



Deep Learning School

Обработка естественного языка и аудио

Введение.

План лекции

- Задачи обработки естественного языка, обработки аудио
- Векторные представления слов
 - BOW
 - Tf-idf
 - LSA
- Эмбединги слов:
 - Контекстные эмбединги
 - Word2Vec
 - GloVe, FastText
- Эмбединги фраз и предложений
- Использование эмбедингов
- Понятие эмбединга в общем виде

В этом видео

- **Задачи обработки естественного языка, обработки аудио**
- Векторные представления слов
 - BOW
 - Tf-idf
 - LSA
- Эмбединги слов:
 - Контекстные эмбединги
 - Word2Vec
 - GloVe, FastText
- Эмбединги фраз и предложений
- Использование эмбедингов
- Понятие эмбединга в общем виде

Обработка естественного языка

Обработка естественного языка

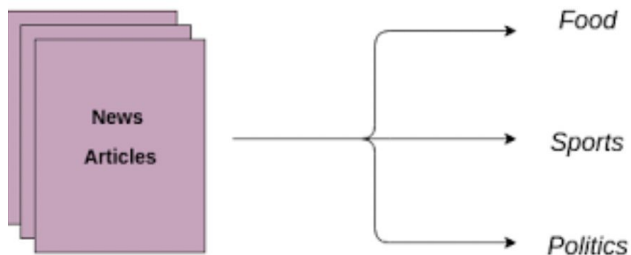
(Natural Language Processing, NLP) включает в себя задачи, связанные с текстами на естественном языке.

Это как задачи понимания языка, так и задачи генерации текстов на естественном языке.

А некоторые задачи включают оба аспекта.

Задачи NLP

- Классификация текста (авторство, жанр, и т.д.)
- Анализ настроения (sentiment analysis)



Задачи NLP

- Генерация текста
- Машинный перевод



Write 2 haikus extolling the virtues of AI and ChatGPT



AI, a gift from the gods

Efficient, precise, and quick

A machine of wonder

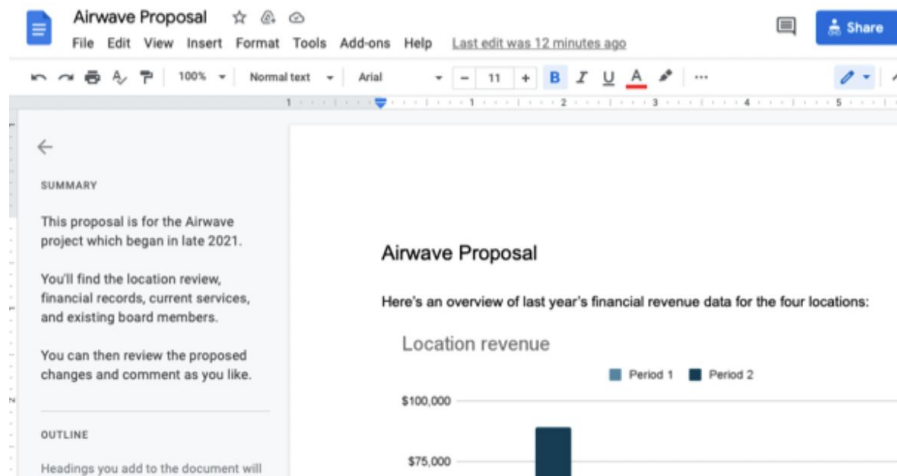
ChatGPT, a friend indeed

Always ready with an answer

A helpful guide, indeed

Задачи NLP

- Вопросно-ответные системы (question answering)
- Суммаризация текста
- Диалоговые системы



Задачи NLP

- Распознавание именованных объектов
(Named Entity Recognition, NER)

Hi, My name is Aman Kharwal **PERSON**

I am from India **GPE**

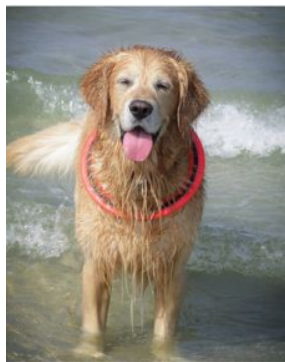
I want to work with Google **ORG**

Steve Jobs **PERSON** is My Inspiration

Задачи NLP

Мультимодальность (CV+NLP)

- Описание изображения/видео (image/video captioning)
- Visual question answering
- Распознавание текста на изображении (Optical Character Recognition, OCR)



Captioning Model

A happy dog is standing in the ocean

Задачи NLP

Мультимодальность (CV+NLP)

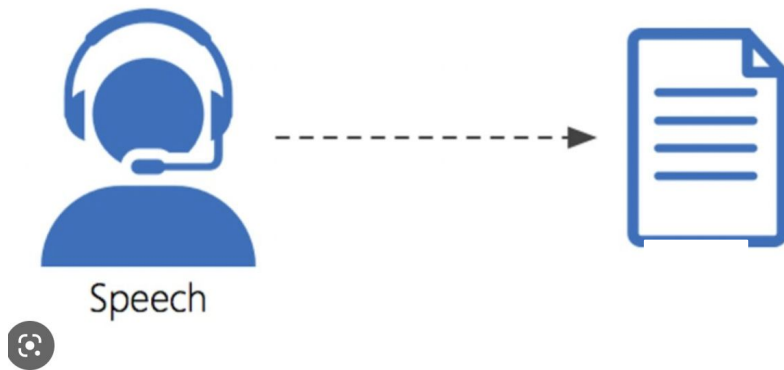
- Генерация изображений/видео по текстам



Фото ежика, сидящего в маленькой лодочке посреди озера. На нем гавайская рубашка и соломенная шляпа. Ежик читает книгу. На его фоне — листья.

Обработка аудио

- Классификация аудио, анализ настроения;
- Audio-to-text, text-to-audio;
- Улучшения качества аудио (speech enhancement);
- Разделение источников звука;



Итоги видео

В этом видео мы познакомились с некоторыми примерами задач обработки естественного языка (NLP) и аудио

В следующих видео мы узнаем, в каком виде представляется текст для подачи на вход нейросети.



Deep Learning School

Векторные представления слов.

План видео

- Задачи обработки естественного языка, обработки аудио
- **Векторные представления слов**
 - **BOW**
 - **Tf-idf**
 - **LSA**
- Эмбединги слов:
 - Контекстные эмбединги
 - Word2Vec
 - GloVe, FastText
- Эмбединги фраз и предложений
- Использование эмбедингов
- Понятие эмбединга в общем виде

Векторные представления слов

Как представлять текст в компьютере?

Текст — это набор слов. Как представлять слова?

One-hot encoding

Создадим словарь фиксированного размера.

Например, $n=50.000$ слов.

Словарь

1.	cat
2.	dog
3.	...
i.	mother
	...

One-hot encoding

Создадим словарь фиксированного размера.

Например, $n=50.000$ слов.

Словарь

1.	cat
2.	dog
3.	...
i.	mother
	...

Размер каждого
вектора = n (50.000)

cat = [1, 0, 0, ..., 0]
dog = [0, 1, 0, ..., 0]
mother = [0, 0, 0, ..., 1, ..., 0]

↑
i-я координата

One-hot encoding

Недостатки:

- Векторы слов не отражают смысл слова.
Нельзя измерить “похожесть” двух слов по смыслу;
- Векторы довольно разрежены, требуют много лишней памяти;
- Размер словаря ограничен.
Слова, не попавшие в словарь, не могут быть обработаны.
- При изменении размера словаря нужно пересчитывать векторы заново.

Мешок слов (Bag of Words, BoG)

На основе one-hot кодирования слов можно построить кодирование предложений.

Вектор предложения — сумма векторов его слов.

Мешок слов (Bag of Words, BoG)

Словарь

- | | |
|-------|----------|
| 1. | a |
| 2. | and |
| 14. | are |
| 145. | cat |
| 257. | dog |
| 678. | is |
| 1537. | sleeping |

- 1. a cat and a dog are sleeping**
- 2. a dog is walking**

BoW для этих предложений:

- | | |
|----|--|
| 1. | [2, 1, 0, ... 1, 0, ..., 0, 1, 0, ... 1, 0, ..., 0, ..., 1, ... 0] |
| 2. | [1, 0, 0, ... 0, 0, ..., 0, 0, 0, ... 1, 0, ..., 1, ..., 0, ... 0] |
| | ↑ ↑ ↑ ↑ ↑ ↑ ↑ |
| | 1 2 14 145 257 678 1537 |

Bag of Words

Наследуются все недостатки one-hot encoding:

- Векторы предложений не очень хорошо отражают смысл предложения. Порядок слов не учитывается;
- Векторы довольно разрежены, требуют много лишней памяти;
- Фиксированный размер словаря.
Слова, не попавшие в словарь, не могут быть обработаны.
- При изменении размера словаря нужно пересчитывать векторы заново.

Bag of Words

Еще один недостаток BoW — различные слова могут иметь разную важность для текста.

1. **a cat and a dog are sleeping**
2. **a dog is walking**

BoW для этих предложений:

1. [2, 1, 0, ... 1, 0, ..., 0, 1, 0, ... 1, 0, ..., 0, ..., 1, ... 0]
 2. [1, 0, 0, ... 0, 0, ..., 0, 0, 0, ... 1, 0, ..., 1, ..., 0, ... 0]
- ↑ ↑ ↑ ↑ ↑ ↑
- 1 2 14 145 257 678 1537

Tf-IDF

$\text{tf-idf}(t, d, D)$ — это мера важности слова t для документа d в наборе документов D .

$$\text{tf-idf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

Tf-IDF

Документ #1

Term	Term count
a	1
dog	1
eats	1
meat	1

Документ #2

Term	Term count
a	1
dog	1
hunts	1
cat	1

Tf-IDF

Документ #1

Term	Term count
a	1
dog	1
eats	1
meat	1

Документ #2

Term	Term count
a	1
dog	1
hunts	1
cat	1

Term Frequency $tf(t, d)$ считается для слова и документа

$$tf(t, d) = \frac{n_t}{\sum_k n_k}$$

← количество раз, которое слово t встречается в документе d

← количество слов в документе d

$$tf("a", doc_1) = 1 / 4 = 0.25$$

$$tf("a", doc_2) = 1 / 4 = 0.25$$

Tf-IDF

Документ #1

Term	Term count
a	1
dog	1
eats	1
meat	1

Документ #2

Term	Term count
a	1
dog	1
hunts	1
cat	1

Inverse document frequency $\text{idf}(t, D)$
считается для слова и набора документов

количество документов в наборе

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d_i \in D \mid t \in d_i\}|}$$

количество документов, в которых встречается слово t

$$\text{idf}(\text{"a"}, D) = \log(2 / 2) = \log(1) = 0$$

Tf-IDF

Документ #1

Term	Term count
a	1
dog	1
eats	1
meat	1

Документ #2

Term	Term count
a	1
dog	1
hunts	1
cat	1

$\text{tf-idf}(t, d, D)$ — это мера важности слова t для документа d в наборе документов D .

