



# Дистрибутивная гипотеза

- Zellig S. Harris: “oculist and eye-doctor... occur in almost the same environments”, “If A and B have almost **identical environments**. . . we say that they are synonyms”
- Самое известное, Джон Ферс:  
**You shall know a word by the company it keeps!**



Кстати,  
Харрис --  
учитель  
Ноама Хомского



Джон Руперт Ферс --  
основатель  
лондонской школы  
ЛИНГВИСТИКИ

Harris, Z. S. (1954). Distributional structure. *Word*, 10, 146–162. Reprinted in J. Fodor and J. Katz, *The Structure of Language*, Prentice Hall, 1964  
Z. S. Harris, *Papers in Structural and Transformational Linguistics*, Reidel, 1970, 775–794

Firth, J. R. (1957). A synopsis of linguistic theory 1930– 1955. In *Studies in Linguistic Analysis*. Philological Society. Reprinted in Palmer, F. (ed.) 1968.  
*Selected Papers of J. R. Firth*. Longman, Harlow

# Слова в похожих контекстах, “близки по смыслу”

Ничто из того,	что было сказано,	не было существенным.
Ничто из <b>всего</b> ,	что было <b>произнесено</b> ,	не было <b>важным</b> .

Я купил *ИКС* в ближайшем хозмаге.

Пришёл домой, натянул *ИКС* на балконе, повесил брюки.

Заключённые спустились по *ИКС* из окна камеры.

Что можно подставить вместо *ИКС*?

# Что такое “похожесть”?

- **first-order co-occurrence**

(syntagmatic association)

Слова рядом в тексте: “выпил” и “лемонадик”/”  
водичка”/”чаёк”

- **second-order co-occurrence**

(paradigmatic association)

У слов похожие соседи: “Татры” и “Карпаты”,  
“любимец” и “питомец”

# Что такое “похожесть”?

## many faces of similarity

- dog -- cat

- dog -- poodle

- dog -- animal

- dog -- bark

- dog -- leash

- dog -- chair

- dog -- dig

- dog -- god

- dog -- fog

- dog -- 6op

same POS

edit distance

same letters

rhyme

shape

# Каждому слову - по вектору “смысла”

Зачем?

1. **Главное:** что-то вроде transfer learning: вместо BoW  
(мы таким образом используем информацию из большого стороннего корпуса [и это очень полезно])
2. Инструмент подбора синонимов и других “связанных” в некотором смысле пар слов
3. Средство изучения языка!
  - а. Пример: семантическая эволюция для историков:  
<https://nlp.stanford.edu/projects/histwords/>  
(есть и куда более ранние работы)
4. **Fun!** квизы (odd one out), [переписывание романов](#) и другое

# План занятия

## 1. Разреженные векторы

- a. Подход “термы-документы”
- b. Подход “термы-термы”
  - i. Построение
  - ii. HAL
- c. Взвешивание
- d. Вычисление семантической близости
- e. Оценка качества

## 2. Плотные векторы

- a. Матричное разложение
- b. “Предиктивные” подходы

Идеи: как научиться искать похожие по смыслу?



# Уже сталкивались: term-document matrix

Только сейчас нас интересуют не столбцы, а **строки**  
(**векторы слов, а не документов**)

	<b>Земфира -- Небомореоблака</b>	<b>Небо -- Википедия</b>	<b>Фабрика -- Море зовёт</b>	<b>Евгений Онегин Глава 1</b>	<b>Анастасия -- Королева золотого песка</b>
<b>небо</b>	6	60		2	
<b>море</b>	6		10	4	1
<b>облако</b>	6	18			
<b>любовь</b>				6	
<b>песок</b>			1		2

# Уже сталкивались: term-document matrix

Только сейчас нас интересуют не столбцы, а **строки**  
(**векторы слов, а не документов**)

	Земфира -- Небомореоблака	Небо -- Википедия	Фабрика -- Море зовёт	Евгений Онегин Глава 1	Анастасия -- Королева золотого песка
небо	6	60		2	
море	6		10	4	1
облако	6	18			
любовь				6	
песок			1		2

Уже

Только  
(векто

```
>>> import numpy as np
>>> sea = np.array([6,0,10,4,1])
>>> sand = np.array([0,0,1,0,2])
>>> cloud = np.array([6,18,0,0,0])

>>> cosine = lambda x,y: x.dot(y) / np.linalg.norm(x) / np.linalg.norm(y)

>>> cosine(sea, sand) > cosine(sea, cloud)
True
>>> cosine(sea, sand) > cosine(sand, cloud)
True
```

небо

6

60

2

море

6

10

4

1

облако

6

18

любовь

6

песок

1

2

# Обсуждения: term-document matrix

- Для этого способа нужно *много* хороших представительных документов, иначе не заработает
- Размерность сильно зависит от размера коллекции текстов
- Очень важна “тематическая однородность”
- Может, разобьём документы на поддокументы? Скажем, на предложения? **НЕТ** (почему?)

# Обсуждения: term-document matrix

- Для этого способа нужно *много* хороших представительных документов, иначе не заработает
- Размерность сильно зависит от размера коллекции текстов
- Очень важна “тематическая однородность”
- Может, разобьём документы на поддокументы?  
Скажем, на предложения? **НЕТ** (почему?)

А вот мысль **смотреть на меньший контекст**

(который может неоднократно встречаться в разных документах) -- ОК

# План занятия

## ~~1. Разреженные векторы~~

- ~~a. Подход “термы-документы”~~
- b. Подход “термы-термы”
  - i. Построение
  - ii. HAL
- c. Взвешивание
- d. Вычисление семантической близости
- e. Оценка качества

## 2. Плотные векторы

- a. Матричное разложение
- b. “Предиктивные” подходы

# Лучше: word-word (word-context) matrix

Считаем, сколько раз слово оказалось в одном контексте  
(например, в окне  $[-2, 2]$ ) с другим

...в Адмиралтейском районе города Петербурга. 37-летний гражданин попал в [руки **наряда** **полиции** около полуночи] рядом со зданием...

