

Глубокое обучение и вообще

Андронов Дмитрий и Соловей Владислав

26 июля 2022 г.

Посиделка 7: Из слов в вектора и обратно

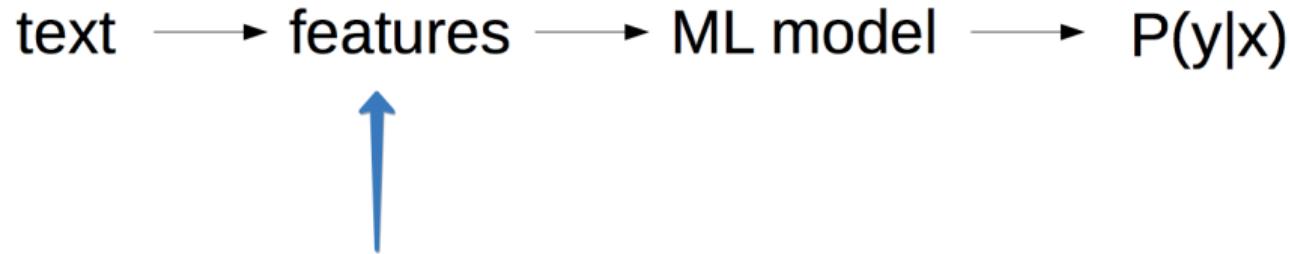
Agenda

- Небольшое введение в анализ текстов
- Представления для текстов, эмбединги
- Классификация текстов

NLP (Natural Language Processing)

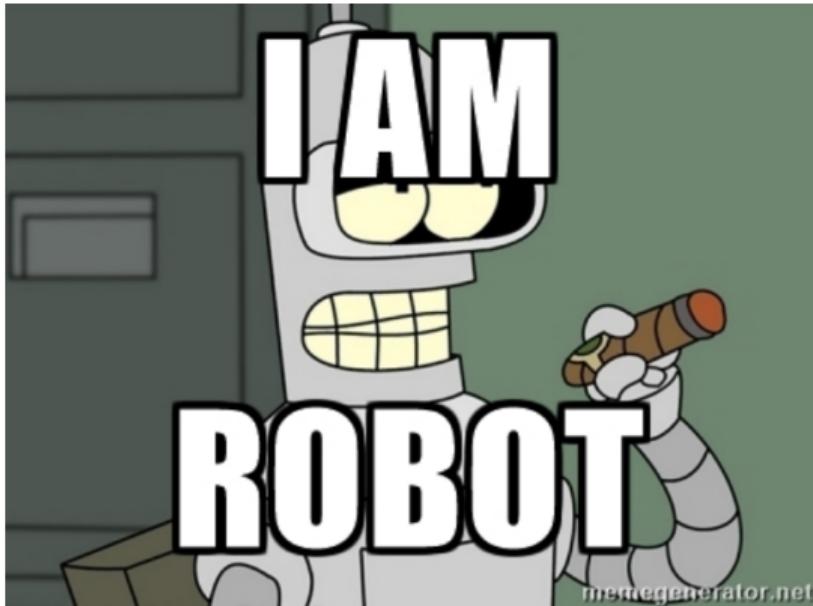


Модели на текстах



Как представить текст
в виде, который могла
бы понять модель?

Что такое текст?



- Текст (документ) — это последовательность токенов (слов)
- Токен (слово) — это последовательность символов

Мешок слов

1. Нежился на пляже
2. Копали яму на пляже
3. Копал картошку
4. Ел картошки и картошку



	нежиться	пляж	копать	яма	картошка	есть
1	1	1	0	0	0	0
2	0	1	1	1	0	0
3	0	0	1	0	1	0
4	0	0	0	0	2	1

Мешок слов

- В словаре много слов, у нас будет слишком много признаков
- В векторе нет никакой информации о смысле слова
- Семантически похожие тексты могут иметь очень разные представления
- Непонятно, что делать, если в тексте появляется какое-то новое слово

Гипотеза мешка слов

- **Гипотеза мешка слов:** нам плевать на взаимное расположение слов. Порядок слов в предложении никак не сказывается на его смысле. **Следуя гипотезе, мы теряем часть информации.**
- Рассматриваем каждое слово, как переменную \Rightarrow большое пространство признаков. Нужно его урезать \Rightarrow **Теряем ещё информацию.**
- Хотим маленькое пространство признаков и много информации в нём!

Лемматизация

Мы хотим уменьшить словарь слов, но сохранив как можно больше информации. Лемматизация - процесс перевода слов по какому-то словарю. Для корректной работы нужен постоянно обновляющийся и большой словарь слов.

Стемминг

Отрубаем по заданному правилу разные куски слова. Всякие приставки суффиксы и тому подобное. Не зависим от словаря слов, но как и любые эвристики могут привести к непредсказуемым результатам: когда-то слишком сильно слово изменим, когда-то наоборот не сможем привести к нормальной форме.

Стоп-слова

Стоп-слова — это часто используемые слова, которые не вносят никакой дополнительной информации в текст. Слова типа "в", "не", "о" ("the", "is", "a") не несут никакой ценности и только добавляют шум в данные.

В библиотеке NLTK есть встроенный список стоп-слов, который можно использовать, чтобы удалить стоп-слова из текста. Однако это не универсальный список стоп-слов для любой задачи, мы также можем создать свой собственный набор стоп-слов в зависимости от сферы.

Tf-idf

- tf (term frequency): считаем частоту слова в каждом документе корпуса
- Если слово встречается в документе часто, но оно не стоп-слово, оно важное
- Способы посчитать:
 - частота слова
 - $\log(\text{частота})$
 - любой свой способ

Tf-idf

- **idf (inverse document frequency)**: считаем вес для редких слов в разрезе всего корпуса текстов
- Слово встретилось только в этом документе, а в других нет, значит оно важное и описывает природу этого документа
- Способы посчитать:
 - $idf = 1$
 - $idf = \log \frac{N}{n_t}$
 - $idf = \log \left(1 + \frac{N}{n_t} \right)$
 - любой свой способ

Tf-Idf Encoding

1. Нежился на пляже
2. Копали яму на пляже
3. Копал картошку
4. Ел картошки и картошку

Tf

	нежиться	пляж	копать	яма	картошка	есть
1	1/6	1/6	0	0	0	0
2	0	1/6	1/6	1/6	0	0
3	0	0	1/6	0	1/6	0
4	0	0	0	0	2/6	1/6

X

Idf

	нежиться	пляж	копать	яма	картошка	есть
1	$\ln 4$	$\ln 2$	0	0	0	0
2	0	$\ln 2$	$\ln 2$	$\ln 4$	0	0
3	0	0	$\ln 2$	0	$\ln 2$	0
4	0	0	0	0	$\ln 2$	$\ln 4$

=

Tf-idf

	нежиться	пляж	копать	яма	картошка	есть
1	0.23	0.11	0	0	0	0
2	0	0.11	0.11	0.23	0	0
3	0	0	0.11	0	0.11	0
4	0	0	0	0	0.23	0.23

=

Tf-Idf Encoding

- выбирая пороговые значения tf и idf можем решать какое число фичей взять в модель
- idf убивает стоп-слова из-за того, что они встречаются почти в каждом документе
- tf убивает редкие слова

Анализ текстов в одном слайде



Порядок слов неважен

Неважен слов порядок

Слов порядок неважен

Он был хорошим человеком и джедаем.
Люди держат деньги в банке.
Падаван, дай мне огурец из банки.