$$\text{tf-idf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

$$\text{tf}(\text{"a"}, \text{doc}_1) = 1 / 4 = 0.25$$

$$\text{tf}(\text{"a"}, \text{doc}_2) = 1 / 4 = 0.25$$

$$\text{idf}(\text{"a"}, D) = \log(2 / 2) = \log(1) = 0$$

$$\text{tf-idf}(\text{"a"}, \text{doc}_1, D) = 0.25 * 0 = \mathbf{0}$$

$$\text{tf-idf}(\text{"a"}, \text{doc}_2, D) = 0.25 * 0 = \mathbf{0}$$

Tf-IDF

Документ #1

Term	Term count
a	1
dog	1
eats	1
meat	1

Документ #2

Term	Term count
a	1
dog	1
hunts	1
cat	1

$\text{tf-idf}(t, d, D)$ — это мера важности слова ***t*** для документа ***d*** в наборе документов ***D***.

$$\text{tf-idf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

$$\text{tf}(\text{"meat"}, \text{doc_1}) = 1 / 4 = 0.25$$

$$\text{tf}(\text{"meat"}, \text{doc_2}) = 0 / 4 = 0$$

$$\text{idf}(\text{"meat"}, D) = \log(2 / 1) = \log(2) \approx 0.3$$

$$\text{tf-idf}(\text{"meat"}, \text{doc_1}, D) = 0.25 * 0.3 = \mathbf{0.75}$$

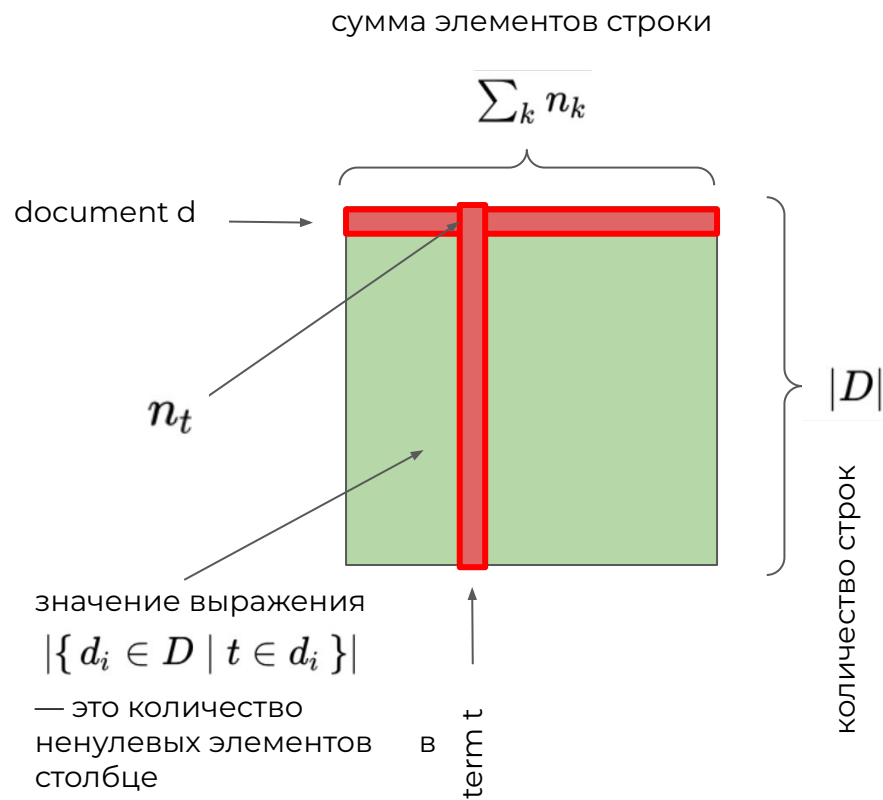
$$\text{tf-idf}(\text{"meat"}, \text{doc_2}, D) = 0 * 0.3 = \mathbf{0}$$

Tf-idf

$$\text{tf}(t, d) = \frac{n_t}{\sum_k n_k}$$

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d_i \in D \mid t \in d_i\}|}$$

$$\text{tf-idf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$



Tf-idf

На основе TF-IDF значений слов можно
построить векторы документов

1. **a dog eats meat**
2. **a dog hunts cat**

TF-IDF векторы для этих предложений:

- | | | | |
|----|-----------------------------|------------------------------------|------------|
| 1. | [0, ..., 0, ... 0.075, ... | 0, ..., 0, ..., | 0, ...] |
| 2. | [0, ..., 0, ... 0, ..., | 0.075, ... 0.075, ..., 0.075, ...] | |
| | ↑ ↑ ↑ | ↑ | ↑ ↑ |
| | a dog eats | hunts | cat meat |

Tf-idf

Еще примеры использования TF-IDF:

- Ранжирование поисковой выдачи;
- Выделение ключевых слов и суммаризация текста.

Tf-idf

Плюсы:

- Векторы имеют больший смысл, чем при BoW;
- Возможность решать такие задачи, как ранжирование документов и выделение ключевых слов;

Недостатки:

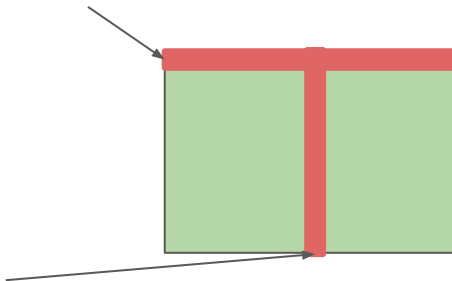
- Векторы довольно разрежены;
- Фиксированный размер словаря.
- При изменении коллекции документов векторы нужно пересчитывать.

Латентный семантический анализ

Матрица
документы-слова
размера $n \times k$

документ

слово



Латентный семантический анализ

Матрица
документы-слова
размера $n \times k$

документ

слово



\approx

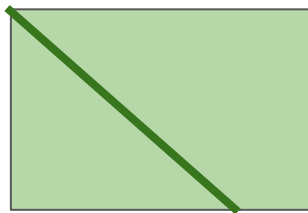
SVD

\approx



$U (k \times k)$

\times



$S (k \times n)$
диагональная матрица

\times



$V^T (n \times n)$

Латентный семантический анализ

Матрица
документы-слова
размера $n \times k$

документ

слово

\approx

SVD
 \approx

вектор
документа

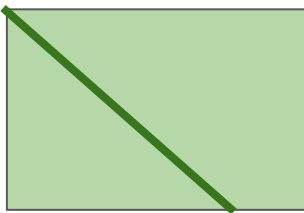
d

$U (k \times k)$

\times

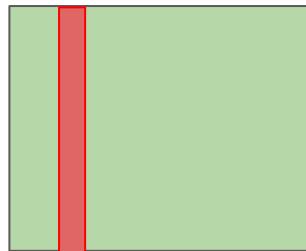
вектор слова

w



$S (k \times n)$
диагональная матрица

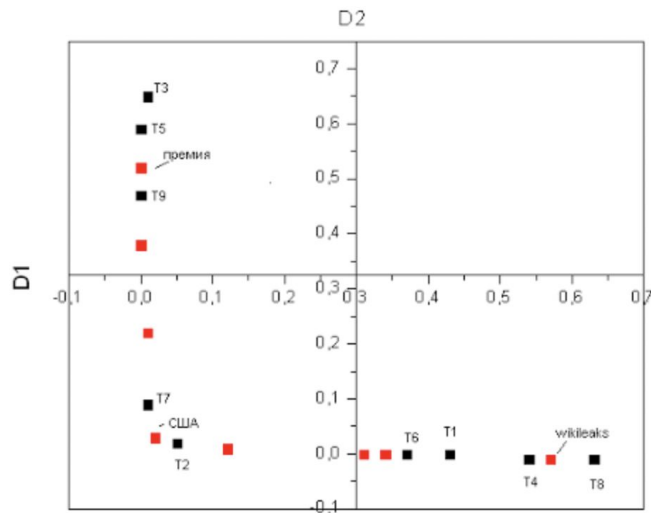
\times



$V^T (n \times n)$

Латентный семантический анализ

Применим метод снижения размерности к векторам документов и визуализируем их точками на плоскости. Видно, что точки разбились на три кластера по темам.



Латентный семантический анализ

Матрица
документы-слова
размера $n \times k$

документ

слово

\approx

SVD

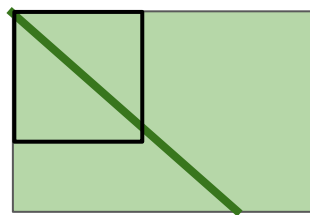
\approx

вектор
документа

d

$U (k \times k)$

\times



$S (k \times n)$
диагональная матрица

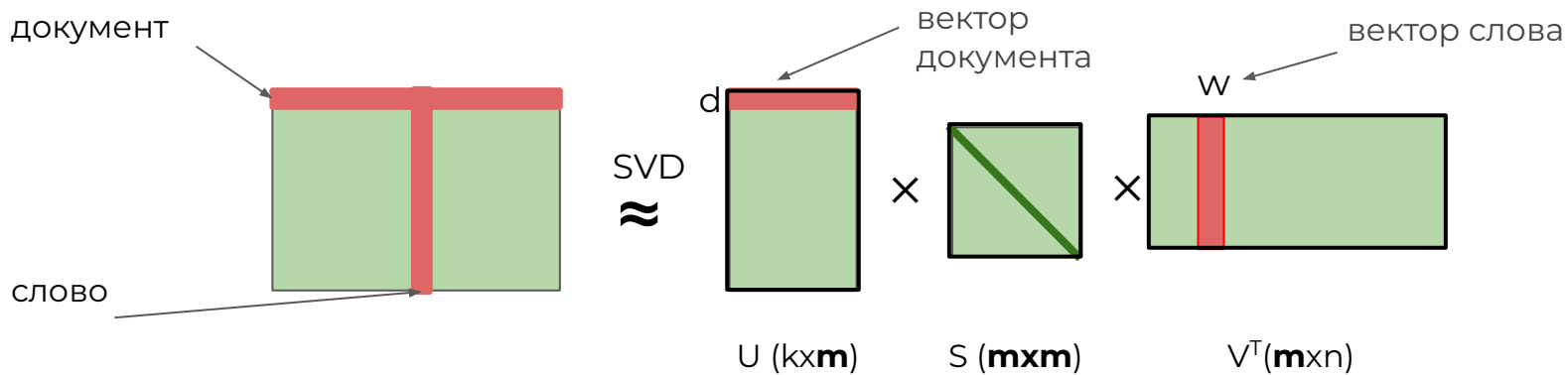
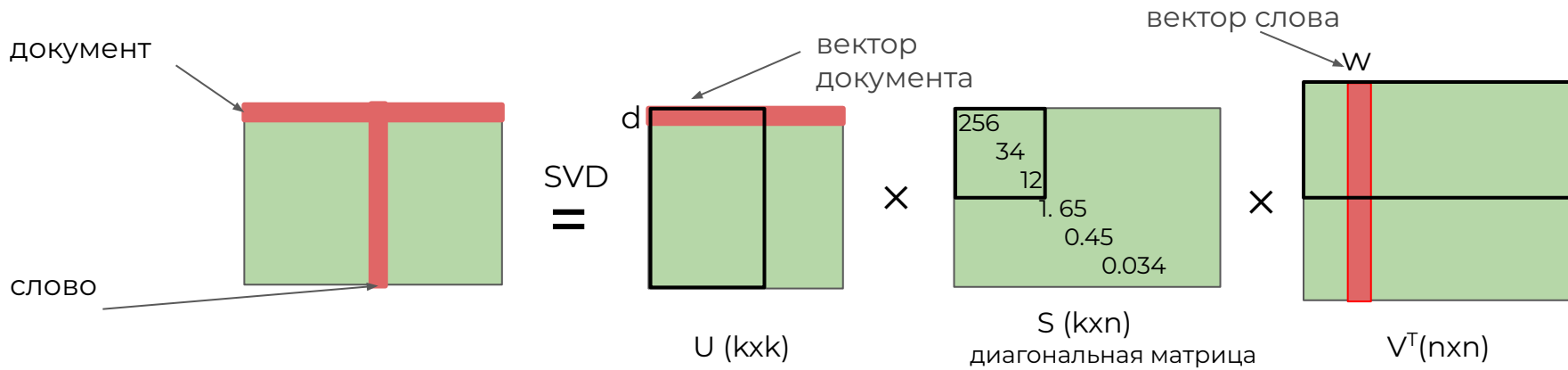
\times

