Глубинное обучение

Введение в NLP. Дистрибутивная семантика. Эмбеддинги. Языковые модели. Word2vec, GloVe, FastText

Даниил Водолазский

ВШЭ

15 сентября 2021 г.

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

Содержание

Что такое NLP

Определение

Обработка естественного языка (Natural Language Processing, NLP) — общее направление искусственного интеллекта и математической лингвистики. Оно изучает проблемы компьютерного анализа и синтеза текстов на естественных языках. Применительно к искусственному интеллекту анализ означает понимание языка, а синтез — генерацию грамотного текста. Решение этих проблем будет означать создание более удобной формы взаимодействия компьютера и человека.

— Википедия

Два основных подхода в NLP

- **Рационалистский**
  - **явно** закладывается лингвистическое знание;
  - **вручную** закодированные правила на основе имеющегося опыта;
  - **символьные** методы со строгими ограничениями;

Два основных подхода в NLP

- **Рационалистский**
  - **явно** закладывается лингвистическое знание;
  - **вручную** закодированные правила на основе имеющегося опыта;
  - **символьные** методы со строгими ограничениями;
- **Эмпирический**
  - **автоматически** извлекается лингвистическое знание из корпуса текстов;
  - разработка на основе **данных**;
  - **вероятностные** оценки и ранжирование.

Введение в NLP

- Distributional Semantics course at NSU (2018, 2019)
  Distributional semantics is one of the most important notions in contemporary computational linguistics and NLP: representations of meaning exploiting distributional semantics framework are used in almost any NLP system. Therefore, deep understanding of distributional semantics and models based on it is crucial for resolving cutting-edge NLP tasks. The main idea of this course is to give such understanding to the listener.

  Course materials: `https://github.com/disemantics/course2019`
  VK Group: `http://vk.com/disemantics`
- Some examples are taken from
  `http://wordspace.collocations.de/doku.php/course:esslli2018:start`

Содержание

Дистрибутивная гипотеза

Цитата

You shall know a word by the company it keeps

— Firth, 1957

Дистрибутивная гипотеза

Цитата

You shall know a word by the company it keeps

— Firth, 1957

- Впервые сформулирована в 1950-х Харрисом.

Дистрибутивная гипотеза

Цитата

You shall know a word by the company it keeps

— Firth, 1957

- Впервые сформулирована в 1950-х Харрисом.
- Значение слова определяется контекстами, в которых оно употребляется.

Дистрибутивная гипотеза

Цитата

You shall know a word by the company it keeps

— Firth, 1957

- Впервые сформулирована в 1950-х Харрисом.
- Значение слова определяется контекстами, в которых оно употребляется.
- Семантически схожие слова встречаются в похожих контекстах.

Пример. Что значит ’**bardiwac**’?

- He handed her her glass of **bardiwac**.
- Beef dishes are made to complement the **bardiwacs**.
- Nigel staggered to his feet, face flushed from too much **bardiwac**.
- Malbec, one of the lesser-known **bardiwac** grapes, responds well to Australia’s sunshine.
- I dined off bread and cheese and this excellent **bardiwac**.
- The drinks were delicious: blood-red **bardiwac** as well as light, sweet Rhenish.

Пример. Что значит ’**bardiwac**’?

- He handed her her glass of **bardiwac**.
- Beef dishes are made to complement the **bardiwacs**.
- Nigel staggered to his feet, face flushed from too much **bardiwac**.
- Malbec, one of the lesser-known **bardiwac** grapes, responds well to Australia’s sunshine.
- I dined off bread and cheese and this excellent **bardiwac**.
- The drinks were delicious: blood-red **bardiwac** as well as light, sweet Rhenish.

Bardiwac — крепкий красный алкогольный напиток из винограда.

Содержание

Дистрибутивно-семантические модели

Определение

Дистрибутивно-семантическая модель (distributional semantic model, DSM) — отмасштабированная и/или преобразованная матрица совстречаемости M, такая то каждая строка x представляет распределение слова среди столбцов-контекстов.

Два основных типа ДСМ:

- **подсчётные (count-based)** модели: используют напрямую подсчитанные статистики по корпусу для получения векторов;
- **предсказательные (prediction-based)** модели: учат векторные представления при помощи нейронной сети.

Матрица «термин — документ»

В **матрице «термин — документ»** каждая строка представляет слово в словаре, а каждая колонка — документ из некоторой коллекции документов.

	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	115	81	71	91
fool	37	58	2	5
wit	21	15	2	3

The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

Матрица «термин — документ»

В **матрице «термин — документ»** каждая строка представляет слово в словаре, а каждая колонка — документ из некоторой коллекции документов.

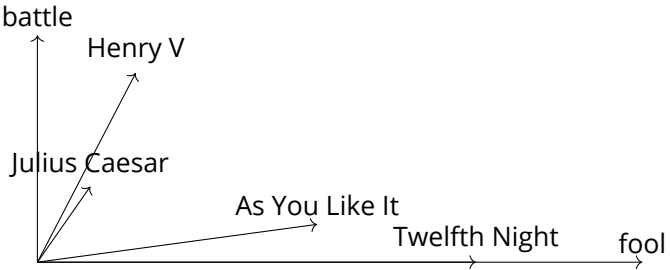
	As You Like It	**Twelfth Night**	**Julius Caesar**	**Henry V**
battle	1	0	7	13
good	115	81	71	91
fool	37	58	2	5
wit	21	15	2	3

The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

Вместо матрицы «термин — документ» можно также использовать матрицы **«термин — термин»** (a. k. a. **матрица совстречаемости слов**) и **«документ — документ»**.

Матрица «термин — документ»

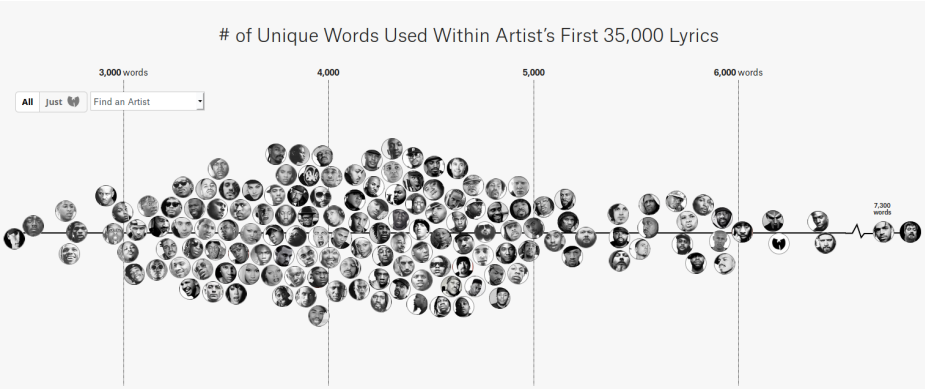
Два слова / документа **похожи**, если их векторы похожи.



Vector space projection on 2-dimensional fool—battle space.

