Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Санкт-Петербургский государственный университет»

Кафедра системного программирования

Кузьмина Илия Викторовна

# Рекомендательные системы на основе профиля интересов пользователей

Курсовая работа

Научный руководитель:
Николенко С. И.

Санкт-Петербург
2016

# Оглавление

# Введение

Количество информации, публикуемой в социальных сетях, постоянно растёт, что делает невозможной ручную обработку документов, так как её скорость существенно ограничена. В связи с этим, требуется реализация алгоритмов, которые будут автоматически обрабатывать имеющиеся объёмы текстов.

Одной из основных задач для социальных сетей является построение профиля интересов пользователя. В данном случае, профиль интересов определяется как некоторый набор тем, для каждой из которых указана вероятность быть интересной конкретному пользователю. Каждая тема представляет собой множество слов, используемых в рассматриваемых текстах.

На основе профиля интересов пользователей может быть реализована рекомендательная система – алгоритм, прогнозирующий положительные оценки пользователя на непросмотренные им ранее публикации в социальной сети. Кроме того, профиль интересов может быть использован для определения релевантных рекламных объявлений.

Большинство известных методов определения тем в текстах основываются на построении и анализе тематических моделей. Этот процесс называется тематическим моделированием (topic modeling[6]).

Множество текстовых документов, которые используются для построения тематической модели, обычно называют корпусом. В данной работе в качестве корпуса использовался набор публикаций в социальной сети "Одноклассники", а также lib.rus.ec и русскоязычный фрагмент "Википедии".

# 1. Цели и постановка задач

Целью данной работы является определение интересующих каждого пользователя тем и создание рекомендательной системы, основывающейся на методах вероятностного тематического моделирования Для достижения поставленной цели необходимо решение следующих задач:

1. Изучение предметной области тематического моделирования
2. Реализация алгоритма построения профиля интересов пользователя
3. Реализация рекомендательного алгоритма
4. Оценка качества рекомендательного алгоритма

# 2. Тематическое моделирование

## 2.1. Векторное представление слов

Построение векторной модели является важным инструментом обработки естественного языка (natural language processing[**?**]). Суть распределённого представления слов (Distributed word representation) – отобразить множество слов из заданного словаря в семантическое пространство, размерность которого может варьироваться для достижения наилучших результатов. В данной работе использовалась модель Word2Vec[3], размерность была выбрана равной 200.

## 2.2. Основные гипотезы тематического моделирования

- Порядок слов в текстовом документе неважен (bag of words[4])
- Все слова можно привести к начальной форме

# 3. Модель Vec2Topic

## 3.1. Назначение

Основная идея Vec2Topic[2]- выделить из темы, представляющей собой набор слов, несколько слов, которые лучше всего её характеризуют (topical words), далее эти слова будут называться "особыми". Предполагается, что особые слова, во-первых, встречаются в контексте многих других слов, поэтому одна из характеристик слова, именуемой степенью (degree), - количество уникальных слов, которые встречаются с данным в одном предложении. Вторая важная характеристика - глубина слова (depth), о которой будет рассказано в следующем разделе.

## 3.2. Иерархическая кластеризация [5]

Имея векторную модель представления слов, можно кластеризовать слова-векторы в 200-мерном семантическом пространстве. Теперь понятия "кластер" и "тема" будут отождествляться. В данной работе использовался стандартный алгоритм кластеризации Mini-Batch K-means. Иерархическая кластеризация строит дерево кластеров (Рис. 1) следующим образом: на каждом шаге находятся два самых близких к друг другу кластера и объединяются их в один. Если два кластера с номерами x и y сливаются и образуют кластер с номером z, то из вершины z проводятся два ребра: в вершины x и y. Таким образом, в результате все исходные кластеры будут объединены в один, соответствующий корню дерева кластеров. В полученном дереве посчитаем глубины каждого листа, соответствующего какому-то из исходных кластеров. Обозначим за глубину слова (depth) глубину кластера, в котором оно лежит.