вектор слова

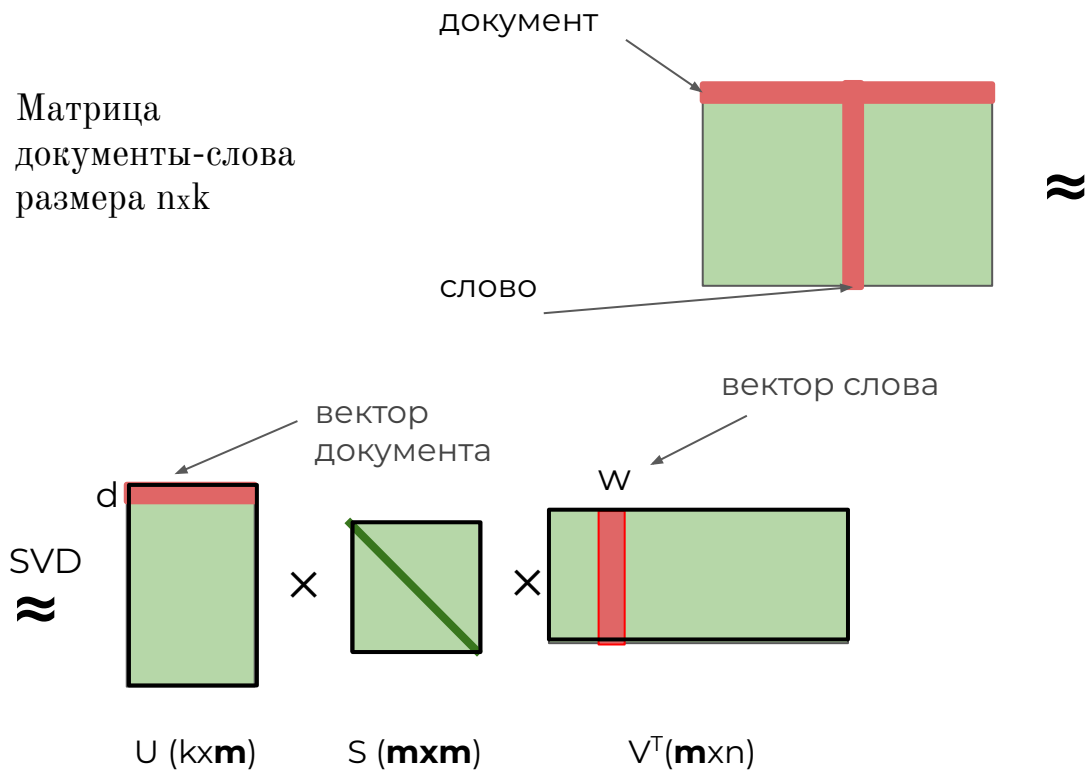
w



$V^T (n \times n)$



Латентный семантический анализ



Латентный семантический анализ

Плюсы:

- Векторы имеют смысл;
- Возможность уменьшать размер эмбедингов без существенной потери качества.

Недостатки:

- Большая вычислительная сложность при большом количестве документов;
- Фиксированный размер словаря;
- При изменении коллекции документов векторы нужно пересчитывать;
- Вероятностная модель метода не соответствует реальности.

Итоги видео

В этом видео мы разобрали несколько вариантов векторных представлений слов, обсудили их достоинства и недостатки:

- Bag of Words (BoW);
- Tf-idf;
- Латентный семантический анализ.

В следующем видео мы обсудим еще несколько вариантов представлений слов в векторном виде и узнаем, что такое эмбединги.



Deep Learning School

Контекстные векторы слов. Эмбединги слов

План видео

- Задачи обработки естественного языка, обработки аудио
- Векторные представления слов
 - BOW
 - Tf-idf
 - LSA
- **Эмбединги слов:**
 - **Контекстные эмбединги**
 - **Word2Vec**
 - GloVe, FastText
- Эмбединги фраз и предложений
- Использование эмбедингов
- Понятие эмбединга в общем виде

Контекстные векторы слов

Рассмотрим три предложения и слова-кандидаты для пропусков:

1. Маша ездит на _____
2. Колесо _____ было проколото
3. У _____ красивая белая рама

	1	2	3
Велосипед	+	+	+
Мотоцикл	+	+	+
Машина	+	+	-
Лошадь	+	-	-

Контекстные векторы слов

Идея — смысл слова определяется контекстом.

Построим векторы слов на основе контекста:

	a	horse	ride	bicycle	frame	rose
a		256	45	230	90	134
horse	256		137	4	2	5
ride	45	137		120	34	3
bicycle	230	4	120		76	2
frame	90	2	34	76		0
rose	134	5	3	2	0	

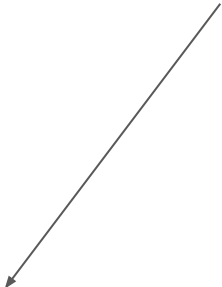
Контекстные векторы слов

Идея — смысл слова определяется контекстом.

Построим векторы слов на основе контекста:

	a	horse	ride	bicycle	frame	rose
a		256	45	230	90	134
horse	256		137	4	2	5
ride	45	137		120	34	3
bicycle	230	4	120		76	2
frame	90	2	34	76		0
rose	134	5	3	2	0	

Вектор слова — строка матрицы



Контекстные векторы слов

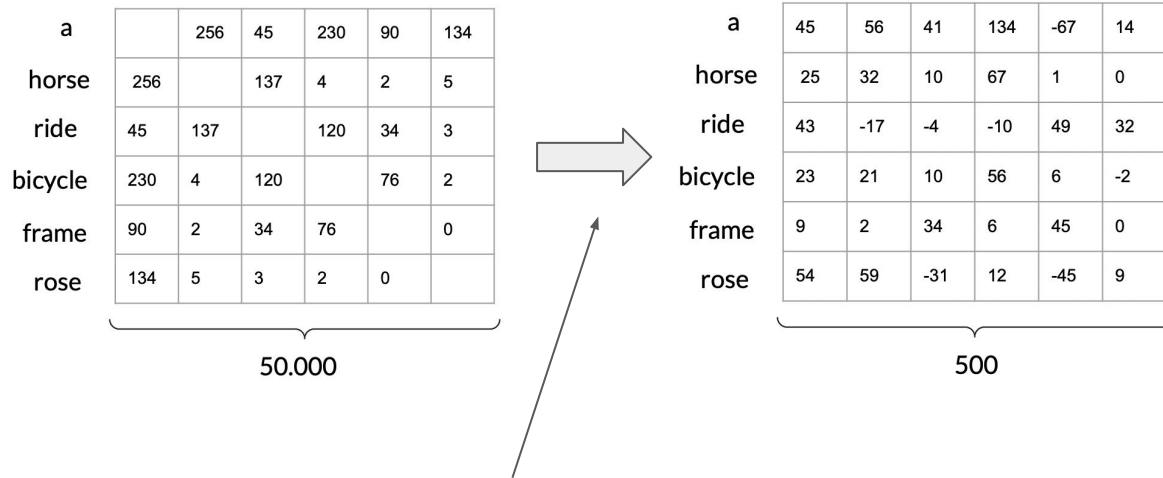
Плюсы:

- Векторы начинают отражать смысл слов! Их можно сравнивать на схожесть по расстоянию (cosine distance/MSE);

Недостатки:

- Векторы все еще довольно разрежены, требуют много лишней памяти;
- Размер словаря ограничен. Слова, не попавшие в словарь, не могут быть обработаны;
- При изменении размера словаря нужно пересчитывать векторы заново;
- Векторы редких слов не очень информативны.

Контекстные векторы слов



Сингулярное разложение

Матрица слова-
слова размера $n \times n$

слово

слово

\approx

вектор слова

вектор слова

W

SVD

\approx

d

\times

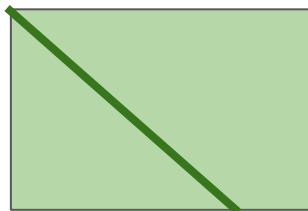
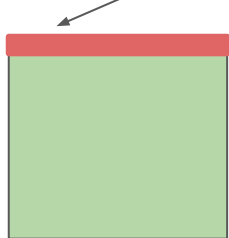
\times

$U (k \times k)$

$S (k \times n)$

диагональная матрица

$V^T (n \times n)$



Латентный семантический анализ

Матрица слова-
слова размера $n \times n$

слово

слово

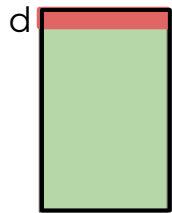
\approx

вектор слова

вектор слова

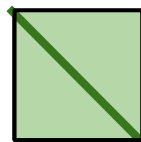
SVD

\approx



U ($k \times m$)

\times



S ($m \times m$)

\times



V^T ($m \times n$)

Эмбе́ддинги слов

Все, что мы делали до сих пор — на основе каких-то соображений строили векторы/матрицы слов/документов, которые как-то отражали смысл слов/документов.

А что если попытаться **выучить** векторы слов/документов?

Эмбединги слов

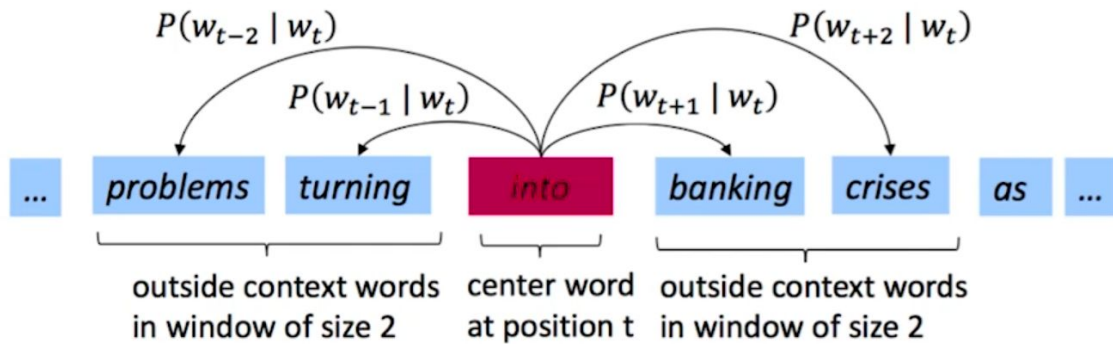
Что мы хотим:

Мы хотим выучить векторы слов небольшой размерности, которые отражали бы смысл слов: их можно было бы сравнивать между собой с помощью некой метрики.

