Анализ текстов

2. Векторная модель

Екатерина Черняк, к.т.н

echernyak@hse.ru

Национальный Исследовательский Университет – Высшая Школа Экономики
НУЛ Интеллектуальных систем и структурного анализа

September 13, 2018

Векторная модель

Векторная модель (Vector Space Model, VSM)

Алгебраическая модель представления документов (или любых других объектов) векторами слов.

- Векторная модель коллекции документов: документ – это вектор, состоящий из частот слов, в нем встречающихся
  - Информационный поиск
- Векторная модель слов (иначе дистрибутивная семантика (distributional semantics)): слово – это вектор, состоящий из частот слов, встречающихся в его окрестности
  - Математическая модель семантики интересна и сама по себе, и как вспомогательное средство
- Определение сходства между документами, между словами
- Снижение размерности
  - Модели скрытых тем
  - Эмбеддинги

Что такое информационный поиск?

Основные понятия

- **Документ** – любой контент, чаще всего – текст
- **Запрос** – небольшой текст, написанный пользователем для выражения его информационных нужд
- **Релевантность** – некая функция, показывающая, насколько данный документ соответствует запросу (то есть, насколько документ удовлетворяет информационные нужды пользователя)

Основные задачи

- **Индексация** – обработка и хранение текстов в хранилище, создание индекса
- **Поиск** – поиск в хранилище текста, наиболее релевантного запросу

Релевантность документов запросу

Релевантность – это сложно:

- Зависит от намерений пользователя
- Зависит от места, времени, используемого устройства
- Зависит от других документов, найденных по этому запросу

Очень большое упрощение:

- D – множество документов
- Q – множество запросов
- Теоретико-множественный подход: $R : D \times Q \to \{0, 1\}$ – бинарная функция релевантности
- Алгебраический подход: $R : D \times Q \to [0, 1]$ – задает ранжирование документов
- Вероятностный подход: $P(R|d, q)$ – когда-нибудь в следующий раз :)

Формальная модель

Чтобы построить модель **представления** текста, нужно определить из чего текст состоит: например, из слов или их производных (термов). Термами могут быть исходные слова, леммы, стемы, буквенные n-граммы и т.д..

на	дворе	трава	траве	дрова	не	руби	двора
3	1	1	2	2	1	1	1
на	двор	трава	дрова	не	рубить		
3	2	3	2	1	1		
на	двор	трав	дров	не	руб		
3	2	3	2	1	1		

Обозначения:

- T – множество термов, всего термов – $|T| = N$
- $d = (w_1, w_2, ..., w_N)$ – **векторное представление текста**
- Вес

$$w_i = \begin{cases} 0, & \text{если } t_i \notin d \\ > 0, & \text{иначе} \end{cases}$$

Индекс

- **Прямой индекс** – список слов (и их позиций) в документе
- **Инвертированный индекс** – список документов (и позиций в них), в (на) которых встречается слово

	d_1	d_2	...	d_M
t_1	w_{11}	w_{12}	...	w_{1M}
t_2	w_{21}	w_{22}		...
...	...			
t_N	w_{N1}	w_{N2}		w_{NM}

Поиск по запросу

Считаем нерелевантными документы, не содержащие ни одного терма из запроса.

Пусть $q = t_i$ – запрос состоит из одного терма.

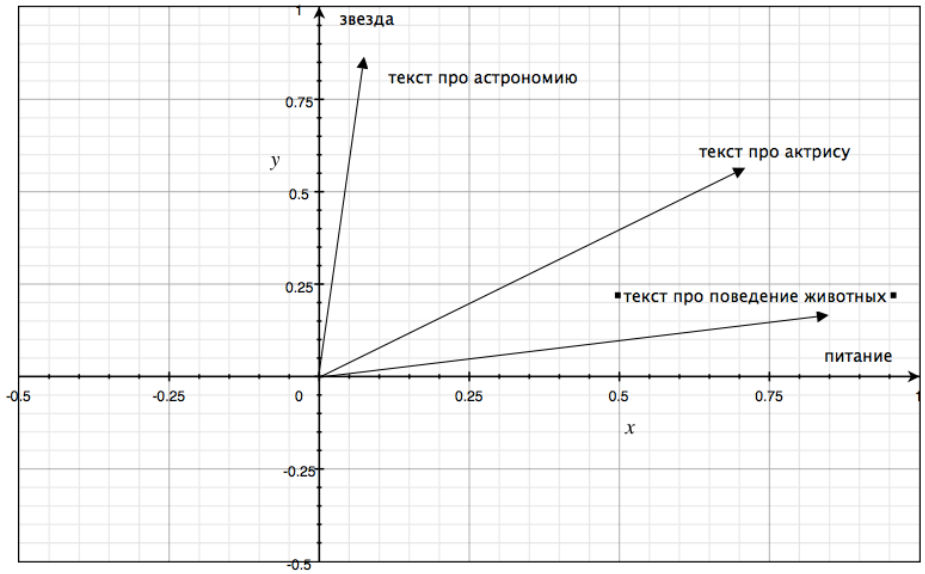
- Найти все документы d_j, такие что $w_{ij} > 0$
- Отсортировать их по убыванию w_{ij}
- Вернуть пользователю отсортированный список “релевантных” запросу документов

Пусть $q = t_1, \ldots t_k$ – запрос состоит из нескольких термов.

- **Дизъюнктивный поиск**: вернуть список документов, содержащих хотя бы один терм из запроса
- **Коньюнктивный поиск**: вернуть список документов, содержащих все термы из запроса

Векторная модель

- Каждый текст – это вектор в N-мерном пространстве термов (N – количество термов в проиндексированной коллекции текстов)
- Запрос – такой же вектор, как и любой текст
- Релевантность документа запросу определяется по сходству документа и запроса, иначе говоря, по близости соответсвующих им векторов друг другу



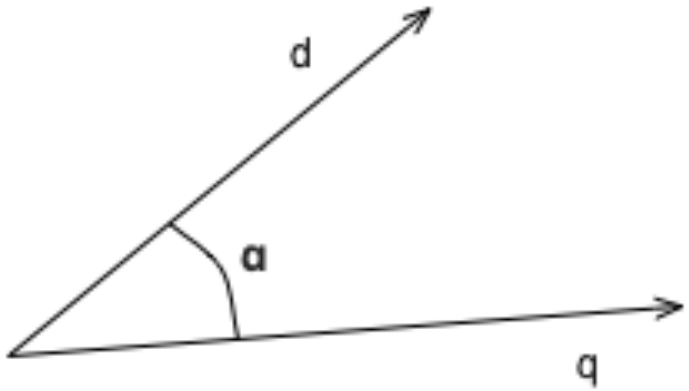
Векторная модель

$$d = (w_1^d, \ldots, w_N^d)$$

$$q = (w_1^q, \ldots, w_N^q)$$

$$\cos\alpha = \mathtt{sim}(q, d) = \frac{q \times d}{||q||||d||} = \frac{\sum_i w_i^q \times w_i^d}{\sqrt{(\sum_i w_i^q)^2}\sqrt{(\sum_i w_i^d)^2}}$$