1. токенизация
2. очистка от стоп-слов

[~~он~~, был, хорошим, человеком, ~~и~~, джедаем]
[люди, держат, деньги, ~~в~~, банке]
[падаван, дай, ~~мне~~, огурец, ~~из~~, банки]

5. ОНЕ или tf-idf,
а затем
моделирование

лемматизация

[быть, хороший, человек, джедай]
[человек, держать, деньги, банк]
[падаван, дать, огурец, банка]

3. нормализация

стемминг

[бы, хорош, чел, джед]
[люд, держ, ден, банк]
[падаван, дай, огур, банк]

4. очистка от слишком редких слов

Как сделать модель лучше?

МОЖЕТ ХВАТИТ ПРИМЕРОВ С ХУРМОЙ

УНИГРАММЫ:

1. МОЖЕТ
2. ХВАТИТ
3. ПРИМЕРОВ
4. С
5. ХУРМОЙ

МОЖЕТ ХВАТИТ ПРИМЕРОВ С ХУРМОЙ

БИГРАММЫ:

1. МОЖЕТ ХВАТИТ
2. ХВАТИТ ПРИМЕРОВ
3. ПРИМЕРОВ С
4. С ХУРМОЙ

МОЖЕТ ХВАТИТ ПРИМЕРОВ С ХУРМОЙ

ТРИГРАММЫ:

1. МОЖЕТ ХВАТИТ ПРИМЕРОВ
2. ХВАТИТ ПРИМЕРОВ С
3. ПРИМЕРОВ С ХУРМОЙ

- Сейчас мы используем слова как фичи
- Почему бы не использовать пары из слов?
- В общем виде n-граммы - последовательности из n слов
- Словарь разрастётся, но результат, возможно, улучшится

https://vas3k.ru/blog/machine_translation/

Дистрибутивная гипотеза

- **Дистрибутивная гипотеза:** слова с похожим смыслом будут встречаться в похожих контекстах.
- Мы можем попытаться заложить в векторное представление слова информацию о его контексте
- Есть разные подходы: count-based и prediction-based, мы поговорим о втором

Из слов в вектора и обратно



Embedding

- **Ключевое понятие лекции** - Embedding (эмбединг).
- Embedding - векторное представление любой сущности (токен, предложение, картинка, узел графа и т.д.)

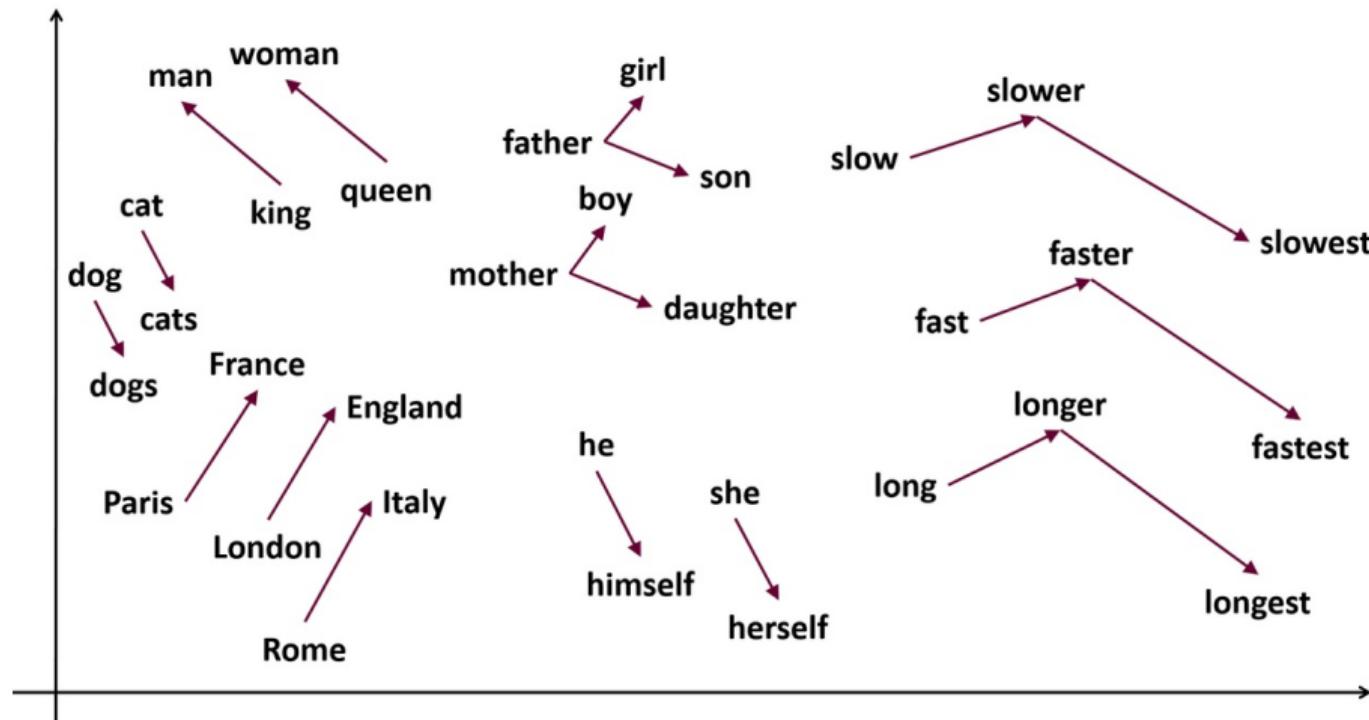
Words embeddings

- **Наша цель:** хотим научить компьютер понимать слова
- Идея! Давайте превратим наши слова в вектора размера d
- На вектора понакладываем хотелок!



