

Работа с текстовыми данными

Артем Зраев

Занятие № 7

План занятия



1. Примеры задач, решаемых в NLP
2. Задача классификации
3. «Классический» пайплайн решения задачи
4. Извлечение признаков. BoW
5. Предобработка: фильтрация, стемминг/лемматизация
6. Извлечение признаков. BoW: тематическое моделирование
7. Векторные представления слов
8. Практика (bigARTM на заголовках новостей)

Прикладные задачи

1. Машинный перевод
2. Задача классификации
3. NER
4. Тематическое моделирование
5. Text summarization
6. Диалоговые системы
7. Etc





Извлекаемая информация: персоны, компании, адреса, даты, места работы, действия и т.д

Так говорила в июле 1805 года известная Анна Павловна Шерер, фрейлина и приближенная императрицы Марии Феодоровны, встречая важного и чиновного князя Василия, первого приехавшего на ее вечер. Анна Павловна кашляла несколько дней, у нее был грипп, как она говорила (грипп был тогда новое слово, употреблявшееся только редкими).

- Физ. лица ● Организации ● События ● Гео-объекты
- Объекты времени ● Денежные единицы ● Бренды

Тематическое моделирование



1. Создание рубрик документов (и не только)
2. Поиск информации
3. Выявление трендов
4. Как подзадача в классификации и рекомендациях

Можно рассматривать тематическое моделирование как нечеткую кластеризацию.

Text summarization



Source Text: Peter and Elizabeth took a taxi to attend the night party in the city.

While in the party, Elizabeth collapsed and was rushed to the hospital.

Summary: Elizabeth was hospitalized after attending a party with Peter.



Задача классификации (пример)



Nonsense? kiss off, geek. What I said is true. I'll have your account terminated.

TOXIC

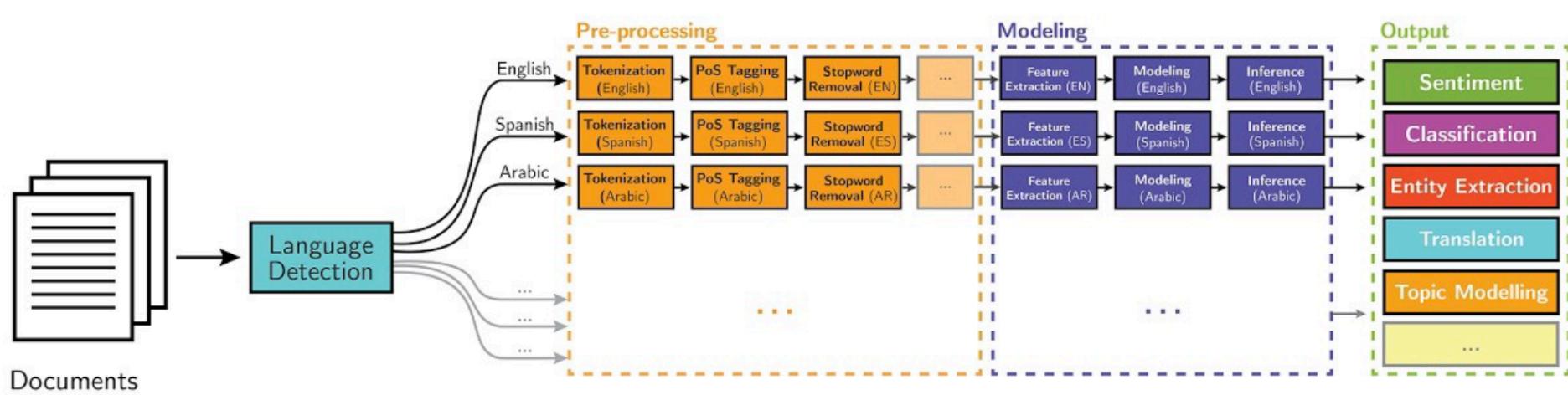
"Ban one side of an argument by a bullshit nazi admin and you get no discussion because the islamist editors feel they ""won""."