$$\hat{d}_q = \arg\max_d sim(d, q)$$

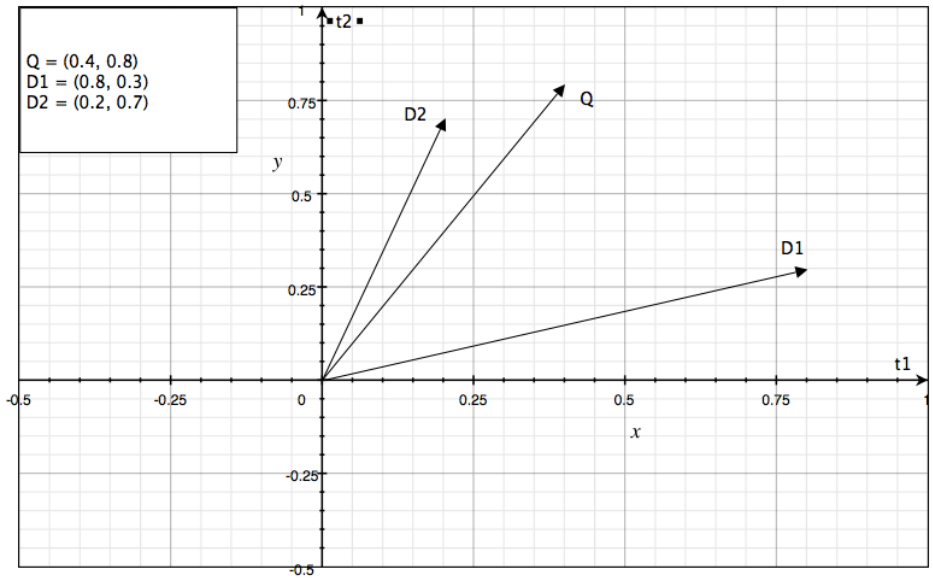


Векторная модель

$$\cos\alpha = \texttt{sim}(q,d) = \frac{q \times d}{||q||||d||} = \frac{\sum_i w_i^q \times w_i^d}{\sqrt{(\sum_i w_i^q)^2}\sqrt{(\sum_i w_i^d)^2}}$$

$$\texttt{sim}(q, d_2) = \frac{0.4 \times 0.2 + 0.8 \times 0.7}{\sqrt{0.4^2 + 0.8^2}\sqrt{0.2^2 + 0.7^2}} = 0.98$$

$$\texttt{sim}(q, d_1) = \frac{0.4 \times 0.8 + 0.8 \times 0.3}{\sqrt{0.4^2 + 0.8^2}\sqrt{0.8^2 + 0.3^2}} = 0.74$$



Как определить веса w?

Схема взвешивания *tfidf*

$$tfidf(t, d, D) = tf(t, d) \times idf(t, D) = f_{t,d} \times \log \frac{|D|}{n_t + 1},$$

где $|D|$ – количество документов, n_t – количество документов, содержащих терм t

Векторная модель: плюсы и минусы

Плюсы:

- Использование весов позволяет улучшить поиск
- Возможен дизъюнктивный поиск
- Косинус – приятная и легко интерпретируемая с точки зрения математики функция
- Множество экспериментов показало, что косинусная функция – хорошая мера релевантности

Минусы

- Между термами бывают сильные семантические связи, поэтому термы никак не могут соответствовать ортогональным векторам
- *IDF* долго вычисляется

Что дальше? Необходим учет полисемии, синонимов и около-синонимов, аббревиатур, различных написаний одного и того же слова.

Снижение размерности в векторной модели

Введем обозначения

- M – количество термов
- $N = |D|$ – количество документов
- C – матрица терм-документ размера $M \times N$, $\mathtt{rank}(C) \leq \min M, N$

И вспомним кое-что из линейной алгебры:
Пусть A квадратная матрица размера $M \times M$. В уравнении $Ax = \lambda x$ – x – собственный вектор, λ – собственные значения, которые находятся с помощью характеристического уравнения $(A - \lambda I_M)x = 0$. Число собственных значений совпадает с рангом матрицы – количеством линейно независимых строк (или столбцов).

Теорема о сингулярном разложении матрицы

Пусть r – это ранг матрицы C размера $M \times N$. Сингулярным разложением матрицы C назовем уравнение

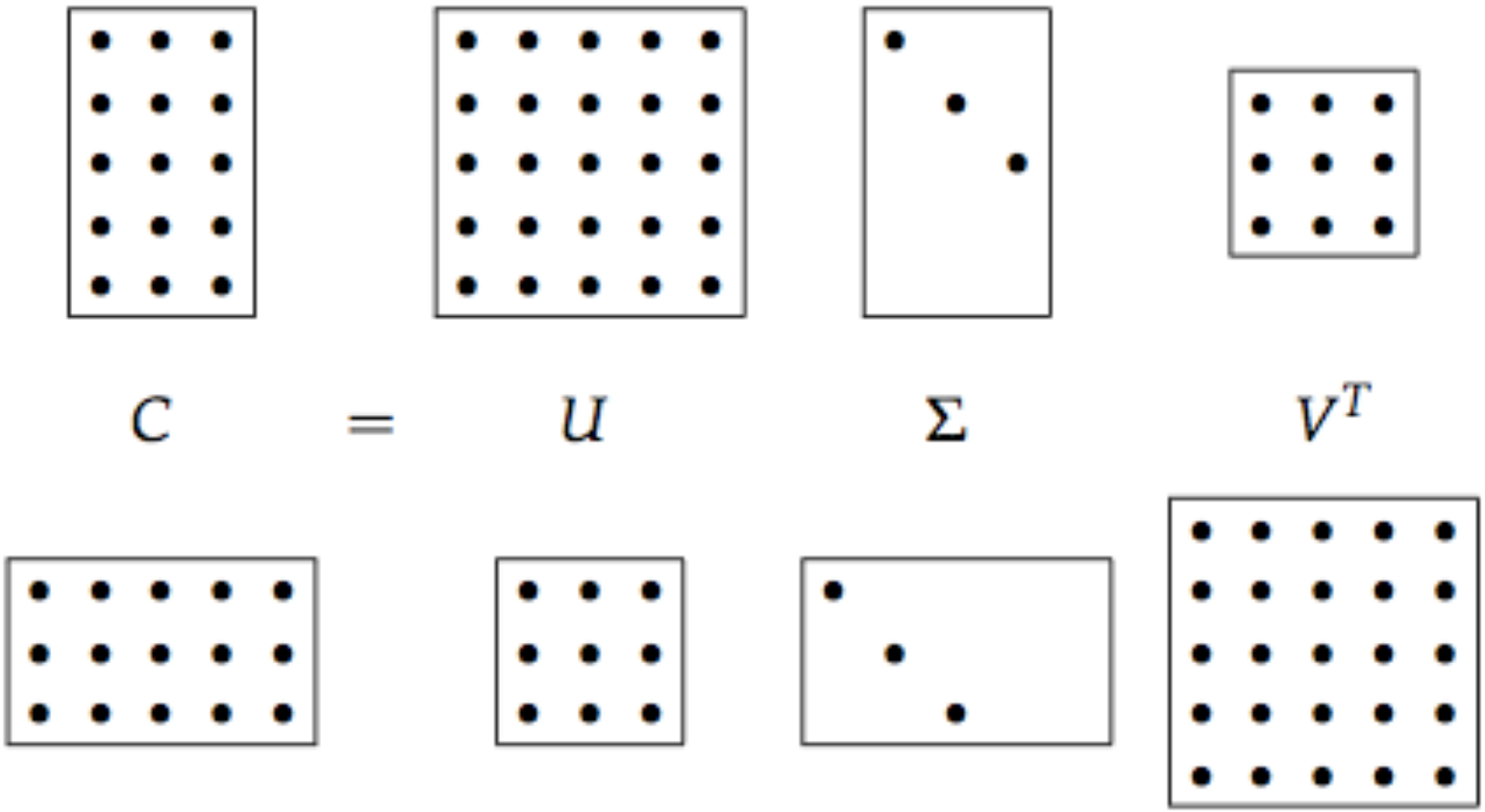
$$C = U\Sigma V^T,$$

где

- Собственные значения $\lambda_1, \ldots, \lambda_r$ матрицы CC^T совпадают с собственными значениями матрицы $C^T C$
- Пусть $\sigma_i = \sqrt{\lambda_i}$, $1 \leq i \leq r$, причем $\lambda_{i+1} \leq \lambda_i$. Тогда Σ – матрица размера $M \times N$, $\Sigma_{ii} = \sigma_i$, $1 \leq i \leq r$.