...взрыв прогремел в ночь на среду у входа  
[в **здание** **полиции** в городе] Хельсингборг...

...неизвестные с холодным оружием злоумышленники напали  
[на **наряд** **полиции** на автозаправке]...

В [Выборге **зданию** **полиции** угрожает огонь]...

Получаем огромные разреженные векторы

	наряд	город	полиция	здание
наряд	x	...	...	...
город	...	x	...	...
полиция	2	1	x	2
здание	...	...	...	...
..				
милиция	3	0	1	4

**У похожих слов заполнены примерно одни и те же ячейки**

# Лучше: word-word (word-context) matrix

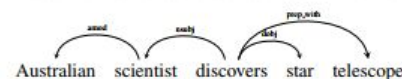
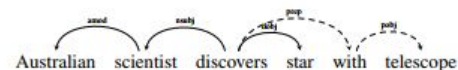
Важно: ‘co-occurrence’ можно определить по-разному

Например, можно брать не просто окно, а какой-то “синтаксически мотивированный” кусок предложения

СМ. "Dependency-Based Word Embeddings", Omer Levy and Yoav Goldberg, 2014  
(там, правда, речь о плотных векторах, о которых мы ещё не говорили)

От выбора контекста зависят свойства полученного вектора

1. Малое окно -- близость будет “синтаксической”
2. Большое окно -- близость будет “схватывать смысл”



WORD	CONTEXTS
australian	scientist/amod <sup>-1</sup>
scientist	australian/amod, discovers/nsubj <sup>-1</sup>
discovers	scientist/nsubj, star/dobj, telescope/prep_with
star	discovers/dobj <sup>-1</sup>
telescope	discovers/prep_with <sup>-1</sup>



# План занятия

## ~~1. Разреженные векторы~~

~~а. Подход “термы-документы”~~

~~б. Подход “термы-термы”~~

~~i. Построение~~

~~ii. HAL~~

с. Взвешивание

д. Вычисление семантической близости

е. Оценка качества

## 2. Плотные векторы

а. Матричное разложение

б. “Предиктивные” подходы

# Пример: HAL (Hyperspace Analogue to Language)

Олдскульный пример: тоже подход с окном, но увеличиваем счётчики для всех пар слов, попавших в окно

Так словам, находящимся близко друг к другу, больше “перепадёт”

**Table 1**  
**Example Matrix for “The Horse Raced Past the Barn Fell”**  
**(Computed for Window Width of Five Words)**

	barn	fell	horse	past	raced	the
<PERIOD>	4	5	0	2	1	3
barn	0	0	2	4	3	6
fell	5	0	1	3	2	4
horse	0	0	0	0	0	5
past	0	0	4	0	5	3
raced	0	0	5	0	0	4
the	0	0	3	5	4	2

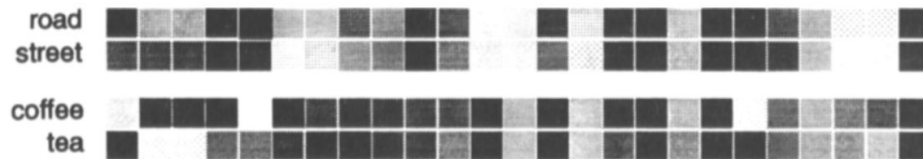


Figure 1. Gray-scaled 25-element co-occurrence vectors.

# Пример: HAL (Hyperspace Analogue to Language)

**Table 2**  
**Five Nearest Neighbors for Target Words**  
**From Experiment 1 (*n1 ... n5*)**

Target	<i>n1</i>	<i>n2</i>	<i>n3</i>	<i>n4</i>	<i>n5</i>
jugs	juice	butter	vinegar	bottles	cans
leningrad	rome	iran	dresden	azerbaijan	tibet
lipstick	lace	pink	cream	purple	soft
triumph	beauty	prime	grand	former	rolling
cardboard	plastic	rubber	glass	thin	tiny
monopoly	threat	huge	moral	gun	large

# План занятия

## ~~1. Разреженные векторы~~

~~а. Подход “термы-документы”~~

~~б. Подход “термы-термы”~~

~~і. Построение~~

~~іі. НАЕ~~

с. Взвешивание

д. Вычисление семантической близости

е. Оценка качества

## 2. Плотные векторы

а. Матричное разложение

б. “Предиктивные” подходы

# Недостатки “просто счётчиков”

Счётчики дадут **слишком** большие значения, например, служебным частям речи и другим общеупотр. словам, при этом -- те не очень информативны, так как встречаются со всем подряд

ВОПРОС: есть идеи, как повлиять на вес **weight(word, context)**, чтобы бесполезные co-occurences имели малый вес?

# Взвесим “важностью”

Для этого у нас уже есть как минимум два хороших средства

Для случая термов-документов (обсуждалось ранее):

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D)$$

...Можно и просто **idf**

Для случая термов-термов:

$$\text{pmi}(x; y) \equiv \log \frac{p(x, y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)}.$$

**Выкидывать стоп-слова надо аккуратно!**

# PMI-weighted word-context matrix

Оценивать вероятности -- счётчиками попаданий контекстов в одно окно для данного слова

contexts



	наряд	город	полиция	здание
наряд	x	...	...	...
город	...	x	...	...
полиция	2	1	x	2
здание	...	...	...	...
..				
милиция	3	0	1	4

words



$$\text{PMI}(w, c) = \log_2 \frac{P(w, c)}{P(w)P(c)}$$

$p(w) = \text{count}(\text{полиция}, *) / \text{all} =$   
сумма строки “полиция” / сумма эл. матрицы

