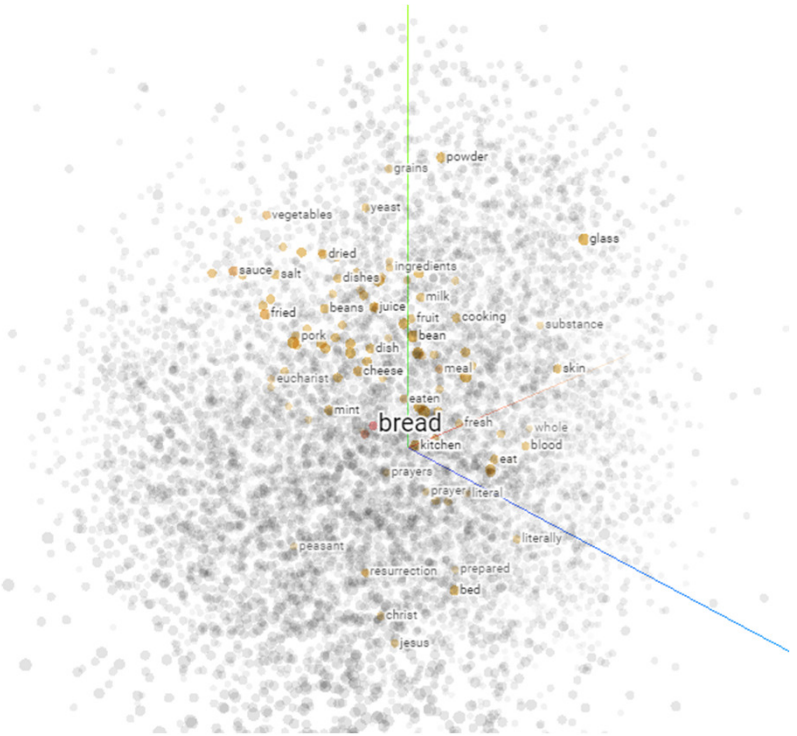
Использование математики для семантического анализа текста

Хафизов Ф.Т.
2021.10.16



http://projector.tensorflow.org/

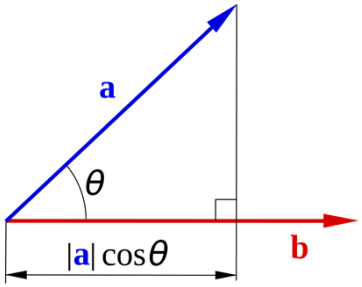
Наша цель:

Показать, что **школьная математика** может быть хорошим инструментом для решения сложных задач

Математика, которая нам понадобится

1. Косинус, вектор, скалярное произведение, матричное умножение
2. Максимизация функции (многих переменных) путём итераций.

Вектор, скалярное произведение, матричное умножение



Косинусное расстояние между векторами := $\cos\theta$

$$\vec{a} \cdot \vec{b} := |\vec{a}|\,|\vec{b}|\,\cos\theta$$

$$\vec{a} \cdot \vec{b} = (a_1, a_2) \cdot (b_1, b_2) = a_1 b_1 + a_2 b_2$$

$$\begin{bmatrix} 6 & 9 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = (6,9) \cdot (-1,1) = 6(-1) + 9(1) = 3$$

$$\begin{bmatrix} 6 & 9 \end{bmatrix} \begin{bmatrix} -1 & 1 & -2 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 6 & -3 \end{bmatrix}$$

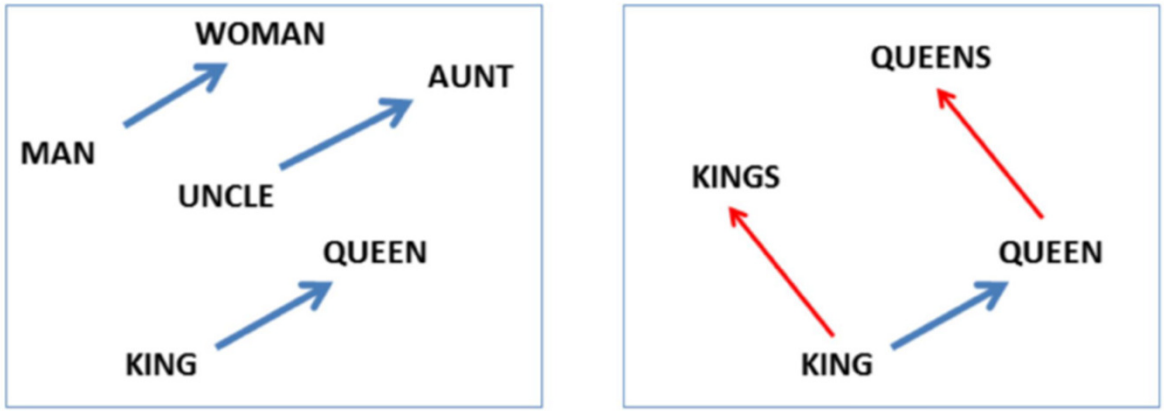
Содержание

1. Вводная часть
   - Какой смысл у слова “смысл”?
   - Как мы узнаём смысл слов?
2. Постановка задачи
3. Алгоритм *Skip-gram* (базовая версия) [1,2]
4. Демонстрация кода
5. Инженерные решения
6. Любопытные примеры