Σ – матрица сингулярных значений, U – матрица левых сингулярных векторов, V – матрица правых сингулярных векторов.

Сингулярное разложение матрицы

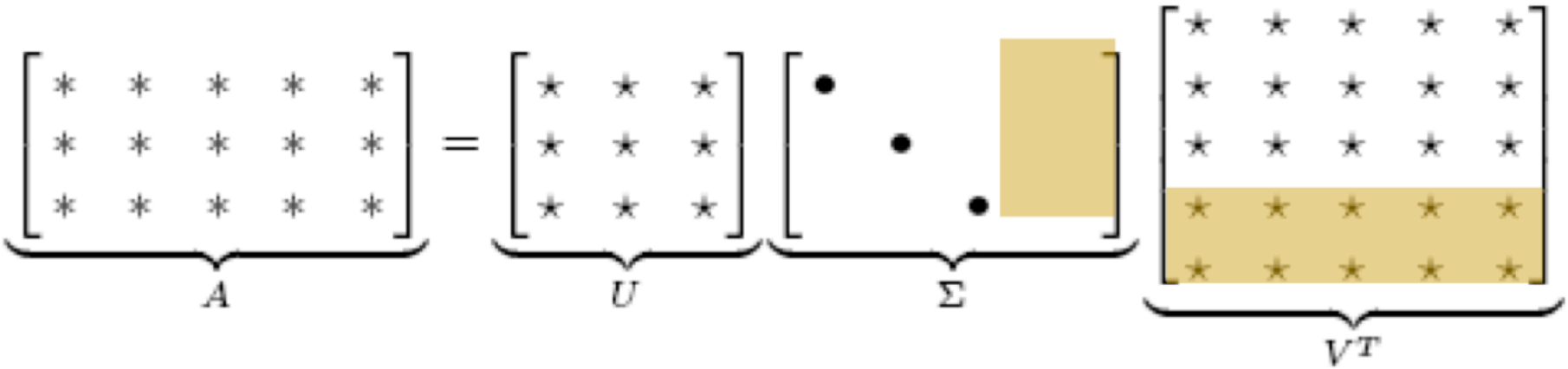


Сингулярное разложение матрицы

reduced SVD

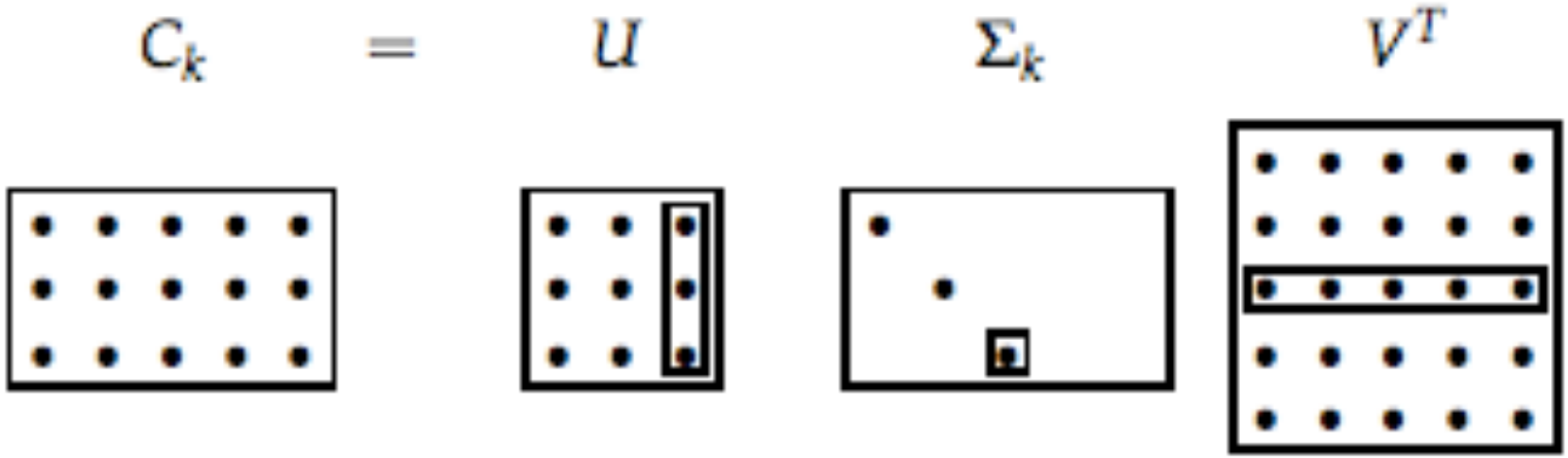
```
In[1]:  U, s, V = numpy.linalg.svd(C, full_matrices=False)
```

$C - M \times N, U - M \times M, s - M \times 1, V - M \times N$



Аппроксимация матрицы C матрицей меньшего ранга

1. Найти $C = U\Sigma V^T$
2. В матрице Σ обнулить $r - k$ наименьших значений и получить Σ_k
3. Искомая апроксимация: $C_k = U\Sigma_k V^T = \sum_{i=1}^{k} \sigma_i u_i v_i^T$

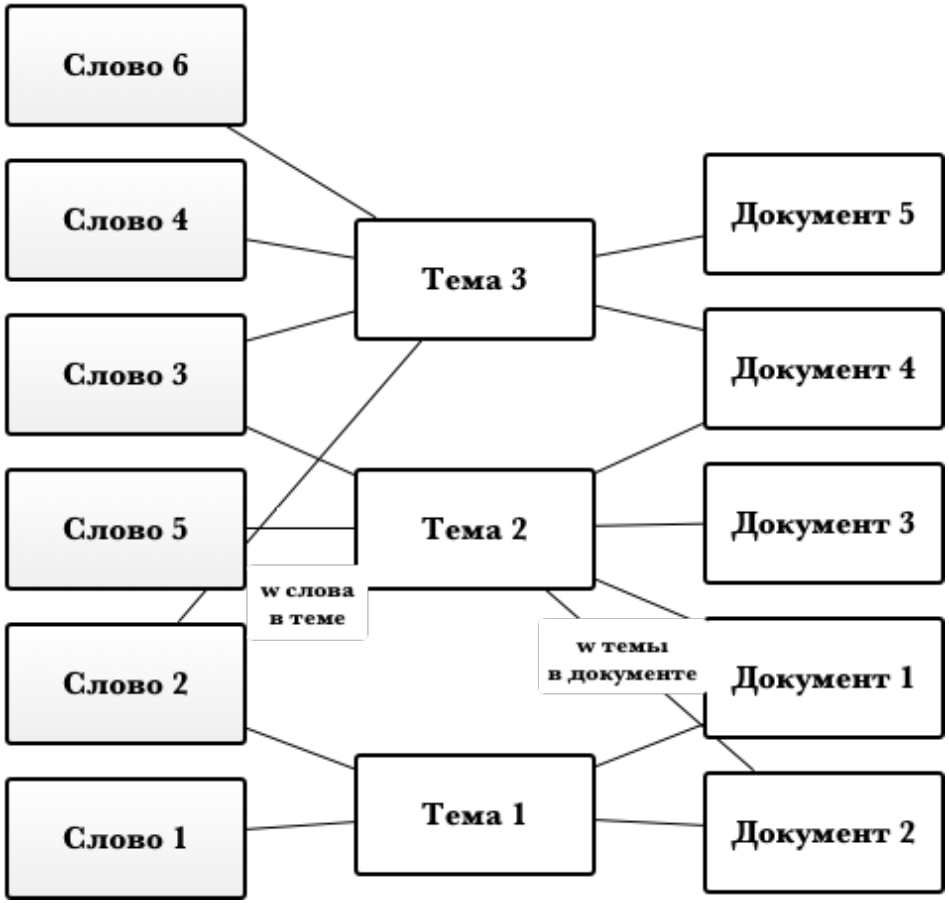


Латентно-семантический анализ

ЛСА помогает определить семантически близкие документы (посвященные одной теме), но непохожие в векторном пространстве (поскольку в них встречаются разные слова). Используем сингулярное разложение для создания нового векторного пространства, в котором у семантически похожие документы будут друг другу ближе.

