделей
- Индекс Девиса-Боулдина (DBI) Введем несколько обозначений:
  - S_i – мера различия элементов внутри кластера i, в случае графов ее полагают равной диаметру кластера, а в случае векторного представления – равна дисперсии внутри кластера
  - M_{ij} – мера различий между кластерами i и j, в случае графов ее полагают равной длине кратчайшего пути среди всех вершин соответствующих кластеров, в случае же векторного представления равна евклидовому расстоянию между центроидами кластеров i и j

- $R_{ij} = \frac{S_i + S_j}{M_{ij}}$
- $D_i = \max_{j \neq i} R_{ij}$

Тогда DBI определяется как:

$$DBI = \frac{1}{N} \sum_{i=1}^{N} D_i,$$

где N - число кластеров

- Нормализованная взаимная информация (NMI) Определим энтропию кластеризции U для N элементов:

$$H(U) = \sum_{i=1}^{|U|} P_U(i) \log(P_U(i)),$$

где $P_U(i) = |U_i|/N$ – вероятность, что объект, выбранный случайно из U попадет в кластер U_i.
Также определим взаимную информацию двух кластеризаций U и V как:

$$MI(U, V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} P(i, j) \log(\frac{P(i, j)}{P_U(i) P_V(j)}),$$

где $P(i, j) = |U_i \cap V_j|/N$ – вероятность, что случайно выбранный элемент попадет одновременно в кластеры U_i и V_j
Тогда нормализованная взаимная информация двух кластеризаций U и V определяется как:

$$NMI(U, V) = \frac{MI(U, V)}{\sqrt{H(U) H(V)}}$$

- Скорректированный индекс Ренда (ARI):
Пусть C - "истинная" кластеризация для N элементов, а K - кластеризация, которую оцениваем относительно C, тогда:

  - a – число пар элементов, которые находятся в одном и том же кластере и в C, и в K
  - b – число пар элементов, которые находятся в разных кластерах и в C, и в K

В этом случае можно определить индекс Ренда (RI) как:

$$RI = \frac{a + b}{\binom{N}{2}}$$

Однако такая метрика не гарантирует, что случайная кластеризация в среднем будет близка к 0. Чтобы исправить это, введем скорректированный индекс Ренда:

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]},$$

где - E[RI] - оценка математического ожидания индекса Ренда для случайных кластеризаций, и в свою очередь max(RI) - максимальное значение индекса Ренда среди полученных для случайных кластеризаций.

Для использования последних двух метрик была построена частичная кластеризация (golden standard) небольшого подмножества наиболее популярных тегов (100 самых популярных), которая считалась заведомо "истинной", и по которой и будет в дальнейшем выбран алгоритм, дающий наилучшие результаты.

3.3. Тематическое моделирование

3.3.1. Введение

Построение тематической модели по коллекции документов сводится к оптимизационной задаче стохастического матричного разложения. В общем случае она имеет бесконечное множество решений, то есть является некорректно поставленной. В этом случае её решение можно сделать устойчивым, добавив к основному критерию дополнительный критерий — регуляризатор, учитывающий специфику предметной области.

Аддитивная регуляризация тематических моделей (*additive regularization for topic modeling*, ARTM) — это многокритериальный подход, в котором к основному критерию добавляется взвешенная сумма регуляризаторов. APTM позволяет комбинировать тематические модели, суммируя регуляризаторы. Благодаря свойству аддитивности, оптимизация любых моделей и их комбинаций производится одним и тем же итерационным процессом, называемым EM-алгоритмом. Для добавления регуляризатора в модель достаточно знать его частные производные по параметрам модели. EM-алгоритм хорошо масштабируется, поскольку каждая его итерация — это один линейный проход по коллекции, а число итераций, требуемых для сходимости процесса, как правило, невелико [10].

3.3.2. Построение тематической модели

Пусть D – множество всех постов Stackoverflow (в терминах тематического моделирования – множество документов), W – множество тегов (*словарь* тематической модели). Каждый пост-документ представляет из себя набор тегов $w_1, \ldots, w_{n_d}$ из словаря W.

Вероятностное пространство. Предполагается, что существует конечное множество тем T, и каждое вхождение тега w в документ d связано с некоторой темой $t \in T$. Коллекция документов рассматривается как случайная и независимая выборка троек (w_i, d_i, t_i), i = 1, . . . , n из дискретного распределения p(w, d, t) на конечном вероятностном пространстве W x D x T. Теги w и документы d являются наблюдаемыми переменными, тема $t \in T$ является латентной (скрытой) переменной.