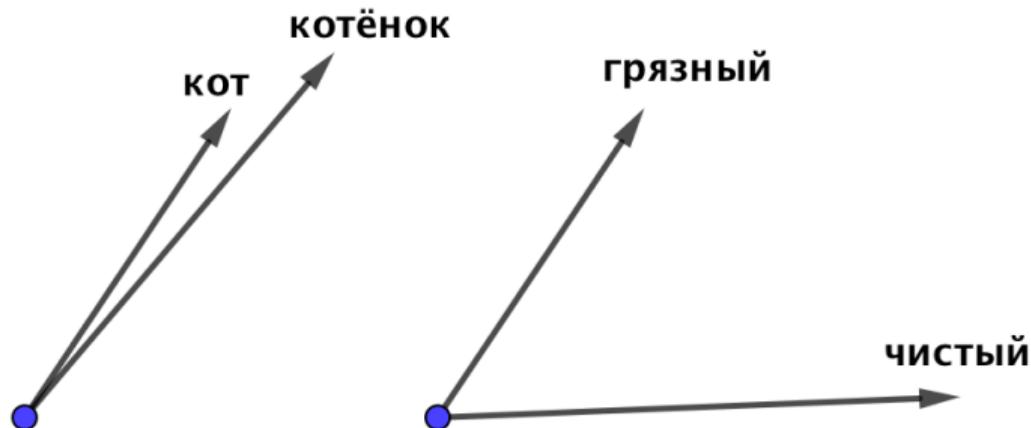
Хотелка первая

- Хотим, чтобы модель улавливала семантические свойства слов



Хотелка вторая

- Модель понимала, где близкие по смыслу слова: кот, котёнок, кошка, тигр, лев, ...



Хотелка третья

- Арифметика!



$$-\text{pink silhouette} + \text{blue silhouette} = \text{Game of Thrones character}$$



Томаш Миколов

- Звучит как магия, но правда работает
- В 2013 году модель предложена чешским аспирантом Томашем Миколовым
- После работал в Google, сейчас ушёл в Facebook



<https://arxiv.org/abs/1301.3781>

Идея word2vec

- **Основная идея:** мы хотим упаковать информацию о контексте слов в вектора
- **Как это сделать:** будем обучать вектора, пытаясь предсказать контекст, в котором встречается слово (skip-gram)

Идея word2vec

- **Основная идея:** мы хотим упаковать информацию о контексте слов в вектора
- **Как это сделать:** будем обучать вектора, пытаясь предсказать контекст, в котором встречается слово (skip-gram)
- **CBOW (непрерывный мешок слов)** — в нём мы по заданному контексту слова пытаемся предсказать слово
- **Skip-gram** — по заданному слову пытаемся предсказать его контекст

w2v верхнеуровнево

... I saw a cute grey cat playing in the garden ...

- Собрали большой корпус из текстов

w2v верхнеуровнево

... I saw a $cute$ $grey$ cat playing in the garden ...
 w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

- Собрали большой корпус из текстов
- Идём по тексту скользящим окном

w2v верхнеуровнево

... I saw a $cute$ $grey$ cat playing in the garden ...
 w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}
context central context
words word words

- Собрали большой корпус из текстов
- Идём по тексту скользящим окном

w2v верхнеуровнево

$$P(w_{t-2} | \text{w}_t) \quad P(w_{t-1} | \text{w}_t) \quad P(w_{t+1} | \text{w}_t) \quad P(w_{t+2} | \text{w}_t)$$

... **I** **saw** **a** **cute** **grey** cat playing in the garden ...

w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

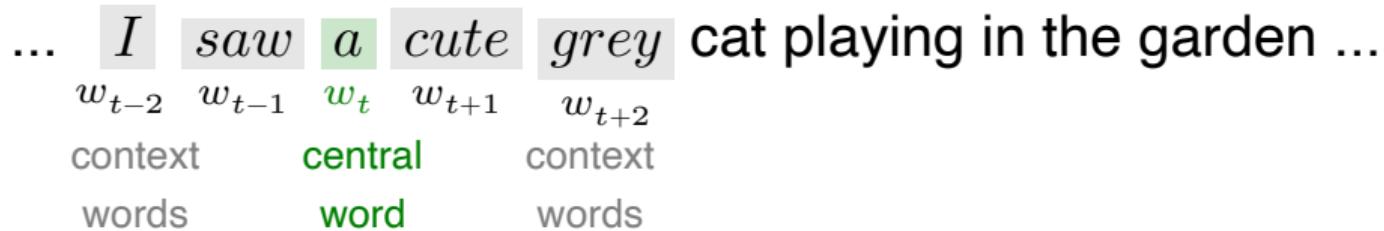
context central context

words word words

- Собрали большой корпус из текстов
- Идём по тексту скользящим окном
- Расчитаем вероятность встретить контекст при условии что центральное слово фиксировано

w2v верхнеуровнево

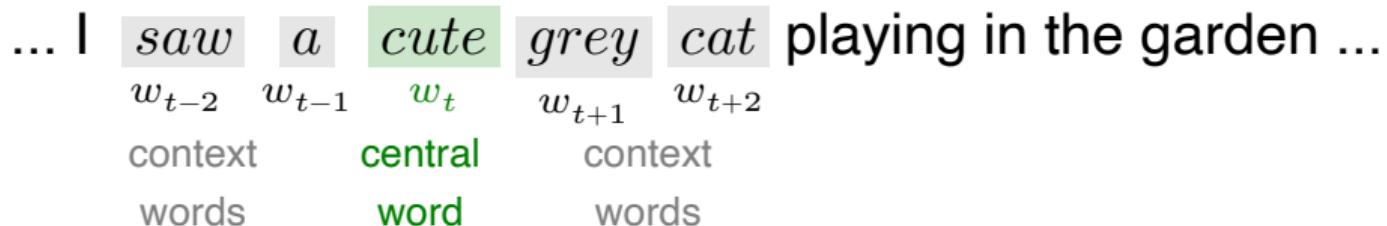
$$P(w_{t-2} | \textcolor{teal}{w_t}) \quad P(w_{t-1} | \textcolor{teal}{w_t}) \quad P(w_{t+1} | \textcolor{teal}{w_t}) \quad P(w_{t+2} | \textcolor{teal}{w_t})$$



- Собрали большой корпус из текстов
- Идём по тексту скользящим окном
- Расчитаем вероятность встретить контекст при условии что центральное слово фиксировано
- Вектора w_i должны максимизировать вероятности

w2v верхнеуровнево

$$P(w_{t-2} | w_t) \quad P(w_{t-1} | w_t) \quad P(w_{t+1} | w_t) \quad P(w_{t+2} | w_t)$$



