Прикладные задачи анализа данных

Семинар 4

Ключевые слова, n-граммы, коллокации

Национальныи Исследовательский Университет
Высшая Школа Экономики

8 февраля 2018

С чем будем работать

28 резонансных событий 2017 года
100 текстов на каждое событие[1]

0. Власти Петербурга согласились передать РПЦ Исаакиевский собор.
1. Дональд Трамп вступил в должность президента США.
2. Скоропостижно скончался постпред России при ООН Виталий Чуркин.
3. Вышел фильм Навального «Он Вам не димон»
4. CNN показала фильм «Владимир Путин — самый влиятельный человек в мире».
5. Умер Дэвид Рокфеллер

...

27. Единый день голосования

[1] ссылка на папку с данными

С чем будем работать

Для начала будем искать ключевые слова
в 100 текстах на тему
«Дональд Трамп вступил в должность президента США»

Код и данные для следующих трех слайдов: ссылка

Самые частые леммы без стоп-слов

nltk.freqdist:
('трамп', 595),
('президент', 491),
('год', 441),
('который', 428),
('инаугурация', 358),
('сша', 352),
('дональд', 316),
('один', 284),
('россия', 251),
('наш', 223),
('январь', 212)

Томита-парсер + nltk.freqdist

```
S -> Adj<gnc-agr[1]> Noun<gnc-agr[1], rt>;
```

('избранный президент', 87),
('белый дом', 69),
('прямая трансляция', 43),
('наша страна', 31),
('этот год', 31),
('соединенный штат', 28),
('новый президент', 27),
('конституционный суд', 25),
('прошлый год', 23),
('весь мир', 21),
('предвыборная кампания', 21)

Самые частые биграммы

nltk.bigrams + nltk.freqdist
(без POS-тегов, лемматизации и мер ассоциации)

(('дональд', 'трамп'), 165), (('дональда', 'трампа'), 133),
(('президента', 'сша'), 125),
(('в', 'году'), 87), (('в', 'россии'), 68),
(('избранного', 'президента'), 59),
(('инаугурация', 'трампа'), 55),
(('москва', 'января'), 51),
(('по', 'делу'), 50),
(('в', 'должность'), 47),
(('об', 'этом'), 46),
(('в', 'вашингтоне'), 45),
(('президент', 'сша'), 44),
(('и', 'в'), 40), (('млрд', 'руб'), 40), (('во', 'время'), 39),

Поиск разрывных коллокаций

Для пары слов вычисляем ожидаемое расстояние и среднеквадратическое отклонение

$$\overline{d} = \frac{\sum_{i=1}^{n} d_i}{n}$$

$$s^2 = \frac{\sum_{i=1}^{n} (d_i - \overline{d})^2}{n-1}$$

n - число раз, когда два слова встретились
d_i - смещение между словами
$\overline{d}$ - выборочное среднее смещений

Хорошо работает в языках с фикс. порядком слов

Поиск разрывных коллокаций

Пример[3]

Найти среднеквадратическое отклонение смещения слов «играть» и «роль» (необоходима лемматизация):

1. Он играет второстепенную роль в известном спектакле.
2. Они играют ведущую роль в новом исследовании.
3. Она играет очень странную роль в этом деле.

[3] В прежложениях имитирован строгий порядок слов

Примеры

s	$\bar{d}$	Count	Word 1	Word 2
0.43	0.97	11657	New	York
0.48	1.83	24	previous	games
0.15	2.98	46	minus	points
0.49	3.87	131	hundreds	dollars
4.03	0.44	36	editorial	Atlanta
4.03	0.00	78	ring	New
3.96	0.19	119	point	hundredth
3.96	0.29	106	subscribers	by
1.07	1.45	80	strong	support
1.13	2.57	7	powerful	organizations
1.01	2.00	112	Richard	Nixon
1.05	0.00	10	Garrison	said

Чем меньше дисперсия, тем лучше словосочетание

Меры ассоциации биграмм - PMI

Pointwise Mutual Information

$PMI(w_1, w_2) = \log \frac{f(w_1,w_2)N}{f(w_1)f(w_2)}$, где w_i - слова, $f(.)$ - частоты

- «проверяет» независимость появления двух слов в тексте
- значения зависят от размера корпуса
- завышает значимость редких словосочетаний (решение - порог отчесечения по частоте)
- «определяет» предметную область: выделяет терминологические сочетания

