УДК 004 **Информационные технологии**

Шагиахмитов Данис Радикович, *магистр, Казанский (Приволжский) федеральный университет, г. Казань*

МЕТОДЫ ИЗВЛЕЧЕНИЯ ПРИЗНАКОВ ИЗ ТЕКСТОВЫХ ДОКУМЕНТОВ

Аннотация: Выделение признаков из текстовых документов является одной из задач информационного поиска. Полученные из текста признаки в дальнейшем используются при поиске и ранжировании документов из набора признаков строится поисковый индекс корпуса тестовых документов.

Для выделения признаков используются как классические алгоритмы, так и подходы из машинного обучения. В статье приведен обзор алгоритмов извлечения признаков из текстовых документов.

Ключевые слова: информационный поиск, текстовые документы, признаки, векторная модель.

Annotation: Selecting features from text documents is one of the tasks of information search. The attributes obtained from the text are then used for searching and ranking documents.the search index of the test document corpus is built from the set of attributes.

Both classical algorithms and machine learning approaches are used to identify features. The article provides an overview of algorithms for extracting features from text documents.

Keywords: information search, text documents, features, vector model.

1. Введение

Извлечение признаков из текстовых документов необходимо для того чтобы сопоставить каждому документу некий набор характеристик, которые описывают этот документ [1]. Пара текстовый документ, набор характеристик позволяет оценить насколько документы похожи друг на друга, к какой тематике принадлежит документ, выполнить поиск документа по текстовому запросу пользователя если рассматривать запрос как очень короткий текстовый документ.

Разделим текст на отдельные слова и подсчитаем количество вхождений каждого слова в документ. Получим такую структуру:

"министерство" : 25

"в" : 14

"финансов" : 7

"рф" : 4

Такой подход имеет проблемы, в выборку попадают очень частотные слова, которые не несут никакой полезной информации о содержании документа, например: предлоги, суффиксы, причастия, междометия, цифры, частицы. Такие слова называют шумовыми словами или стоп-словами и удаляют из общей выборки так как они не несут полезной информации. Вторая проблема при таком подходе что слова не находятся в своей нормальной форме, содержат окончания и при подсчете количества не попадают в одну группу. Слова перед подсчетом нужно нормализировать, например, стеммировать его. Стеммирование это процесс нахождения основы слова, основа слова не обязательно совпадает с морфологическим корнем слова. Слова, прошедшие через процесс стеммирования называются термами. Также вместо стеммирования можно использовать более сложный механизм лемматизации. Применим эти приемы и еще раз построим выборку.

"министр" : 23

"финанс" : 10

"рф" : 5

"правительств" : 4

Выборка стала более содержательной, такой подход называется Bag-of-Terms – мешок термов.

2. TF-IDF

Присвоим каждому терму, обнаруженному в документе, вес зависящий от количества появлений этого терма в данному документе. Такая схема взвешивания называется частотой терма (term frequency) и обозначается как $tf_{t,d}$,

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

где t обозначает терм, d — документ, n_t есть число вхождений слова t в документ, а в знаменателе — общее число слов в данном документе. В рамках такой модели подсчета веса порядок следования термов в документе игнорируется, а основное значение придается количеству вхождений каждого терма. Мы предполагаем, что два документа с одинаковым наборам термов с одинаковыми весами схожи между собой.

Подсчет частоты терма описанный выше имеет серьезный недостаток. При ранжировании веса всех термов считаются одинаково важными. Но на самом деле некоторые термы имеют малую или нулевую различительную силу при определении релевантности. Например, коллекция документов об автомобильной промышленности, скорее всего, содержит терм "авто" практически в каждом документе. Для того чтобы устранить указанный недостаток, нужно ввести механизм ослабления влияния терма, который встречается в коллекции слишком часто.

Решение состоит в том, чтобы уменьшить вес терма tf на некий коэффициент, который увеличивается по мере увеличения его частоты в коллекции. Для этого используется еще один параметр документная частота. Документная частота — это число документов, в которых встречается данный терм. Используем документную

частоту для коррекции веса tf. Для этого определим еще один параметр: обратная документная частота терма t. Параметр определяется по формуле:

$$idf = log\frac{N}{df},$$

где N — количество документов в коллекции, df — документная частота терма.

Обратная документная частота редко встречающегося терма является большой, в то время как для часто встречающегося термина она невелика.

Комбинируя частоту терма (tf) и его обратную документную частоту можно вычислить вес для каждого терма в документе. Модель взвешивания *tf-idf* вычисляет вес каждого терма по формуле [2]:

$$tf - idf = tf_{t,d} \times idf_t$$

Вес вычисленный по модели $tf - idf$ имеет следующие свойства:

1. Вес достигает максимального значения, если терм / встречается много раз в небольшом количестве документов.
2. Вес уменьшается, если терм встречается в каком-то документе лишь несколько раз или встречается во многих документах.
3. Вес достигает минимального значения, если термин встречается практически во всех документах.

3. Функция BM25

Функция BM25 представляет собой современную модификацию *tf-idf* подобной функции ранжирования. BM25 это не одна функция, а семейство функций с различными компонентами и параметрами. Одна из распространенных форм этой функции описана ниже.