- Собрали большой корпус из текстов
- Идём по тексту скользящим окном
- Расчитаем вероятность встретить контекст при условии что центральное слово фиксировано
- Вектора w_i должны максимизировать вероятности

w2v верхнеуровнево

$$P(w_{t-2} | w_t) \quad P(w_{t-1} | w_t) \quad P(w_{t+1} | w_t) \quad P(w_{t+2} | w_t)$$

... I saw a $cute$ $grey$ cat $playing$ in the garden ...

w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

context central context

words word words

- Собрали большой корпус из текстов
- Идём по тексту скользящим окном
- Расчитаем вероятность встретить контекст при условии что центральное слово фиксировано
- Вектора w_i должны максимизировать вероятности

w2v верхнеуровнево

$$P(w_{t-2} | w_t) \quad P(w_{t-1} | w_t) \quad P(w_{t+1} | w_t) \quad P(w_{t+2} | w_t)$$

... I saw a *cute* *grey* *cat* *playing* *in* the garden ...

w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

context central context

words word words

- Собрали большой корпус из текстов
- Идём по тексту скользящим окном
- Расчитаем вероятность встретить контекст при условии что центральное слово фиксировано
- Вектора w_i должны максимизировать вероятности

Правдоподобие модели

- word2vec пытается максимизировать правдоподобие текстов:

$$L(\theta) = \prod_{t=1}^T \prod_{\substack{j \neq 0 \\ -m \leq j \leq m}} P(w_{t+j} | w_t, \theta)$$

- Чтобы получить функцию потерь, прологарифмируем и умножим на -1 :

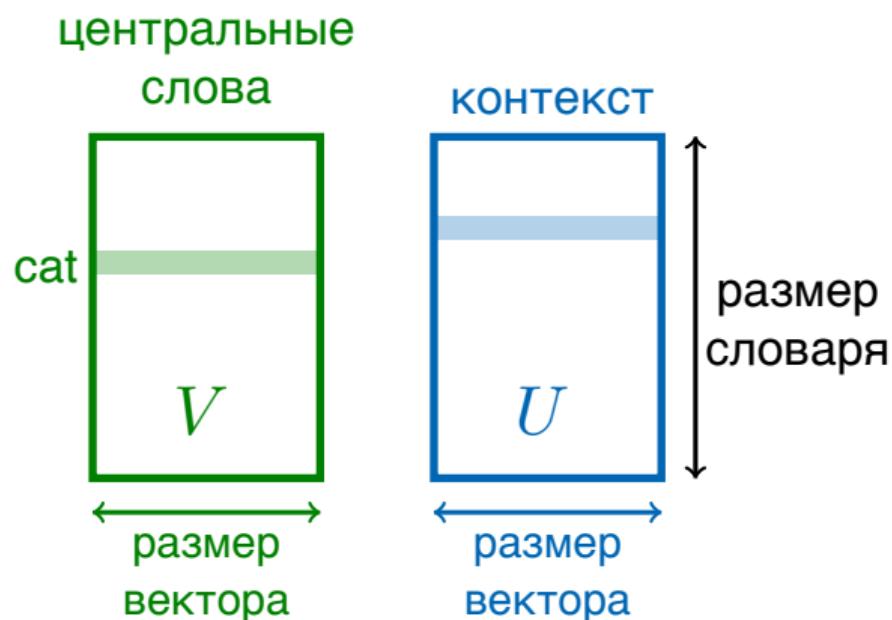
$$Loss = J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{j \neq 0 \\ -m \leq j \leq m}} \log P(w_{t+j} | w_t, \theta)$$

- Осталось договориться как мы будем считать вероятность

Как считать вероятность?

Для каждого слова w будем обучать два вектора:

- v_w — когда это слово центральное
- u_w — когда это слово входит в контекст
- после обучения обычно используют только вектора слов, V
- про вектора контекста, U , забывают



Как считать вероятность?

Для центрального слова c и контекстного слова o :

$$P(o | c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

- обычный softmax
- чем выше $u_o^T v_c$ тем больше вероятность встретить настоящее слово o контексте центрального c
- хотим, чтобы вероятности для пар слов, встречающихся в данных, были высокими, а для пар не встречающихся в данных низкими

Обучение модели



Один шаг обучения в деталях

$P(w_{t-2} | w_t) \quad P(w_{t-1} | w_t) \quad P(w_{t+1} | w_t) \quad P(w_{t+2} | w_t)$

... I saw a $cute \ grey \ cat \ playing \ in \ the \ garden \ ...$

$w_{t-2} \quad w_{t-1} \quad w_t \quad w_{t+1} \quad w_{t+2}$

context central context

words word words

$$Loss = J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{j \neq 0 \\ -m \leq j \leq m}} \log P(w_{t+j} | w_t, \theta)$$

$$\theta = \{U, V\}$$

Один шаг обучения в деталях

$$P(w_{t-2} | w_t) \quad P(w_{t-1} | w_t) \quad P(w_{t+1} | w_t) \quad P(w_{t+2} | w_t)$$

... I saw a *cute* *grey* *cat* *playing* *in* the garden ...

w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

context central context

words word words

$$Loss = J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{\substack{j \neq 0 \\ -m \leq j \leq m}} J_{t,j}(\theta)$$

$J_{t,j}(\theta)$ — потери для слова j в окне t

Один шаг обучения в деталях

$$P(w_{t-2} | w_t) \quad P(w_{t-1} | w_t) \quad P(w_{t+1} | w_t) \quad P(w_{t+2} | w_t)$$

... I saw a **cute** **grey** **cat** **playing** **in** the garden ...

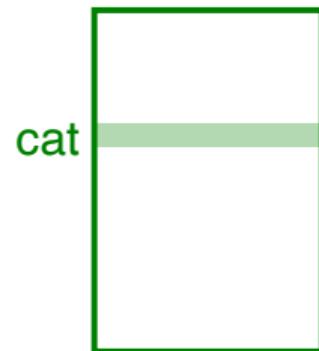
w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

context central context

words word words

Потери на одном слове:

$$-\log P(\text{cute} | \text{cat}) = -\log \frac{\exp(u_{\text{cute}}^T v_{\text{cat}})}{\sum_w \exp(u_w^T v_{\text{cat}})}$$



Один шаг обучения в деталях

$$-\log P(\text{cute} \mid \text{cat}) = -\log \frac{\exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}})}{\sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})} = -\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} + \log \sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})$$

Один шаг обучения в деталях

$$-\log P(\text{cute} \mid \text{cat}) = -\log \frac{\exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}})}{\sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})} = -\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} + \log \sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})$$



$$\mathbf{u}_{w_1}^T \mathbf{v}_{\text{cat}}$$

$$\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}}$$

$$\mathbf{u}_{w_3}^T \mathbf{v}_{\text{cat}}$$

$$\mathbf{u}_{w_4}^T \mathbf{v}_{\text{cat}}$$

...