## 3.3. Мера слов и тем

Как было сказано выше, мера слова зависит от двух характеристик: степени и глубины, и вычисляется по следующей формуле:

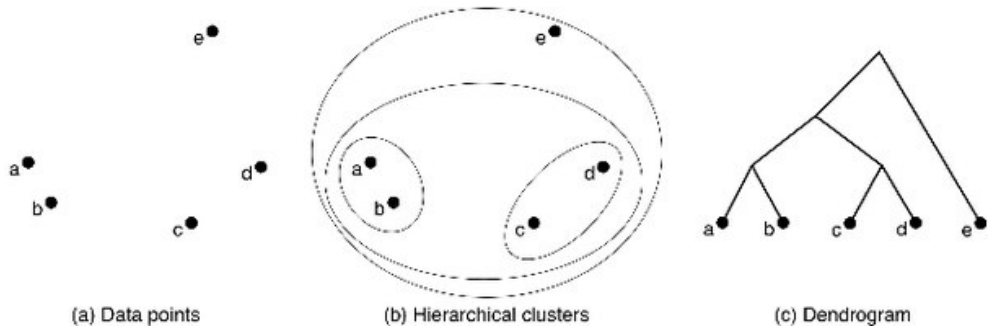


Рис. 1: Иерархическая кластеризация

$$Score(v) = \frac{depth(v)}{\max_{u \in V} depth(u)}$$

Мера темы (кластера) вычисляется как среднее мер по всем словам, входящим в данный кластер.

$$Score(t) = \frac{\sum_{v \in t} score(v)}{|t|}$$

# 4. Данные

Исходные данные можно разделить на две категории:

1. Данные о публикациях. Они хранились в 200 файлах размером 45-50 Мб и представлялись в формате

   **group_id author_id post_id timestamp_iso text**

2. Данные о проявлениях положительного интереса пользователей к публикациям, которые хранились в одном файле формата .csv размером 12 Гб в следующем виде:

   **(group_id, post_id) {(user_id, timestamp)...}**;

где

- group_id – уникальный идентификатор группы
- author_id – идентификатор автора поста (может быть равным 0, если автор неизвестен)
- post_id – идентификатор публикации
- timestamp_iso – время публикации в формате iso
- user_id – уникальный идентификатор пользователя
- timestamp – количество миллисекунд, прошедших с 1 января 1970 года, до того момента, когда пользователь выразил позитивную оценку к данной публикации

Так как часть данных была излишней, а также с ними было неудобно работать, они были переструктурированы. Теперь для каждого пользователя известен список понравившихся ему публикаций, и для каждого сообщества был создан отдельный файл, в котором тексты были отсортированы по дате публикации. Ниже представлен пример, иллюстрирующий новое структурирование данных (Рис. 2)

| user_id | (group_id, post_id) |
|---|---|
| 184549376: | (250953, 674035) (282748, 4555578) (214341, 504639) |
| 144703491: | (52750, 10961512) (242141, 11520519) |

Рис. 2: Пример переформатированных данных

Для реализации рекомендательного алгоритма желательно знать не только положительные оценки публикаций, но и отрицательные, для каждого пользователя был составлен список предположительно негативных оценок: публикация считается неинтересной пользователю, если она была размещена в сообществе непосредственно перед какой-либо интересной этому пользователю публикацией (и сама таковой не является). Формат данных о предположительно неинтересных пользователю публикациях аналогичен тому, который используется для хранения списка интересных публикаций.

Также в данной работе использовалась уже построенная модель Word2Vec, в которой содержалось 3232617 различных слов. После приведения слов к нормальной форме в модели осталось 953946 слов.

# 5. Рекомендательная система

## 5.1. Построение модели

953946 векторов для слов, оставшихся после приведения к начальной форме, были разделены на 2000 кластеров. Для кластеризации использовался стандартный алгоритм Mini Batch K-means[8], так как из всех известных алгоритмов кластеризации только он рассчитан на такое количество точек. Алгоритм имеет сложность $O(n * K * I * f)$[1], где n – количество точек (векторов) в пространстве, K – заранее заданное количество кластеров (в данном случае – 2000), I – количество итераций, f – размерность векторов (200). Далее, для каждого слова нужно было посчитать его меру (score).