- Близкие по значению термы формируют одно измерение (т.н. “тему”) в пространстве меньшей размерности
- За счет снижения размерности уменьшается и количество шума

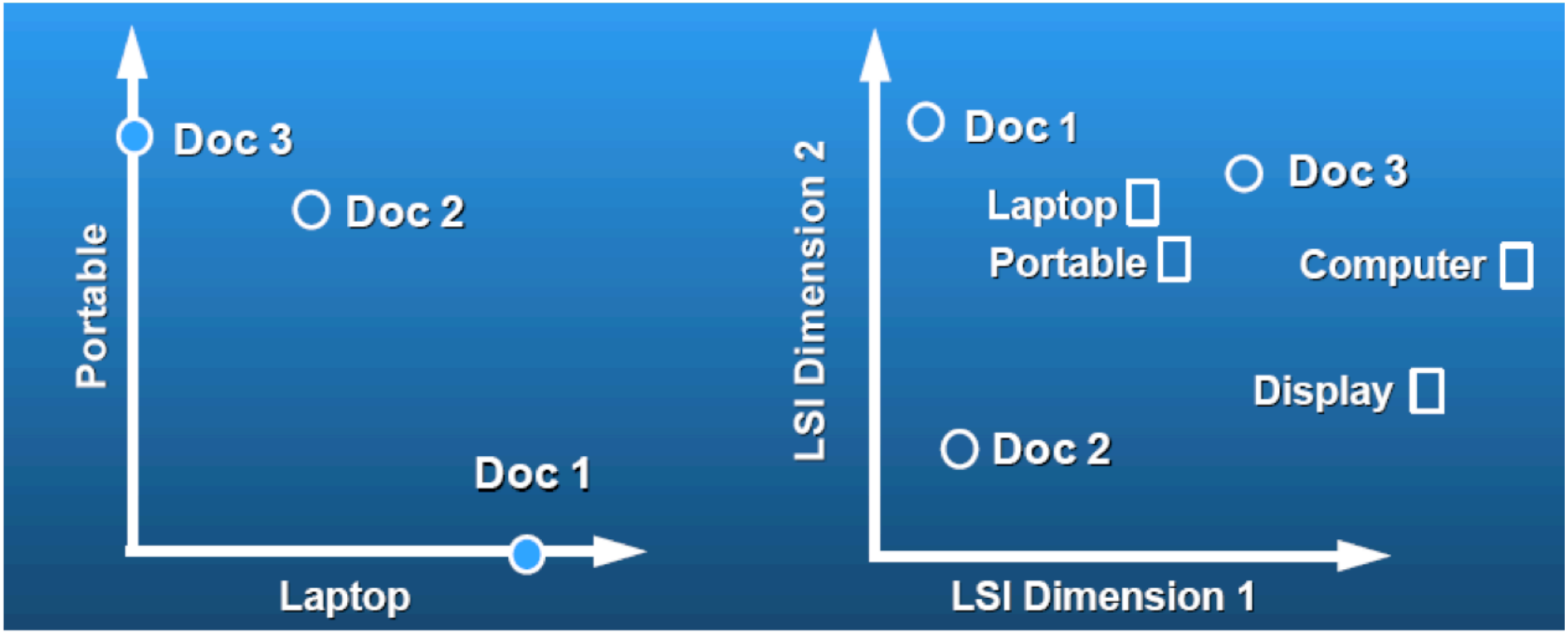
Латентно-семантический анализ



$$C_k = U\Sigma_k V^T$$

- k – число скрытых тем
- U_k – веса слов в темах
- V_k^T – веса тем в документах

Латентно-семантический анализ (Susan Dumais)



Латентно-семантический анализ

1. От матрицы терм-документ C переходим к ее апроксимации $C_k = U_k * \Sigma_k * V_k^T$
2. Преобразование вектора-запроса q: $q_k = \Sigma_k^{-1} U_k^T q$

LSI в gensim

```
lsi = lsimodel.LsiModel(corpus_tfidf, dictionary, num_topics)
```

Упражнения

1. Поиск в векторной модели (пример из работ Dumais)
2. Визуализация векторной модели

Эмпирическая проверка

В своих работах Susan Dumais показала, что использование ЛСА в задаче поиска в среднем повышает качество результатов. Что это значит?

Проводится стандартная проверка:

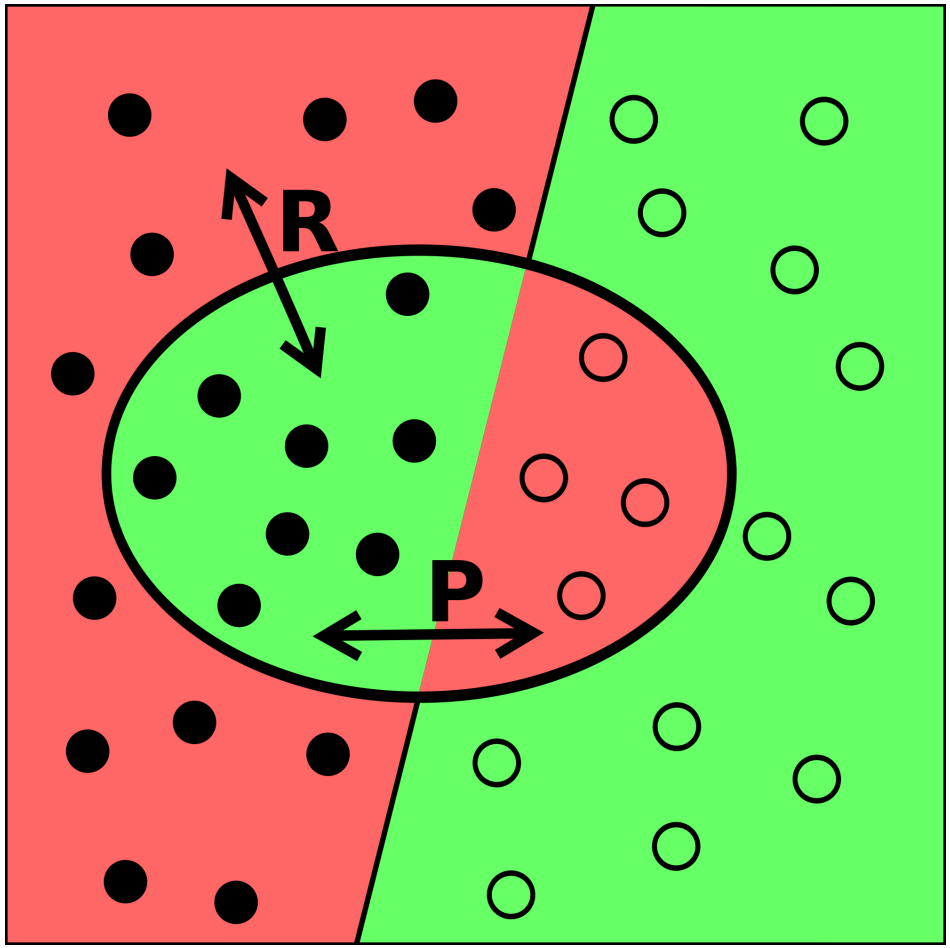
- Есть размеченные коллекции (запросы и релевантные им документы) TREC 1, 2, 3
- Есть baseline: $tf - idf$ и косинусная мера близости
- Есть меры качества: точность и полнота