Гипотеза условной независимости. Предполагается, что появление тегов-слов в документе d по теме t зависит от темы, но не зависит от документа d, и описывается общим для всех документов распределением $p(w|t)$. Это предположение, называемое гипотезой условной независимости, допускает три эквивалентных представления:

$$\begin{aligned} p(w|d,t) &= p(w|t) \\ p(d|w,t) &= p(d|t) \\ p(d,w|t) &= p(d|t)p(w|t) \end{aligned} \tag{1}$$

Вероятностная порождающая модель выражает вероятности $p(w|d)$ появления тегов-слов w в документах d через распределения $p(w|t)$ и $p(t|d)$. Согласно формуле полной вероятности и гипотезе условной независимости:

$$p(w|d) = \sum_{t \in T} p(w|t)p(t|d) \tag{2}$$

Вероятностная модель (2) описывает процесс порождения коллекции D по известным распределениям $p(w|t)$ и $p(t|d)$. Построение тематической модели — это обратная задача: по известной коллекции D требуется оценить параметры модели $\phi_{wt} = p(w|t)$ и $\theta_{td} = p(t|d)$.

Обычно число тем $|T|$ много меньше $|D|$ и $|W|$, и задача сводится к поиску приближённого представления заданной матрицы частот $F = (f_{wd})_{WxD}$, $f_{wd} = \frac{n_{dw}}{n_d}$ в виде произведения $F \approx \Phi\Theta$ двух неизвестных матриц меньшего размера — матрицы слов-тегов и тем $\Phi = (\phi_{wt})_{WxT}$ и матрицы тем документов $\Theta = (\theta_{td})_{TxD}$. Все три матрицы F, Φ, Θ являются стохастическими, то есть имеют неотрицательные нормированные столбцы f_d, ϕ_t, θ_d, представляющие дискретные распределения [10].

3.3.3. Адаптивная регуляризация

Задача стохастического матричного разложения является некорректно поставленной, так как множество её решений в общем случае бесконечно. Если $F = \Phi\Theta$ — решение, то $F = (\Phi S)(S^{-1}\Theta)$ также является решением для всех невырожденных S, при которых матрицы $\Phi' = \Phi S$ и $\Theta' = S^{-1}\Theta$ являются стохастическими. Существует общий подход к решению некорректно поставленных обратных задач, называемый регуляризацией. Когда оптимизационная задача недоопределена, к основному критерию добавляют дополнительный критерий — регуляризатор, по возможности учи-

тывающий специфические особенности данной задачи и знания предметной области [10].

Аддитивная регуляризация тематических моделей (APTM) основана на введении дополнительных критериев-регуляризаторов $R_i(\Phi, \Theta)$, $i = 1, \ldots, r$, и максимизации их линейной комбинации с логарифмом правдоподобия $L(\Phi, \Theta)$:

$$\begin{aligned} &R(\Phi, \Theta) = \sum_{i=1}^{k} r_i R_i(\Phi, \Theta), \\ &L(\Phi, \Theta) \to max_{\Phi,\Theta}, \\ &\sum_{w \in W} \phi_{wt} \in \{0, 1\}, \phi_{wt} \geq 0, \\ &\sum_{t \in T} \theta_{td} \in \{0, 1\}, \theta_{td} \geq 0, \end{aligned} \tag{3}$$

где r_i — неотрицательные коэффициенты регуляризации. Оптимизация взвешенной суммы критериев является широко распространённым приёмом в многокритериальной оптимизации. Преобразование вектора критериев в один скалярный критерий называется скаляризацией.

Если $\phi_t = 0$, то тема t исключается из тематической модели. Таким образом, в постановку задачи закладывается возможность определять оптимальное число тем, при условии, что исходно было задано избыточное число тем. Заметим, что к удалению темы t может приводить не только обнуление t-го столбца матрицы Θ, но и обнуление t-й строки матрицы Θ.

Дивергенция Кульбака–Лейблера или KL-дивергенция активно используется в ARTM при построении регуляризаторов. KL-дивергенция — это несимметричная функция расстояния между дискретными распределениями $P = (p_i)_{i=1}^n$ и $Q = (q_i)_{i=1}^n$:

$$KL(P||Q) \equiv KL_i(p_i||q_i) = \sum_{i=1}^{n} p_i \ln(\frac{p_i}{q_i}) \tag{4}$$

Если P — эмпирическое распределение, а $Q(\alpha)$ — параметрическое семейство (модель) распределений, то минимизация KL-дивергенции эквивалентна максимизации правдоподобия:

$$KL(P \,||\, Q(\alpha)) = \sum_{i=1}^{n} p_i \ln(\frac{p_i}{q_i(\alpha)}) \to min_\alpha \iff \sum_{i=1}^{n} p_i \ln(q_i(\alpha)) \to max_\alpha \tag{5}$$