1. **Глубина слова.** Так как в стандартных библиотеках алгоритмы иерархической кластеризации не возвращают дерево кластеров, а результатом их работы является набор меньшего количества кластеров, полученных слиянием поданных на вход кластеров, необходимо было самостоятельно реализовать алгоритм иерархической кластеризации с сохранением построенного дерева.

2. **Степень слова.** Для вычисления степени необходимо посчитать количество уникальных слов, которые встречаются с данным в одном предложении. Невозможно хранить все пары слов, встречающихся в одном предложении, так как в этом случае объём занимаемой памяти намного превысит объём оперативной памяти. Эта проблема была решена следующим образом:

    - Вначале все тексты публикаций, сгруппированные в 200 файлах, были отнормированы и очищены от знаков препинания, каждое предложение размещено в отдельной строке.
    - Далее, для каждого файла в отдельности были посчитаны все пары слов, которые встречаются внутри одного предложения, и записаны в файл в лексикографическом порядке.

- В результате, было создано 200 файлов с отсортированными парами слов. Количество уникальных пар слов было посчитано с помощью аналога метода двух указателей, только вместо двух их было 200. Отсюда были найдены степени всех слов.
- Так как обработка разных текстовых файлов независима, процесс вычисления пар слов из одного предложения был распараллелен.

При известных мерах слов были посчитаны меры тем, как среднее арифметическое мер всех входящих в него слов. Утверждается, что чем больше мера слова, тем больше оно характеризует тему, к которой относится.

## 5.2. Построение профиля интересов пользователя

В публикациях, интересующих конкретного пользователя, может быть намного меньше слов, чем в словаре модели Word2Vec, и распределение интересов по темам будет отличаться от распределения по всем текстам в целом, которое было построено на предыдущем шаге. Для каждого пользователя были оставлены только те темы, слова из которых встречаются в интересных ему публикациях, и для каждой из такой тем была построена мера:

$$Score^{u}_{dislike}(t) = \frac{\sum_{v \in t, v \in d(u)} score(v)}{\#(v \in t, v \in d_{like}(u))}$$

, где $d_{like}(u)$ - публикации, которым пользователь $u$ выразил положительную оценку.

На рисунке приведены примеры построенных профилей пользователей (Рис. 3)

## 5.3. Рекомендательный алгоритм

Профиль интересов для каждого пользователя уже построен. Аналогично, можно построить профиль слов и тем, предположительно неин-

- **Topic #10: 0.57949**
  торт глазурь крем тесто корж противень
  **Topic #6: 0.54294**
  молоко яйцо мука сахара ложка сметана желатина какао-порошок масло приготовление столовый сахар крахмал ванильный маргарин лимонный ванилин заварной ст щепотка
- **Topic #21: 0.69140**
  пензенский пенза алексий азовский пресвятой
  **Topic #16: 0.67656**
  богоматерь общецерковный святой божий матерь чудотворный
- **Topic #6: 0.43481**
  арбидол биопарокс композитум неврохель валерианохель траумель канефрон лимфомиозот
  **Topic #21: 0.42425**
  лекарственный зверобой экстракт боярышник пассифлора страстоцвет бузина

Рис. 3: Примеры профилей интересов пользователей

тересных пользователю, по словам из документов $d_{dislike}(u)$. О том, как получить список этих документов, было рассказано в разделе 4.

Теперь, по содержанию новой публикации ($p$), ещё непросмотренной пользователем $u$, нужно определить, будет ли она ему интересна. Для этого посчитаем две величины:

$$P(like|u) = \sum_t score^u_{like} * W(t)$$

$$P(dislike|u) = \sum_t score^u_{dislike} * W(t)$$

, где $W(t)$ – количество слов в тексте публикации $p$, относящихся к теме t.