---

https://publications.hse.ru/articles/71840267

Меры ассоциации биграмм - t-score

T-Score

$$T\text{-}score(w_1, w_2) = \frac{f(w_1,w_2) - f(w_1) * f(w_2)}{f(w_1,w_2)/N}$$

- является модифицированным ранжированием по частоте
- не преувеличивает значимость редких коллокаций (поэтому в пороге нет необходимости)
- выделяет «общеязыковые» устойчивые сочетания

---

https://publications.hse.ru/articles/71840267

Меры ассоциации биграмм - χ^2, LLR

χ^2

	w_2	not w_2
w_1	O_{11}	O_{12}
not w_1	O_{21}	O_{22}

$O_{11} = f(w_1, w_2)$ – **наблюдаемая** частота
$E_{11} = \frac{O_{11}+O_{12}}{N} \times \frac{O_{11}+O_{21}}{N} \times N$ – **ожидаемая** частота

$$\chi^2 = \sum_{i\in 1,2, j\in 1,2} \frac{(O_{ij}-E_{ij})^2}{E_{ij}} = \frac{N(O_{11}O_{22}-O_{12}O_{21})^2}{(O_{11}+O_{12})(O_{11}+O_{21})(O_{12}+O_{22})(O_{21}+O_{22})}$$

LLR
$\log \frac{L(H_1)}{L(H_2)}$ – log-likelihood ratio

Меры ассоциации биграмм в NLTK

- `bigram_measures.pmi`
- `bigram_measures.student_t`
- `bigram_measures.chi_sq`
- `bigram_measures.likelihood_ratio`

Данные, код, результаты: ссылка

TextRank — приложение алгоритма PageRank к задачам обработки естественного языка:

1. построение графа на основе исходного текста (вершины V - слова, ребра E - связи)
2. вычисление весов вершин по мерам центральности:

PageRank [Brin, Page, 1998]

$$PR(V_i) = (1 - d) + d \times \sum_{V_j \in In(V_i)} \frac{PR(V_j)}{|Out(V_j)|}$$

HITS [Kleinberg, 1999]

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j)$$
$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j)$$

3. извлечение цепочек слов с большим весом

---

https://nlpub.ru/TextRank

Построение графа. Будет выполняться построение взвешенного неориентированного графа в виде $G = (V, E)$, где V — множество слов, E — множество связей между ними.

В качестве V можно принять множество всех уникальных лемм исходного текста. Поскольку большинство терминов являются именными группами [1], то множество слов стоит ограничить только леммами, образованными от имён существительных и прилагательных.

Множество E строится путём последовательного сканирования текста заданным окном из $N \in [2, 10]$ слов. На каждой итерации для пары слов вычисляется величина связи $WC(w_1, w_2)$, обратно зависящая от расстояния между словами:

$$WC(w_1, w_2) = \begin{cases} 1 - \frac{d(w_1, w_2) - 1}{N - 1}, & \text{если } d(w_1, w_2) \in (0, N), \\ 0, & \text{если } d(w_1, w_2) \geqslant N, \end{cases}$$

где w_1 и w_2 — слова, $d(w_1, w_2)$ — расстояние между словами, N — размер окна.

---

http://koost.eveel.ru/science/CSEDays2012.pdf

Ранжирование вершин графа. После генерации графа G необходимо вычислить TextRank — значение стационарного распределения случайного блуждания для каждой вершины $t \in V$ с учётом весов связей [2]:

$$TR(t_i) = (1-d) + d \cdot \sum_{t_j \in In(t_i)} \frac{w_{ji}}{\sum_{t_k \in Out(t_j)} w_{jk}} \cdot TR(t_j),$$

где d — фактор затухания, $In(t)$ — множество вершин, входящих в t, $Out(t)$ — множество вершин, исходящих из t, w_{ij} — вес ребра (t_i, t_j). Для неориентированного графа принято $In(t) \equiv Out(t)$.

http://koost.eveel.ru/science/CSEDays2012.pdf

Алгоритм TextRank

Рис. 1. Граф текста аннотации данной статьи

http://koost.eveel.ru/science/CSEDays2012.pdf

После вычисления TextRank необходимо составить множество C кандидатов в термины из первых T слов из списка вершин, упорядоченных по убыванию значения TextRank. Существует ряд подходов к определению $T \equiv |C|$: принять постоянное значение T, использовать $T = \frac{1}{3}|V|$, и т. д.