Меры качества в информационном поиске

$$\text{точность} = \texttt{precision} = \mathrm{P} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

$$\text{полнота} = \texttt{recall} = \mathrm{R} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

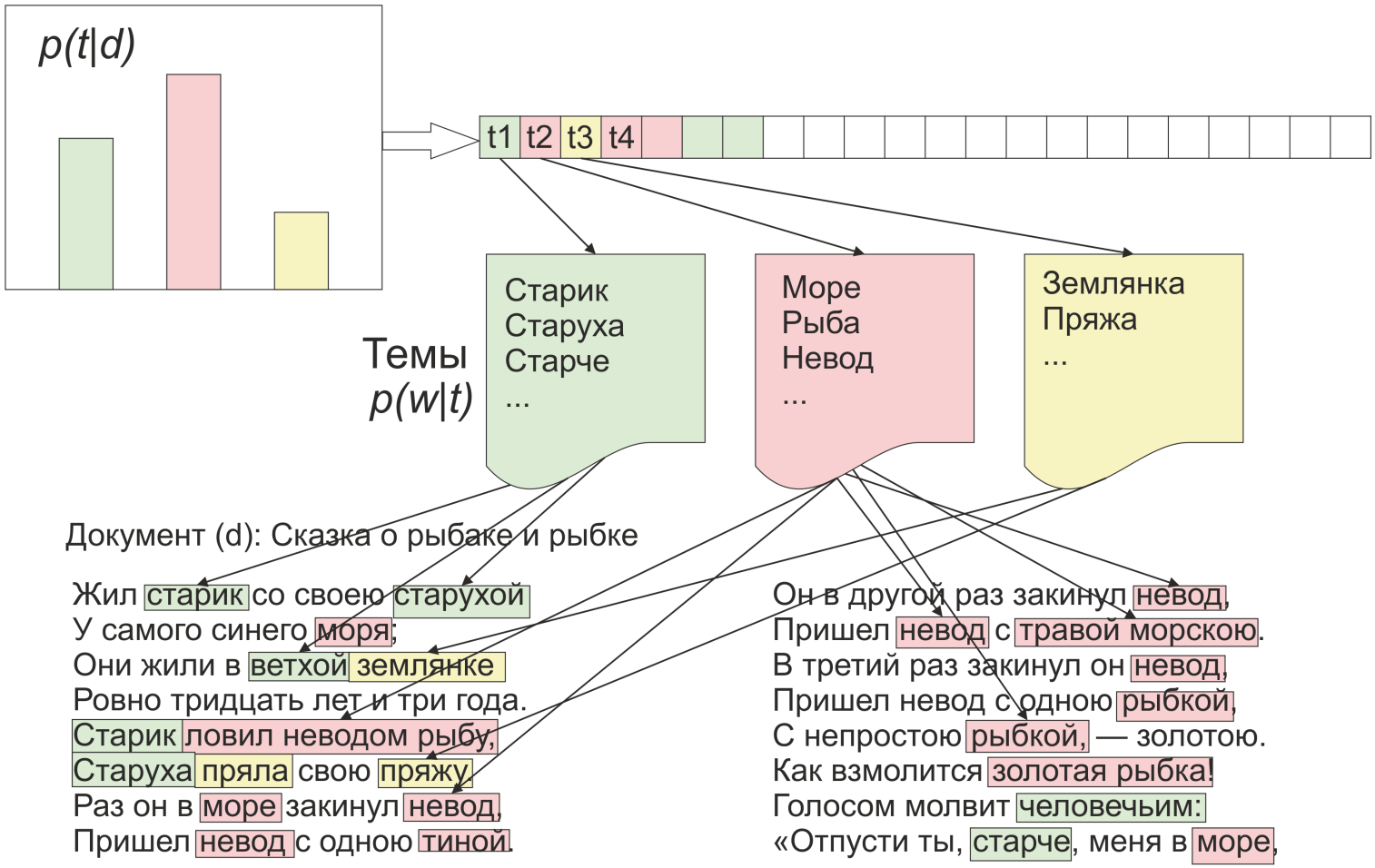
$$\texttt{F-measure} = \frac{2\mathrm{PR}}{\mathrm{P} + \mathrm{R}}$$



Латентно-семантический анализ: другие приложения

- “soft clustering”: каждое измерение в пространстве меньшей размерности – это кластер, V_k^T – вектора принадлежности к кластерам
- Рекомендательные системы: вместо термов – объекты, вместо документов – пользователи
- Примитивная невероятностная модель скрытых тем
- Приложения в других областях (например, в компьютерном зрении.

Вероятностное тематическое моделирование



Лекция К. В. Воронцова. Тематическое моделирование и разведочный информационный поиск

Неформальная постановка задачи

- Найти темы (или топики в документах) без использования обучения
- Задача сходна задаче бикластеризации: есть слова и документы, надо найти бикластеры (слова, характерные для определенных документов)
- В одном документе может быть несколько тем
- Темы состоят из слов
- Самый известный алгоритм – LDA, латентное размещение Дирихле – предполагает генеративный процесс порождения документов: сначала генерируются темы, затем – документы. Число тем заранее зафиксировано

Формулировка задачи

Тема – это набор слов, неслучайно встретившихся в неком подмножестве документов из корпуса

- $w \in W$ – множество слов (наблюдаемы)
- $d \in D$ – множество документов (наблюдаемы)
- $t \in T$ – множество тем (скрыты!)
- $p(w|t)$ – тема определяется словами
- $p(d|t)$ – документ определяется темами
- слова и документы независимы: $p(w|d,t) = p(w|t)$

Главное уравнение:

$$P(w|d) = \sum_t P(d|t)P(w|t)$$

или

$$P(w,d) = DT$$

Это задача матричной факторизации, которая не имеет единственного решения!

Генеративный процесс LDA [Blei, 2003]

Генерация документа:

1. Задать распределение тем (нужно знать распределение тем по колдлекции)
2. Для каждой позиции слова из документа
   1. Выбрать тему (нужно знать распределение тем по документу
   2. Выбрать слово из темы (нужно знать распределение слов по темам)

Для анализа коллекции текстов нужно оценить:

1. Распределение слов по темам (мультиномиальное распределение)
2. Распределение тем по документам (распределение Дирихле)
3. Распределение тем по коллекции (распределение Дирихле)

Латентное размещение Дирихле, LDA [Blei, 2003]

Векторы документов и векторы тем порождаются распределениями Дирихле.