$p(c) = \text{count}(*, \text{здание}) / \text{all} =$   
сумма столбца “здание” / сумма эл. матрицы

$p(w, c) = \text{count}(\text{полиция}, \text{здание}) / \text{all} = 2 /$   
сумма эл. матрицы

# Положительная PMI (Positive PMI)

Мы часто имеем дело с очень редкими словами (одно на миллион, например), и ясно, что оценивать независимость событий с вероятностями порядка  $10^{-6}$  -- так себе идея :(

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^W \sum_{j=1}^C f_{ij}} \quad p_{i*} = \frac{\sum_{j=1}^C f_{ij}}{\sum_{i=1}^W \sum_{j=1}^C f_{ij}} \quad p_{*j} = \frac{\sum_{i=1}^W f_{ij}}{\sum_{i=1}^W \sum_{j=1}^C f_{ij}}$$

$$\text{PPMI}_{ij} = \max\left(\log_2 \frac{p_{ij}}{p_{i*} p_{*j}}, 0\right)$$



# Проблема: (P)PMI “любит” редкие события

Для этого Omer Levy, Yoav Goldberg, Ido Dagan в 2015 был предложен трюк

$$\text{PPMI}_{\alpha}(w, c) = \max(\log_2 \frac{P(w, c)}{P(w)P_{\alpha}(c)}, 0)$$

$$P_{\alpha}(c) = \frac{\text{count}(c)^{\alpha}}{\sum_c \text{count}(c)^{\alpha}}$$

Вдохновением послужили похожие идеи из реализаций word2vec и GloVe  
На всех задачах неплохое качество показало значение 0.75

# Как ещё взвешивать векторы?

t-критерий Стьюдента: оцениваем, насколько далеко друг от друга наблюдаемое среднее и ожидание

“Можем ли отклонить эту гипотезу?”

$$t = \frac{\bar{x} - \mu}{\sqrt{\frac{s^2}{N}}}$$



$$P(a, b) = P(a)P(b)$$

$$\text{t-test}(a, b) = \frac{P(a, b) - P(a)P(b)}{\sqrt{P(a)P(b)}}$$

*Кстати, так же, как и PMI, эту статистику можно использовать для поиска коллокаций*

Почему так можно?

Manning, C. D. and Schütze, H. (1999). "Foundations of Statistical Natural Language Processing. MIT Press.

Curran, J. R. (2003). From Distributional to Semantic Similarity. PhD thesis

# План занятия

## ~~1. Разреженные векторы~~

~~а. Подход “термы-документы”~~

~~б. Подход “термы-термы”~~

~~і. Построение~~

~~іі. НАЕ~~

~~е. Взвешивание~~

d. Вычисление семантической близости

e. Оценка качества

## 2. Плотные векторы

a. Матричное разложение

b. “Предиктивные” подходы

# Как вычислять близость векторов

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Другой взгляд:

1. скалярное произведение как “взвешенное пересечение множеств”
2. знаменатель как средство борьбы с любовью скалярного произведения “раздуваться” из-за больших значений отдельных координат

# Как вычислять близость векторов - 2

“Мягкий” вариант расстояния Жаккара (контекст = элемент множества)

$$\text{sim}_{\text{Jaccard}}(\vec{v}, \vec{w}) = \frac{\sum_{i=1}^N \min(v_i, w_i)}{\sum_{i=1}^N \max(v_i, w_i)}$$

Можно нормализовать векторы так, чтобы сумма элементов каждого была равна 1, и посмотреть на их **сходство как распределений** по KL-divergence

$$D(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

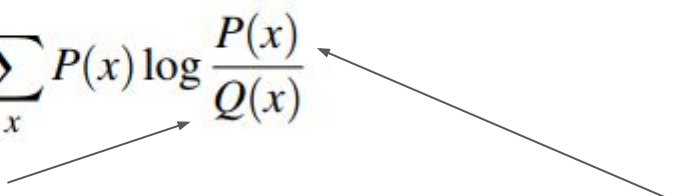
Всё ли здесь ок?

# Как вычислять близость векторов - 2

“Мягкий” вариант расстояния Жаккара (контекст = элемент множества)

$$\text{sim}_{\text{Jaccard}}(\vec{v}, \vec{w}) = \frac{\sum_{i=1}^N \min(v_i, w_i)}{\sum_{i=1}^N \max(v_i, w_i)}$$

Можно нормализовать векторы так, чтобы сумма элементов каждого была равна 1, и посмотреть на их сходство как распределений по KL-divergence

$$D(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$


**У нас много нулей, на которые нельзя делить\* и в которых не задан log**

---

\* если вы не на матанализе

# Как вычислять близость векторов\* - 3

Есть симметричное и конечное расстояние на основе дивергенции Кульбака-Лейблера:

$$D(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

Дивергенция Йенсена-Шеннона, сумма KL-d от каждого из распределений до “среднего”

$$JS(P||Q) = D(P|\frac{P+Q}{2}) + D(Q|\frac{P+Q}{2})$$

В нашем случае вычисляется так

$$\text{sim}_{JS}(\vec{v}||\vec{w}) = D(\vec{v}|\frac{\vec{v}+\vec{w}}{2}) + D(\vec{w}|\frac{\vec{v}+\vec{w}}{2})$$

# План занятия

## ~~1. Разреженные векторы~~

- ~~а. Подход “термы-документы”~~
- ~~б. Подход “термы-термы”~~
  - ~~i. Построение~~
  - ~~ii. НАЕ~~
- ~~в. Взвешивание~~
- ~~г. Вычисление семантической близости~~
- е. Оценка качества