Такие выученные векторы мы будем называть *эмбедингами слов*.

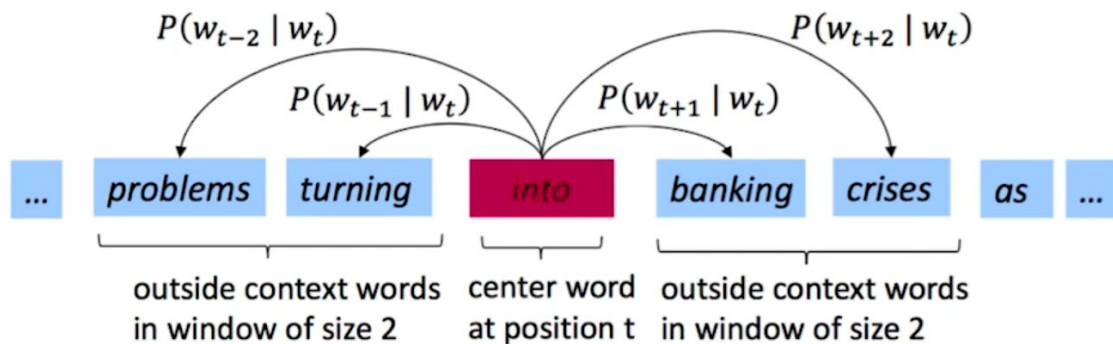
Word2Vec

Мы будем учить нейросеть по слову предсказывать слова, которые могут находиться в контексте (стоять вокруг этого слова).



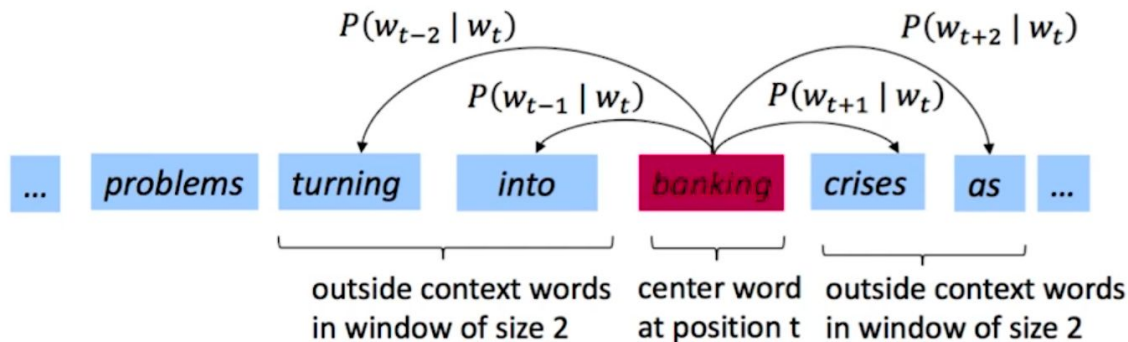
Word2Vec

Наш датасет — набор текстов. Мы будем идти по датасету скользящим окном размера 5, и в каждом положении окна по центральному слову учить нейросеть предсказывать слова, находящиеся в текущем окне.



Word2Vec

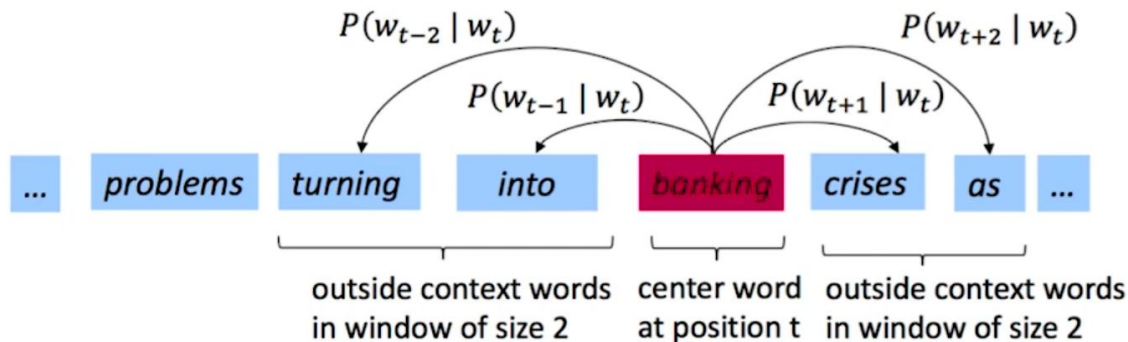
Наш датасет — набор текстов. Мы будем идти по датасету скользящим окном размера 5, и в каждом положении окна по центральному слову учить нейросеть предсказывать слова, находящиеся в текущем окне.



Word2Vec

Формализуем задачу:

- Ставится задача классификации.
Количество классов — размер словаря n .
- На вход нейросеть принимает слово, выдает n значений —
распределение на слова в словаре.
- Лосс-функция — кросс-энтропия между распределением,
выданным сетью, и верным распределением (one-hot вектором)

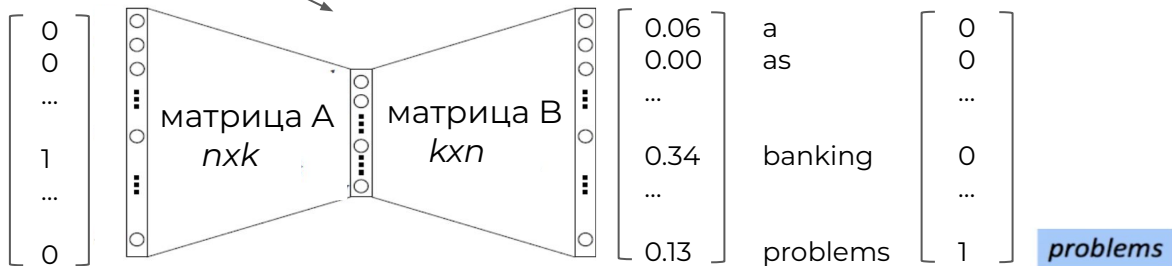


Word2Vec

нет функции
активации

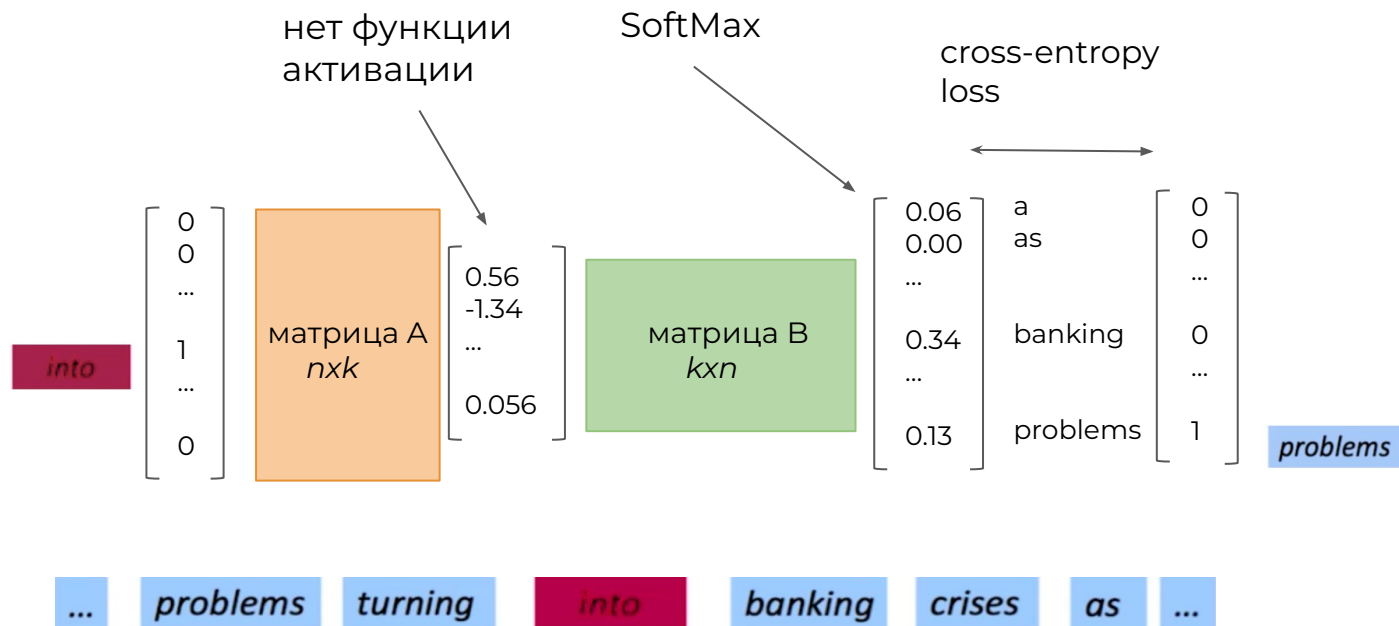
SoftMax

cross-entropy loss

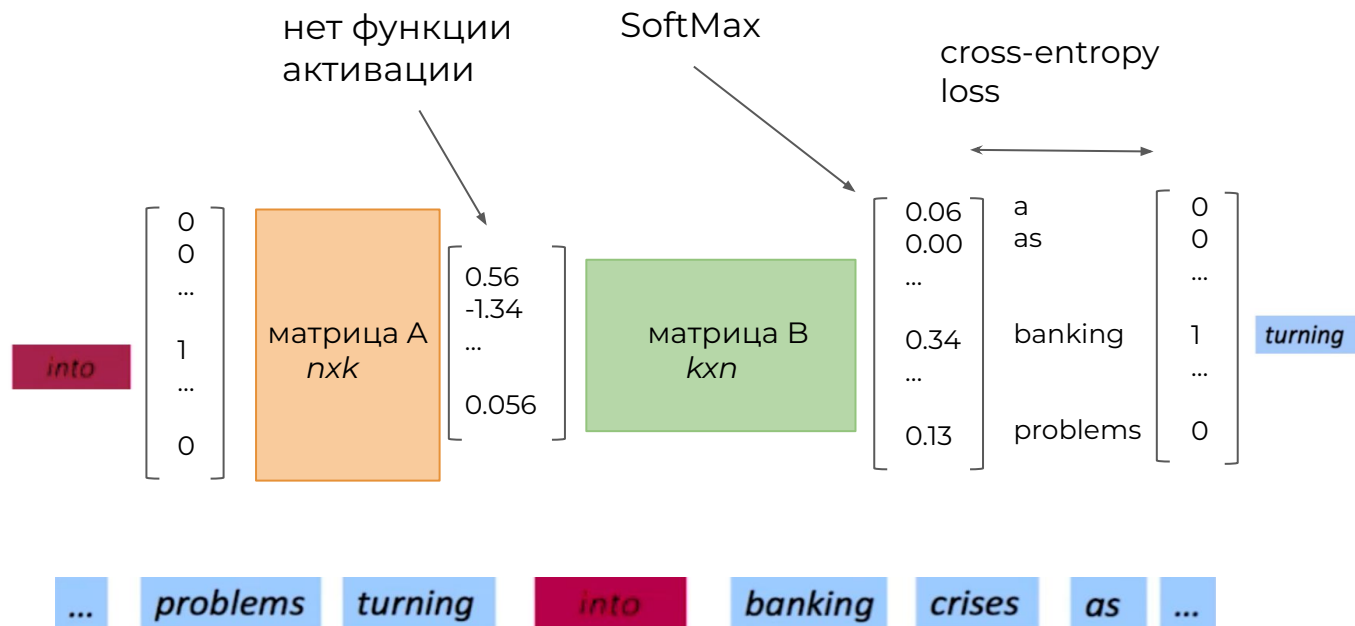


... problems turning into banking crises as ...