Один шаг обучения в деталях

$$-\log P(\text{cute} \mid \text{cat}) = -\log \frac{\exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}})}{\sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})} = -\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} + \log \sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})$$



V



U

$$\begin{aligned} \mathbf{u}_{w_1}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(\mathbf{u}_{w_1}^T \mathbf{v}_{\text{cat}}) \\ \mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}}) \\ \mathbf{u}_{w_3}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(\mathbf{u}_{w_3}^T \mathbf{v}_{\text{cat}}) \\ \mathbf{u}_{w_4}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(\mathbf{u}_{w_4}^T \mathbf{v}_{\text{cat}}) \\ \dots & \quad \dots \end{aligned}$$

Один шаг обучения в деталях

$$-\log P(\text{cute} \mid \text{cat}) = -\log \frac{\exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}})}{\sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})} = -\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} + \log \sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})$$



V



U

$$\begin{aligned} u_{w_1}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(u_{w_1}^T \mathbf{v}_{\text{cat}}) \rightarrow \sum_w \exp(u_w^T \mathbf{v}_{\text{cat}}) \\ \mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}}) \\ u_{w_3}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(u_{w_3}^T \mathbf{v}_{\text{cat}}) \\ u_{w_4}^T \mathbf{v}_{\text{cat}} &\rightarrow \exp(u_{w_4}^T \mathbf{v}_{\text{cat}}) \\ \dots & \quad \dots \end{aligned}$$

Один шаг обучения в деталях

Функция потерь:

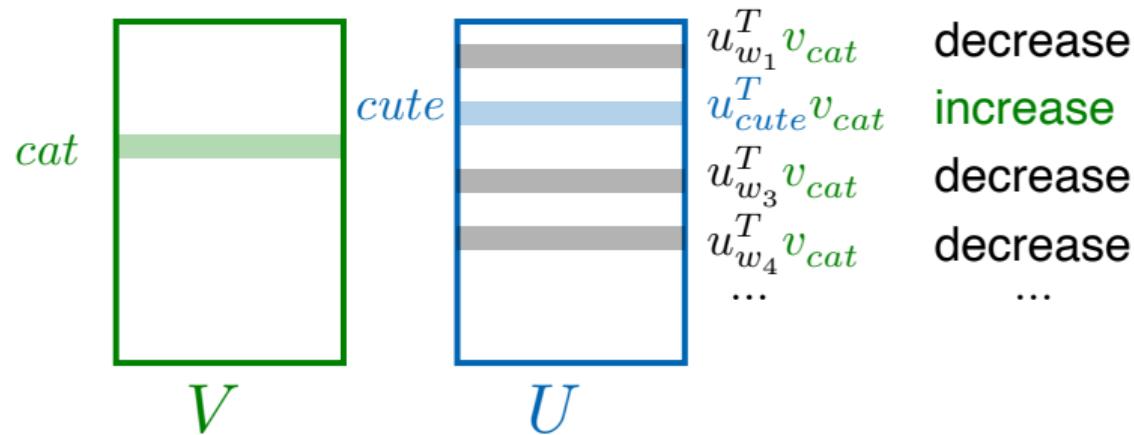
$$J_{t,j}(\theta) = -\log P(\text{cute} \mid \text{cat}) = -\log \frac{\exp(\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}})}{\sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})} = -\mathbf{u}_{\text{cute}}^T \mathbf{v}_{\text{cat}} + \log \sum_w \exp(\mathbf{u}_w^T \mathbf{v}_{\text{cat}})$$

Шаг градиентного спуска:

$$\begin{aligned}\mathbf{v}_{\text{cat}} &= \mathbf{v}_{\text{cat}} - \gamma \cdot \frac{\partial J_{t,j}(\theta)}{\partial \mathbf{v}_{\text{cat}}} \\ \mathbf{u}_w &= \mathbf{u}_w - \gamma \cdot \frac{\partial J_{t,j}(\theta)}{\partial \mathbf{u}_w} \quad \forall w \in V\end{aligned}$$

Один шаг обучения в деталях

$$J_{t,j}(\theta) = -\log P(\text{cute} \mid \text{cat}) = -\log \frac{\exp(u_{\text{cute}}^T v_{\text{cat}})}{\sum_w \exp(u_w^T v_{\text{cat}})} = -u_{\text{cute}}^T v_{\text{cat}} + \log \sum_w \exp(u_w^T v_{\text{cat}})$$