## 2. Плотные векторы

- а. Матричное разложение
- б. “Предиктивные” подходы



# Как оценивать качество

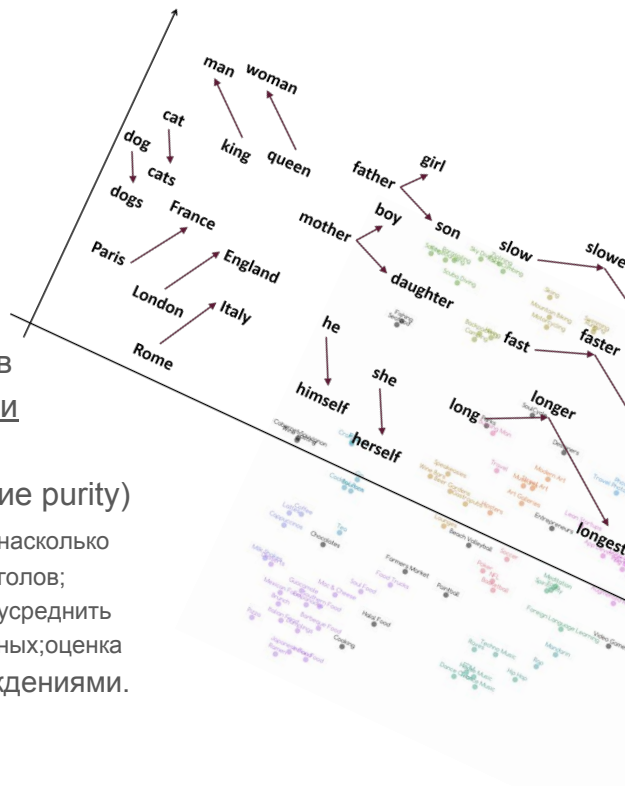
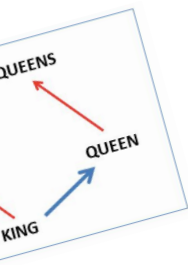
## 1. Extrinsic evaluation

лучший для практиков способ оценки качества модели. Примеры:

- классификация [коротких] текстов
- любая другая полезная задача : )

## 2. Intrinsic evaluation

- mainstream: проверка на датасетах “близких” в разных смыслах слов
- mainstream: проверка на задачах аналогии (как синтаксической, так и семантической)
- кластеризация слов, у которых уже проставлены метки (и вычисление purity)
- \*selectional preference: пусть у нас есть выборка с экспертными суждениями о том, насколько типичны те или иные существительные как дополнения и подлежащая при выбранных глаголах; выберем по корпусу 20 наиболее “ожидаемых” существительных для данных глаголов и усреднить для них **векторы**; вычислим косинусное расстояние от векторов из суждений до усреднённых; оценка качества – корреляция Спирмана между расстояниями и человеческими суждениями.



# План занятия

## ~~1. Разреженные векторы~~

~~а. Подход “термы-документы”~~

~~б. Подход “термы-термы”~~

~~і. Построение~~

~~іі. НАЕ~~

~~с. Взвешивание~~

~~д. Вычисление семантической близости~~

~~е. Оценка качества~~

## 2. Плотные векторы

а. Матричное разложение

б. “Предиктивные” подходы

## “Плотные” векторы

- от десятков тысяч к сотням измерений
- мало нулей
- отход от концепции “координата=слово”

# Но зачем?

Разреженные приписывают каждой координате слово, поэтому

- модели, которым их подают на вход, сложно обучать: большое число параметров усложняет модель
- труднее выучить синонимию, так как у синонимов-контекстов просто разные координаты

# Основные подходы

1. Матричное разложение
2. “Предиктивные”, “нейронные” подходы
3. Кластеризация слов

# План занятия

## ~~1. Разреженные векторы~~

- ~~а. Подход “термы-документы”~~
- ~~б. Подход “термы-термы”~~
  - ~~i. Построение~~
  - ~~ii. НАЕ~~
- ~~в. Взвешивание~~
- ~~г. Вычисление семантической близости~~
- ~~е. Оценка качества~~

## ~~2. Плотные векторы~~

- ~~а. Матричное разложение~~
- ~~б. “Предиктивные” подходы~~

# Матричное разложение

Попробуем понять интуитивно, что и зачем делаем

- 1) понижаем размерность, при этом надеемся сохранить закономерности, которые есть в исходных данных (например, синонимию),
- 2) от такой “проекции” можно оставить только “наиболее значимые” координаты (те, в которых больший разброс значений)

# SVD: сингулярное разложение

Любую матрицу можно представить в виде

$$A = USV^T$$

где матрица **S** -- **диагональная** (и имеет то же число строк и столбцов, что и A), значения на диагонали - сингулярные числа, а **U**, **V** -- **ортогональные**

## Теорема Эркarta-Янга

лучшее приближение (по норме Фробениуса) **A** среди матриц ранга **k** -- это сингулярное разложение, в котором мы в **S** оставили только **k** первых диагональных элементов (если они упорядочены по невозрастанию).



# Приближение матрицей меньшего ранга

Можно поставить задачу немного иначе

**W**: матрица: **w** слов x **m** измерений

“латентного пространства”, при этом

- столбцы ортогональны др. др.
- столбцы упорядочены по убыванию разброса значений в координатах в новом пространстве

**Σ**: диагональная матрица **m** x **m**, каждое значение на диагонали отражает “важность” измерения

**C**: матрица: **m** x **c**

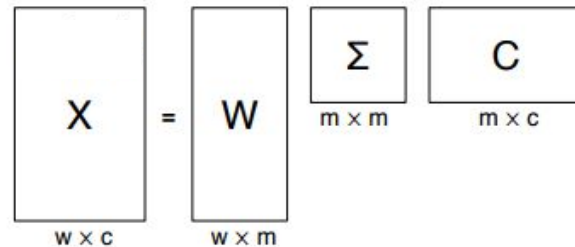
$$\begin{array}{c} \boxed{\text{X}} \\ w \times c \end{array} = \begin{array}{c} \boxed{\text{W}} \\ w \times m \end{array} \begin{array}{c} \boxed{\Sigma} \\ m \times m \end{array} \begin{array}{c} \boxed{\text{C}} \\ m \times c \end{array}$$