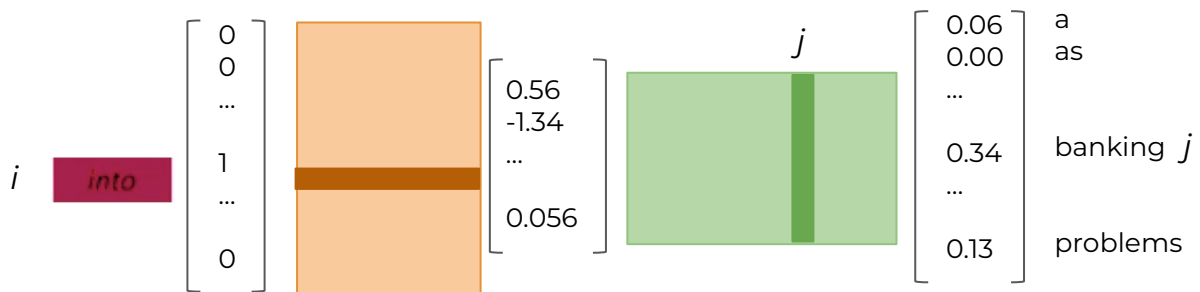
Word2Vec



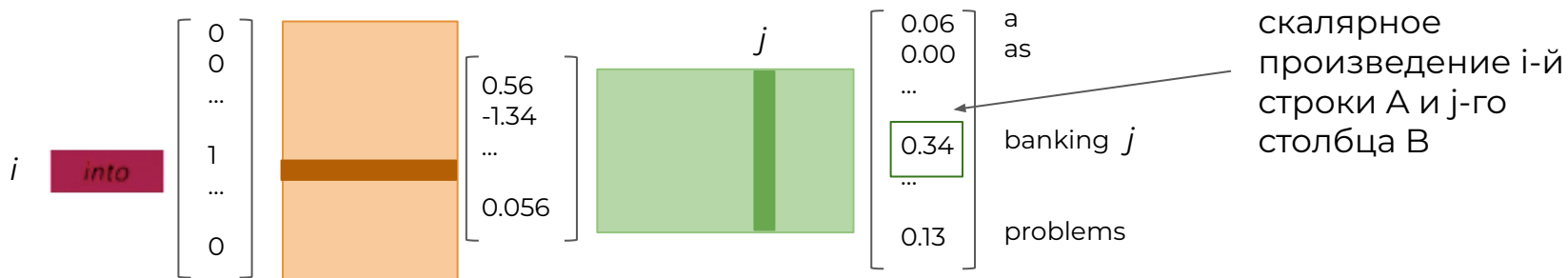
Word2Vec



Word2Vec

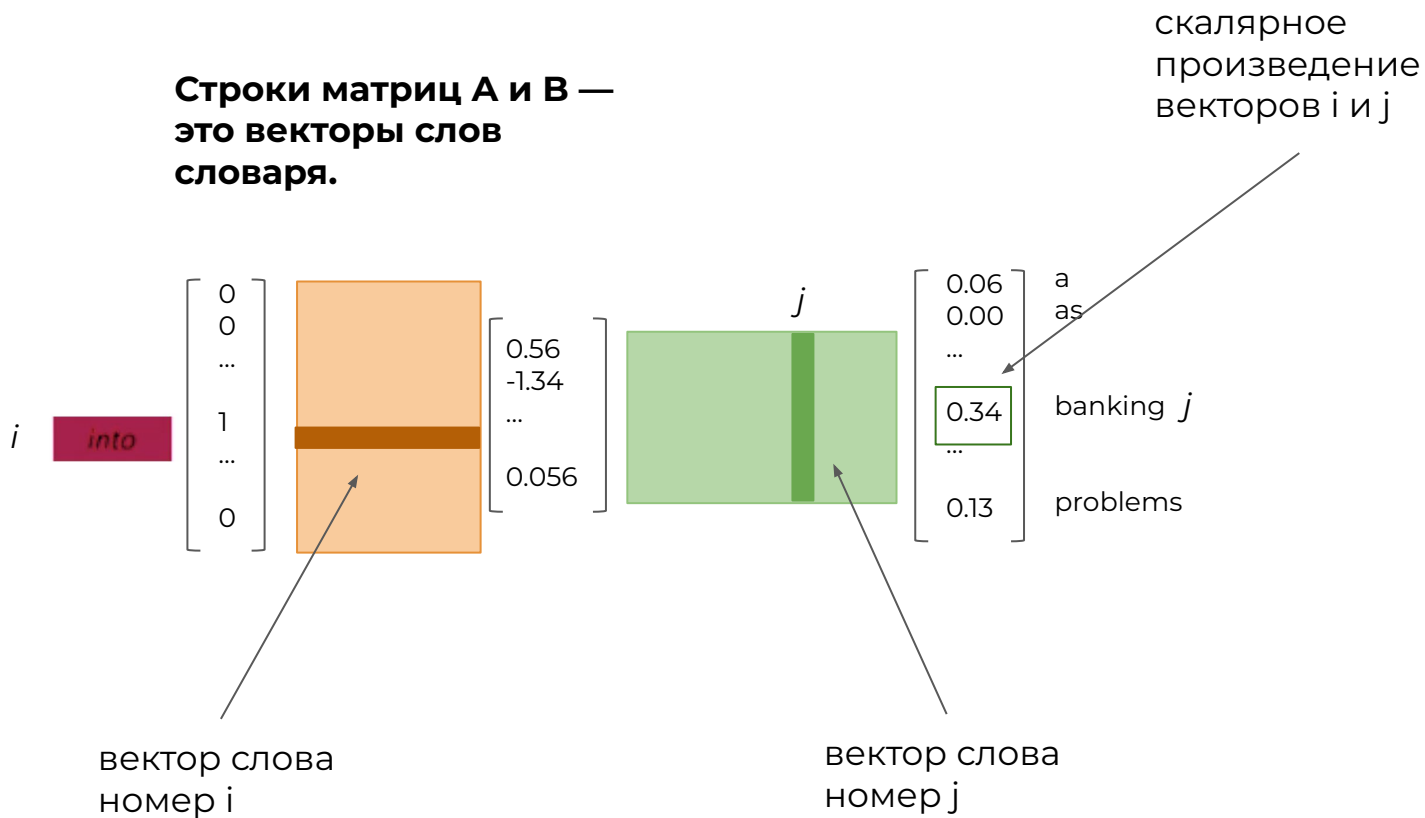


Word2Vec



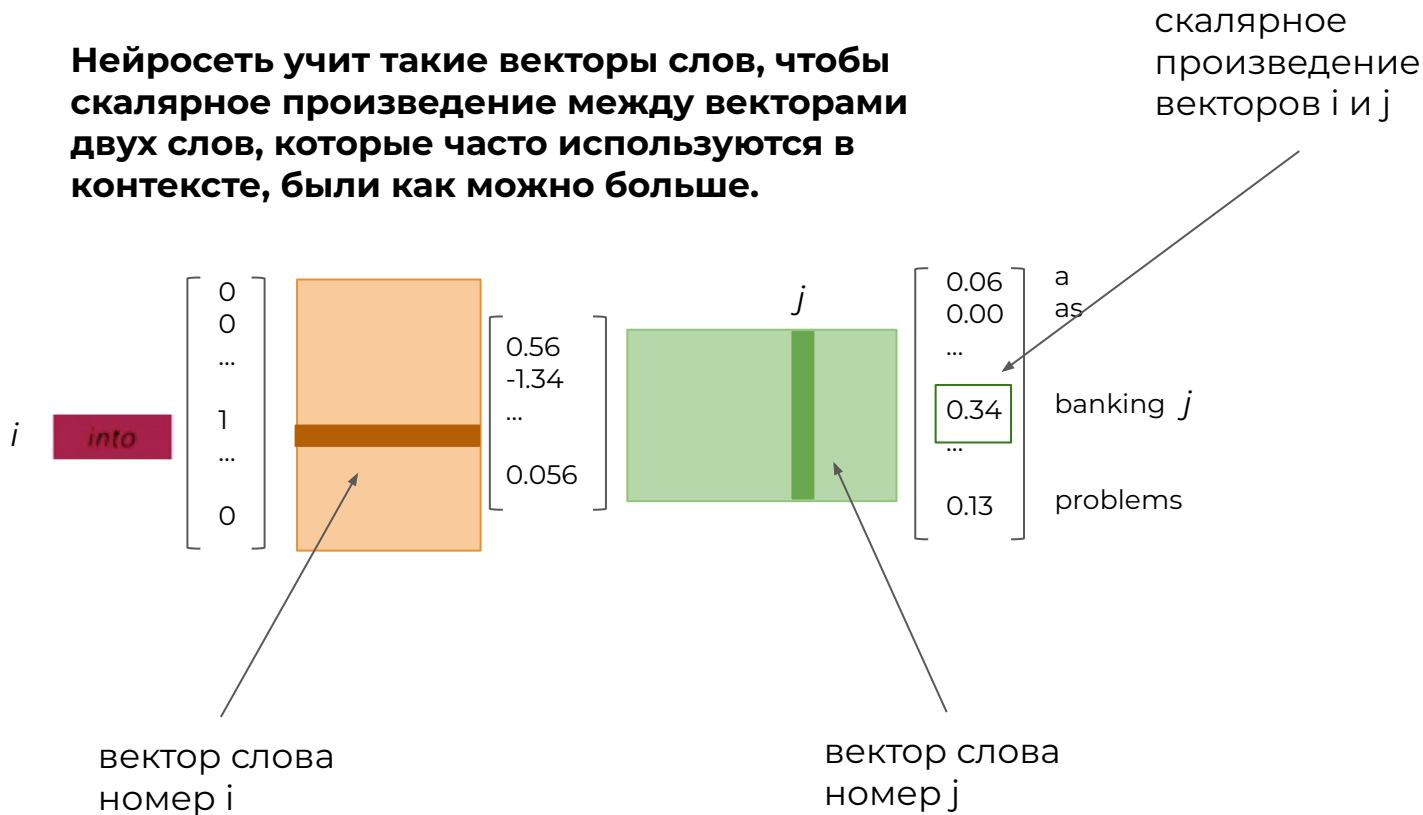
Word2Vec

Строки матриц **A** и **B** —
это векторы слов
словаря.