[1] https://arxiv.org/pdf/1301.3781.pdf Mikolov 2013.09
[2] https://arxiv.org/pdf/1310.4546.pdf Mikolov 2013.10

Пример «вычисления» смысла слов [1,2]

$v(\text{king}) - v(\text{man}) + v(\text{woman}) \approx v(\text{queen})$



Вопрос-1: Может ли компьютер распознать смысл слов?

А что мы подразумеваем под «**смыслом**» слов?

Как определить смысл слова они?

- Представители власти отказались дать разрешение демонстрантам, т.к., они боялись беспорядков.

- Представители власти отказались дать разрешение демонстрантам, т.к., они призывали к беспорядкам.

[Терри Виноград]

Как мы распознаём значения слов?

1. **Рифек** получают из молока путём брожения.
2. Чтобы разнообразить привычный способ приготовления блинов попробуем испечь тонкие блины на **рифеке** с дырочками.
3. **Рифек** обладает уникальным набором бактерий и грибков, входящих в его состав.
4. «Заплатите за **рифек**, Шура, — сказал Паниковский, — потом сочтемся».
5. Благодаря своему сложному составу, **рифек** может препятствовать развитию в кишечнике патогенной флоры.
6. Известным популяризатором **рифека** в России был ялтинский врач и климатолог В. Н. Дмитриев.
7. **Рифек** оказывает пробиотическое воздействие, то есть благоприятно влияет на микрофлору кишечника и обмен веществ в целом.

Ответ на Вопрос-1

- Компьютеры (пока) не могут распознавать смысл слов так, как люди.

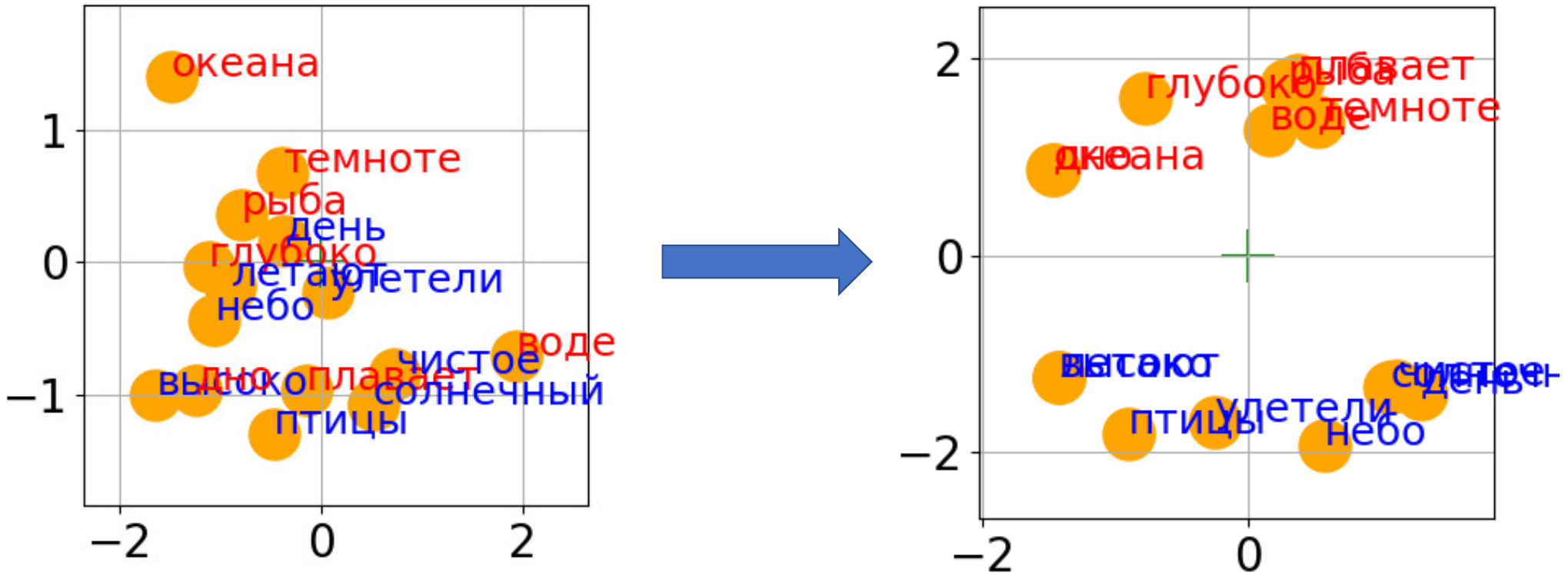
- Однако, компьютеры могут оценить близость слов по анализу контекста.