# Truncated SVD

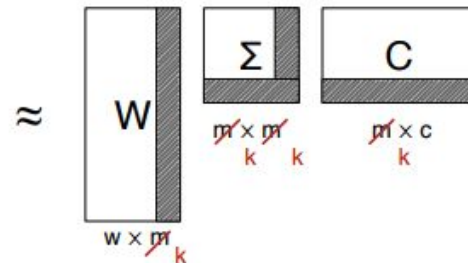
Оставляем только top K измерений

И тогда наши векторные представления слов - это соответствующие строки матрицы  $W_k$  - то есть k-мерные векторы

1) SVD

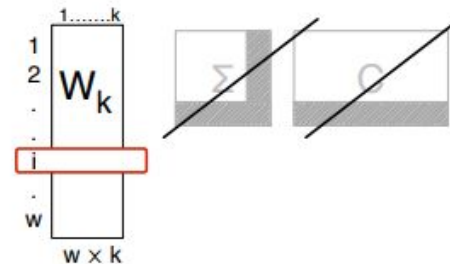


2) Truncation:



3) Embeddings:

embedding for word i:



# LSA: Латентный семантический анализ

	<i>access</i>	<i>document</i>	<i>retrieval</i>	<i>information</i>	<i>theory</i>	<i>database</i>	<i>indexing</i>	<i>computer</i>
Doc 1	x	x	x			x	x	
Doc 2				x*	x			x*
Doc 3			x	x*				x*

Просто применим SVD ( $m = \text{сотни}$ ) к терм-документной матрице, в которой в качестве весов запишем произведение:

локального веса

$$\log f(i, j) + 1$$

и глобального веса

$$1 + \frac{\sum_j p(i, j) \log p(i, j)}{\log D}$$

для всех термов  $i$  во всех документах  $j$

# Truncated SVD for term-term PPMI matrix

Просто применяем SVD к матрице слов-контекстов и обрезаем несколько измерений, подбирая  $k$  вручную. Иногда работает лучше, чем разреженный вариант.

Прочие замечания по SVD как способу получения векторных представлений:

- $(W\Sigma)^T$  также можно рассматривать как векторные представления (но это не особо работает)
- Отрезание до  $k$ , вероятно, помогает лучше обобщать и избавляться от неважной информации
- Часто отбрасывают и **несколько первых измерений**, в некоторых задачах помогает

Но он вычислительно труден

# План занятия

## ~~1. Разреженные векторы~~

- ~~а. Подход “термы-документы”~~
- ~~б. Подход “термы-термы”~~
  - ~~i. Построение~~
  - ~~ii. НАЕ~~
- ~~в. Взвешивание~~
- ~~г. Вычисление семантической близости~~
- ~~е. Оценка качества~~

## ~~2. Плотные векторы~~

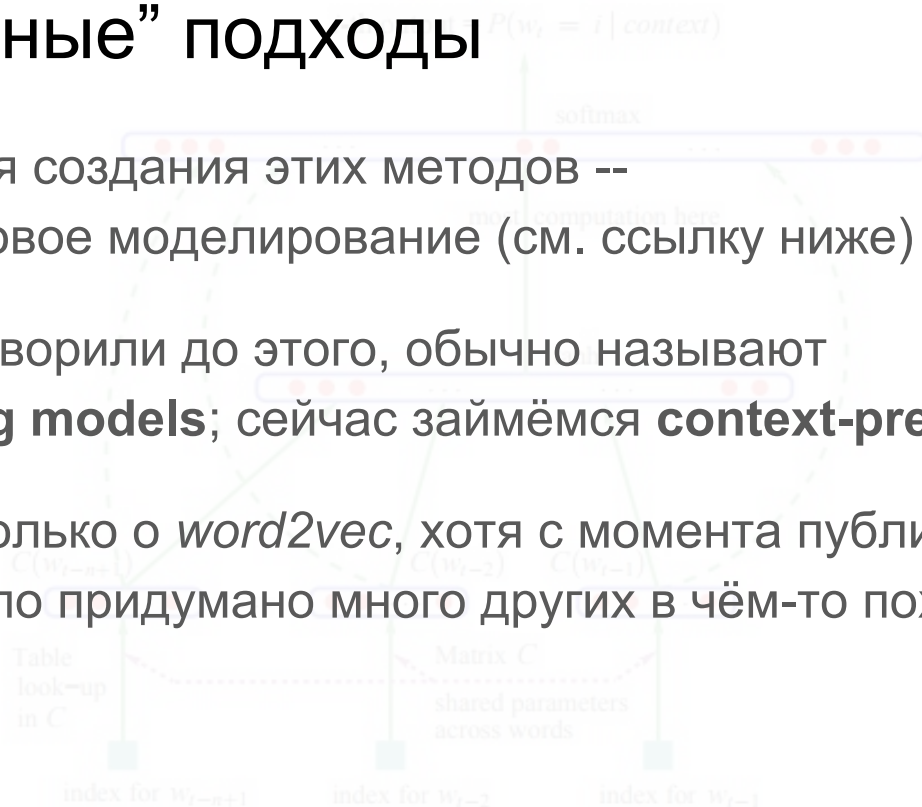
- ~~а. Матричное разложение~~
- ~~б. “Предиктивные” подходы~~

# “Предиктивные” подходы

Вдохновение для создания этих методов --  
нейронное языковое моделирование (см. ссылку ниже)

Всё, о чём мы говорили до этого, обычно называют  
**context-counting models**; сейчас займёмся **context-predicting models**

Мы поговорим только о *word2vec*, хотя с момента публикации статьи и инструмента было придумано много других в чём-то похожих моделей (e.g. fastText)