TOXIC
 OBSCENE
 INSULT

Why can you put English for example on some players but others people don't like it :- why?

SAFE

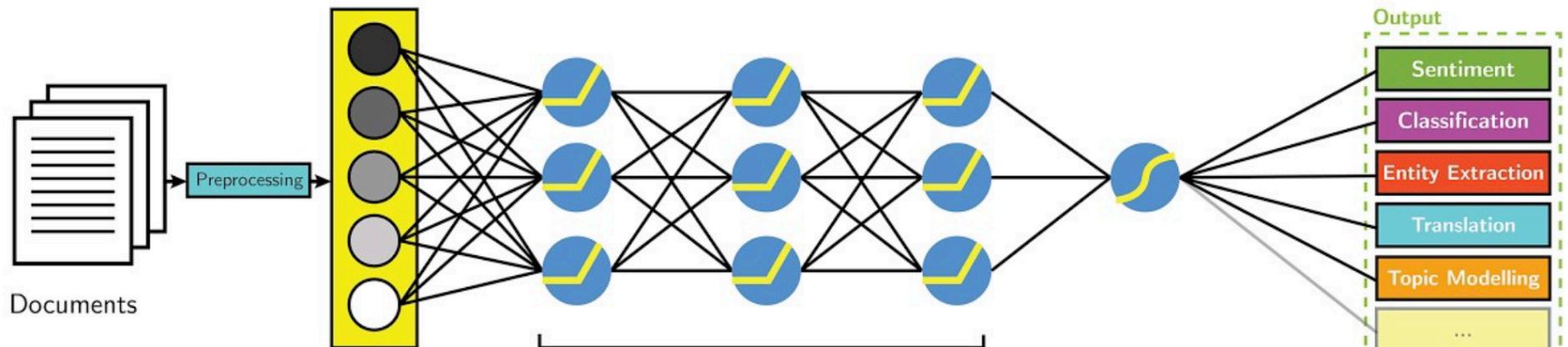
«Классический» пайплайн решения задач



«Современный» пайплайн решения задач



Deep Learning-based NLP



Задача: классификация заявок в ТП



Дано:

1. Есть пользователи, у которых что-то ломается
2. Есть техподдержка, которая это чинит
3. Заявки прилетают в местный helpdesk

Задача: классификация заявок в ТП



Проблема:

Заявки приходится распределять руками между подразделениями, что требует чтения глазами и времени (не считая необходимости держать людей на такой рутинной и тяжелой работе)

Задача: классификация заявок в ТП



Как будем решать:

Сделать модель, которая будет распределять заявки по подразделениям автоматически (хотя бы какую-то их часть, достаточную для снижения нагрузки на людей)

Извлечение признаков. BoW



```
In [1]: import pandas as pd  
from sklearn.feature_extraction.text import CountVectorizer
```

```
In [2]: texts = ["Недоступен сайт ya.ru, не пингуется",  
             "не работает монитор",  
             "Не могу подключиться к серверу по ssh",  
             "на мониторе потемнения"]
```

```
In [3]: cv = CountVectorizer(lowercase=False)  
tdata = cv.fit_transform(texts)  
ft = cv.get_feature_names()
```

```
In [4]: pd.DataFrame(tdata.todense(),  
                  columns=ft)
```

Out[4]:

	ru	ssh	ya	He	Недоступен	могу	монитор	мониторе	на	не	пингуется	по	подключиться	потемнения	работает	сайт	серверу
0	1	0	1	0		1	0	0	0	0	1	1	0		0	0	0
1	0	0	0	0		0	0	1	0	0	1	0	0		0	1	0
2	0	1	0	1		0	1	0	0	0	0	0	1		1	0	0
3	0	0	0	0		0	0	0	1	1	0	0	0		1	0	0