- векторы документов $\theta_d = \big(p(t|d) : \ t \in T\big)$ порождаются одним и тем же вероятностным распределением на нормированных $|T|$-мерных векторах; это распределение удобно взять из параметрического семейства распределений Дирихле $\mathrm{Dir}(\theta, \alpha), \ \alpha \in \mathbb{R}^{|T|}$;
- векторы тем $\phi_t = \big(p(w|t) : \ w \in W\big)$ порождаются одним и тем же вероятностным распределением на нормированных векторах размерности $|W|$; это распределение удобно взять из параметрического семейства распределений Дирихле $\mathrm{Dir}(\theta, \beta), \ \beta \in \mathbb{R}^{|W|}$.

Генеративный процесс LDA [Blei, 2003]

1. Выбрать распределение темы для документа i : $\theta_i \sim \mathrm{Dir}(\alpha)$, $1 \leq i \leq |D|$, $\mathrm{Dir}(\alpha)$ – распределение Дирихле с параметром α
2. Выбрать распределение слов для темы k : $\varphi_k \sim \mathrm{Dir}(\beta)$, $1 \leq k \leq K$, K – **заданное число тем**, $\mathrm{Dir}(\beta)$ – распределение Дирихле с параметром β
3. Для каждой позиции слова i, j, $1 \leq j \leq N_i$ (N_i – количество слов в документе d_i, $1 \leq i \leq |D|$)
    1. Выбрать тему $t_{i,j} \sim \mathrm{Multinomial}(\theta_i)$
    2. Выбрать слово $w_{i,j} \sim \mathrm{Multinomial}(\varphi_{z_{i,j}})$

LDA в gensim

```
lda = ldamodel.LdaModel(corpus, dictionary, num_topics)
```

Латентное размещение Дирихле, LDA [Blei, 2003]

5 предложений:

- I like to eat broccoli and bananas.
- I ate a banana and spinach smoothie for breakfast.
- Chinchillas and kittens are cute.
- My sister adopted a kitten yesterday.
- Look at this cute hamster munching on a piece of broccoli.

Распределение тем по предложениям:

- Предложения 1 и 2: 100% тема A
- Предложения 3 и 4: 100% тема B
- Предложение 5: 60% тема A, 40 % тема B

Распределение слов по темам:

- Тема A: 30% broccoli, 15% bananas, 10% breakfast, 10% munching (про еду)
- Тема B: 20% chinchillas, 20% kittens, 20% cute, 15% hamster (милые животные)

Упражнение

Упражнение

1. LSI для извлечения скрытых тем из газетных статей
2. LDA для извлечения скрытых тем из газетных статей

1. Векторная модель
2. Векторная модель в информационном поиске
3. Вероятностное тематическое моделирование
4. Дистрибутивная семантика

Дистрибутивная семантика

Смысл слова [Ludwig Wittgenstein]

Die Bedeutung eines Wortes liegt in seinem Gebrauch.

Distributional hypothesis [J.R.Firth, 1957]

You shall know a word by the company it keeps!

Что такое “бардьюк”?

- Он подал ей бокан бардьюка.
- Бардьюк подают к блюдам из говядины.
- Ноги у него заплетались, а лицо горело от выпитого бардьюка.
- Виноград сорта бардьюк выращивают в Австралии.
- К простому ужину из хлеба и сыра я взял бутылку отличного местного бардьюка.
- Напитки были прекрасны: кроваво-красный бардьюк и легкое белое рейнское.

doc#50340	так Остро ощущает, когда он влюблен. Эта	компульсия	может существовать и при отсутствии влюбленности
doc#336729	психологии оракул гороскоп такое понятие —	компульсия	— гороскоп притяжение к оракуул, реализующееся
doc#878536	(обсессии); навязчивое поведение (	компульсия	); оппозиционное поведение;
doc#1000748	характерна другая сила тяги, так называемая «	компульсия	». - И что означает это слово в применении
doc#1000748	учителей очень образно описывал, что такое «	компульсия	». Это влечение, сравнимое с жизненно важными
doc#1369221	обеспечиваются равновесие, гомеостазис. Иногда	компульсия	лучше устраняется посредством ее «взрыва
doc#2553333	борьбы с ними. Навязчивое влечение (	компульсия	) — стремление, вопреки разуму, воле и чувствам
doc#3060833	есть субъективный компонент — влечение, или	компульсия	, и объективный — ритуал (вызванные влечением
doc#3575480	рука поднимается вверх и запускается ваша	компульсия	, само ощущение будет буквально утягивать
doc#3575480	направлении. Не то, чтобы у вас исчезла	компульсия	, просто у вас появляется компульсия быть
doc#3575480	исчезла компульсия, просто у вас появляется	компульсия	быть более таким, каким вы хотите быть.
doc#4796843	вещи, которые усиливает эту компульсию, и	компульсия	потеряет всю свою силу. Сама компульсия
doc#4796843	компульсия потеряет всю свою силу. Сама	компульсия	это только то, что лежит на поверхности
doc#4796843	их основе тревога или нечто подавленное.	Компульсия	является защитным механизмом против чувства
doc#4796843	жизни тревогу или депрессию (Чаще всего	компульсия	приводится в действие тревогой-беспокойством
doc#4796843	начните прорабатывать эти чувства. Скоро ваша	компульсия	, независимо от того что вы делаете, уйдет
doc#4796843	непомерную, но полезную службу, которую	компульсия	выполняет для вас. Поблагодарите ее вовлекая
doc#4796843	После того как я обработал эту тревогу,	компульсия	к курению никогда больше не возвращалась
doc#4900615	максимально успешного лечения заболеваний]	КОМПУЛЬСИЯ	compulsion [непреодолимое побуждение совершать
doc#5703937	родственными». Эти близнецы — навязчивость (	компульсия	) и торможение — знакомы каждому, кто испытывал

Дистрибутивная семантика

Векторная модель: матрицы терм-контекст или терм-терм

	$context_1$	$context_2$	...	$context_{\|C\|}$
$term_1$	f_{11}	f_{12}		$f_{1\|C\|}$
$term_2$	f_{21}	f_{22}		$f_{2\|C\|}$
...				
$term_{\|V\|}$	$f_{\|V\|1}$	$f_{\|V\|2}$		$f_{\|V\|\|C\|}$

Термы $|V|$ – обычно индивидуальные слова, чаще всего существительные

Контексты $|C|$:

- Поверхностные: слова или символы в пределах окна
- Текстуальные: тексты целиком или элементы текста (абзацы, предложения)
- Синтаксические: специфические синтаксические связи, чаще всего *subj-of, obj-of*

Веса в дистрибутивной модели

Веса f_{ij}

- бинарный вес: 1, если $term_i$ и $context_j$ встречаются рядом, 0, иначе
- частота: сколько раз $term_i$ и $context_j$ встречаются рядом
- **Positive Pointwise Mutual Information** [Niwa, Nitta, 1994]:

$$PMI(term_i, context_j) = \log_2 \frac{P(term_i, context_j)}{P(term_i)P(context_j)}$$

$$PPMI(term_i, context_j) = \begin{cases} PMI, & \text{if } PMI > 0. \\ 0, & \texttt{иначе}. \end{cases}$$

ММП оценки вероятностей

$$PPMI(term_i, context_j) = \max(\log_2 \frac{P(term_i, context_j)}{P(term_i)P(context_j)}, 0)$$

- $P(term_i, context_j) = \frac{f_{ij}}{\sum_{n=1}^{|V|}\sum_{m=1}^{|C|} f_{nm}}$
- $P(term_i) = \frac{\sum_{m=1}^{|C|} f_{im}}{\sum_{n=1}^{|V|}\sum_{m=1}^{|C|} f_{nm}}$
- $P(context_j) = \frac{\sum_{m=1}^{|V|} f_{nj}}{\sum_{n=1}^{|V|}\sum_{m=1}^{|C|} f_{nm}}$