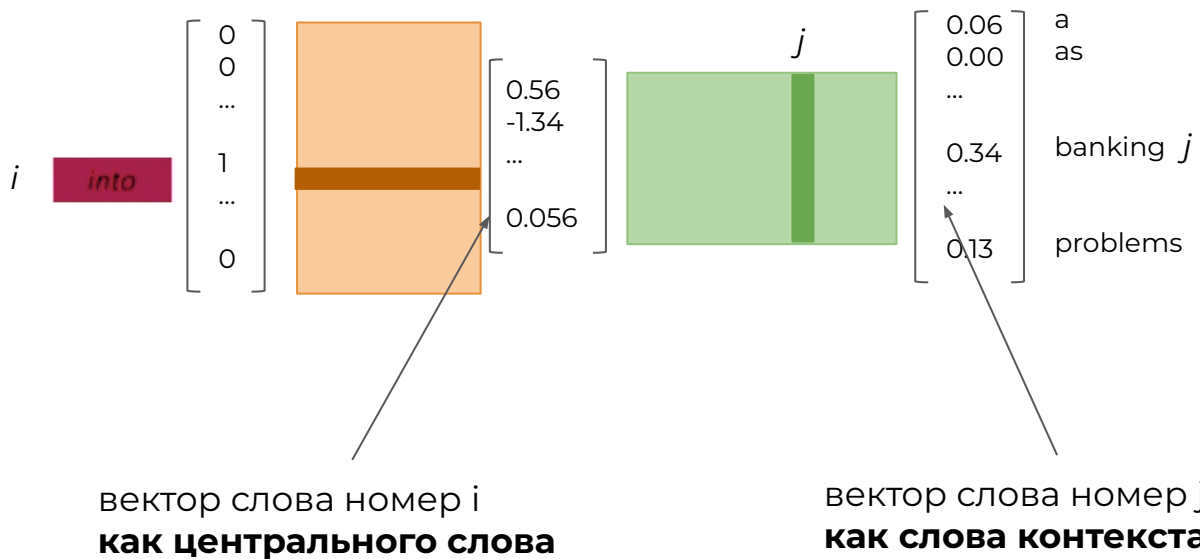


Word2Vec

Нейросеть учит такие векторы слов, чтобы скалярное произведение между векторами двух слов, которые часто используются в контексте, были как можно больше.



Word2Vec



Word2Vec

После обучения сети мы получаем векторы размера k для всех слов в словаре.

Размер k мы можем задавать сами.

Эти векторы содержат смысл слов. Их можно сравнивать между собой с помощью косинусного расстояния.

Косинусное расстояние — это нормализованное скалярное произведение двух векторов.

$$\text{similarity}(A, B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n A_i^2} \times \sqrt{\sum_{i=1}^n B_i^2}}$$

Word2Vec

На векторах word2vec можно проводить векторную арифметику:

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

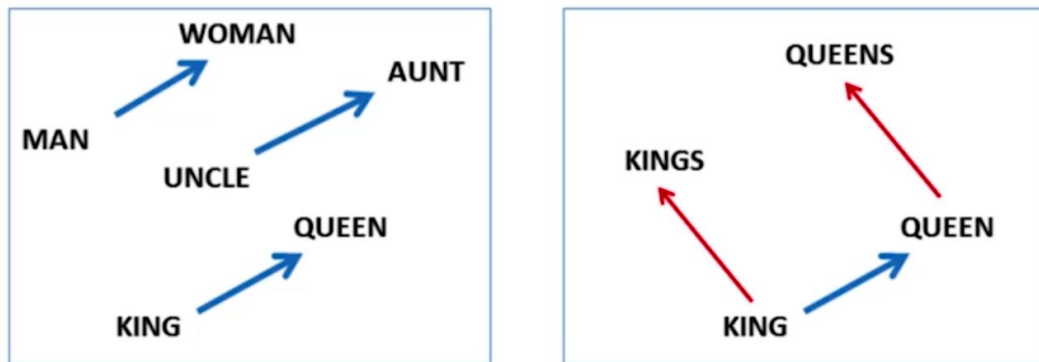
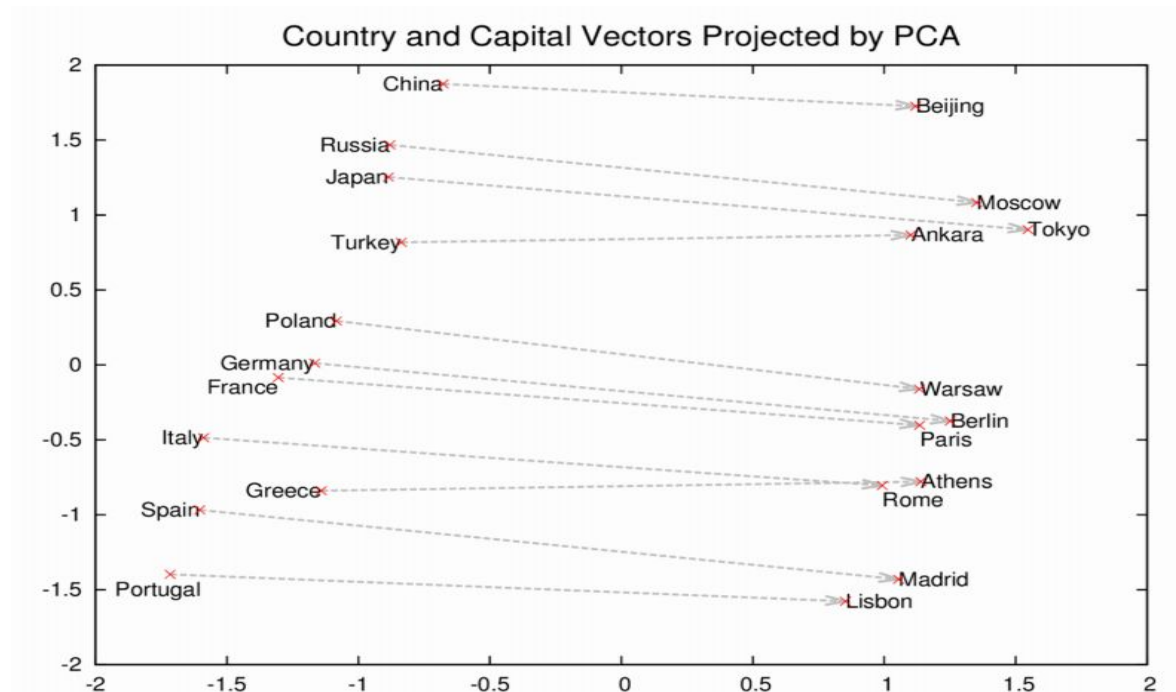
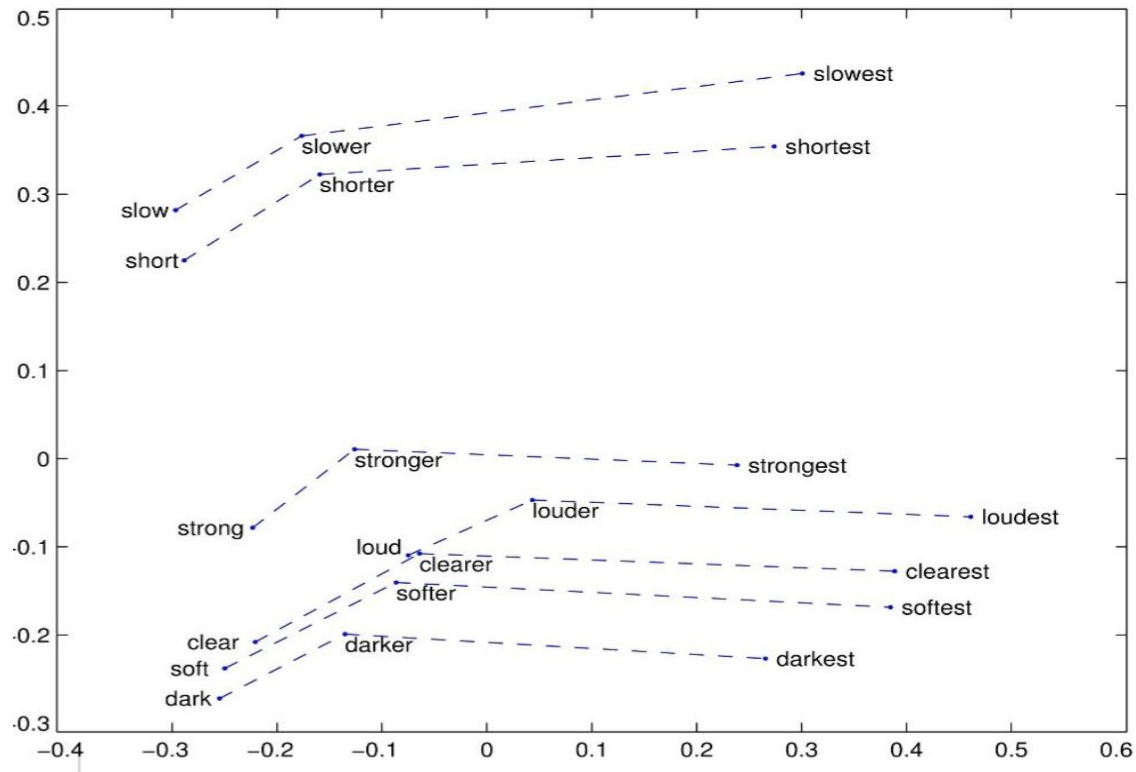


Иллюстрация эмбедингов после
понижения размерности

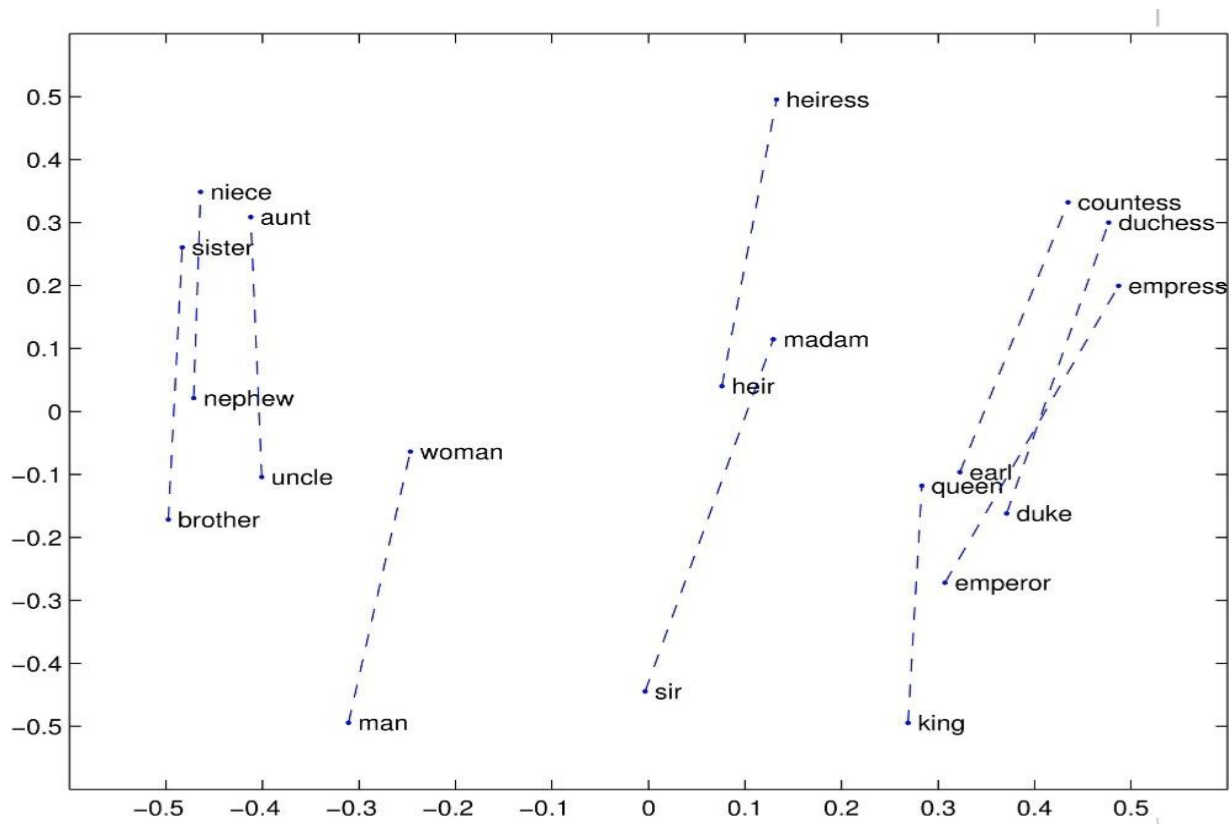
Word2Vec



Word2Vec



Word2Vec



Word2Vec

Преимущества:

- Векторы отражают смысл слов;
- Размерность векторов не зависит от размера словаря;
- При добавлении документов векторы можно дообучить.