Постановка задачи:

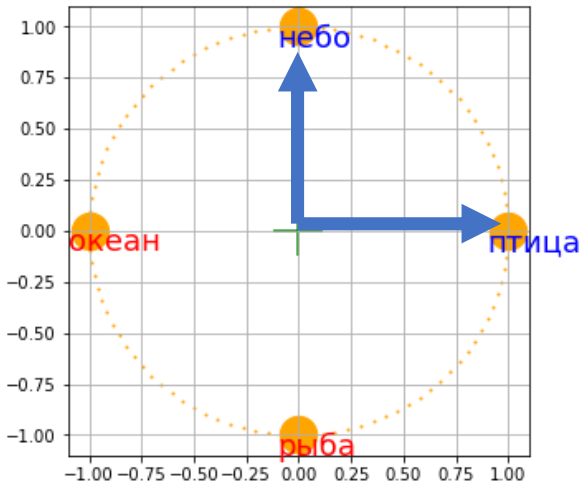
Разделить слова в соответствии с их контекстом

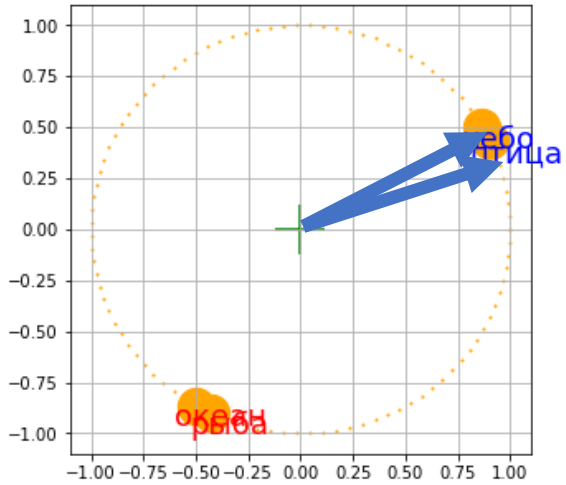
Замечание: задачу будем решать, представляя слова в виде векторов на плоскости

Векторное вложение слов, отражающее сематнику слов в заданном контексте.



Что происходит с косинусом угла между векторами одного цвета?





Метод
(алгоритм *skip-gram* [1,2])

Контекстные пары (0/3)

	center	context
0	рыба	плавает

- Рыба плавает глубоко в воде.
- Дно океана очень глубоко.
- Рыба плавает в темноте.
- Птицы улетели в небо.
- Птицы летают очень высоко.
- В солнечный день небо чистое.

Контекстные пары (1/3)

	center	context
0	рыба	плавает
1	рыба	глубоко

- Рыба плавает глубоко в воде.
- Дно океана очень глубоко.
- Рыба плавает в темноте.
- Птицы улетели в небо.
- Птицы летают очень высоко.
- В солнечный день небо чистое.

Контекстные пары (2/3)

	center	context
0	рыба	плавает
1	рыба	глубоко
2	плавает	рыба

- Рыба плавает глубоко в воде.
- Дно океана очень глубоко.
- Рыба плавает в темноте.
- Птицы улетели в небо.
- Птицы летают очень высоко.
- В солнечный день небо чистое.

Контекстные пары (3/3)

	center	context
0	рыба	плавает
1	рыба	глубоко
2	плавает	рыба
3	плавает	глубоко
4	плавает	воде
...		
39	небо	солнечный
40	небо	день
41	небо	чистое
42	чистое	день
43	чистое	небо

- Рыба плавает глубоко в воде.
- Дно океана очень глубоко.
- Рыба плавает в темноте.
- Птицы улетели в небо.
- Птицы летают очень высоко.
- В солнечный день небо чистое.