# Лирическое отступление: хайпанём немножечко?

В 2013 году команда исследователей из Google опубликовала статью с новым способом построения векторных представлений, которые

- 1) отлично определяют схожесть слов
- 2) сохраняют некоторые отношения как разности между векторами

Благодаря PR-машине гугла все программисты узнали о существовании дистрибутивной семантики :)

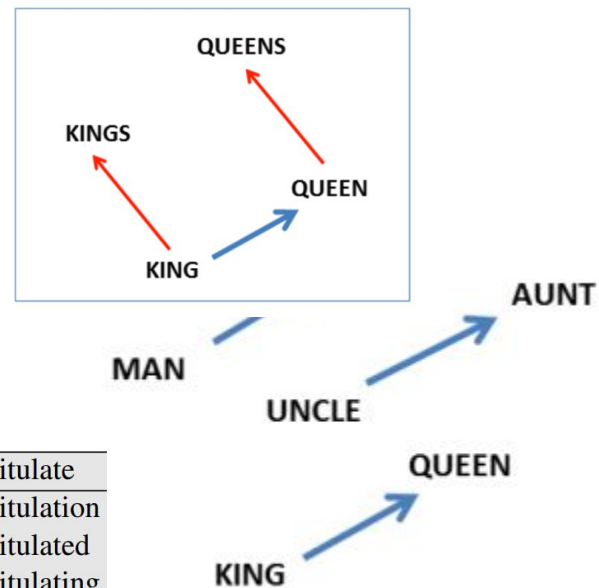
<b>target:</b>	Redmond	Havel	ninjutsu	graffiti	capitulate
	Redmond Wash.	Vaclav Havel	ninja	spray paint	capitulation
	Redmond Washington	president Vaclav Havel	martial arts	grafitti	capitulated
	Microsoft	Velvet Revolution	swordsmanship	taggers	capitulating

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space

// In Proceedings of Workshop at ICLR, 2013

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic Regularities in Continuous Space Word Representations

// In Proceedings of NAACL HLT, 2013



# word2vec - это семейство алгоритмов

**SGNS:** Skip-grams with Negative Sampling

предсказываем “оконные” контексты по слову

**CBOW:** Continuous Bag-of-Words

предсказываем слово по “оконному” контексту, *рассматривать не будем*

Забегая вперёд -- Т. Mikolov:

**Skip-gram:** works well with small amount of the training data, represents well even rare words or phrases.

**CBOW:** several times faster to train than the skip-gram, slightly better accuracy for the frequent words



# skip-grams

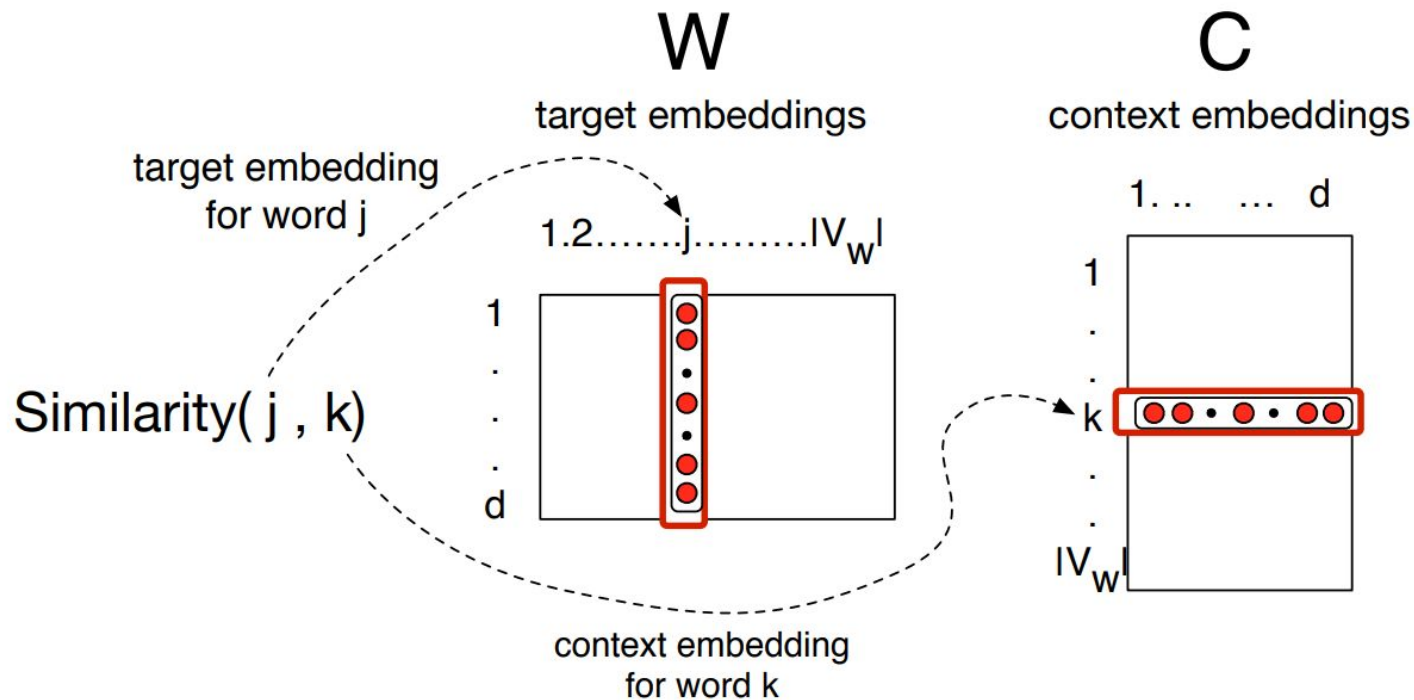
Идём по тексту с окном  $2L$ , и учимся предсказывать контекстные слова для данного слова -- то есть для слова  $w_t$  оцениваем вероятности появления в его окрестностях слов  $w_{t-L} w_{t-L+1} \dots w_{t-1} w_{t+1} \dots w_{t+L}$ .