Здесь явно что-то не так

Извлечение признаков. BoW



```
In [5]: cv = CountVectorizer(lowercase=True)
```

```
In [6]: tdata = cv.fit_transform(texts)
ft = cv.get_feature_names()
```

```
In [7]: pd.DataFrame(tdata.todense(),
                    columns=ft)
```

Out[7]:

	ru	ssh	я	могу	монитор	мониторе	на	не	недоступен	пингуется	по	подключиться	потемнения	работает	сайт	серверу
0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	1	0
1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0
2	0	1	0	1	0	0	0	1	0	0	1	1	0	0	0	1
3	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0

Еще какие проблемы с таким представлением?

Извлечение признаков. BoW



TF = (Number of time the word occurs in the text) / (Total number of words in text)

IDF = (Total number of documents / Number of documents with word t in it)

TF-IDF = TF * IDF

```
In [8]: from sklearn.feature_extraction.text import TfidfTransformer  
  
In [9]: transformer = TfidfTransformer()  
transformed_weights = transformer.fit_transform(tdata)  
weights = np.asarray(transformed_weights.mean(axis=0)).ravel().tolist()  
weights_df = pd.DataFrame({'term': ft, 'weight': weights})
```

```
In [10]: weights_df.sort_values(by='weight', ascending=False).head(10)
```

Out[10]:

	term	weight
7	не	0.240088
4	монитор	0.161126
13	работает	0.161126
5	мониторе	0.144338
6	на	0.144338
12	потемнения	0.144338
0	ru	0.107509
1	ssh	0.107509
2	я	0.107509
3	могу	0.107509

Словом с самым большим значением оказалась частица не. Давайте исправим это!

Извлечение признаков. BoW



Просто уберем частицы из предложений.

```
In [14]: cv = CountVectorizer(stop_words=[u"и", u"на", u"по"])
tdata = cv.fit_transform(texts)
ft = cv.get_feature_names()
```

```
In [15]: transformer = TfidfTransformer()
transformed_weights = transformer.fit_transform(tdata)
weights = np.asarray(transformed_weights.mean(axis=0)).ravel().tolist()
weights_df = pd.DataFrame({'term': ft, 'weight': weights})
```

Посмотрим как теперь выглядит наша матрица

```
In [16]: pd.DataFrame(tdata.todense(),
                     columns=ft)
```

```
Out[16]:
ru ssh ya могу монитор мониторе недоступен пингуется подключиться потемнения работает сайт серверу
0 1 0 1 0 0 0 1 1 0 0 0 1 0
1 0 0 0 0 1 0 0 0 0 0 1 0 0
2 0 1 0 1 0 0 0 0 1 0 0 0 1
3 0 0 0 0 0 1 0 0 0 1 0 0 0
```

	term	weight
4	монитор	0.176777
5	мониторе	0.176777
9	потемнения	0.176777
10	работает	0.176777
1	ssh	0.125000
3	могу	0.125000
8	подключиться	0.125000
12	серверу	0.125000
0	ru	0.111803
2	ya	0.111803

Что делать со словоформами?

Извлечение признаков. BoW



```
In [15]: import pymorphy2
```

```
morph = pymorphy2.MorphAnalyzer()
```

Пройдемся в цикле (так делать нехорошо, но для наших целей сойдет)

```
In [29]: texts_morph = []
for text in texts:
```

```
    texts_morph.append(" ".join([morph.parse(word)[0].normal_form for word in text.split()]))
```

```
In [30]: cv = CountVectorizer(stop_words=[u"не", u"на", u"по"])
tdata = cv.fit_transform(texts_morph)
ft = cv.get_feature_names()
```

```
In [31]: transformer = TfidfTransformer()
```

```
transformed_weights = transformer.fit_transform(tdata)
weights = np.asarray(transformed_weights.mean(axis=0)).ravel().tolist()
weights_df = pd.DataFrame({'term': ft, 'weight': weights})
```

```
In [32]: pd.DataFrame(tdata.todense(),
                     columns=ft)
```