Случайное вложение W_0

	vocab	init_embedding
0	воде	[1.94, -0.7]
1	высоко	[-1.64, -0.99]
2	глубоко	[-1.12, -0.04]
3	день	[-0.36, 0.17]
4	дно	[-1.24, -0.96]
5	летают	[-0.9, -0.17]
6	небо	[-1.06, -0.43]
7	океана	[-1.49, 1.4]
8	плавает	[-0.15, -0.97]
9	птицы	[-0.47, -1.29]
10	рыба	[-0.79, 0.35]
11	солнечный	[0.52, -1.07]
12	темноте	[-0.38, 0.68]
13	улетели	[0.07, -0.23]
14	чистое	[0.73, -0.84]

Обучение

Для пары (‘плавает’, ‘глубоко’)
плавает : x ---> x = W_0[8,:] = (-0.1, -1.0)
глубоко : y ---> y’ = W_1[:,2] = $(-0.8, 1.6)^t$

W_0 =
```
[ 1.9, -0.7]
[-1.6, -1. ]
[-1.1, -0. ]
[-0.4,  0.2]
[-1.2, -1. ]
[-0.9, -0.2]
[-1.1, -0.4]
[-1.5,  1.4]
x [-0.1, -1. ]
[-0.5, -1.3]
[-0.8,  0.4]
[ 0.5, -1.1]
[-0.4,  0.7]
[ 0.1, -0.2]
[ 0.7, -0.8]
```

y’

W_1 =
```
[ 0.2 -1.4 -0.8  1.3 -1.5 -1.4  0.6 -1.5  0.4 -0.9  0.3  1.1  0.5 -0.3  1.1]
[ 1.3 -1.2  1.6 -1.4  0.9 -1.2 -1.9  0.9  1.8 -1.8  1.7 -1.3  1.4 -1.7 -1.3]
```

x.y’

x.W_1 = [-1.32, 1.34, -1.52, 1.27, -0.75, 1.34, 1.84, -0.75, -1.84, 1.89, -1.73, 1.19, -1.45, 1.73, 1.19]

exp(x.W_1) = [0.27, 3.82, 0.22, 3.56, 0.47, 3.82, 6.3 , 0.47, 0.16, 6.62, 0.18, 3.29, 0.23, 5.64, 3.29]

Целевая функция

exp(x.W_1) =

```
[0.27, 3.82, 0.22, 3.56, 0.47, 3.82, 6.3 , 0.47, 0.16, 6.62, 0.18, 3.29, 0.23, 5.64, 3.29]
```

‘глубоко’ ‘плавает’

$$P(w_{context}, w_{center}) = P(w_y, w_x) := \frac{\exp(x \cdot y')}{\sum_{v=1}^{|V|} \exp(x \cdot y'_v)}$$

$$U(W_0, W_1) := \frac{1}{T} \sum_{t=1}^{T} \ln P(y_{t+j}, x_t)$$

Алгоритм Skip-Gram пытается максимизировать целевую функцию $U(W_0, W_1)$, которая зависит от векторного вложения наших слов $w_1, w_2, w_3, \ldots, w_T$ выбранных из текста.

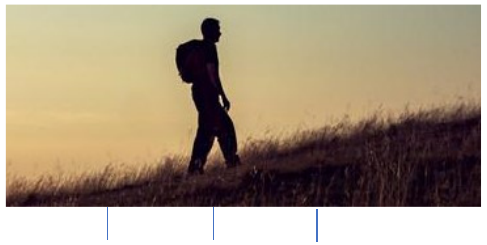
$$\max_{W_0, W_1} U(W_0, W_1)$$

|V|=15 (Размер словаря), T=1-44 (число контекстных пар).

Итерационная максимизация функций

$$f(2+\epsilon) \approx f(2) + \epsilon f'(2)$$

если $f'(2) > 0$, то $f(2.1) \approx f(2) + 0.1 f'(2) > f(2)$



В случае $U(a_1,a_2)$ роль положительной производной играет **<u>градиент</u>**

$$\nabla U(a_1, a_2) := \left(\frac{\partial U}{\partial a_1}, \frac{\partial U}{\partial a_2} \right)$$

Итерационная максимизация функции многих переменных

$\vec{x} = [t_1, t_2]$

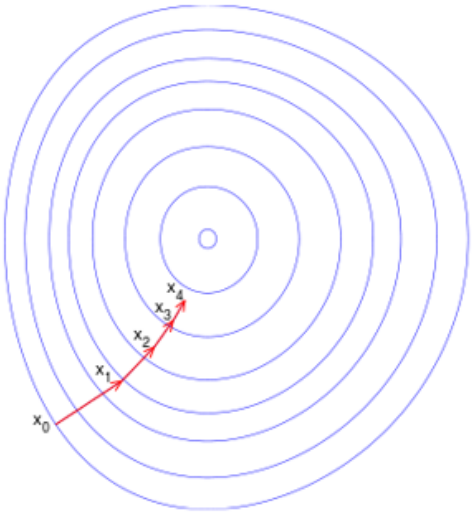
$\nabla U := \left(\frac{\partial U}{\partial t_1} , \frac{\partial U}{\partial t_2} \right)$

Пример: для $U(\vec{x}) = \sin(t_1) + 5t_2, \quad \nabla U(t_1, t_2) = \left(\cos(t_1) , 5 \right).$

$\vec{x}_{n+1} = \vec{x}_n + \epsilon \left(\nabla U(\vec{x}_n) \right)$

$U(\vec{x}_{n+1}) > U(\vec{x}_n)$

Если $\vec{x} = (t_1, t_2) = (\pi/2, 1)$, то $U(\vec{x}) = 6$ и вектор $\nabla U(t_1, t_2) = (0, 5)$ указывает направление наибольшего возрастания функции U из точки $(\pi/2, 1, 6)$.