Сборка словосочетаний. Необходимо извлечь из текста все последовательности слов, состоящих из элементов множества C. Поскольку $\forall t \in C : \exists! TR(t) \in \mathbb{R}$, то общий вес извлечённой последовательности определяется суммой весов всех составляющих её слов.

При сборке словосочетаний стоит учитывать два момента: 1) последовательность обязана иметь в составе хотя бы одно имя существительное; 2) при обнаружении вложенности одной последовательности в другую рассматривается только последовательность с бóльшим весом.

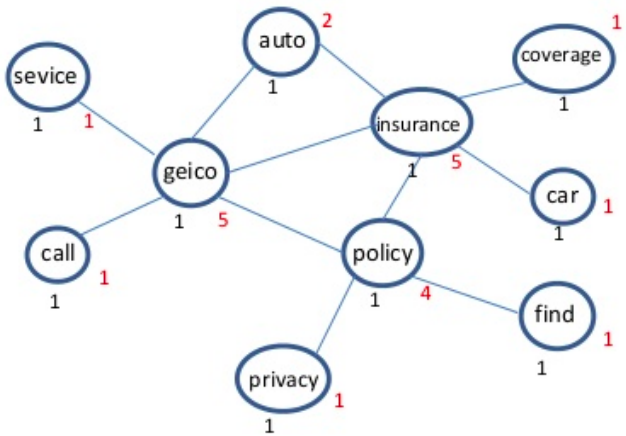
---

http://koost.eveel.ru/science/CSEDays2012.pdf

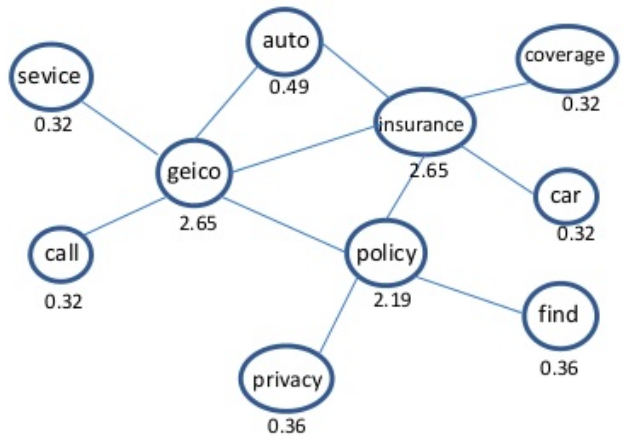
Пример ранжирования вершин

$$S(V_i) = (1-d) + d * \sum_{j \in ngbr(V_i)} \frac{1}{|degree(V_j)|} S(V_j)$$

d is the damping factor that usually set to 0.85



first iteration



iterations

$$Score(geico) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{1} * 1}_{\text{service}} + \underbrace{\frac{1}{1} * 1}_{\text{call}} + \underbrace{\frac{1}{2} * 1}_{\text{auto}} + \underbrace{\frac{1}{5} * 1}_{\text{insurance}} + \underbrace{\frac{1}{4} * 1}_{\text{policy}}\right) = 2.65$$