Бинарная матрица

$$W_{w,d} = \operatorname{sgn} \#(w, d).$$



▸ Источник

TF-IDF

Term frequency—inverse document frequency или **TF-IDF** — численная статистика, которая пытается указать, насколько важно слово в данном документе.
Частота термина (term frequency, TF) пропорциональна тому, сколько раз слово w встречалось в документе. Простейший — и наиболее предпочтительный — вариант — просто брать число вхождений слова w в документ d:

$$TF_{w,d} = \#(w, d).$$

Обратная документная частота (inverse document frequency, IDF) (1972) — множитель, повышающий веса редким словам и понижающий — более частотным. Обозначим $D_{|w} := \{d \in D : w \in d\}$.
Тогда

$$IDF_w = \log \frac{|D|}{|D_{|w}|}.$$

Тогда для элементов матрицы TF-IDF имеем:

$$W_{w,d} = TF_{w,d} \cdot IDF_w.$$

PMI. Извлечение коллокаций

Определение

Поточечная взаимная информация (pointwise mutual information, PMI) (1989) между целевым словом w и контекстным словом c определяется так:

$$PMI_{w,c} = \log_2 \frac{P(w,c)}{P(w)P(c)}.$$

PMI. Извлечение коллокаций

Определение

Поточечная взаимная информация (pointwise mutual information, PMI) (1989) между целевым словом w и контекстным словом c определяется так:

$$PMI_{w,c} = \log_2 \frac{P(w,c)}{P(w)P(c)}.$$

- **Числитель** показывает, как часто слова встречались вместе.

PMI. Извлечение коллокаций

Определение

Поточечная взаимная информация (pointwise mutual information, PMI) (1989) между целевым словом w и контекстным словом c определяется так:

$$PMI_{w,c} = \log_2 \frac{P(w,c)}{P(w)P(c)}.$$

- **Числитель** показывает, как часто слова встречались вместе.
- **Знаменатель** показывает, как часто мы бы ожидали встретить пару слов, если бы каждое появлялось независимо.

PMI. Извлечение коллокаций

Определение

Поточечная взаимная информация (pointwise mutual information, PMI) (1989) между целевым словом w и контекстным словом c определяется так:

$$PMI_{w,c} = \log_2 \frac{P(w,c)}{P(w)P(c)}.$$

- **Числитель** показывает, как часто слова встречались вместе.
- **Знаменатель** показывает, как часто мы бы ожидали встретить пару слов, если бы каждое появлялось независимо.
- **Отношение** дает оценку того, насколько пара слов встречается чаще, чем при случайном выборе.

PMI. Извлечение коллокаций

Определение

Поточечная взаимная информация (pointwise mutual information, PMI) (1989) между целевым словом w и контекстным словом c определяется так:

$$PMI_{w,c} = \log_2 \frac{P(w,c)}{P(w)P(c)}.$$

- **Числитель** показывает, как часто слова встречались вместе.
- **Знаменатель** показывает, как часто мы бы ожидали встретить пару слов, если бы каждое появлялось независимо.
- **Отношение** дает оценку того, насколько пара слов встречается чаще, чем при случайном выборе.

PMI. Извлечение коллокаций. Пример

w	c	$\#(w)$	$\#(c)$	$\#(w,c)$	PMI
puerto	rico	1938	1311	1159	10.0349081703
hong	kong	2438	2694	2205	9.72831972408
los	angeles	3501	2808	2791	9.56067615065
carbon	dioxide	4265	1353	1032	9.09852946116
san	francisco	5237	2477	1779	8.83305176711
ice	hockey	5607	3002	1933	8.6555759741
it	the	283891	3293296	3347	-1.72037278119
are	of	234458	1761436	1019	-2.09254205335
this	the	199882	3293296	1211	-2.38612756961
is	of	565679	1761436	1562	-2.54614706831
a	and	984442	1375396	1457	-2.92239510038
in	and	1187652	1375396	1537	-3.05660070757
of	and	1761436	1375396	1190	-3.70663100173

The results of applying PMI to search collocations.

Косинусная близость

Чтобы определить сходство между двумя целевыми словами u и v, нам нужна функция, принимающая два таких вектора и возвращающая меру сходства векторов. Наиболее распространенной мерой сходства является **косинус** угла между векторами:

$$\cos(u, v) = \frac{u \cdot v}{||u|| \cdot ||v||} = \frac{\sum_{i=1}^{N} u_i v_i}{\sqrt{\sum_{i=1}^{N} u_i^2} \sqrt{\sum_{i=1}^{N} v_i^2}},$$

Косинусная близость

Чтобы определить сходство между двумя целевыми словами u и v, нам нужна функция, принимающая два таких вектора и возвращающая меру сходства векторов. Наиболее распространенной мерой сходства является **косинус** угла между векторами:

$$\cos(u, v) = \frac{u \cdot v}{||u|| \cdot ||v||} = \frac{\sum_{i=1}^{N} u_i v_i}{\sqrt{\sum_{i=1}^{N} u_i^2} \sqrt{\sum_{i=1}^{N} v_i^2}},$$

$$simlarity(u, v) = \frac{1}{2} \left(1 - \cos(u, v)\right).$$

Факторизация матриц. SVD

Сингулярное разложение (singular-value decomposition, SVD) (1936) — наиболее известный способ факторизации матриц.

Теорема

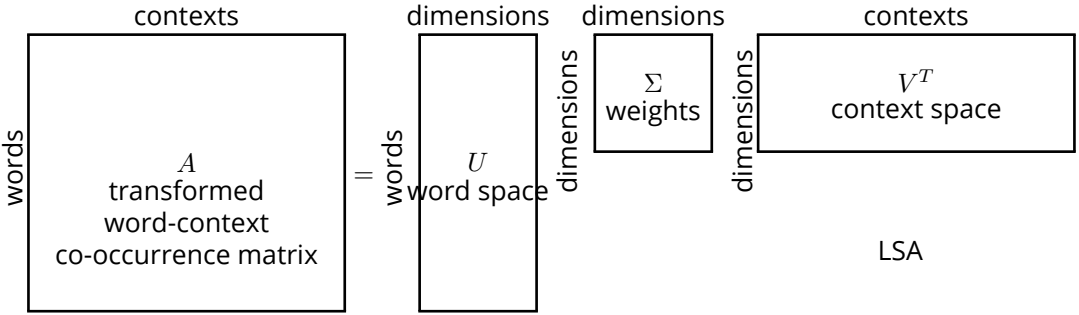
Suppose A is a $m \times n$ real-valued matrix. Then there exists a factorization, called a «singular value decomposition» of A, of the form

$$A = U\Sigma V^\top$$

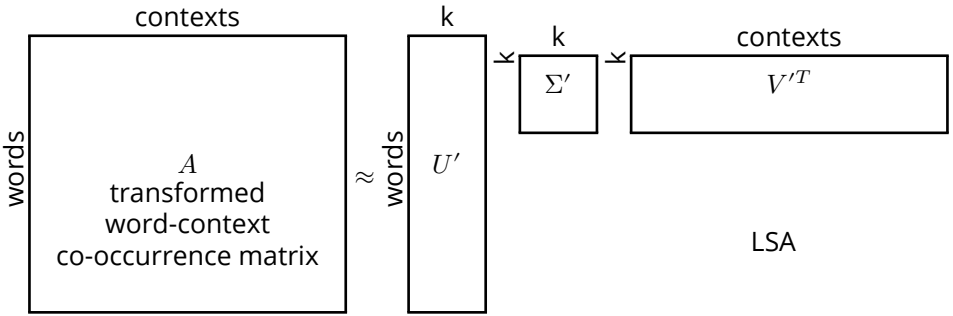
where U is an $m \times m$ orthogonal matrix, Σ is a diagonal $m \times n$ matrix with non-negative real numbers on the diagonal, V is an $n \times n$ orthogonal matrix.

The diagonal entries σ_i of Σ are known as the **singular values** of A.

SVD. Иллюстрация



SVD. Снижение размерности



Извлечение скрытых признаков

SVD применяется для получения низкоразмерных представлений.