Пусть дан запрос Q, содержащий слова $q_1, \ldots\ldots, q_n$, тогда функция BM25 даёт следующую оценку релевантности документа D запросу Q:

$$score(D,Q) = \sum_{i=1}^{n} idf(q_i) * \frac{f(q_i,D) * (k_1+1)}{f(q_i,D) + k_1 * (1 - b + b * \frac{|D|}{avgdl})}$$

$idf(q_i, D)$ — обратная частота слова q_i в документе D

$f(q_i, D)$ частота слова q_i в документе D,

$|D|$ - количество слов в документе,

$avgdl$ - средняя длина документа в коллекции, k_1и b свободные коэффициенты обычно выбирают как $k_1 = 2$ и $b = 0.75$ и «подгоняют» их значения под корпус документов по которому идет поиск.

Существует много вариантов функции BM25, например, BM25F которая рассчитывает релевантность отдельно для определенных зон документа или BM25L которая обеспечивают лучшие результаты на очень длинных документах [3].

Данная функция используется во многих системах полнотекстового поиска например Sphinx или Apache Lucene.

4. Word embeddings(Word2Vec, FastText)

Word embeddings — это общее название для подходов в обработке естественного языка при котором каждому слову из словаря сопоставляется вектор значений. Такие векторы строятся на основе теоретической основы из дистрибутивной семантики. Рассмотрим несколько моделей использующих такой подход.

Word2vec — инструмент, используемый для анализа семантики естественных языков, основанный на дистрибутивной семантике, машинном обучении и векторном представлении слов.

Word2vec принимает на вход большой корпус текстов, а на выходе сопоставляет каждому слову вектор. Word2vec вычисляет векторное представление слов, «обучаясь» на входных текстах [4]. Обучая нейронные сети, Word2Vec максимизирует косинусную меру близости между векторами слов, которые встречаются в похожих контекстах и минимизирует косинусную меру между словами которые не встречаются.

$$similarity = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B)^2}}$$

где A и B — два вектора, расстояние между которыми вычисляется.

$$P(\omega_o|\omega_c) = \frac{e^{s(\omega_o,\omega_c)}}{\sum_{\omega_i \epsilon V} e^{s(\omega_i,\omega_c)}}$$

где ω_o — вектор целевого слова, ω_c — это некий вектор контекста вычисленный путем усреднения из векторов слов окружающих нужное слово, $s(\omega_1, \omega_2)$ — это функция, которая сопоставляет двум векторам одно число, например, это может быть упоминавшееся выше косинусное расстояние.

Полученные векторные представления слов могут быть использованы для решения практических задач, например, нахождения близкого по смыслу слова по отношению к изучаемому слову, исправление опечаток, оценка важности слов в запросе, кластеризация текстов и т.д.

В word2vec существует две модели обучения первая Continuous Bag of Words (CBOW) в данном подходе нейронная сеть предсказывает исходное слово по его контексту. Нейронная сеть обучается, используя стохастический градиентный спуск через метод обратного распространения ошибки.

Вторая модель обучения Skip-gram в таком подходе по центральному слову предсказывается окружающий его контекст. Как указывает своей статье автор Word2Vec Т. Миколов модель Continuous Bag of Words (CBOW) в несколько раз быстрее обучается чем Skip-gram.

Модель CBOW больше подходит для коротких корпусов, а модель Skip-gram для объемных корпусов текстов.

FastText — это способ построения векторного представления в которой также используются модели CBOW и Skip-gram. Основным недостатоком word2vec является то, что для слов, не встречающиеся в обучающей выборке невозможно вычислить вектор. FastText решает эту проблему с помощью N-грамм символов [5]. Алгоритм проходит скользящим окном по корпусу текстов, но рассматривает не

слова, а символьные n-граммы. Например, 3-граммами для слова яблоко являются ябл, бло, лок, око. Пусть имеется словарь $G_w \subset \{1, \dots, G\}$ n-грамм, связываем векторное представления z_g с каждым n-граммом g. И представляем слово как сумму векторных представлений его n-грамм. Тогда близость двух слов в FastText определяется как

$$s(w, c) = \sum_{g \in G_w} z_g^T \, v_c$$

FastText использует подход hashing trick используя при расчетах хеши n-граммов, это сильно ускоряет вычисления, fastText работает быстрей чем word2vec [6].

Библиографический список:

1. Маннинг К. Введение в информационный поиск / К. Маннинг, П. Рагхаван, Х.М. Шютце. — М.:«Вильямс», 2011. — 528 с.

2. NLPX Tales of Data Science, TF-IDF с примерам кода [Электронный ресурс] http://nlpx.net/archives/57 (Дата обращения: 31.03.2020).

3. Yuanhua Lv, ChengXiang Zhai When documents are very long, BM25 fails, Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2011, doi: 10.1145/2009916.2010070.

4. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space – [Электронный ресурс] In Proceedings of Workshop at International Conference on Learning Representations (ICLP) – 2013, URL: http://arxiv.org/abs/1301.3781 (дата обращения: 21.03.2020).

5. Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, Bag of Tricks for Efficient Text Classification [Электронный ресурс] https://arxiv.org/abs/1607.01759 (дата обращения: 21.03.2020).

6. Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov. Enriching Word Vectors with Subword Infromation [Электронный ресурс] https://arxiv.org/abs/1607.04606 (дата обращения: 21.03.2020).