# 6. Инструменты

Основным языком разработки был выбран Python3, так как для него создано большое количество библиотек машинного обучения, а также на нём удобно обрабатывать текстовые данные без возникающих в предыдущей версии языка трудностей с различными кодировками. В работе использовались следующие библиотеки:

- Pymorphy2[10] – для приведения слов к начальной форме и определения частей речи.
- Gensim[7] – для загрузки модели Word2Vec из бинарного файла.
- Scikit-learn[9] – для кластеризации векторов-слов.

Алгоритм, вычисляющий степень всех слов, был написан на C++, в целях уменьшения времени исполнения.

# 7. Оценка результатов

Оценить качество построения профиля интересов пользователей не представляется возможным иным способом, кроме как через оценку рекомендательного алгоритма. Для этого были выбраны 2000 пользователей, которые положительно оценили не менее 20 публикаций, и для которых список предположительно неинтересных публикаций имеет длину не менее 20. Эти публикации были разделены на две части - тренировочный набор данных и тестовый набор данных. Тренировочный набор составлял 80% от всех имеющихся публикаций каждого типа. Для оценки точности рекомендательного алгоритма были посчитаны величины precision и recall, вычисляющиеся по формулам:

$$precision = \frac{tp}{tp + fp}$$

$$recall = \frac{tp}{tp + fn}$$

, где $tp$ – количество истинно-положительных прогнозов, $fp$ – количество ложно-положительных прогнозов и $fn$ – количество ложно-отрицательных прогнозов.

Эти величины получились равными 0.68 и 0.74 соответственно.

# 8. Результаты

В ходе работы для достижения поставленной цели были решены следующие задачи:

1. Обработаны пользовательские данные о публикациях и положительных оценках
2. Изучены современные методы построения тематических моделей
3. Построена тематическая модель Vec2Topic (задача включает в себя построение дерева кластеров и вычисление степеней слов)
4. Реализован рекомендательный алгоритм

# 9. Благодарности

Хочу выразить благодарность руководившему моей работой в рамках менторской программы компании EMC Галинскому Руслану Борисовичу за оказанную им помощь в решении возникавших технических трудностей, связанных с реализацией алгоритмов для больших объёмов данных, в освоении нового языка программирования и новой предметной области.

# Список литературы


[1] Agrawal Siddharth. MACHINE LEARNING – MINI BATCH K-MEANS. — 2013. — URL: https://algorithmicthoughts.wordpress.com/2013/07/26/machine-learning-mini-batch-k-means (online; accessed: 26.07.2013).

[2] Ramandeep S. Randhawa Parag Jain Gagan Madan. Topic Modeling Using Distributed Word Embeddings. — 2016. — URL: ''http://arxiv.org/pdf/1603.04747v1.pdf'' (online; accessed: 15.03.2016,).

[3] Tomas Mikolov Ilya Sutskever Kai Chen Greg Corrado Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. — 2013.

[4] Wikipedia. Bag-of-words model // Wikipedia, the free encyclopedia. — 2016. — URL: https://en.wikipedia.org/wiki/Bag-of-words_model (online; accessed: 16.04.2016).

[5] Wikipedia. Hierarchical clustering // Wikipedia, the free encyclopedia. — 2016. — URL: https://en.wikipedia.org/wiki/Hierarchical_clustering (online; accessed: 17.05.2016).

[6] Wikipedia. Topic model // Wikipedia, the free encyclopedia. — 2016. — URL: https://en.wikipedia.org/wiki/Topic_model (online; accessed: 03.05.2016).

[7] models.word2vec – Deep learning with word2vec, 2016. — URL: http://radimrehurek.com/gensim/models/word2vec.html (online; accessed: 26.02.2016).

[8] scikit learn. — Mini Batch K-means, 2014. — URL: http://scikit-learn.org/stable/modules/clustering.html#mini-batch-k-means (online; accessed: 05.03.2016).

[9] scikit-learn – Machine Learning in Python, 2013. — URL: http://scikit-learn.org/stable/ (online; accessed: 02.03.2016).

[10] Морфологический анализатор pymorphy2, 2013. — URL: http://pymorphy2.readthedocs.io/en/0.2/index.html (дата обращения: 02.12.2015).