Вопрос-2: Можно ли упростить алгоритм?

$$W_1 = W_0^t$$

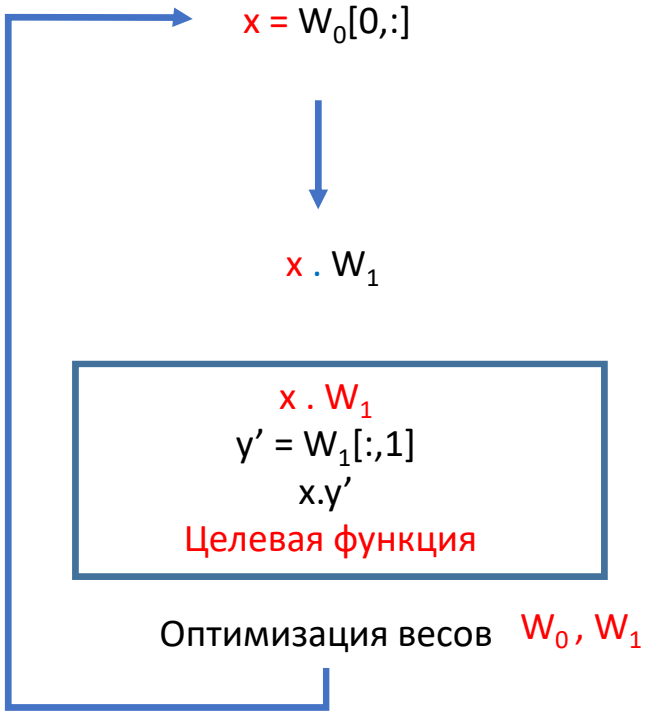
Для пары (‘плавает’, ‘глубоко’)
плавает : x ---> x = W_0[8,:] = (-0.1, -1.0)
глубоко : y ---> y’ = W_1[:,1] = $(-1.4, -1.2)^t$

W_0 =

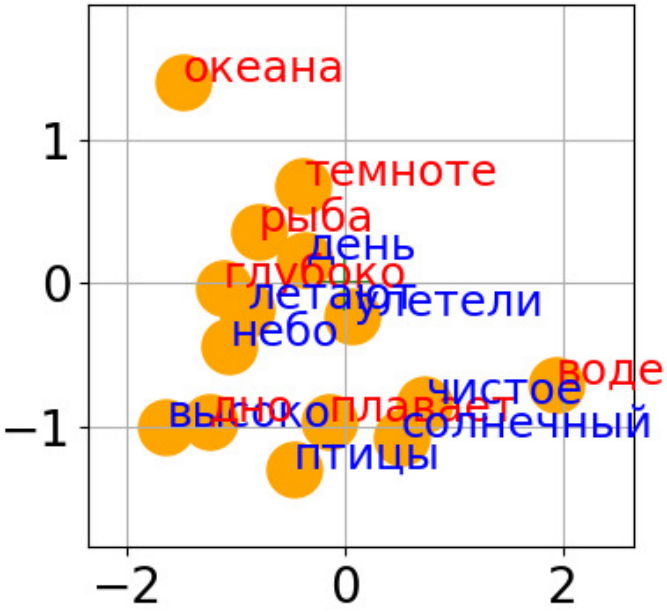
```
[ 1.9, -0.7]
[-1.6, -1. ]
[-1.1, -0. ]
[-0.4,  0.2]
[-1.2, -1. ]
[-0.9, -0.2]
[-1.1, -0.4]
[-1.5,  1.4]
[-0.1, -1. ]
[-0.5, -1.3]
[-0.8,  0.4]
[ 0.5, -1.1]
[-0.4,  0.7]
[ 0.1, -0.2]
[ 0.7, -0.8]
```

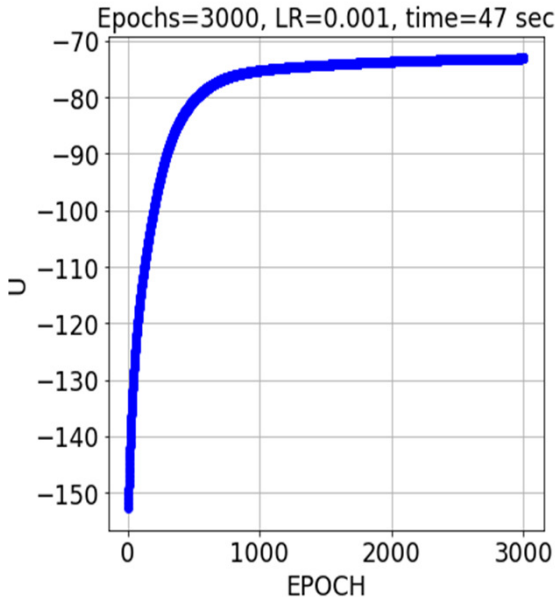
y’