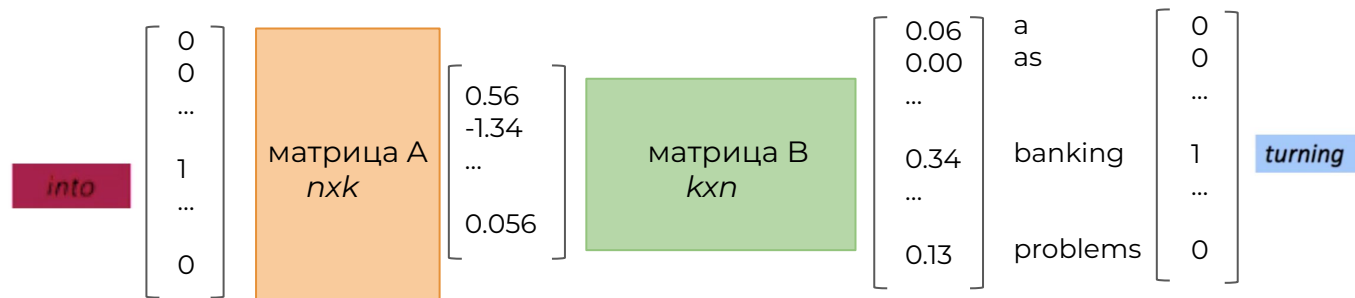
Недостатки:

- Фиксированный размер словаря. При изменении размера словаря документов векторы нужно пересчитывать;
- Для редких слов эмбединги получаются неоптимальными;
- Слова, имеющие один корень, обрабатываются нейросетью по-разному.

eat, eater, eating

Word2Vec

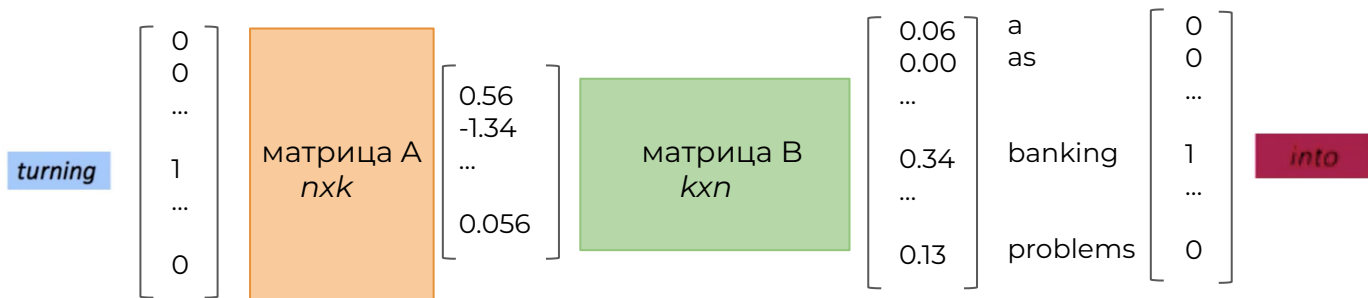
Skip-Gram — предсказание слов контекста по центральному слову



... problems turning into banking crises as ...

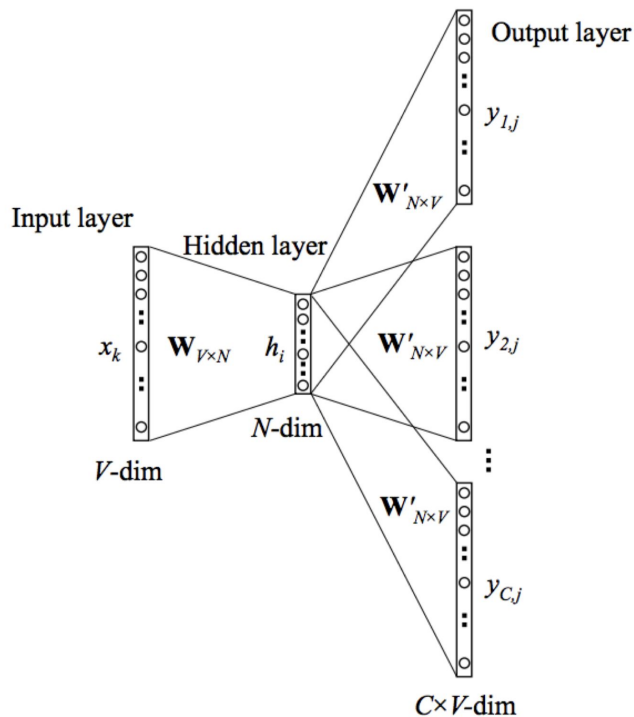
Word2Vec

CBOW — предсказание центрального слова
по словам контекста

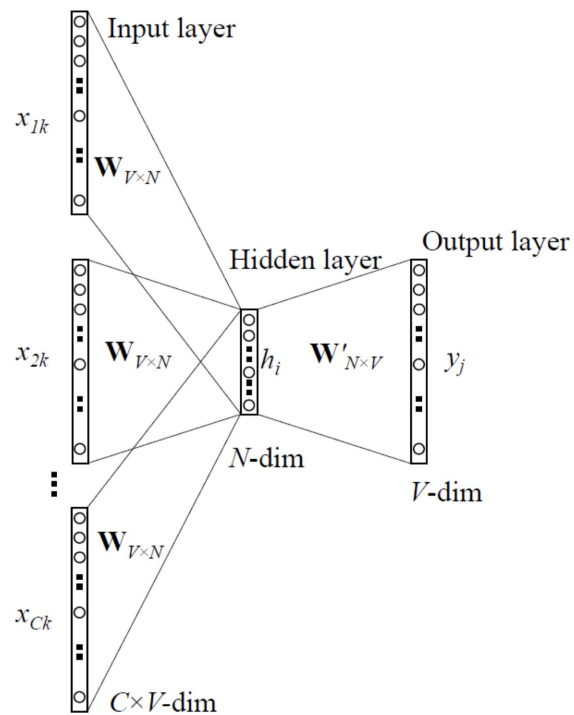


Word2Vec

Skip-gram



CBOW



Итоги видео

В этом видео мы:

- Познакомились с идеей того, что смысл слова можно определять через контекст;
- Построили эмбединги слов `word2vec`.

В следующем видео мы подробнее обсудим проблемы `word2vec` и способы их решения, а также поговорим об использовании эмбедингов.



Deep Learning School

Word2vec: детали. Эмбеддинги предложений

План видео

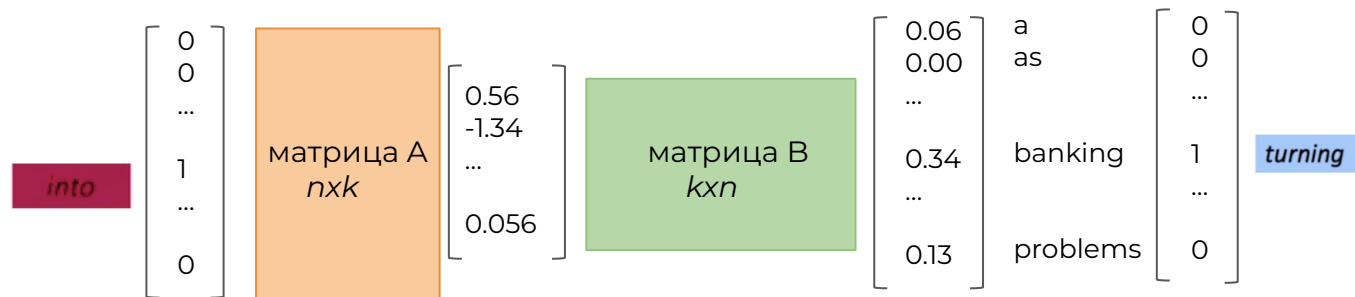
- Задачи обработки естественного языка, обработки аудио
- Векторные представления слов
 - BOW
 - Tf-idf
 - LSA
- Эмбединги слов:
 - Контекстные эмбединги
 - Word2Vec
 - **GloVe, FastText**
- **Эмбединги фраз и предложений**
- **Использование эмбедингов**
- **Понятие эмбединга в общем виде**

Word2Vec

SoftMax

огромное
выражение для
вычисления

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



... problems turning into banking crises as ...

Word2Vec

SoftMax

огромное
выражение для
вычисления

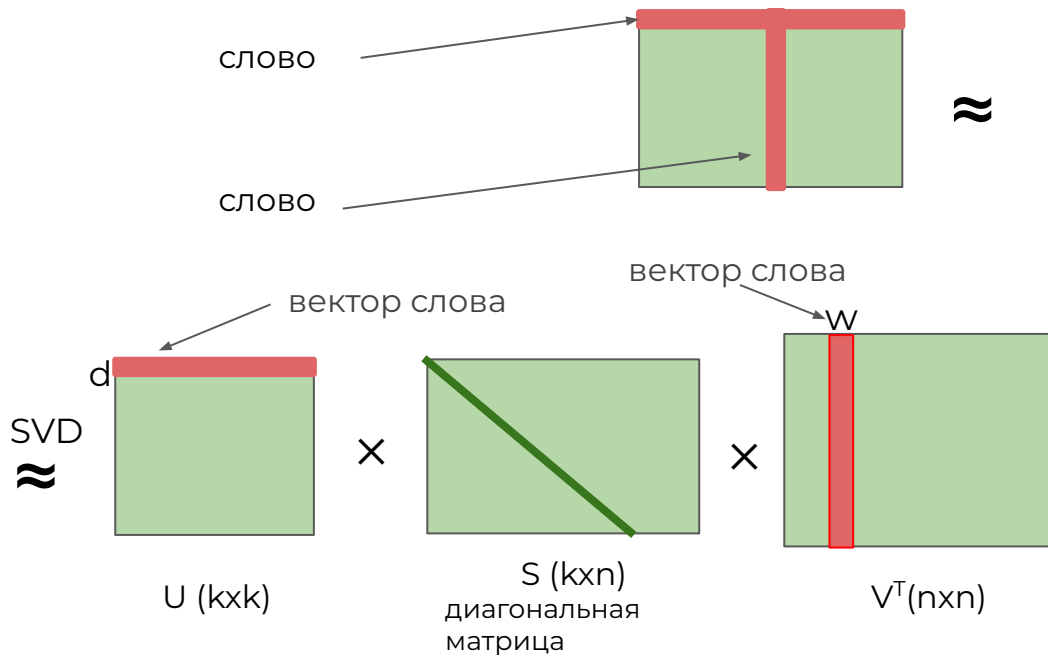
$$P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

Идеи решения проблемы:

- Иерархический SoftMax;
- Negative sampling;

Word2Vec vs SVD

Word2vec с negative sampling получает эмбединги, похожие на эмбединги из SVD-разложения



FastText

Идея — будем строить векторы для частей слов, а не для целых слов.