Извлечение скрытых признаков

SVD применяется для получения низкоразмерных представлений.

- Строки в уменьшенных матрицах U' и V' — **плотные** (k-мерные) векторы.

Извлечение скрытых признаков

SVD применяется для получения низкоразмерных представлений.

- Строки в уменьшенных матрицах U' и V' — **плотные** (k-мерные) векторы.
- ... также известные как векторы **скрытых признаков** или **распределённые представления**.

Извлечение скрытых признаков

SVD применяется для получения низкоразмерных представлений.

- Строки в уменьшенных матрицах U' и V' — **плотные** (k-мерные) векторы.
- ... также известные как векторы **скрытых признаков** или **распределённые представления**.
- Измерения представляют некоторое скрытое (латентное) **'значение'**.

Приложения в NLP и CL:

- **Latent Semantic Indexing (LSI)** (1988): use query vectors to search for the most similar documents (information retrieval).

Извлечение скрытых признаков

SVD применяется для получения низкоразмерных представлений.

- Строки в уменьшенных матрицах U' и V' — **плотные** (k-мерные) векторы.
- ... также известные как векторы **скрытых признаков** или **распределённые представления**.
- Измерения представляют некоторое скрытое (латентное) **'значение'**.

Приложения в NLP и CL:

- **Latent Semantic Indexing (LSI)** (1988): use query vectors to search for the most similar documents (information retrieval).
- **Latent Semantic Analysis (LSA)** (1997): analyze relationships between words and their contexts.

Плюсы и минусы подсчётных моделей

Плюсы

- Хорошо изучены, получено много результатов. Есть связь с другими науками: алгеброй и теорией информации.

Плюсы и минусы подсчётных моделей

Плюсы

- Хорошо изучены, получено много результатов. Есть связь с другими науками: алгеброй и теорией информации.
- Даже в случае маленьких данных иногда можно получить неплохое качество.

Плюсы и минусы подсчётных моделей

Плюсы

- Хорошо изучены, получено много результатов. Есть связь с другими науками: алгеброй и теорией информации.
- Даже в случае маленьких данных иногда можно получить неплохое качество.

Минусы

- Проблема **мешка слов (bag-of-words)**: «A cat is chasing a dog» эквивалентно «A dog is chasing a cat».

Плюсы и минусы подсчётных моделей

Плюсы

- Хорошо изучены, получено много результатов. Есть связь с другими науками: алгеброй и теорией информации.
- Даже в случае маленьких данных иногда можно получить неплохое качество.

Минусы

- Проблема **мешка слов (bag-of-words)**: «A cat is chasing a dog» эквивалентно «A dog is chasing a cat».
- **Неоднозначность** никак не разрешается: одно слово — один вектор.

Плюсы и минусы подсчётных моделей

Плюсы

- Хорошо изучены, получено много результатов. Есть связь с другими науками: алгеброй и теорией информации.
- Даже в случае маленьких данных иногда можно получить неплохое качество.

Минусы

- Проблема **мешка слов (bag-of-words)**: «A cat is chasing a dog» эквивалентно «A dog is chasing a cat».
- **Неоднозначность** никак не разрешается: одно слово — один вектор.
- Для слов, ни разу не попавших в обучающую выборку (**out-of-vocabulary**, OOV words), нельзя получить векторное представление.

Содержание

Языковые модели

Определение

Статистическая языковая модель может быть представлени как **условная вероятность следующего слова** при условии всех предыдущих, так как

$$P(w_1^T) = \prod_{t=1}^{T} P(w_t \mid w_1^{t-1}),$$

где w_t t-е слово и $w_i^j = (w_i, w_{i+1}, \dots, w_{j-1}, w_j)$.

Языковые модели

n**-граммные модели** строят (n-мерные) **таблицы условных вероятностей** для следующего слова при условии *контекстов*, т. е. комбинаций последних $n-1$ слов:

Языковые модели

n**-граммные модели** строят (n-мерные) **таблицы условных вероятностей** для следующего слова при условии *контекстов*, т. е. комбинаций последних $n-1$ слов:

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Языковые модели

n**-граммные модели** строят (n-мерные) **таблицы условных вероятностей** для следующего слова при условии *контекстов*, т. е. комбинаций последних $n-1$ слов:

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Использование такого способа представления слов и их вероятностей потребует хранить $O\left(|W|^n\right)$ элементов. Это называется **проклятием размерности**.

Языковые модели

n-граммные модели строят (n-мерные) **таблицы условных вероятностей** для следующего слова при условии *контекстов*, т. е. комбинаций последних $n-1$ слов:

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Использование такого способа представления слов и их вероятностей потребует хранить $O(|W|^n)$ элементов. Это называется **проклятием размерности**. На практике $n = 2, 3, 4$.

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w
- **контекст**: w_{t-n+1}^{t-1}

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w
- **контекст**: w_{t-n+1}^{t-1}

Отличия:

- **порядок слов** имеет значение;

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w
- **контекст**: w_{t-n+1}^{t-1}

Отличия:

- **порядок слов** имеет значение;
- возможно **генерировать** тексты.

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w
- **контекст**: w_{t-n+1}^{t-1}

Отличия:

- **порядок слов** имеет значение;
- возможно **генерировать** тексты.

Но:

- по-прежнему **разреженные**;

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w
- **контекст**: w_{t-n+1}^{t-1}

Отличия:

- **порядок слов** имеет значение;
- возможно **генерировать** тексты.

Но:

- по-прежнему **разреженные**;
- требуют много **памяти**.

Статистические языковые модели

$$P(w_t \mid w_1^{t-1}) \approx P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\#(w_t, w_{t-n+1}^{t-1})}{\sum_{w \in W} \#(w, w_{t-n+1}^{t-1})}.$$

Связь в матрицей «слово — контекст»:

- **слово**: w
- **контекст**: w_{t-n+1}^{t-1}

Отличия:

- **порядок слов** имеет значение;
- возможно **генерировать** тексты.

Но:

- по-прежнему **разреженные**;
- требуют много **памяти**.

Neural Probabilistic Language Model

A **neural probabilistic language model (NPLM)** was first proposed by Bengio et al. in 2003.

The approach is as follows:

1. **associate** with **each word** in the vocabulary an **embedding** (a real-valued vector in $\mathbb{R}^d$);

Neural Probabilistic Language Model

A **neural probabilistic language model (NPLM)** was first proposed by Bengio et al. in 2003.

The approach is as follows:

1. **associate** with **each word** in the vocabulary an **embedding** (a real-valued vector in $\mathbb{R}^d$);
2. **express** the **joint probability** function of word sequences in terms of the **embeddings** of these words in the sequence,

Neural Probabilistic Language Model

A **neural probabilistic language model (NPLM)** was first proposed by Bengio et al. in 2003.

The approach is as follows:

1. **associate** with **each word** in the vocabulary an **embedding** (a real-valued vector in $\mathbb{R}^d$);
2. **express** the **joint probability** function of word sequences in terms of the **embeddings** of these words in the sequence,
3. learn **simultaneously** the word **embeddings** and **parameters** of that probability function.

Neural Probabilistic Language Model

A **neural probabilistic language model (NPLM)** was first proposed by Bengio et al. in 2003.

The approach is as follows:

1. **associate** with **each word** in the vocabulary an **embedding** (a real-valued vector in $\mathbb{R}^d$);
2. **express** the **joint probability** function of word sequences in terms of the **embeddings** of these words in the sequence,
3. learn **simultaneously** the word **embeddings** and **parameters** of that probability function.