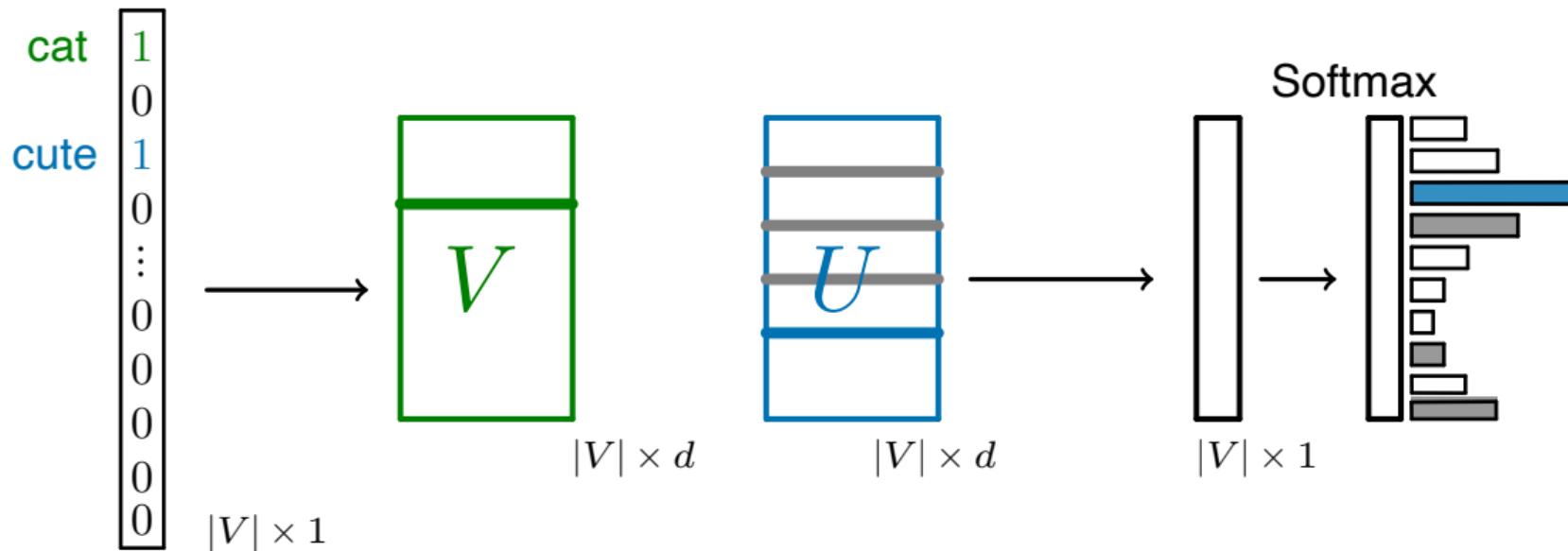


Обновляем один v_{cat} и каждый u_w , то есть всего $|V| + 1$ параметр

Negative Sampling

- Нужно обновлять много параметров \Rightarrow медленное обучение
- Многие слова вместе не встречаются, поэтому большая часть вычислений избыточная
- Давайте максимизировать вероятность типичного контекста и минимизировать вероятность нетипичного контекста

Negative Sampling



Обновляем один v_{cat} , один u_{cute} , и k случайных u_w , то есть всего $k + 2$ параметра

Negative Sampling функция потерь

Старая функция потерь:

$$J_{t,j}(\theta) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_w \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_w \exp(u_w^T v_{cat}) \rightarrow \min_{U,V}$$

Negative Sampling функция потерь

Старая функция потерь:

$$J_{t,j}(\theta) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_w \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_w \exp(u_w^T v_{cat}) \rightarrow \min_{U,V}$$

Новая функция потерь:

$$J_{t,j}(\theta) = -\log \sigma(u_{cute}^T v_{cat}) - \sum_{w \in \{w_1, \dots, w_k\}} \log(1 - \sigma(u_w^T v_{cat})) \rightarrow \min_{U,V}$$

Negative Sampling функция потерь

Старая функция потерь:

$$J_{t,j}(\theta) = -\log \frac{\exp(u_{cute}^T v_{cat})}{\sum_w \exp(u_w^T v_{cat})} = -u_{cute}^T v_{cat} + \log \sum_w \exp(u_w^T v_{cat}) \rightarrow \min_{U,V}$$

Новая функция потерь:

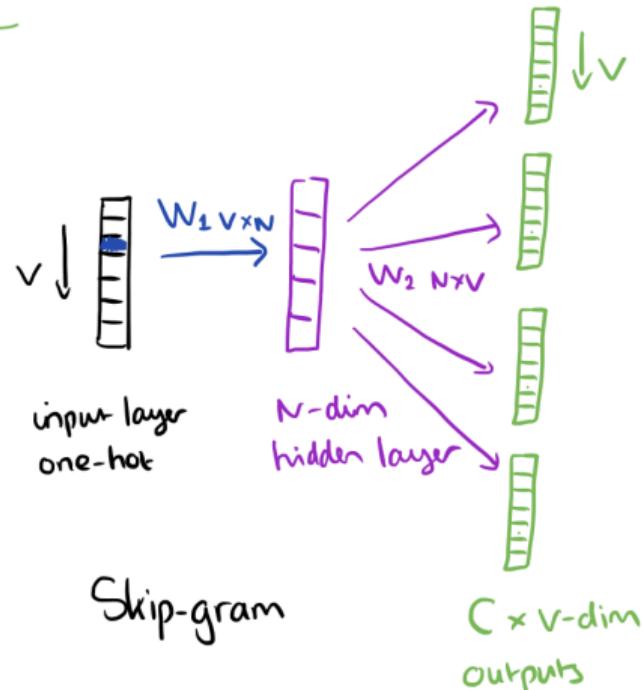
$$J_{t,j}(\theta) = -\log \sigma(u_{cute}^T v_{cat}) - \sum_{w \in \{w_1, \dots, w_k\}} \log \sigma(-u_w^T v_{cat}) \rightarrow \min_{U,V}$$

Skip-gram

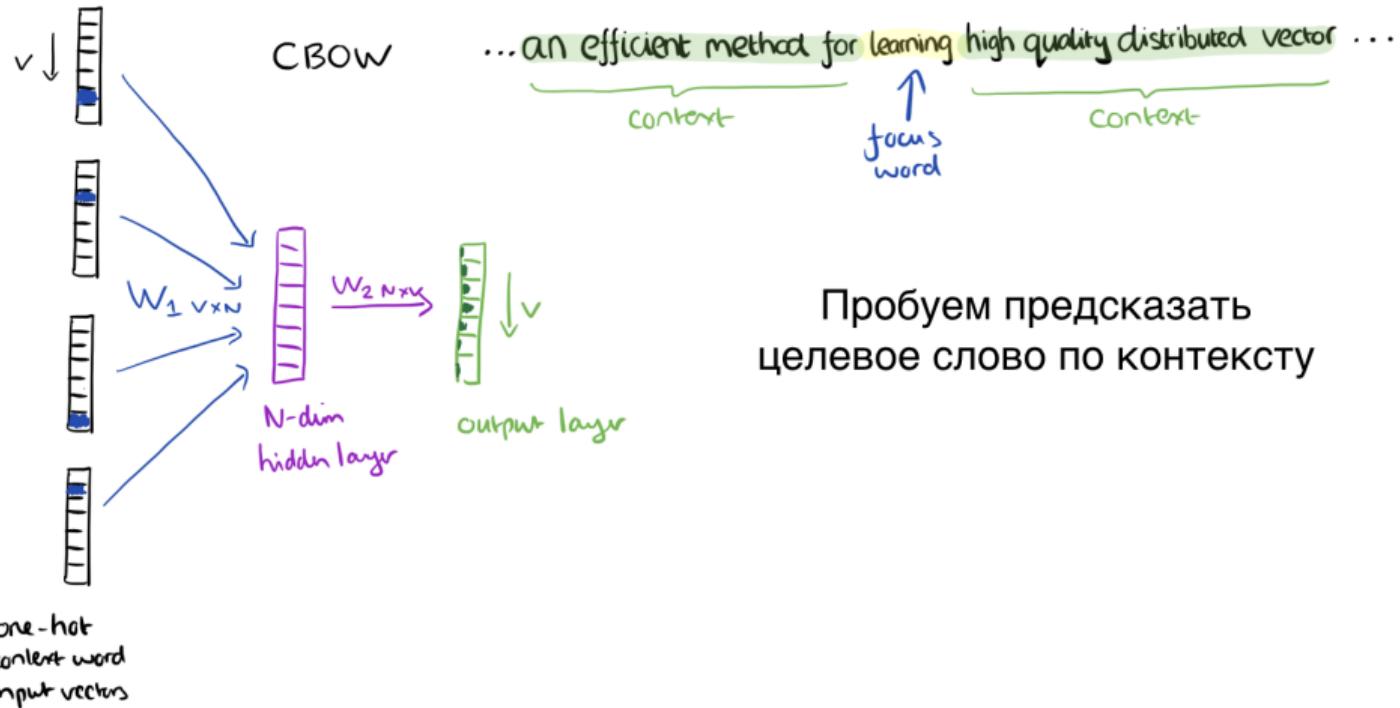
...an efficient method for learning high quality distributed vector ...