(P)PMI придает больший вес редким событиям. Дополнительная корректировка на $\alpha = 0.75$:

$$P_\alpha(context_j) = \frac{(\sum_{m=1}^{|V|} f_{nj})^\alpha}{\sum_{m=1}^{|C|}(\sum_{m=1}^{|V|} f_{nm})^\alpha}$$

Меры близости

Для определения близости между термами

- косинусная мера близости:

$$cos(term_i, term_j) = \frac{t_i \times t_j}{||t_i||||t_j||} = \frac{\sum_k f_{ik} \times f_{jk}}{\sqrt{(\sum_k f_{ik})^2}\sqrt{(\sum_k f_{jk})^2}}$$

- мера Жаккара:

$$jc(term_i, term_j) = \frac{\sum_k min(f_{ik}, f_{jk})}{\sum_k max(f_{ik}, f_{jk})}$$

Туториал Марко Барони

От разреженных векторов к эмебддингам (embedding)

Счетные модели: сингулярное разложение матрицы $P(PMI)$:
$X = U\Sigma V$

- $\Phi = U_k\Sigma_k$
- $\Phi = U_k\sqrt{\Sigma}_k$

  Φ – вектора слов

От разреженных векторов к эмебддингам (embedding)

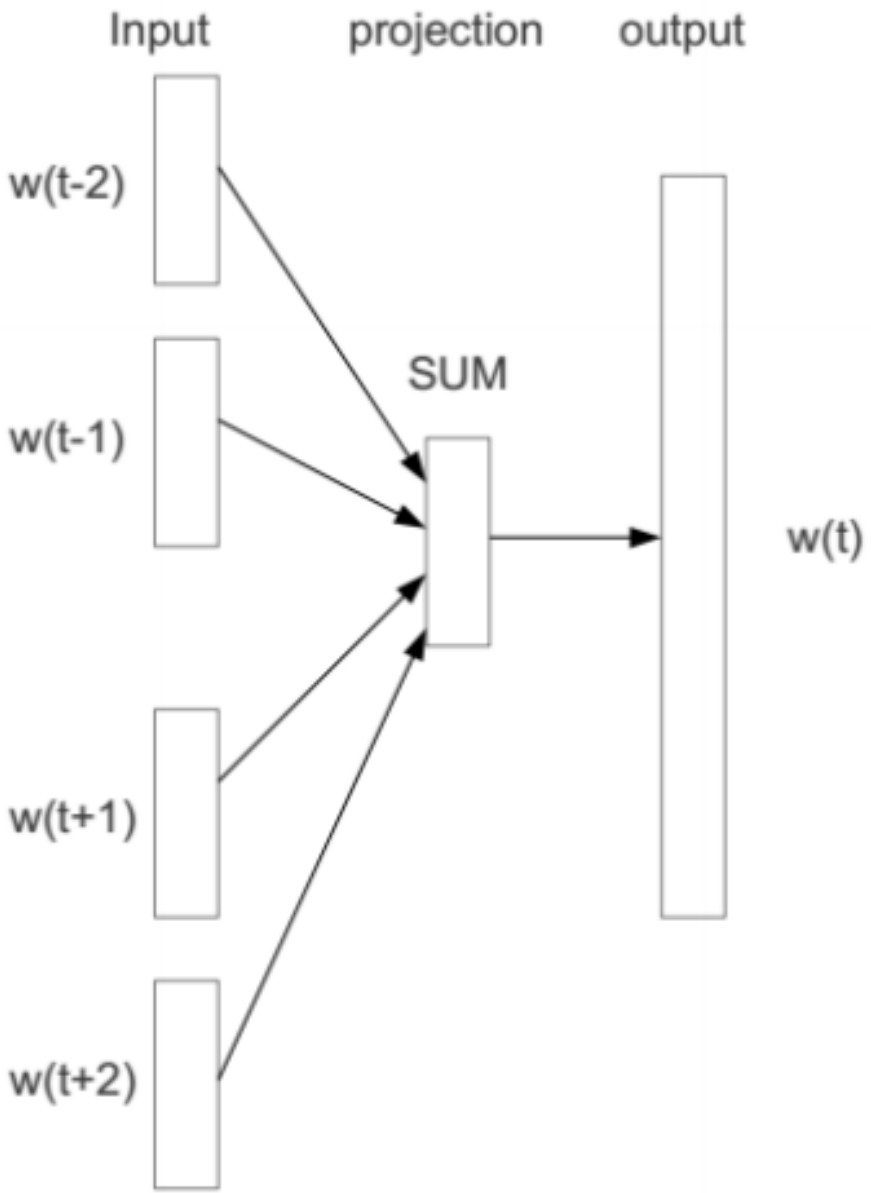
- Предективные модели:
  - Skip-gram, CBOW aka word2vec [Mikolov et al., 2013]
    `http://rusvectores.org/ru/`
    `https://www.tensorflow.org/tutorials/word2vec`
  - Dependency embeddings [Levi et al., 2015]
    `https://bitbucket.org/yoavgo/word2vecf`
  - GloVe [Pennington et al., 2014]
    `https://github.com/facebookresearch/fastText`
  - FastText [Joulin et al., 2016]
    `https://github.com/facebookresearch/fastText`
  - AdaGram [Bartunov et al., 2016]
    `https://github.com/sbos/AdaGram.jl`
    `http://adagram.ll-cl.org`
  - SenseGram [Pelevina et al., 2016]
    `https://github.com/tudarmstadt-lt/sensegram`
  - StarSpace [Wu, 2017]
    `https://github.com/facebookresearch/StarSpace`
  - Poincare embeddings [Nickel et al., 2017]

Как оценить качество?

- По внутренним свойствам векторов:
  - аналогии
  - синонимы
  - “лишние слова”
- По внешним задачам:
  - Классификация и кластеризация
  - POS-tagging

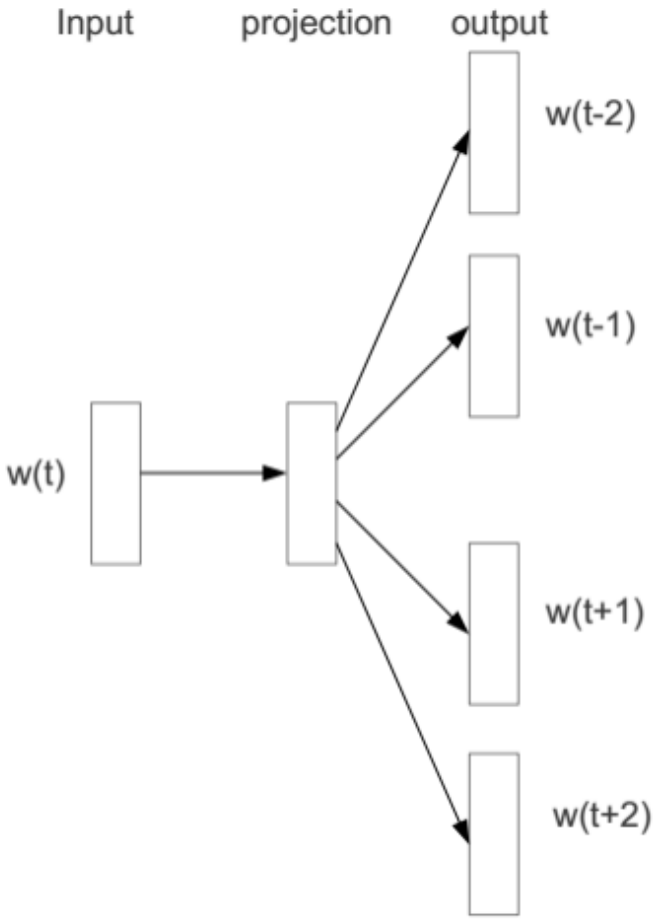
Continuous bag-of-words model (CBOW) [Mikolov, 2013]

- Входной слой: контекст слова (+, – 2 слова слева и справа)
- Слой проекции: линейный
- Выходной слой: вектор слова



skip-gram [Mikolov, 2013]

- Обратная задача: предсказание векторов контекста по данному слову
- Выходной слой: вектор слов
- Если объем обучающего множества и количество эпох достаточно велики, то вектора, полученные для одного и того же слова с помощью CBOW и skip-gram архитектуры похожи
- Негативное сэмплирование: не все отрицательные контексты важны



Quora: How does word2vec work?