NPLM. Intuition

The aim is to learn a good model for $f(w_t, \dots, w_{t-n+1}) = P(w_t \mid w_{t-n+1}^{t-1})$.
To do so, the function is decomposed into two parts:

NPLM. Intuition

The aim is to learn a good model for $f(w_t, \dots, w_{t-n+1}) = P(w_t \mid w_{t-n+1}^{t-1})$.
To do so, the function is decomposed into two parts:

1. A **mapping** C from **any element** w of W to a **real vector** $C(w) \in \mathbb{R}^d$ (the distributed feature vectors associated with each word in the vocabulary). In practice, C is represented by a $|W| \times d$ matrix of free parameters.

NPLM. Intuition

The aim is to learn a good model for $f(w_t, \dots, w_{t-n+1}) = P(w_t \mid w_{t-n+1}^{t-1})$.
To do so, the function is decomposed into two parts:

1. A **mapping** C from **any element** w of W to a **real vector** $C(w) \in \mathbb{R}^d$ (the distributed feature vectors associated with each word in the vocabulary). In practice, C is represented by a $|W| \times d$ matrix of free parameters.
2. The **probability function over words** (expressed with C):
   $g : (i, C(w_{t-1}), \dots, C(w_{t-n+1})) \to P(w_t = w^{(i)} \mid w_{t-n+1}^{t-1})$;

$$f(w^{(i)}, w_{t-1}, \dots, w_{t-n+1}) = g(i, C(w_{t-1}), \dots, C(w_{t-n+1})).$$

NPLM. Architecture

$$L = -\frac{1}{T}\sum_{t=1}^{T}\log P(w_t \mid w_{t-n+1}^{t-1}) + R(\Theta) \longrightarrow \min_{\Theta},$$

where Θ is a set of all parameters, and R is a regularizer.

NPLM. Architecture

$$L = -\frac{1}{T}\sum_{t=1}^{T}\log P(w_t \mid w_{t-n+1}^{t-1}) + R(\Theta) \longrightarrow \min_{\Theta},$$

where Θ is a set of all parameters, and R is a regularizer.

The probabilities are expressed by applying **SoftMax** to the **unnormalized log-probabilities** y_w:

$$P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\exp y_{w_t}}{\sum_{w \in W} \exp y_w},$$

NPLM. Architecture

$$L = -\frac{1}{T}\sum_{t=1}^{T}\log P(w_t \mid w_{t-n+1}^{t-1}) + R(\Theta) \longrightarrow \min_{\Theta},$$

where Θ is a set of all parameters, and R is a regularizer.

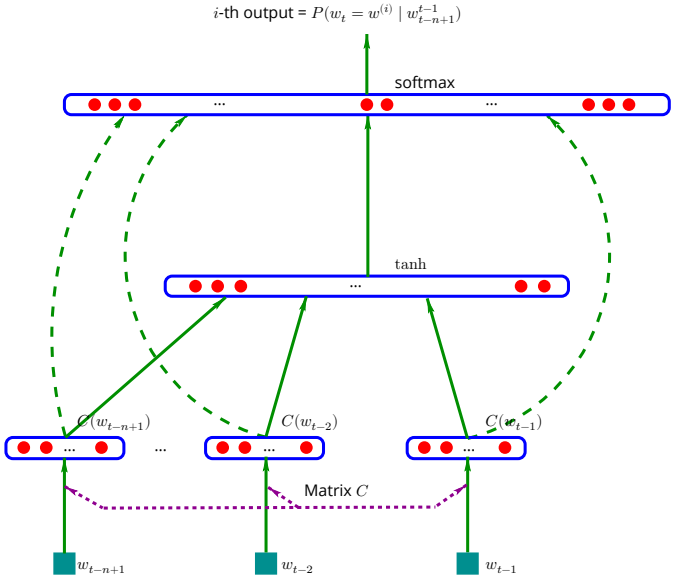
The probabilities are expressed by applying **SoftMax** to the **unnormalized log-probabilities** y_w:

$$P(w_t \mid w_{t-n+1}^{t-1}) = \frac{\exp y_{w_t}}{\sum_{w \in W} \exp y_w},$$

$$y = b + Wx + U\tanh(d + Hx),$$

where W is optionally zero (no direct connections), and

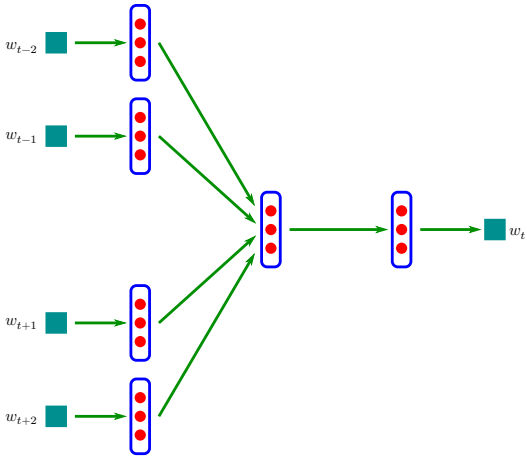
$$x = [C(w_{t-1}), C(w_{t-2}), \ldots, C(w_{t-n+1})].$$


i-th output = $P(w_t = w^{(i)} \mid w_{t-n+1}^{t-1})$
softmax
tanh
$C(w_{t-n+1})$
$C(w_{t-2})$
$C(w_{t-1})$
Matrix C
w_{t-n+1}
w_{t-2}
w_{t-1}

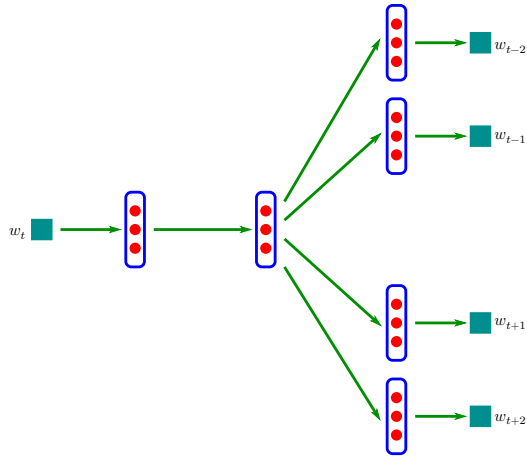
Содержание

Word2Vec

Рассмотрим две модели для получения векторных представлений слов, которые могут быть **эффективно обучены** на больших объёмах текстовых данных. Это **Skip-gram (SG)** и **Continuous bag-of-words model (CBOW)**, представленные в работах Mikolov et al., 2013.



Architecture of the CBOW model



Architecture of the Skip-gram model

Continuous Skip-gram Model

Пусть дана последовательность слов $w_1, w_2, w_3, \dots, w_T$. Цель модели **Skip-gram** — максимизировать средний логарифм вероятности

$$L = \frac{1}{T} \sum_{t=1}^{T} \sum_{c \in C_t} \log P(c \mid w_t),$$

где $C_t = \cup_{j=-win, j \neq 0}^{win} w_{t+j}$, а win — размер тренировочного окна (который может быть функцией от центрального слова w_t).

Continuous Skip-gram Model

Пусть дана последовательность слов $w_1, w_2, w_3, \dots, w_T$. Цель модели **Skip-gram** — максимизировать средний логарифм вероятности

$$L = \frac{1}{T} \sum_{t=1}^{T} \sum_{c \in C_t} \log P(c \mid w_t),$$

где $C_t = \cup_{j=-win, j \neq 0}^{win} w_{t+j}$, а win — размер тренировочного окна (который может быть функцией от центрального слова w_t).