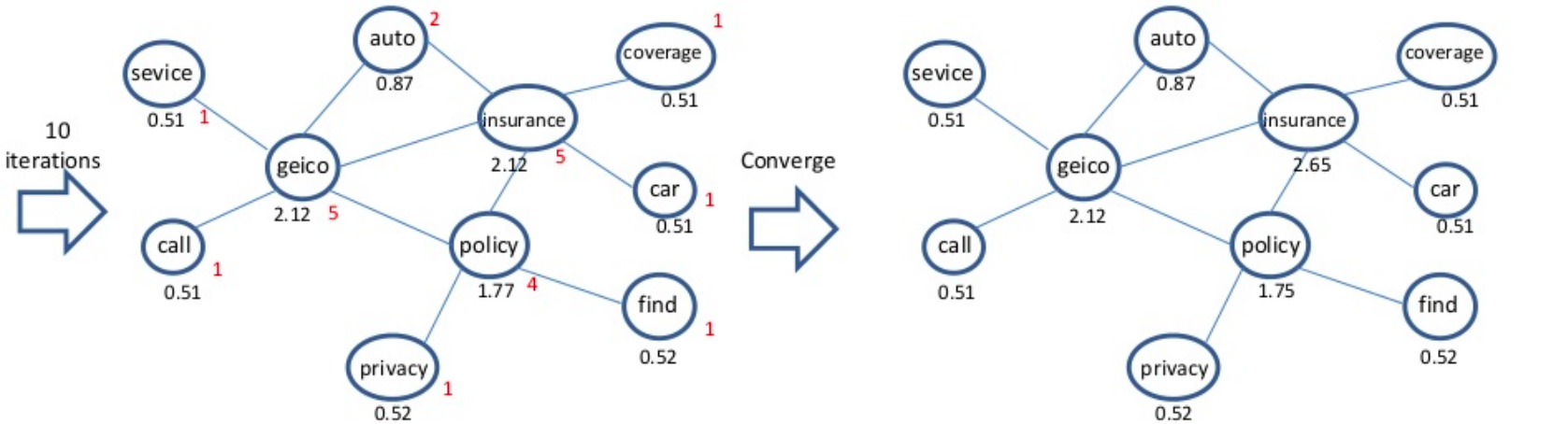
$$Score(policy) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{1} * 1}_{\text{find}} + \underbrace{\frac{1}{1} * 1}_{\text{privacy}} + \underbrace{\frac{1}{5} * 1}_{\text{insurance}} + \underbrace{\frac{1}{5} * 1}_{\text{geico}}\right) = 2.19$$

$$Score(service) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{5} * 1}_{\text{geico}}\right) = 0.32$$

Пример ранжирования вершин

$$S(V_i) = (1-d) + d * \sum_{j \in ngbr(V_i)} \frac{1}{|degree(V_j)|} S(V_j)$$

d is the damping factor that usually set to 0.85



$$Score(geico) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{1} * 0.51}_{\text{service}} + \underbrace{\frac{1}{1} * 0.51}_{\text{call}} + \underbrace{\frac{1}{2} * 0.87}_{\text{auto}} + \underbrace{\frac{1}{5} * 2.12}_{\text{insurance}} + \underbrace{\frac{1}{4} * 1.77}_{\text{policy}}\right) = 2.12$$

Converge Really Quick!
(<= 20 iterations)

$$Score(policy) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{1} * 0.52}_{\text{find}} + \underbrace{\frac{1}{1} * 0.52}_{\text{privacy}} + \underbrace{\frac{1}{5} * 2.12}_{\text{insurance}} + \underbrace{\frac{1}{5} * 2.12}_{\text{geico}}\right) = 1.75$$

$$Score(service) = 0.15 + 0.85 * \left(\underbrace{\frac{1}{5} * 2.12}_{\text{geico}}\right) = 0.51$$

Gensim TextRank

```
In[1]: from gensim.summarization import keywords
In[2]: print(keywords(text))
```

Пример работы TextRank на тексте про Telegram: ссылка

Rapid Automatic Keyword Extraction (RAKE)

1. Фразы-кандидаты – все слова между разделителями
2. Оценка фразы-кандидата p: $\frac{deg(p)}{freq(p)}$
3. Ограничения по частоте и количеству слов в кандидате-фразе

Пример работы RAKE на тексте про Telegram: ссылка

Выделяем ключевые слова, которые очень часто встречаются в данном тексте и гораздо реже встречаются в остальных текстах коллекции

For a term i in document j:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Пример tf-idf

Вычислить значения tf-idf для каждого **существительного** из «коллекции»:

1. Я люблю торты и пирожные. Особенно торты.
2. Я часто покупаю торты и эклеры.
3. Торты вкуснее эклеров. Эклеры слишком сладкие.

ИЛИ

	Doc1	Doc2	Doc3
car	27	4	24
auto	3	33	0
insurance	0	33	29
best	14	0	17

Данные, код и результаты: ссылка

Упражнение

Для вашего текста (скаченного с rubooks или, на крайний случай, winni.txt или alice.txt с предыдущих семинаров) извлечь биграммы по мерам ассоциации:

- `bigram_measures.pmi`
- `bigram_measures.student_t`
- `bigram_measures.chi_sq`
- `bigram_measures.likelihood_ratio`

Можете использовать код отсюда. Попробуйте поменять порог минимальной встречаемости биграмм.