Предсказываем - корректируем модель на основе **отклонения от истинных значений**  
- предсказываем - корректируем - ...

“На пальцах”:

- 1) Каждому слову и каждому контексту - плотный вектор (изначально случайный)
- 2) Показатель “близости” слова и контекста - скалярное произведение этих векторов
- 3) Будем настраивать значения в векторах так, чтобы  $p(v_{\text{context}} | v_{\text{word}})$  (вычисляется на основе скалярного произведения из (2)) для правильных контекстов были больше

# skip-grams



# skip-grams

Раньше близость была основана на косинусном расстоянии, то есть “нормализованном” скалярном произведении; здесь мы хотим того же

$$\text{Similarity}(j,k) \propto c_k \cdot v_j$$

...но на выходе хотим иметь вероятности. Тогда поможет **softmax**

$$p(w_k|w_j) = \frac{\exp(c_k \cdot v_j)}{\sum_{i \in |V|} \exp(c_i \cdot v_j)}$$

**BTW, проблема:** в знаменателе сумма  $|V|$  скалярных произведений (это долго)

Решается с помощью **negative sampling** или **hierarchical softmax**

# skip-grams with negative sampling

Считать одну вероятность, совершая  $|V|m$  операций умножения и  $|V|(m - 1)$  сложения, и вычислять  $|V|+1$  экспоненту -- дорого

Можем несколько упростить:

1. максимизировать сигмоиды скалярных произведений с **правильными контекстами**,
2. минимизировать сигмоиды скалярных произведений со **случайными контекстами** (это и есть negative samples)

$$\sigma(x) = \frac{1}{1+e^x}$$

# skip-grams with negative sampling

$$\sigma(x) = \frac{1}{1+e^x}$$

Пусть у нас окно размера 2,  
“хорошие” контексты

lemon, a [tablespoon of apricot preserves or] jam  
c1 c2 w c3 c4

ВОТ ЭТО ХОТИМ УВЕЛИЧИТЬ

$$\sigma(c1 \cdot w) + \sigma(c2 \cdot w) + \sigma(c3 \cdot w) + \sigma(c4 \cdot w)$$

k = 2 означает, что  
доля “плохих” 1:2

[cement metaphysical dear coaxial  
n1 n2 n3 n4  
apricot attendant whence forever puddle]  
n5 n6 n7 n8

ВОТ ЭТО ХОТИМ УМЕНЬШИТЬ

$$\sigma(n1 \cdot w) + \sigma(n2 \cdot w) + \dots + \sigma(n8 \cdot w)$$

# skip-grams with negative sampling

Выпишем итоговую ошибку для каждой пары (слово, контекст)

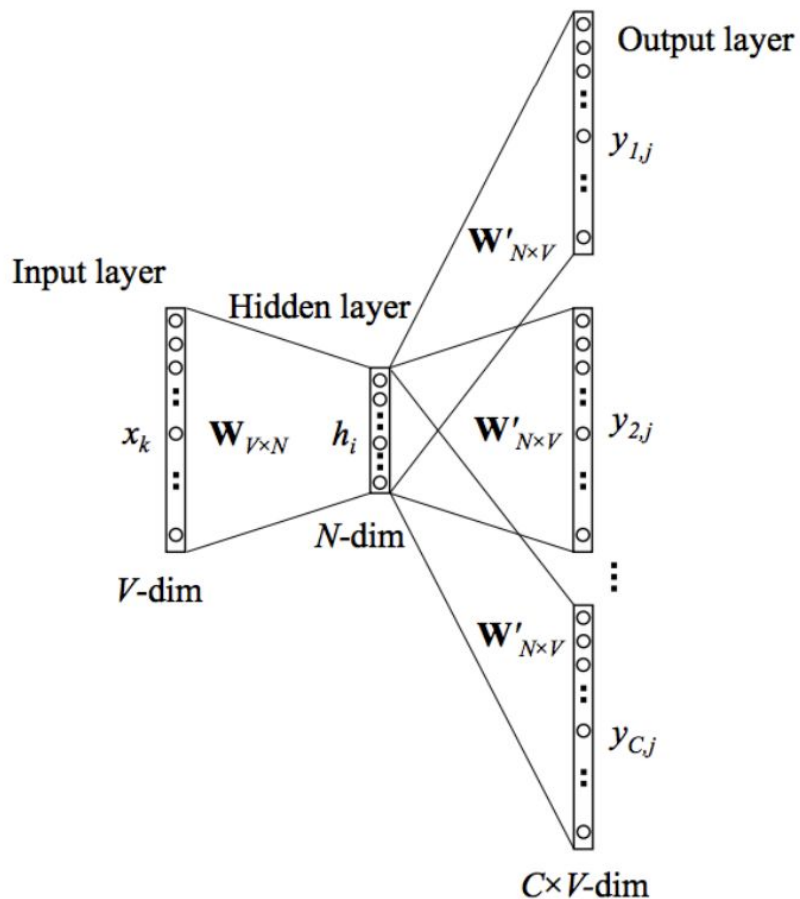
$$\log \sigma(c \cdot w) + \sum_{i=1}^k \mathbb{E}_{w_i \sim p(w)} [\log \sigma(-w_i \cdot w)]$$