В модели Skip-gram каждое слово w ассоциировано с двумя обучаемыми векторами, u_w и v_w. Это “входной” и “выходной” векторы для w. Вероятность правильно предсказать контекстное слово c при условии центрального слова w_t определяется как

Continuous Skip-gram Model

Пусть дана последовательность слов $w_1, w_2, w_3, \dots, w_T$. Цель модели **Skip-gram** — максимизировать средний логарифм вероятности

$$L = \frac{1}{T} \sum_{t=1}^{T} \sum_{c \in C_t} \log P(c \mid w_t),$$

где $C_t = \cup_{j=-win, j \neq 0}^{win} w_{t+j}$, а win — размер тренировочного окна (который может быть функцией от центрального слова w_t).

В модели Skip-gram каждое слово w ассоциировано с двумя обучаемыми векторами, u_w и v_w. Это “входной” и “выходной” векторы для w. Вероятность правильно предсказать контекстное слово c при условии центрального слова w_t определяется как

$$P(c \mid w_t) = \frac{\exp\left(u_{w_t}^\top v_c\right)}{\sum_{c' \in W} \exp\left(u_{w_t}^\top v_{c'}\right)}.$$

Negative Sampling

Определение

Определим **отрицательное сэмплирование** целевой функцией

$$\log \sigma(u_w^\top v_c) + \sum_{j=1}^{k} \mathbb{E}_{c_{n,j} \sim P_n} \log \sigma(-u_w^\top v_{c_{n,j}}),$$

которая используется, чтобы заменить $\log P(c \mid w_t)$ в целевой функции Skip-gram.

Negative Sampling

Определение

Определим **отрицательное сэмплирование** целевой функцией

$$\log \sigma(u_w^\top v_c) + \sum_{j=1}^{k} \mathbb{E}_{c_{n,j} \sim P_n} \log \sigma(-u_w^\top v_{c_{n,j}}),$$

которая используется, чтобы заменить $\log P(c \mid w_t)$ в целевой функции Skip-gram.

Таким образом, наша задача — отличить настоящее слово c от случайных сэмплов c_n из шумового распределения $P_n(w)$ с помощью логистической регрессии, где для каждого слова есть k отрицательных примеров.

Skip-gram Connection to Count-Based Models

SGNS embeds both words and their contexts into a low-dimensional space $\mathbb{R}^d$, resulting in the word and context matrices U and V. Let $M = UV^\top$. An implicit matrix M of dimensions $|W| \times |W|$ is factorized into two smaller matrices, and $M_{w,c} = u_w^\top v_c$.

Skip-gram Connection to Count-Based Models

SGNS embeds both words and their contexts into a low-dimensional space $\mathbb{R}^d$, resulting in the word and context matrices U and V. Let $M = UV^\top$. An implicit matrix M of dimensions $|W| \times |W|$ is factorized into two smaller matrices, and $M_{w,c} = u_w^\top v_c$.

So, each cell of matrix M contains a quantity $f(w, c)$ reflecting the strength of association between that particular word-context pair. What can we say about the association function $f(w, c)$? In other words, which matrix is SGNS factorizing?

Skip-gram Connection to Count-Based Models

SGNS embeds both words and their contexts into a low-dimensional space $\mathbb{R}^d$, resulting in the word and context matrices U and V. Let $M = UV^\top$. An implicit matrix M of dimensions $|W| \times |W|$ is factorized into two smaller matrices, and $M_{w,c} = u_w^\top v_c$.

So, each cell of matrix M contains a quantity $f(w, c)$ reflecting the strength of association between that particular word-context pair. What can we say about the association function $f(w, c)$? In other words, which matrix is SGNS factorizing?

Theorem

Levy and Goldberg (2014) has shown that $M_{w,c} = f(w, c) = \log 2 \cdot PMI_{w,c} - \log k$, where k is a number of negative samples.

Рассуждение по аналогии на уровне эмбеддингов. Интуиция

Чтобы найти слово, относящееся к *small* так же, как *biggest* относится к *big*, можно просто вычислить вектор

$$v^* = v_{biggest} - v_{big} + v_{small}.$$

Рассуждение по аналогии на уровне эмбеддингов. Интуиция

Чтобы найти слово, относящееся к *small* так же, как *biggest* относится к *big*, можно просто вычислить вектор

$$v^* = v_{biggest} - v_{big} + v_{small}.$$

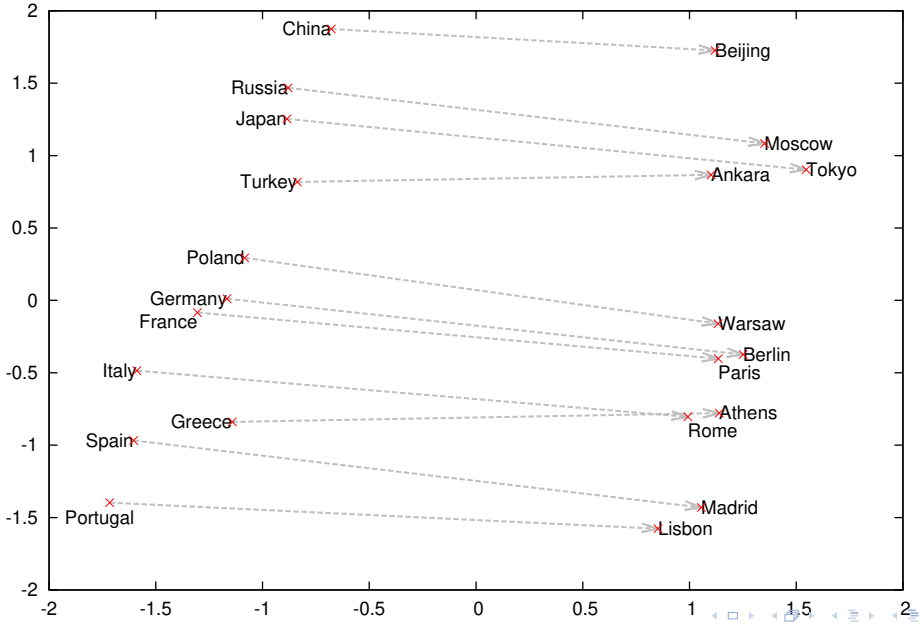
Ещё пример:

$$v_{Paris} - v_{France} + v_{Italy} = v_{Rome}.$$

Примеры выученных отношений

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Examples of the word pair relationships, using the best word vectors from Skip-gram model trained on 783M words with 300 dimensionality.


Country and Capital Vectors Projected by PCA
China
Beijing
Russia
Moscow
Japan
Tokyo
Turkey
Ankara
Poland
Warsaw
Germany
Berlin
France
Paris
Italy
Athens
Greece
Rome
Spain
Madrid
Portugal
Lisbon
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

Содержание

GloVe

The **Global Vector (GloVe)** model (2014) aims to combine the **count-based matrix factorization** and the **context-based skip-gram model** together.
We will define the co-occurrence probability as:

$$P(w_k \mid w_i) = \frac{\#(w_i, w_k)}{\#(w_i)}.$$

Let $w_i =$ “ice” and $w_j =$ “steam”. The third word $w_k =$ “solid” is related to “ice” but not “steam”, and thus $P(w_k \mid w_i)/P(w_k \mid w_j) \to \infty$.
If the third word $w_k =$ “water” is related to both or $w_k =$ “fashion” is unrelated to either of them, the equation above is expected to be close to one.