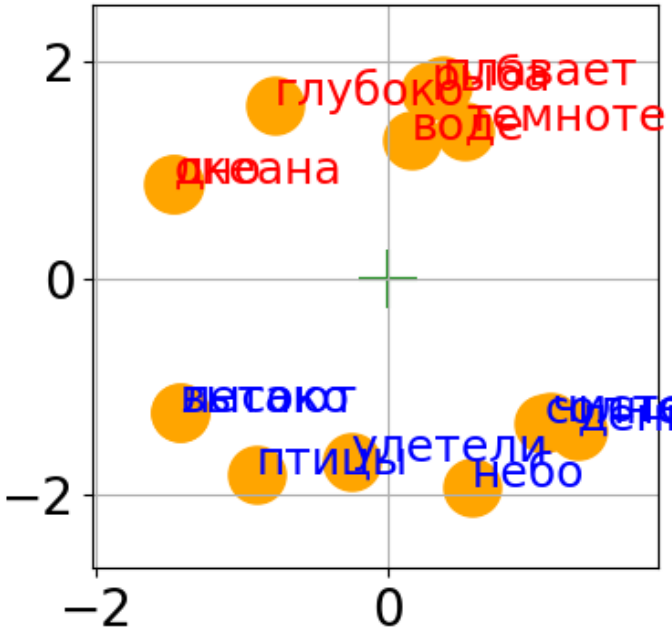
x



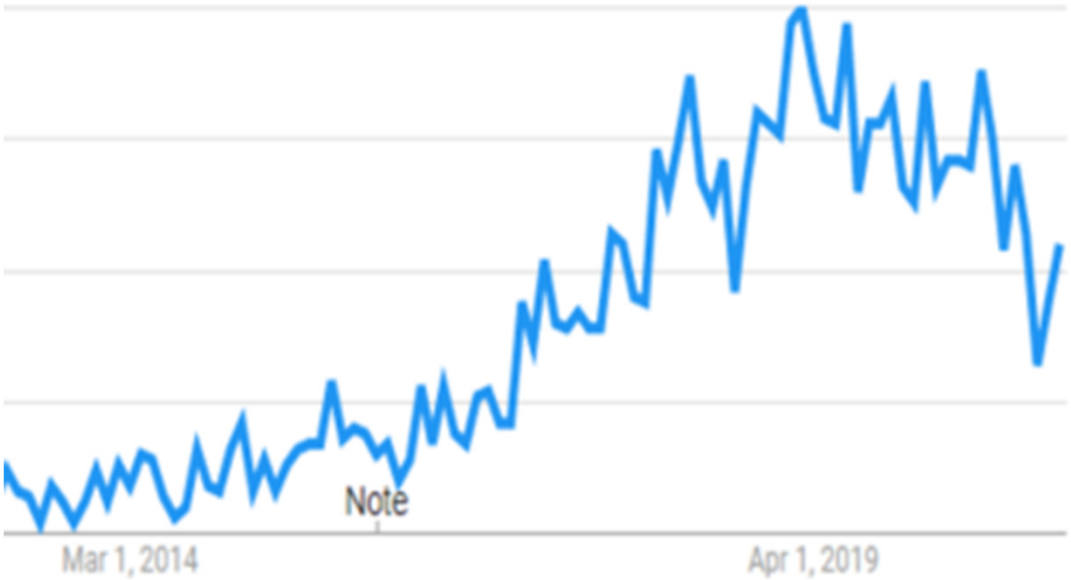
Демонстрация кода





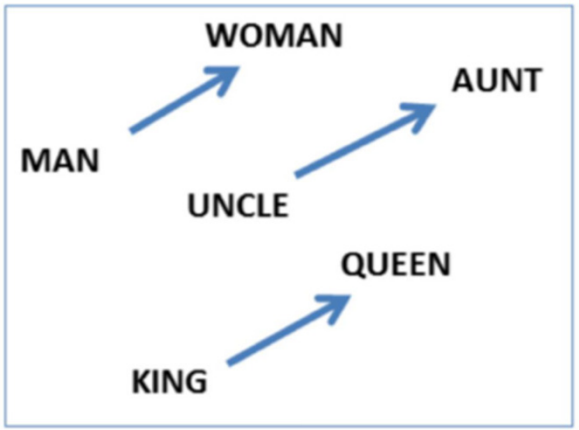


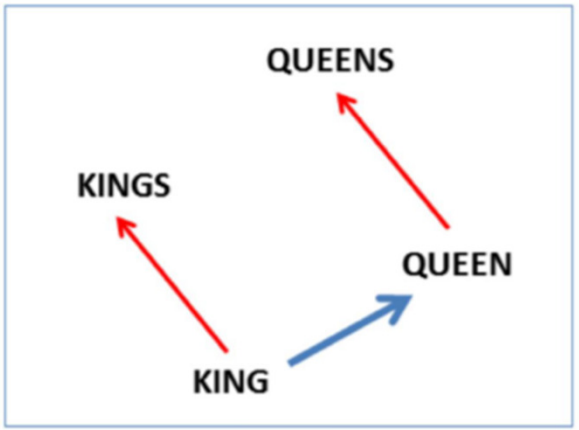
Поиск в сети слов “Word Embedding”