- Делим слова на n-граммы по буквам:
 $\text{apple} = \langle \text{ap}, \text{pp}, \text{ple}, \text{le} \rangle$
- Учим векторы для n-грамм;
- Вектор слова получаем как сумму векторов его n-грамм.

FastText

Идея — будем строить векторы для частей слов, а не для целых слов.

- Делим слова на n-граммы по буквам:
apple = $\langle \text{ap, ppl, ple, le} \rangle$
- Учим векторы для n-грамм;
- Вектор слова получаем как сумму векторов его n-грамм.

Плюсы:

- Можно получить более адекватные эмбединги для редких и неизвестных слов;

Недостатки:

- n-грамм может быть очень много. Требуется больше вычислительных ресурсов.

GloVe (Global Vectors)

GloVe использует статистическую информацию о частоте встречаемости слов и фраз в тексте, чтобы улучшить обучение эмбедингов редких слов.

Подробнее можно почитать тут:

<https://towardsdatascience.com/light-on-math-ml-intuitive-guide-to-understanding-glove-embeddings-b13b4f19c010>

Эмбе́ддинги предложений

Как можно использовать идею для получения эмбе́ддингов слов, чтобы получить эмбе́ддинги предложений?

Эмбе́ддинги предложений

Что мы делали со словами:

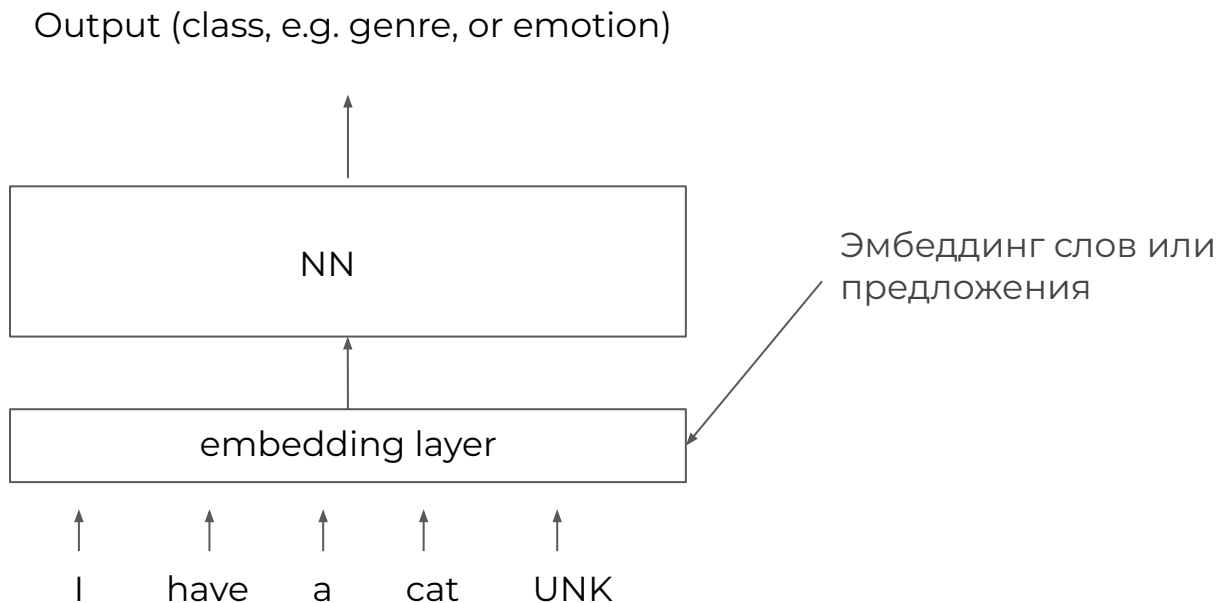
- Предсказывали следующее/предыдущее слово.

Что можно делать для предложений:

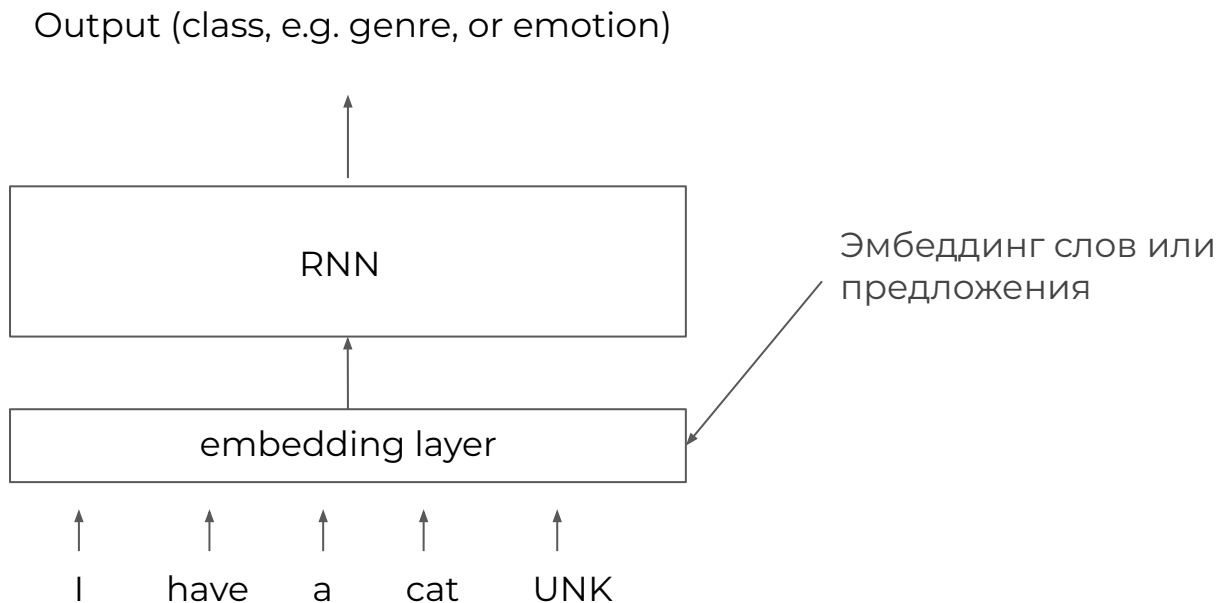
- Предсказывать порядок следования двух предложений (бинарная классификация);
- Может ли предложение А идти после предложения В? (бинарная или многоклассовая классификация)
- Предсказание соединительного слова между двумя предложениями.

Обучаясь на подобные задачи, нейросеть выучивает некую информацию о предложениях.

Классификация текста с помощью эмбедингов



Классификация текста с помощью эмбедингов



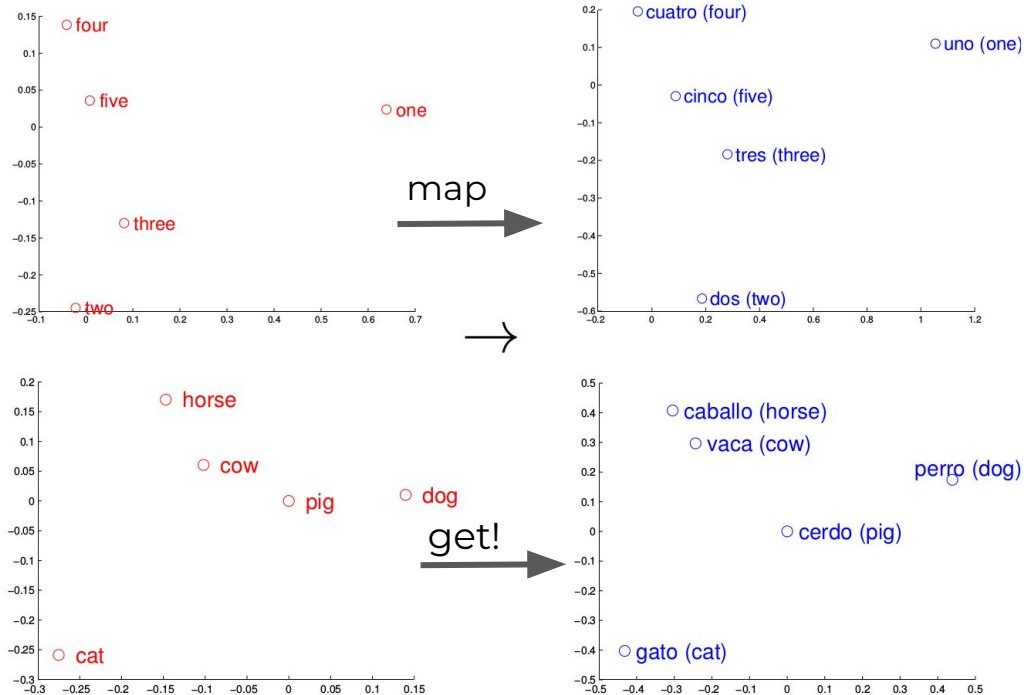
Карта языка

Пусть у нас есть тексты на неизвестном языке,
которые мы хотим научиться понимать.

Как это можно сделать:

- Обучаем эмбединги для слов английского языка;
- Обучаем эмбединги для неизвестного языка
- Находим преобразование f , которое переводит эмбединги английского языка в эмбединги неизвестного языка.

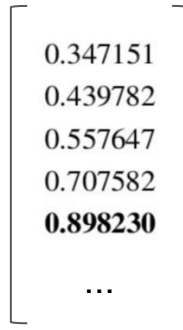
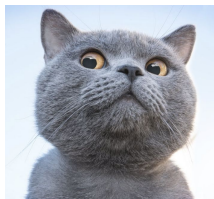
Карта языка



Эмбединги в общем смысле

Эмбединг — векторное представление объекта, которое отражает информацию об объекте.

Выходы слоев моделей, обученных под какую-либо задачу, тоже можно считать эмбедингами.



Эмбединги в общем смысле

Пример: поиск похожих изображений

- Берем предобученную VGG16 на ImageNet;
- Прогоняем все картинки датасета через модель, получаем выходы (эмбединги) предпоследнего слоя модели;
- Для конкретной картинки ищем похожие, сравнивая эмбединг этой картинки с эмбедингами других картинок датасета по MSE/косинусному расстоянию.



Итоги видео

В этом видео мы:

- Разобрали некоторые недостатки word2vec и способы их решения;
- Узнали, как использовать эмбединги для решения задач на текстах;
- Обсудили эмбединги фраз и предложений, а также общий смысл слова эмбединг.