The word meanings are captured by the **ratios of co-occurrence probabilities** rather than the **probabilities themselves**. The global vector models the relationship between two words regarding to the third context word as:

$$F(w_i, w_j, w_k) = \frac{P(w_k \mid w_i)}{P(w_k \mid w_j)}.$$

GloVe

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

GloVe

Since the goal is to learn meaningful word vectors, F is designed to be a function of the linear difference between two words w_i and w_j:

$$F(w_i, w_j, w_k) = F((u_{w_i} - u_{w_j})^\top v_{w_k}).$$

With the consideration of F being symmetric between target words and context words, the final solution is to model F as an exponential function. And we obtain the following equations:

$$F(u_{w_i}^\top v_{w_k}) = \exp(u_{w_i}^\top v_{w_k}) = P(w_k \mid w_i),$$

$$F((u_{w_i} - u_{w_j})^\top v_{w_k}) = \exp((u_{w_i} - u_{w_j})^\top v_{w_k}) = \frac{\exp(u_{w_i}^\top v_{w_k})}{\exp(u_{w_j}^\top v_{w_k})} = \frac{P(w_k \mid w_i)}{P(w_k \mid w_j)}.$$

Finally,

$$u_{w_i}^\top v_{w_k} = \log P(w_k \mid w_i) = \log \frac{\#(w_i, w_k)}{\#(w_i)} = \log \#(w_i, w_k) - \log \#(w_i).$$

GloVe

$$u_{w_i}^\top v_{w_k} = \log P(w_k \mid w_i) = \log \frac{\#(w_i, w_k)}{\#(w_i)} = \log \#(w_i, w_k) - \log \#(w_i).$$

After adding biases, we obtain

$$\log \#(w_i, w_k) = u_{w_i}^\top v_{w_k} + b_{w_i}^u + b_{w_k}^v.$$

The loss function for the GloVe model is designed to preserve the above formula by minimizing the sum of the squared errors

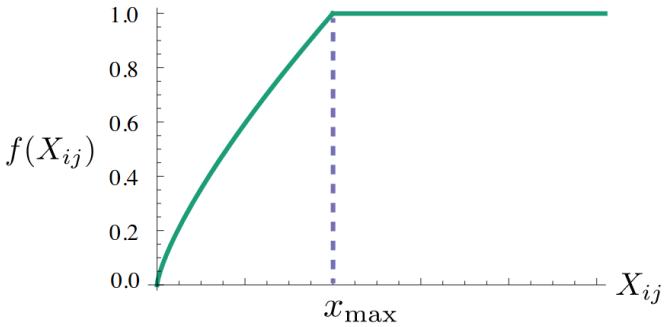
$$L = \sum_{w \in \mathcal{W}} \sum_{c \in \mathcal{W}} f\left(\#(w, c)\right) \left(u_w^\top v_c + b_w^u + b_c^v - \log \#(w, c)\right)^2.$$

GloVe

The paper proposed the following weighting function:

$$f(x) = \begin{cases} \left(\frac{x}{x_{\max}}\right)^{\alpha}, & \text{if } x < x_{\max}, \\ 1, & \text{otherwise} \end{cases}$$

with optimal values $\alpha = 0.75$ and $x_{\max} = 100$.



LexVec

Just like GloVe, **LexVec** (2016) also tries to factorize PPMI matrices, emerging characteristics of both count-based and prediction-based models. Unlike GloVe, it penalizes errors of frequent co-occurrences more heavily, while still treating negative co-occurrences.
The LexVec loss function has two terms:

$$L^{WC}(w,c) = \frac{1}{2}\left(u_w^\top v_c - PPMI^*_{w,c}\right)^2,$$

$$L^{W}(w) = \frac{1}{2}\sum_{j=1}^{k} \mathbb{E}_{c_{n,j}\sim P_n}\left(u_w^\top v_{c_{n,j}} - PPMI^*_{w,c_{n,j}}\right)^2$$

where $PPMI^*$ is an improved PPMI matrix.

Swivel

The central claim of the authors of **Submatrix-wise vector embedding learner (Swivel)** (2016) is that none of the mainstream word embeddings provide any special treatment to **unobserved** word-context co-occurrences, so the ability to capture them helped to outperform other embedding training algorithms in word similarity and word analogy tasks. For co-occurrences that have been observed, Swivel computes the weighted squared error between the embedding dot product and the PMI of w and c:

$$L^+(w,c) = \frac{1}{2} f(\#(w,c)) \left(u_w^\top v_c - PMI_{w,c}\right)^2 =$$
$$= \frac{1}{2} f(\#(w,c)) \left(u_w^\top v_c - \log \#(w,c) - \log T + \log \#(w) + \log \#(c)\right)^2.$$

This encourages $u_w^\top v_c$ to correctly estimate the observed PMI. $f(x) = x^{\frac{1}{2}}$.

Swivel

If $\#(w, c) = 0$, then $PMI_{w,c} = -\infty$, and the squared error cannot be computed.
Treating $\#(w, c)$ as a sample, we can ask: how significant is it that its observed value is zero?
We address this by smoothing the PMI value as if a single co-occurrence had been observed (i.e., computing PMI as if $\#(w, c) = 1$), and using an asymmetric cost function that penalizes over-estimation of the smoothed PMI:

$$\begin{aligned} L^0(w, c) &= \log\left(1 + \exp\left(u_w^\top v_c - PMI^*_{w,c}\right)\right) = \\ &= \log\left(1 + \exp\left(u_w^\top v_c - \log T + \log \#(w) + \log \#(c)\right)\right). \end{aligned}$$

Here, PMI^* refers to the smoothed PMI computation where $\#(w, c)$'s actual count of 0 is replaced with 1. This loss penalizes the model for over-estimating the objective value; however, it applies negligible penalty—i.e., is noncommittal—if the model under-estimates it.

Содержание

Композициональность

Цитата

The meaning of an utterance is a function of the meaning of its parts and their composition rules

— Frege, 1892

Композициональность

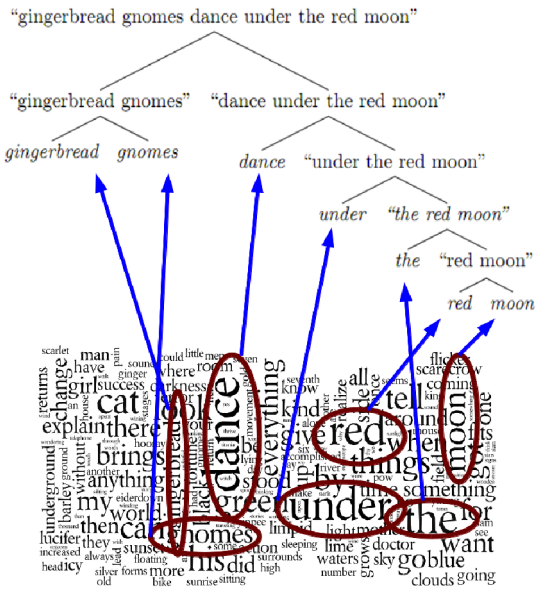
Цитата

The meaning of an utterance is a function of the meaning of its parts and their composition rules

— Frege, 1892

Принцип композициональности много используется в формальной семантике:

$$[[red\ car]] = \lambda x.RED(x) \wedge CAR(x)$$


"gingerbread gnomes dance under the red moon"
"gingerbread gnomes"
"dance under the red moon"
gingerbread
gnomes
dance
"under the red moon"
under
"the red moon"
the
"red moon"
red
moon

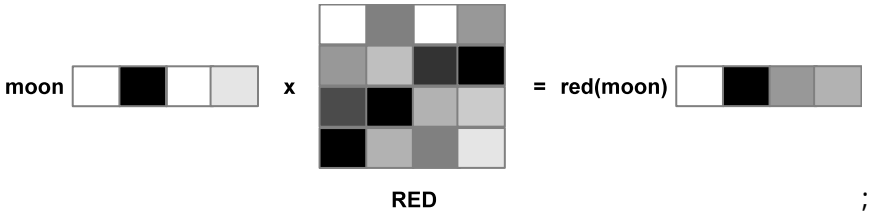
;

Композициональность на фразовом уровне

- Простые и взвешенные **аддитивные модели** — сильные бейзлайны (Mitchell and Lapata, 2010).