Omer Levy, authored several academic papers on word2vec

Written Oct 20, 2014 · Upvoted by Franck Dernoncourt, PhD student in machine learning at MIT and Amund Tveit, PhD in machine learning.

TL;DR

word2vec is not one algorithm.

word2vec is not deep learning.

The recommended algorithm in word2vec, skip-gram with negative sampling, factorizes a word-context PMI matrix.

Skip-Gram with Negative Sampling (SGNS) [Levy, Goldberg]

Статья, в которой формулируется задача оптимизации вероятности пар слово-контекст
Статья, в которой word2vec представлено как факторизация матрицы PPMI

Skip-Gram

По данному слову предсказываем контекст:

$$p(w_{i-k}, \ldots, w_{i+k}|w_i) = \prod_{-k \leq i \leq k, i \neq k} p(w_{i+k}|w_i)$$

Вероятность представляем при помощи softmax:

$$p(w|c) = \frac{e^{\vec{w} \cdot \vec{v}_c}}{\sum_{c \in V} \vec{w} \cdot \vec{v}_c}$$

Оптимизационная задача:

$$L = \sum_{w \in V} \sum_{c \in V} f_{wc} \log p(c|w)$$

Skip-Gram with Negative Sampling (SGNS)

Будем предсказывать не контекст для текущего слово, а да / нет для пары слов:

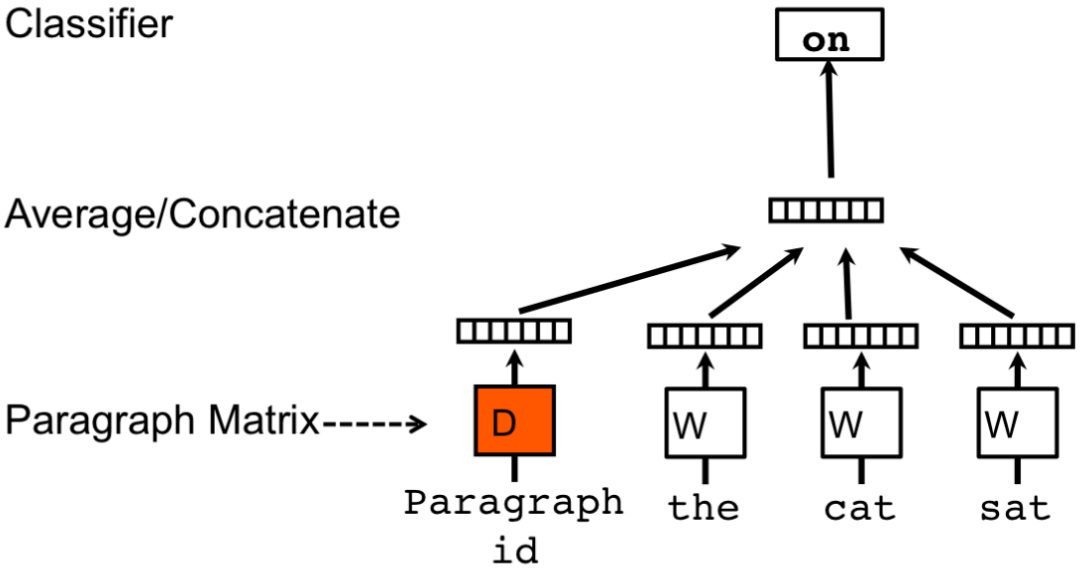
$$L = \sum_{w \in V} \sum_{c \in V} f_{wc} \log \theta(\vec{w} \cdot \vec{v}_c) + k * \log \theta(-\vec{w} \cdot \vec{v}_c)$$

L достигаем максимума при $\vec{w}$, $\vec{v}$ эквивалентным факторизации sPMI.

Насколько похожи два предложения (абзаца)?

Нужно посчитать косинусную меру близости между векторами-предложениями (абзацамм). Как найти вектор-предложение (абзац) ?

1. Усреднить вектора слов, входящих в каждое предложение
2. Doc2vec: тоже самое, что word2vec, только для предложений (абзацев)



word2vec и doc2vec в gensim

word2vec

```
model = word2vec.Word2Vec(sentences, size=100, window=5,
min_count=5, workers=4)
```

doc2vec

```
model = doc2vec.Doc2Vec(sentences, size=100, window=5,
min_count=5, workers=4)
```

Global **Ve**ctors [Pennington, Socher, Manning]

	x = solid	x = gas	x = water	x = random
p(x|ice)	large	small	large	small
p(x|steam)	small	large	large	small
p(x|ice)/p(x|steam)	large	small	1	1

GloVe [Pennington, Socher, Manning]

The training objective of GloVe is to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. Owing to the fact that the logarithm of a ratio equals the difference of logarithms, this objective associates (the logarithm of) ratios of co-occurrence probabilities with vector differences in the word vector space. Because these ratios can encode some form of meaning, this information gets encoded as vector differences as well. For this reason, the resulting word vectors perform very well on word analogy tasks, such as those examined in the word2vec package.

Global Vectors [Pennington, Socher, Manning]

Два требования к векторам слов:

1. $(w_i, w_j) = \log p(i|j)$
2. $w_x(w_a - w_b) \log \frac{p(x|a)}{p(x|b)}$

Задача определения весов:

$$J = \sum_{i,j=1}^{|V|} \alpha(f_{ij})(w_i w_j + b_i - b_j - \log(f_{ij}))^2$$

b – сдвиг для каждого слова

$$\alpha(x) = \begin{cases} (x/x_{max})^{\alpha}, \\ 1, \text{иначе} \end{cases}$$

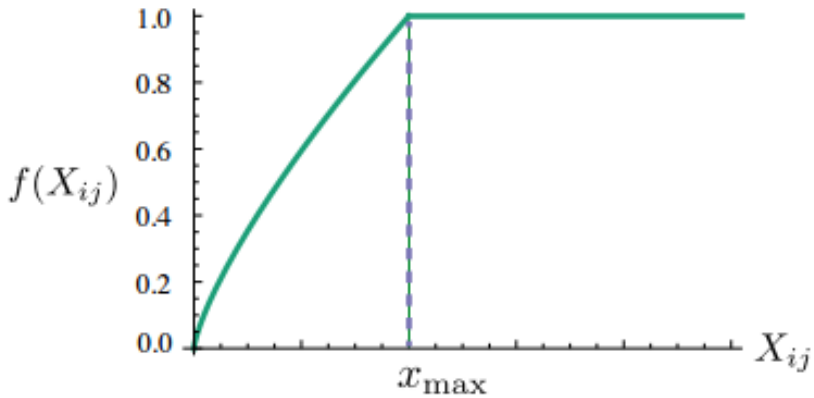


Figure 1: Weighting function f with $\alpha = 3/4$.