Не совсем то же, что softmax, но работает

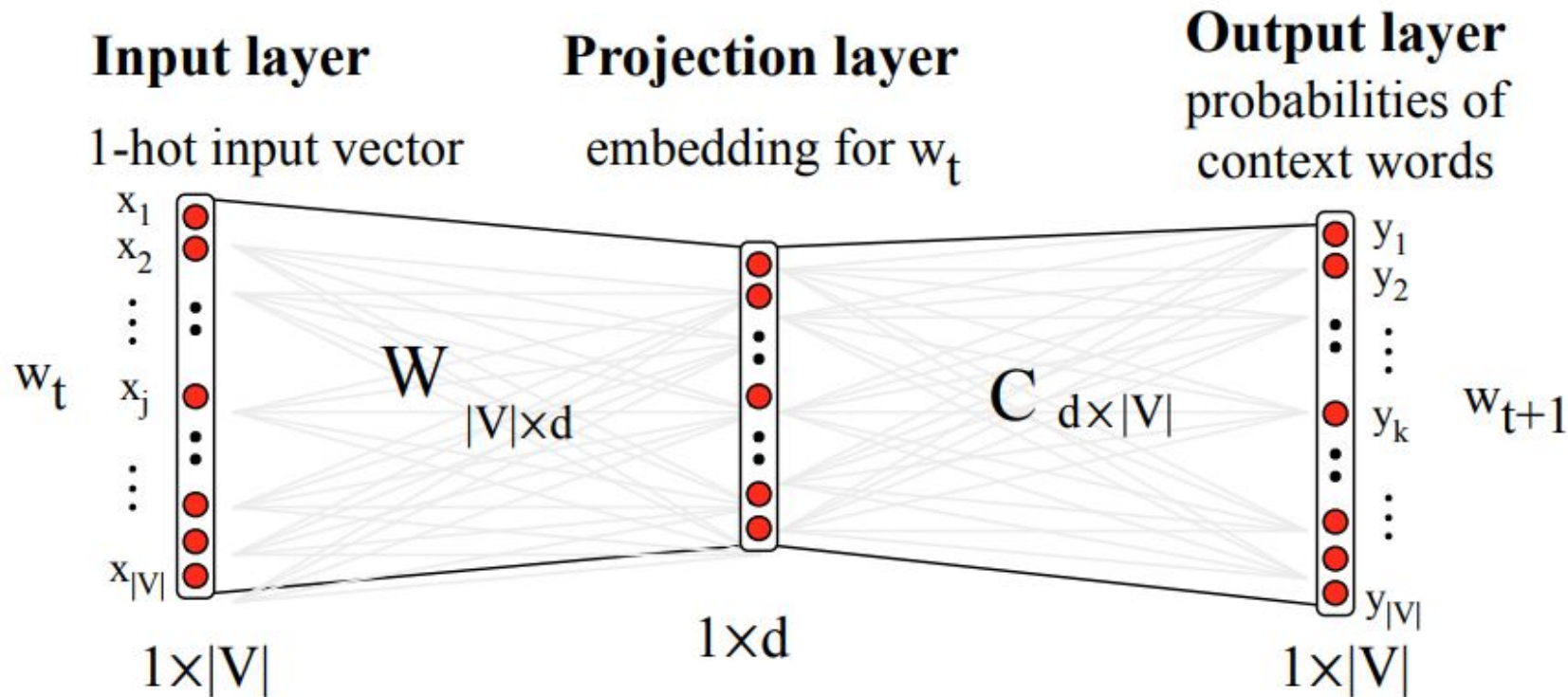
# Взгляд как на “нейронную сеть”

Обучение - обратным  
распространением ошибки, ~ SGD

(см. [тьюториал](#) или [ещё один](#))



# Взгляд как на “нейронную сеть”: один контекст





# Связь с матричным разложением

Доказано, что когда алгоритм skip-gram достигает оптимума, верно, что

$$WC = X^{\text{PMI}} - \log k$$

То есть word2vec - неявное матричное разложение разреженного представления с весами PMI!

Но всё-таки он лучше работает. Почему?

- Introduces many **engineering tweaks** and **hyperparameter settings**
  - May seem minor, but **make a big difference** in practice
  - Their impact is often more significant than the embedding algorithm's

# Инструменты

Открытых моделей/реализаций много, но мейнстрим -

- **gensim**
- **word2vec** (от Google)
- GloVE (Stanford)
- fastText (FacebookAIResearch)
- Реализации в разных библиотеках для глубокого обучения

Готовые векторы для разных языков, например

- [RusVectores](#)
- [Не уверен в этом списке](#) (но можно и погуглить)

# Датасеты

**WordSim-353** - 353 пары существительных с оценкой “близости” от 0 до 10

**SimLex-999** - похоже, но различные части речи + упор на синонимию

**TOEFL dataset** - 80 задачек: слово + ещё четыре, надо выбрать синоним

Также есть датасеты, в которых есть не только слова, но и контексты

## Для русского языка

Переводы стандартных датасетов и данные из тезаурусов

<https://github.com/nlpub/russe-evaluation>

Кстати, ещё не поздно принять участие в соревновании!

<http://russe.nlpub.org/2018/wsi/>

# Also see

Другие популярные векторные представления

**Glove:** J. Pennington, R. Socher, C. Manning. Global Vectors for Word Representation EMNLP2014

**fastText:** P. Bojanowski, E. Grave, A. Joulin, T. Mikolov. Enriching word vectors with subword information, 2016.

Векторные представления для текстов

**doc2vec:** Le Q., Mikolov T. Distributed representations of sentences and documents // ICML-14

Работа с многозначностью слов

**AdaGram:** S. Bartunov, D. Kondrashkin, A. Osokin, D. Vetrov. Breaking Sticks and Ambiguities with Adaptive Skip-gram. International Conference on Artificial Intelligence and Statistics (AISTATS) 2016.

And many more...

# Использованные/рекомендуемые материалы

1. [Martin/Jurafsky, Ch. 15](#)
2. Yoav Goldberg: [word embeddings what, how and whither](#)
3. Статьи на слайдах
4. [Рассказ Валентина Малых из ODS/iPavlov о w2v](#)
5. [Ну совсем хороший туториал](#) по word2vec
6. Wikipedia