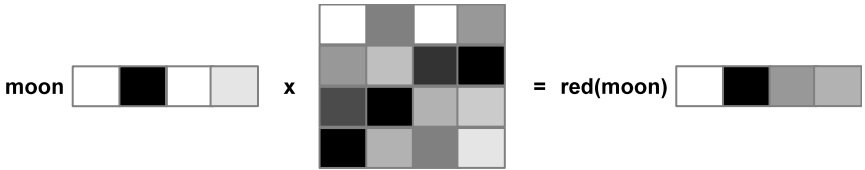
Композициональность на фразовом уровне

- Простые и взвешенные **аддитивные модели** — сильные бейзлайны (Mitchell and Lapata, 2010).
- **Lexical function model:** nouns are vectors, adjectives are matrices (Baroni and Zamparelli, 2010).



Композициональность на фразовом уровне

- Простые и взвешенные **аддитивные модели** — сильные бейзлайны (Mitchell and Lapata, 2010).
- **Lexical function model:** nouns are vectors, adjectives are matrices (Baroni and Zamparelli, 2010).

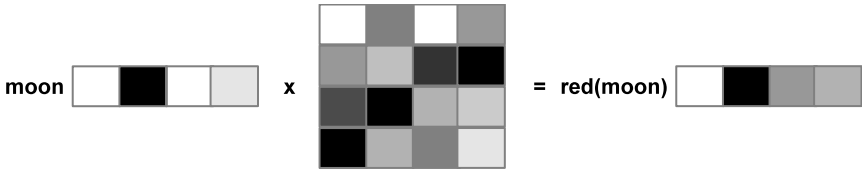


;

- **Compact closed categories** and **Frobenius algebras** (Kartsaklis, 2010–2015)

Композициональность на фразовом уровне

- Простые и взвешенные **аддитивные модели** — сильные бейзлайны (Mitchell and Lapata, 2010).
- **Lexical function model:** nouns are vectors, adjectives are matrices (Baroni and Zamparelli, 2010).

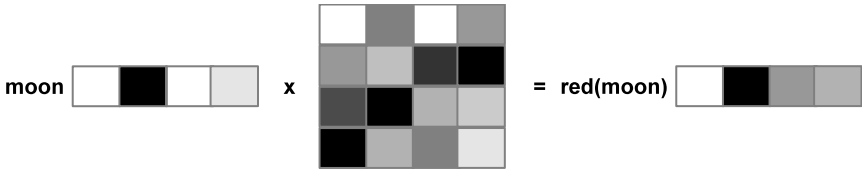


;

- **Compact closed categories** and **Frobenius algebras** (Kartsaklis, 2010–2015)
- Computational Linguistics journal special issue (Dec. 2016)
  `https://www.mitpressjournals.org/toc/coli/42/4`

Композициональность на фразовом уровне

- Простые и взвешенные **аддитивные модели** — сильные бейзлайны (Mitchell and Lapata, 2010).
- **Lexical function model:** nouns are vectors, adjectives are matrices (Baroni and Zamparelli, 2010).



;

- **Compact closed categories** and **Frobenius algebras** (Kartsaklis, 2010–2015)
- Computational Linguistics journal special issue (Dec. 2016)
  `https://www.mitpressjournals.org/toc/coli/42/4`

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** ‘worker accident insurance law’

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** ’worker accident insurance law’
- **Türkleştiremediğimizlerdensinizdir** ’surely you are one of those whom we failed to turn into Turks’

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** ’worker accident insurance law’
- **Türkleştiremediğimizlerdensinizdir** ’surely you are one of those whom we failed to turn into Turks’
- **верхнесредиземноморские** (1 результат в Google) ’upper Mediterranean’

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** 'worker accident insurance law'
- **Türkleştiremediğimizlerdensinizdir** 'surely you are one of those whom we failed to turn into Turks'
- **верхнесредиземноморские** (1 результат в Google) 'upper Mediterranean'
- **двухсотпятидесятидевятиэтажный** '259-floor'

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** ’worker accident insurance law’
- **Türkleştiremediğimizlerdensinizdir** ’surely you are one of those whom we failed to turn into Turks’
- **верхнесредиземноморские** (1 результат в Google) ’upper Mediterranean’
- **двухсотпятидесятидевятиэтажный** ’259-floor’
- **Taumatawhakatangihangakoauauotamateaturipukakapikimaungahoronukupokaiwh**
  ’The summit where Tamatea, the man with the big knees, the slider, climber of mountains, the land-swallower who travelled about, played his kōauau (flute) to his loved one’ (Wikipedia).

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** ’worker accident insurance law’
- **Türkleştiremediğimizlerdensinizdir** ’surely you are one of those whom we failed to turn into Turks’
- **верхнесредиземноморские** (1 результат в Google) ’upper Mediterranean’
- **двухсотпятидесятидевятиэтажный** ’259-floor’
- **Taumatawhakatangihangakoauauotamateaturipukakapikimaungahoronukupokaiwh**
  ’The summit where Tamatea, the man with the big knees, the slider, climber of mountains, the land-swallower who travelled about, played his kōauau (flute) to his loved one’ (Wikipedia).
- dgtl-dt-d-decil-de-ditaliz-dez-**digitalization**, `https://youtu.be/2i4dpkr3wRc?t=10`

Подсловные представления слов

Проблема: как обрабатывать морфологически сложные слова?

- **Arbeiterunfallversicherungsgesetz** ’worker accident insurance law’
- **Türkleştiremediğimizlerdensinizdir** ’surely you are one of those whom we failed to turn into Turks’
- **верхнесредиземноморские** (1 результат в Google) ’upper Mediterranean’
- **двухсотпятидесятидевятиэтажный** ’259-floor’
- **Taumatawhakatangihangakoauauotamateaturipukakapikimaungahoronukupokaiwh**
  ’The summit where Tamatea, the man with the big knees, the slider, climber of mountains, the land-swallower who travelled about, played his kōauau (flute) to his loved one’ (Wikipedia).
- dgtl-dt-d-decil-de-ditaliz-dez-**digitalization**, `https://youtu.be/2i4dpkr3wRc?t=10`

Символьные n-граммы. FastText

FastText (2017) — модель для обучения представлений слов с учётом морфологии.

Символьные n-граммы. FastText

FastText (2017) — модель для обучения представлений слов с учётом морфологии. Каждое слово w представляется как мешок символьных n-грамм. Само слово w тоже включается в множество n-грамм, чтобы получать для него представление. Для слова `where` и $n = 3$ имеем:

`<wh, whe, her, ere, re>, <where>`.