Интересные примеры применения векторного представления слов

$v(\text{king}) - v(\text{man}) + v(\text{woman}) \approx v(\text{queen})$





v(Russia) + v(river) ≈ v(Volga River)
v(Germany) + v(capital) ≈ v(Berlin)

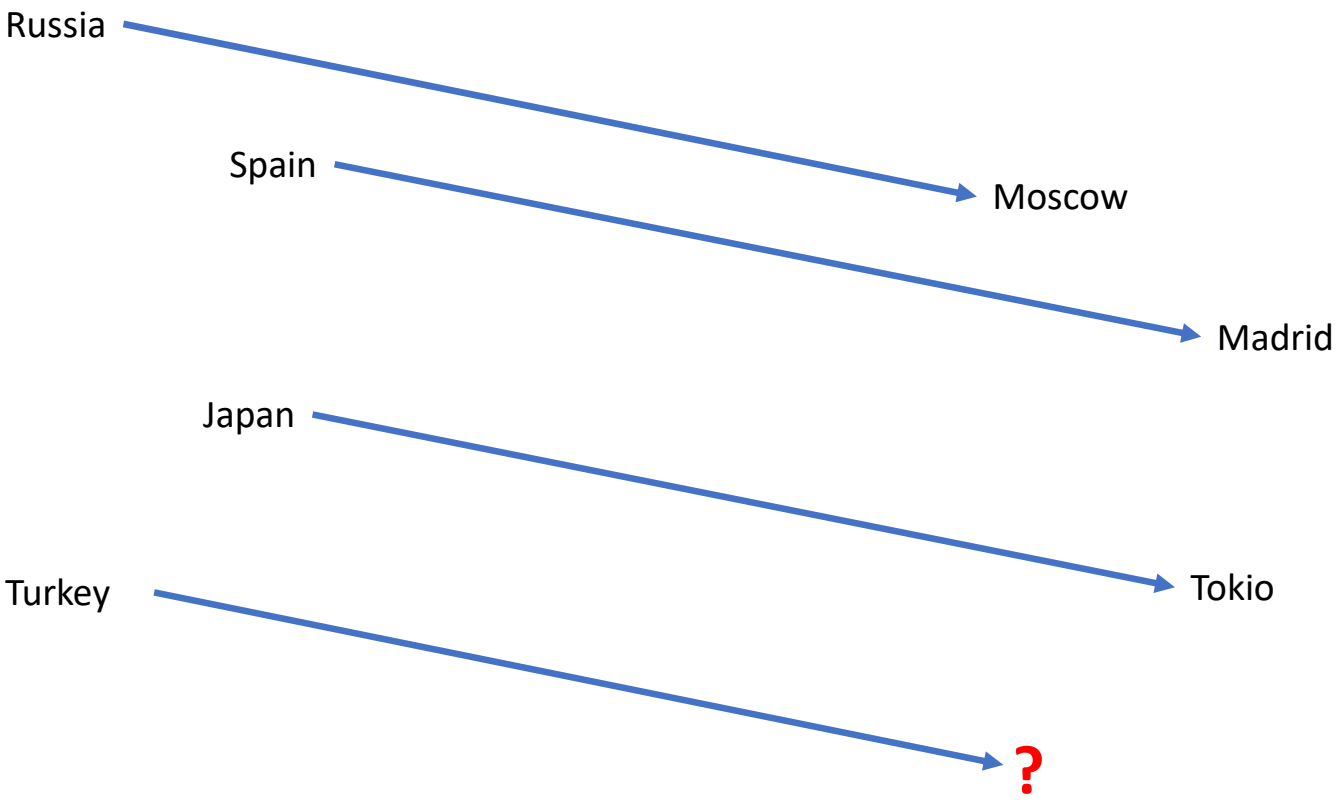
Ref: [1,2]