Пробуем предсказать
Контекст по целевому слову



CBOW



Гиперпараметры

- Размер эмбединга (исторически 300, но варианты 100, 500 тоже возможно)
- Число наблюдений для негативного сэмплирования (для маленьких датасетов 15 – 20, для больших 2 – 5)
- Окно контекста (обычно 5 – 10),
- Очень большое окно — похожесть топиков.

https://lena-voita.github.io/nlp_course/word_embeddings.html

Полезные мысли про обучение

- Довольно легко собрать свой собственный w2v и обучить его, но не стоит делать это. Ваша реализация не будет такой эффективной, как уже существующие специализированные реализации. За последние годы алгоритмы для обучения w2v претерпели существенную эволюцию.
- Реализация w2v из пакета gensim зачастую работает быстрее, чем модели, написанные в стандартных для нейросеток бэкэндах. Это происходит из-за многопоточности и разных умных оптимизаций тонких мест в обучении.

Полезные мысли про обучение

- При достаточно большом корпусе текстов можно не делать лемматизацию. Сетка сама поймёт по контексту, что слова близки и присвоит им похожие вектора.
- Если у вас специфическая задача, в которой встречается специфическая лексика, возьмите предобученную на большом корпусе сетку и дообучите её под свои нужды.
- w2v может выдавать эмбединги только для слов из заданного при обучении словаря. Этот минус можно попытаться побороть и получить другую модель, fasttext.

Проблемы word2vec

- Не умеем работать с новыми словами, которых не было в нашем словаре при обучении
- Не закладываем никакой априорной информации о разных формах одного слова
- Обучаемся на контекст слова, но всё ещё действуем в парадигме мешка слов, никак не учитываем порядок слов
- Не учитываем структуру слов, не умеем обрабатывать опечатки

Как это использовать и где
раздобыть?

Как это использовать

- Можно искать похожие слова
- Можно менять формы слов
- Можно искать определённые отношения
- Можно использовать как признаки для моделей
- Обучение w2v — аналог transfer learning, но обучение идёт на фиктивную задачу, разметка, сделанная вручную, не нужна

Something2vec

- Эмбеддинги (embeddings) – это сопоставление произвольной сущности (например, узла в графе или кусочка картинки) некоторому вектору.
- Любую последовательность можно представить в виде эмбединга
- Последовательность банковских транзакций
- Веб-сессии (последовательность перехода по сайтам)
- Графы взаимосвязей между пользователями
- Любая категориальная переменная: порядок, в котором турист посещал города; порядок, в котором юзер отранжировал сериалы и тп

something2vec



Где взять уже готовое (проект RusVectōrēs)

Модели

В настоящий момент вы можете скачать следующие модели (жирным выделены модели, доступные для использования в веб-интерфейсе):

Таблицу можно (и нужно) пролистывать по горизонтали!

Статус	Скачать	Корпус	Размер корпуса	Объём словаря	Частотный порог	Тэгсет	Алгоритм	Размерность вектора	Размер окна
Текущий	331 Мбайт	Тайга	почти 5 миллиардов слов	237 255	200	Universal Tags	Continuous Skipgram	300	2
Текущий	191 Мбайт	НКРЯ	250 миллионов слов	195 071	20	Universal Tags	Continuous Skipgram	300	5
Текущий	376 Мбайт	НКРЯ и Википедия за декабрь 2017	600 миллионов слов	384 764	40	Universal Tags	Continuous Skipgram	300	2
Текущий	547 Мбайт	Русскоязычные новости с сентября 2013 до ноября 2016	почти 5 миллиардов слов	289 191	200	Universal Tags	Continuous Bag-of-Words	600	2
Текущий	192 Мбайт	Araneum	около 10 миллиардов слов	196 620	400	Universal Tags	Continuous Skipgram	300	2
Текущий	1 Гбайт	Araneum	около 10 миллиардов слов	195 782	400	Нет	fastText CBOW (3.-5-граммы)	300	5
Текущий	675 Мбайт	Araneum	около 10 миллиардов слов	195 782	400	Нет	fastText Skipgram (3-грамм)	300	5

Куча разных моделей для русского языка: <https://rusvectores.org/ru/models/>

Где взять уже готовое (Google)

Google Code Archive

Search this site

Projects Search About

Project word2vec

Source

Issues Tool for computing continuous distributed representations of words.

Wikis

Downloads

Introduction

This tool provides an efficient implementation of the continuous bag-of-words and skip-gram architectures for computing vector representations of words. These representations can be subsequently used in many natural language processing applications and for further research.

Quick start

- Download the code: svn checkout <http://word2vec.googlecode.com/svn/trunk/>
- Run 'make' to compile word2vec tool
- Run the demo scripts: `J/demo-word.sh` and `J/demo-phrases.sh`
- For questions about the toolkit, see <http://groups.google.com/group/word2vec-toolkit>

Project Information

The project was created on Jul 30, 2013.

- License: Apache License 2.0
- 945 stars
- svn-based source control

Labels:

NeuralNetwork MachineLearning
NaturalLanguageProcessing WordVectors
Google

Гуглловская модель для английского языка: <https://code.google.com/archive/p/word2vec/>

Свойства word2vec



Частотность слова

- Высокая Средняя Низкая

НКРЯ и Wikipedia

1. чай noun 0.56
2. пиво noun 0.56
3. самогон noun 0.56
4. лимонад noun 0.53
5. напиток noun 0.53



Частотность слова

- Высокая Средняя Низкая

НКРЯ и Wikipedia

1. преданность noun 0.38
2. доброта noun 0.37
3. нежность noun 0.37
4. упование noun 0.35
5. умиление noun 0.35

<https://rusvectores.org/ru/calculator>

Свойства word2vec

Результат обучения векторных представлений сильно зависит от коллекции документов. Могут возникать неожиданные артефакты.