Символьные n-граммы. FastText

FastText (2017) — модель для обучения представлений слов с учётом морфологии. Каждое слово w представляется как мешок символьных n-грамм. Само слово w тоже включается в множество n-грамм, чтобы получать для него представление. Для слова `where` и $n = 3$ имеем:

`<wh, whe, her, ere, re>, <where>`.

Пусть G_w^n — множество n-грамм в слове w. Сопоставим вектор z_{g^n} каждой из n-грамм g^n. Мы представляем слово через его n-граммы.

Символьные n-граммы. FastText

FastText (2017) — модель для обучения представлений слов с учётом морфологии. Каждое слово w представляется как мешок символьных n-грамм. Само слово w тоже включается в множество n-грамм, чтобы получать для него представление. Для слова `where` и $n = 3$ имеем:

`<wh, whe, her, ere, re>, <where>`.

Пусть G_w^n — множество n-грамм в слове w. Сопоставим вектор z_{g^n} каждой из n-грамм g^n. Мы представляем слово через его n-граммы. Эмбеддинг слова w может быть получен как

$$u_w = \frac{1}{|G_w^n|} \sum_{g^n \in G_w^n} z_{g^n}.$$

Символьные n-граммы. FastText

FastText (2017) — модель для обучения представлений слов с учётом морфологии. Каждое слово w представляется как мешок символьных n-грамм. Само слово w тоже включается в множество n-грамм, чтобы получать для него представление. Для слова `where` и $n = 3$ имеем:

`<wh, whe, her, ere, re>, <where>`.

Пусть G_w^n — множество n-грамм в слове w. Сопоставим вектор z_{g^n} каждой из n-грамм g^n. Мы представляем слово через его n-граммы. Эмбеддинг слова w может быть получен как

$$u_w = \frac{1}{|G_w^n|} \sum_{g^n \in G_w^n} z_{g^n}.$$

Такая простая модель позволяет обмениваться знанием о морфологии между словами, что приводит к более надёжным результатам.

Byte Pair Embeddings и R-граммы

Byte pair encoding (BPE) (1994) — кодирование переменной длины, которое рассматривает текст как последоватлеьность символов и итеративно объединяет наиболее частотную пару символов в один новый.

Byte Pair Embeddings и R-граммы

Byte pair encoding (BPE) (1994) — кодирование переменной длины, которое рассматривает текст как последоватлеьность символов и итеративно объединяет наиболее частотную пару символов в один новый. Поскольку BPE может работать с любой последовательностью символов, не требуется никакого препроцессинга текста.

Byte Pair Embeddings и R-граммы

Byte pair encoding (BPE) (1994) — кодирование переменной длины, которое рассматривает текст как последоватлеьность символов и итеративно объединяет наиболее частотную пару символов в один новый. Поскольку BPE может работать с любой последовательностью символов, не требуется никакого препроцессинга текста.

1. aaabdaaabac
2. ZabdZabac, Z=aa
3. ZYdZYac, Y=ab, Z=aa
4. XdXac, X=ZY, Y=ab, Z=aa

Byte Pair Embeddings и R-граммы

Byte pair encoding (BPE) (1994) — кодирование переменной длины, которое рассматривает текст как последоватлеьность символов и итеративно объединяет наиболее частотную пару символов в один новый. Поскольку BPE может работать с любой последовательностью символов, не требуется никакого препроцессинга текста.

1. aaabdaaabac
2. ZabdZabac, Z=aa
3. ZYdZYac, Y=ab, Z=aa
4. XdXac, X=ZY, Y=ab, Z=aa

В NLP BPE впервые был применён к машинному переводу (2016).

Byte Pair Embeddings и R-граммы

Byte pair encoding (BPE) (1994) — кодирование переменной длины, которое рассматривает текст как последоватлеьность символов и итеративно объединяет наиболее частотную пару символов в один новый. Поскольку BPE может работать с любой последовательностью символов, не требуется никакого препроцессинга текста.

1. aaabdaaabac
2. ZabdZabac, Z=aa
3. ZYdZYac, Y=ab, Z=aa
4. XdXac, X=ZY, Y=ab, Z=aa

В NLP BPE впервые был применён к машинному переводу (2016).

BPEmb (2018) — коллекция предобученных эмбеддингов на основе BPE для 275 языков.

Эмбеддинги на основе морфемной сегментации

Морфемная сегментация — процесс разбиения слов на морфемы.

Эмбеддинги на основе морфемной сегментации

Морфемная сегментация — процесс разбиения слов на морфемы. Эмбеддинг для слова w может быть вычислен как

$$u_w = \frac{1}{|G_w^m|} \sum_{g^m \in G_w^m} z_{g^m},$$

где G_w^m — множество морфем слова w.

Эмбеддинги на основе морфемной сегментации

Морфемная сегментация — процесс разбиения слов на морфемы. Эмбеддинг для слова w может быть вычислен как

$$u_w = \frac{1}{|G_w^m|} \sum_{g^m \in G_w^m} z_{g^m},$$

где G_w^m — множество морфем слова w.
Однако сегментация — это отдельная задача, которую не всегда удаётся решить качественно.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Simple Multiplicative Model (2010): $v_d = v_b \odot v_p$.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Simple Multiplicative Model (2010): $v_d = v_b \odot v_p$.

Weighted Additive Model: $v_d = \alpha v_b + \beta v_p$.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Simple Multiplicative Model (2010): $v_d = v_b \odot v_p$.

Weighted Additive Model: $v_d = \alpha v_b + \beta v_p$.

Dilation Model: $v_d = (\alpha - 1)(v_p^\top v_b)v_p + (v_p^\top v_p)v_b$.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Simple Multiplicative Model (2010): $v_d = v_b \odot v_p$.

Weighted Additive Model: $v_d = \alpha v_b + \beta v_p$.

Dilation Model: $v_d = (\alpha - 1)(v_p^\top v_b)v_p + (v_p^\top v_p)v_b$.

Lexical Function Model (2010): $v_d = P_p v_b$.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Simple Multiplicative Model (2010): $v_d = v_b \odot v_p$.

Weighted Additive Model: $v_d = \alpha v_b + \beta v_p$.

Dilation Model: $v_d = (\alpha - 1)(v_p^\top v_b)v_p + (v_p^\top v_p)v_b$.

Lexical Function Model (2010): $v_d = P_p v_b$. Non-cummutativity: $P_{able}(P_{un}(v_{lock})) \neq P_{un}(P_{able}(v_{lock}))$.

Композициональность на морфологичском уровне

Все модели принимают базовое слово b и возвращают производное слово d, предтавленные в виде векторов. Паттерн (семантический сдвиг) p представляется вектором или матрицей. **Simple Additive Model**: $v_d = v_b + v_p$.

Simple Multiplicative Model (2010): $v_d = v_b \odot v_p$.

Weighted Additive Model: $v_d = \alpha v_b + \beta v_p$.

Dilation Model: $v_d = (\alpha - 1)(v_p^\top v_b)v_p + (v_p^\top v_p)v_b$.

Lexical Function Model (2010): $v_d = P_p v_b$. Non-cummutativity: $P_{able}(P_{un}(v_{lock})) \neq P_{un}(P_{able}(v_{lock}))$.

Full Additive Model: $v_d = Av_b + Bv_p$.

Содержание

Что узнали

- Что такое дистрибутивная семантика, какие бывают простые модели и посложнее.
- Познакомились с нейронными языковыми моделями.
- Увидели лексические и морфологические расширения Word2vec: GloVe, FastText, и др.