Алгебраические уравнения из слов

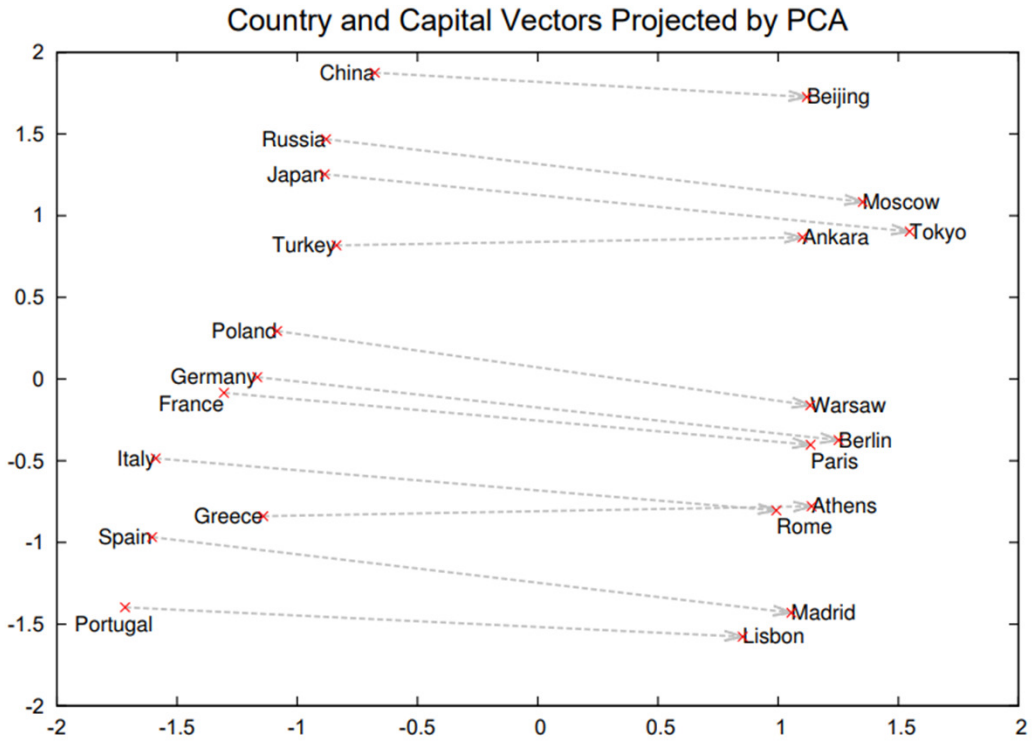
France + (Moscow - Russia) = X

X = ?

Нахождение новых зависимостей



PCA projection of the 1000-dimensional Skip-gram vectors



Mikolov et al, 2013

Ссылки

- [0] Mikolov et al, 2013, Linguistic Regularities in Continuous Space Word Representations, https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/rvecs.pdf
- [1] Mikolov et al 2013.09 https://arxiv.org/pdf/1301.3781.pdf
- [2] Mikolov et al 2013.10 https://arxiv.org/pdf/1310.4546.pdf

Инженерные решения (или как алгоритм становится технологией)

Вопрос-3: Как использовать этот подход на всём тексте Вики?

Сложности

- Огромное количество слов и предложений
- Вычислительная сложность
- Не все слова вносят одинаковый вклад в поиск решения

Инженерные решения

1. Распределённая система вычислений
2. Оптимизация словаря
3. Начинаем выбрасывать, с вероятностью $P(w_i)$, часто повторяющиеся слова

$$P(w_i) = \begin{cases} \sqrt{1 - \frac{t}{f(w_i)}} & , \quad \text{if } f(w_i) > t \\ 0 & , \quad \text{otherwise} \end{cases}$$

4. Дополняем целевую функцию отрицательными примерами

Word Embedding Demo

https://turbomaze.github.io/word2vecjson/

Similar Words

Enter a word and see words with similar vectors.

doctor | List words

doctor	1.0000000000000002
physician	0.7806019127031032
doctors	0.7476568731527384
surgeon	0.6793393714387082
dentist	0.6785442117848048
nurse	0.6319524227288814
psychiatrist	0.614703850361634
medical	0.5671389130686404
clinic	0.5499804910039348
therapist	0.5283346636619084

Word Algebra

Enter all three words, the first two, or the last two and see the words that result.

madrid + (russia - moscow) = Get result

spain	0.7905075552539405
madrid	0.7650632609053115

Word Algebra

Enter all three words, the first two, or the last two and see the words that result.

china + (moscow - russia) = Get result

shanghai	0.779147686225549
china	0.7691120887831315
chinese	0.6720659726186436
hu	0.5964189163973439
yuan	0.5946876191518002