Максимизация правдоподобия (3) эквивалентна минимизации взвешенной суммы дивергенций Кульбака–Лейблера между эмпирическими распределениями $\hat{p}(w|d) = \frac{ndw}{nd}$ и модельными $p(w|d)$, по всем документам d из D:

$$\sum_{d \in D} n_d KL_w(\frac{n_{dw}}{n_d} || \sum_{t \in T} \phi_{wt}\theta_{td}) \to min_{\Phi,\Theta} \tag{6}$$

где весом документа d является его длина n_d. Если веса n_d убрать, то все документы будут искусственно приведены к одинаковой длине. Такая модификация функционала

качества может быть полезна при моделировании коллекций, содержащих документы одинаковой важности, но существенно разной длины.

Весь описанный выше функционал тематического моделирования, реализован в библиотеке BigARTM, а также в этой библиотеке существует возможность добавления регуляризаторов (путем изменения ее исходного кода и перекомпилирования). После обучения тематической модели на данных Stackoverflow, используем матрицу Ф для получения наиболее вероятной темы для каждого тега. Таким образом, каждому тегу было сопоставлен некоторый кластер, в котором все теги имеют достаточно большую вероятность появления вместе в постах Stackoverflow.

3.3.4. Регуляризатор на основе графа тегов

Поскольку при построении графа тегов, была построена мера их схожести (обозначим ее за $S(w_i, w_j)$), используем ее в качестве дополнительного регуляризатора тематической модели. А именно потребуем, чтобы разница между столбцами матрицы Ф для похожих тегов стремилась к минимуму:

$$\begin{array}{l} \sum_{i,j} JSD(p(t|w_i) \,||\, p(t|w_j))S(w_i, w_j) \to min_{\Phi},\ p(t|w_i) = \phi_{w_i t}, \\ JSD(P \,||\, Q) = \frac{1}{2}KL(P \,||\, M) + \frac{1}{2}KL(Q \,||\, M), M = \frac{1}{2}(P + Q) \end{array} \tag{7}$$

где JSD – дивергенция Дженсона-Шеннона.

В следствии чего определим регуляризатор как:

$$R(\Phi, \Theta) = \sum_{t \in T} \sum_{w \in W} S(w_i, w_j)(\phi_{w_i t} \ln(\phi_{w_j t}) + \phi_{w_j t} \ln(\phi_{w_i t})), \tag{8}$$

Данный регуляризатор был реализован с использованием библиотеки BigARTM, а коэффициент регуляризации определялся исходя из метрики NMI получаемых результатов. Таким образом получим комбинацию обоих рассмотренных выше подходов к кластеризации тегов.

3.4. Результаты

Описанные выше алгоритмы кластеризации были реализованы, с применением библиотек BigARTM, igraph, MCL, clv, kernlab и других.

Как видно из таблицы (2), лучшие результаты показал алгоритм BigARTM + RegSim. При этом оценка модулярности и DBI для тематических моделей не производилась, так как эти критерии требуют описания кластеризуемых элементов в виде векторов, либо в виде вершин графа.

Алгоритм	Модулярность	DBI	NMI	ARI
Fast Greedy	0.05	1.97	0.28	0.29
Spectral clustering	0.17	2.4	0.35	0.31
Markov Clustering	0.27	2.46	0.73	0.54
BigARTM			0.32	0.18
BigARTM + RegSim			0.86	0.72

Таблица 2: Пример кластеризации тегов

4. Кластеризация пользователей

После получения кластеризации тегов, для каждого пользователя можно определить, в какой кластер тегов он чаще писал посты:

1. Получим набор кластеров тегов в которые пишет пользователь
2. Выберем самый часто используемый им кластер в качестве основного
3. Определять кластер пользователя можно во временном окне по его постам

Таким образом получим набор временных рядов для каждого кластера тегов. При этом остается возможность наличия пользователей, которые активно пишут в несколько кластеров, но при этом учтен окажется только наиболее популярный из них. Это ограничение текущего подхода может быть преодолено за счет введения порога минимальной активности пользователя в кластере тегов, однако такой подход в данной работе рассматриваться не будет. Кроме того, частично эта проблема решается за счет уменьшения временного окна, в рамках которого будет выбираться основной кластер пользователя.