Википедия

most_similar(россия)
российский 0.5653642416
рф 0.523694574833
украина 0.492026507854
ссср 0.473026573658
финляндия 0.464367419481
most_similar(тролль)
муметь 0.717674195766
гоблин 0.559770524502
великан 0.557757973671
злобный 0.55741250515
гном 0.554968833923

Луркоморье

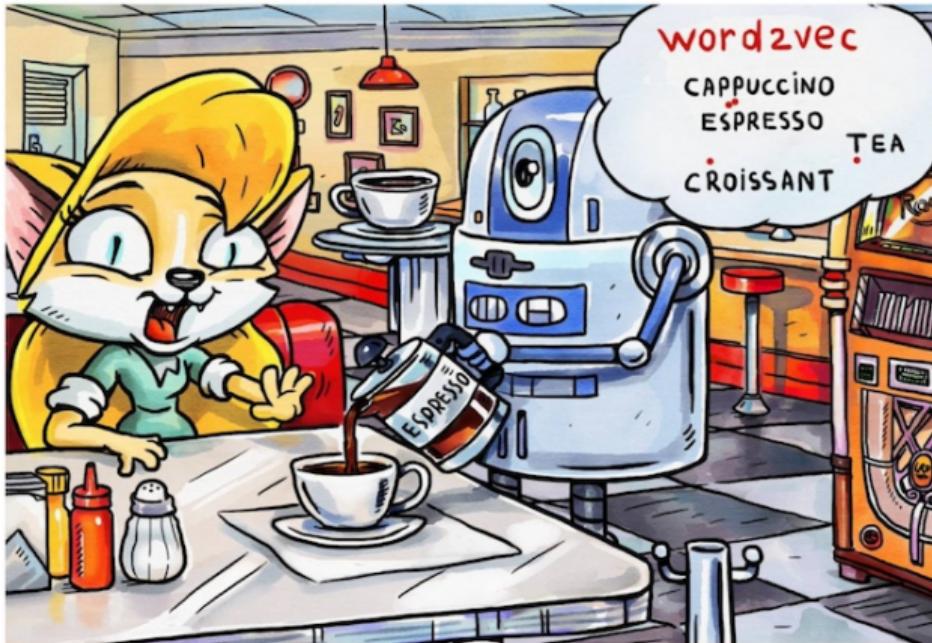
most_similar(россия)
беларусь 0.645048737526
европа 0.622894406319
украина 0.622316598892
рашка 0.619276404381
германия 0.609378278255
most_similar(тролль)
троллинг 0.725703835487
троль 0.660580933094
лжец 0.582996308804
проводокатор 0.57004237175
толстый 0.568691492081

Модель - сексист

```
model.most_similar(u'интеллектуал')
[('моралист', 0.7139864563941956),
('теоретик', 0.6941959857940674),
('литератор', 0.6819325089454651)]  
  
model.most_similar(u'интеллектуалка')
[('бездельница', 0.6617184281349182),
('бунтарка', 0.6578608751296997),
('дилетантка', 0.6419748663902283)]  
  
  
model.most_similar(positive=[u'интеллектуал', u'женщина'], negative=[u'мужчина'])
[('знаменитость', 0.49774518609046936),
('поп-звезда', 0.4860984981060028),
('элита', 0.48200151324272156)]  
  
model.most_similar(positive=[u'ум', u'женщина'], negative=[u'мужчина'])
[('любовь', 0.43659064173698425),
('душа', 0.4330841302871704),
('внешность', 0.4280041456222534)]  
  
model.most_similar(positive=[u'гений', u'женщина'], negative=[u'мужчина'])
[('букашка', 0.4793989062309265),
('химера', 0.4589369595050812),
('душонка', 0.4547439217567444)]
```

<https://nikolenko.livejournal.com/267442.html>

Word2vec всего лишь модель



- Espresso? But I ordered a cappuccino!
- Don't worry, the cosine distance between them is so small that they are almost the same thing.

Откуда взять данные (дампы Википедии)

Wikimedia Downloads

If you are reading this on Wikimedia servers, please note that we have rate limited downloaders and we are capping the number of per-ip connections to 2. This will help to ensure that everyone can access the files with reasonable download times. Clients that try to evade these limits may be blocked. Our mirror sites do not have this cap.

Data downloads

The Wikimedia Foundation is requesting help to ensure that as many copies as possible are available of all Wikimedia database dumps. Please [volunteer to host a mirror](#) if you have access to sufficient storage and bandwidth.

Database backup dumps

A complete copy of all Wikimedia wikis, in the form of wikitext source and metadata embedded in XML. A number of raw database tables in SQL form are also available.

These snapshots are provided at the very least monthly and usually twice a month. If you are a regular user of these dumps, please consider subscribing to [xmldatadumps-l](#) for regular updates.

Дампы википедии для разных языков: <https://dumps.wikimedia.org>

Например, для русского: <https://dumps.wikimedia.org/ruwiki/>

Откуда взять данные (НКРЯ)



НАЦИОНАЛЬНЫЙ
КОРПУС
РУССКОГО
ЯЗЫКА

главная
архив новостей

поиск в корпусе

что такое корпус?

состав и структура

статистика

графики

частоты

морфология

Национальный корпус русского языка

English

На этом сайте помещен корпус современного русского языка общим объемом более 600 млн слов. Корпус русского языка — это информационно-справочная система, основанная на собрании русских текстов в электронной форме.

Корпус предназначен для всех, кто интересуется самыми разными вопросами, связанными с русским языком: профессиональных лингвистов, преподавателей языка, школьников и студентов, иностранцев, изучающих русский язык.

Развитие подкорпусов НКРЯ (основного, поэтического, параллельного, акцентологического, диалектного) в 2015 году осуществлялось при поддержке РГНФ, проекты № 15-04-12018 «Развитие специализированных модулей НКРЯ» и № 14-04-12012 «Корпус диалектных текстов Национального корпуса русского языка. Пополнение и разметка».

[Как пользоваться Корпусом \(инструкция в формате PDF\)](#)

[Подробнее о корпусе](#)

Национальный корпус Русского языка: <http://ruscorpora.ru>

Учим свой w2v