По итогу обработки результатов кластеризации тегов методом BigARTM + RegSim были получены следующие тренды сообществ:

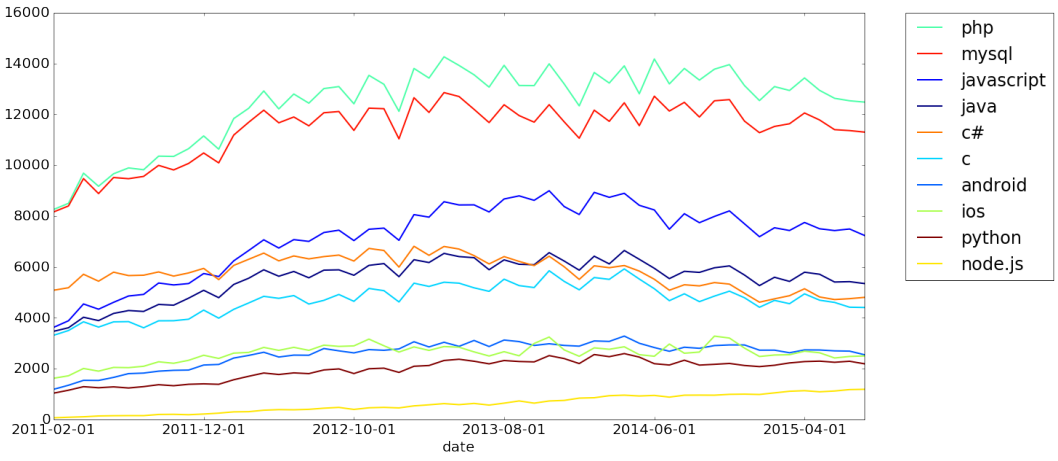

Рис. 6: Примеры поста и тега из Stackoverflow

Заключение

В результате проделанной работы было разработано программное решение, которое для поданной на вход коллекции постов описанного выше вида, строит кластеризацию использованных тегов и временные ряды с размерами сообществ пользователей, использующих соответствующие кластеры тегов. В ходе работы были решены следующие задачи:

1. Были рассмотрены существующие подходы и методы кластеризации тегов.
2. Оценена эффективность выбранных методов, в результате которой лучшим алгоритмом кластеризации графов оказался Марковский алгоритм.
3. Предложен и опробован комбинированный подход (BigARTM + RegSim), который дал лучшие результаты при оценке NMI для заранее заданной базовой кластеризации.
4. Выявлены сообщества для каждого из получившихся кластеров тегов.

В настоящий момент ведется работа по доработке программной архитектуры описанного решения.

Список литературы

[1] Antenucci Dolan. Classification of tweets via clustering of hashtags. — 2011. — URL: http://www.twiki.di.uniroma1.it/pub/ApprAuto/WebHome/AntenucciHandyModiTinkerhess.pdf (online; accessed: 04.06.2017).

[2] Fatemeh Riahi Zainab Zolaktaf Mahdi Shafiei. Finding Expert Users in Community Question Answering. — 2012. — URL: http://www.cs.ubc.ca/~zolaktaf/2012-Riahi-WWW.pdf (online; accessed: 04.06.2017).

[3] Fortunato Santo. Community detection in graphs. — 2010. — URL: https://arxiv.org/pdf/0906.0612.pdf (online; accessed: 04.06.2017).

[4] Gajduk Andrej. Intelligent tag groupping by using an aglomerative clustering algorithm. — 2013. — URL: http://ciit.finki.ukim.mk/data/papers/10CiiT/10CiiT-20.pdf (online; accessed: 04.06.2017).

[5] Gediminas Adomavicius Jesse C. Bockstedt Alok Gupta, Kauffman Robert J. Making Sense of Technology Trends in the Information Technology Landscape: A Design Science Approach. — Dec., 2008. — Vol. 32. — P. 779–809.

[6] Kumar Lov, Sureka Ashish. Thirteen Years of Mining Software Repositories (MSR) Conference - What is the Bibliography Data Telling Us?

[7] Muntean Cristina Ioana. Exploring the Meaning behind Twitter Hashtags through Clustering. — URL: http://hpc.isti.cnr.it/~muntean/papers/twitter-hashtags.pdf (online; accessed: 04.06.2017).

[8] Sparrows and Owls: Characterisation of Expert Behaviour in StackOverflow. — 2014. — URL: http://yangjiera.github.io/works/umap2014_experts.pdf (online; accessed: 04.06.2017).

[9] Vincent D. Blondel Jean-Loup Guillaume Renaud Lambiotte, Lefebvre Etienne. Fast unfolding of communities in large networks. — 2008. — URL: https://arxiv.org/pdf/0803.0476.pdf (online; accessed: 04.06.2017).

[10] В. Воронцов К. Тематическое моделирование в BigARTM: теория, алгоритмы, приложения. — 2015. — URL: http://www.machinelearning.ru/wiki/images/b/bc/Voron-2015-BigARTM.pdf (online; accessed: 04.06.2017).