Out[32]:

	ru	ssh	ya	монитор	мочь	недоступный	пинговаться	подключиться	потемнение	работать	сайт	сервер
0	1	0	1	0	0	1	1	0	0	0	1	0
1	0	0	0	1	0	0	0	0	0	1	0	0
2	0	1	0	0	1	0	0	1	0	0	0	1
3	0	0	0	1	0	0	0	0	1	0	0	0

	term	weight
3	монитор	0.309565
8	потемнение	0.196322
9	работать	0.196322
1	ssh	0.125000
4	мочь	0.125000
7	подключиться	0.125000
11	сервер	0.125000
0	ru	0.111803
2	ya	0.111803
5	недоступный	0.111803

Предобработка текстов



1. Приведение к нижнему регистру
2. Удаление стопслов
3. Лемматизация/стемминг
4. Фильтрация по частоте встречаемости

Извлечение признаков. Тематическое моделирование



Как человек генерирует текст?

1. Новый документ
2. Выбирается тематика из распределения тематик документа
3. Выбирается слово из распределения слов в выбранной тематике
4. Слово записывается
5. Повторить 1-4

Цель вероятностного тематического моделирования - восстановить распределения (а точнее - их параметры) так чтобы вероятность породить данную коллекцию документов была максимальна.

Извлечение признаков. Тематическое моделирование



Допущения:

1. Порядок слов не важен, мы коллекцию документов рассматриваем как набор пар (документ, слово)
2. Фиксированное количество тематик
3. Тематика описывается распределением
4. Документ также описывается распределением
5. Мы допускаем независимость слов от документов (очень сильное допущение!)

Извлечение признаков. Тематическое моделирование



Тематическое моделирование (topic modelling) - приложение машинного обучения к статистическому анализу текстов.

Тема - терминология предметной области, набор слов (униграмм или n-грамм) часто совместно встречающихся в документах.

Тематическая модель исследует скрытую тематическую структуру коллекции текстов:

- * тема t - вероятностное распределение над терминами
- * документ d - вероятностное распределение над темами

Тема - это набор слов, глядя на которые можно сказать, какую предметную область они описывают.

$$p(w|d) = \sum_{t \in T} p(w|t)p(t|d)$$

Извлечение признаков. Тематическое моделирование



	doc_1	doc_2	doc_3	doc_4	doc_5
word_1	1	0	0	0	0
word_2	0	1	0	0	0
word_3	0	0	1	0	0
word_4	0	0	0	1	0
word_5	0	0	0	0	1
word_6	1	0	0	0	0
word_7	0	1	0	0	0
word_8	0	0	0	1	0

=

	topic_1	topic_2	topic_3
word_1	1	0	0
word_2	0	1	0
word_3	0	0	1
word_4	1	0	0
word_5	0	1	0
word_6	0	0	1
word_7	0	1	0
word_8	0	0	1

$$F = p(w|d)$$

$$\Phi = p(w|t)$$

X

	doc_1	doc_2	doc_3	doc_4	doc_5
topic_1	1	0	0	0	0
topic_2	0	1	0	0	0
topic_3	0	0	1	0	0

$$\Theta = p(t|d)$$

Тематическое моделирование. Пример



```
In [1]: import pandas as pd
from nltk.corpus import gutenberg
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.decomposition import LatentDirichletAllocation

import pyLDAvis
import pyLDAvis.sklearn
pyLDAvis.enable_notebook()

%matplotlib inline
```

Доступные произведения

```
In [2]: gutenberg.fileids()
```

```
Out[2]: ['austen-emma.txt',
 'austen-persuasion.txt',
 'austen-sense.txt',
 'bible-kjv.txt',
 'blake-poems.txt',
 'bryant-stories.txt',
 'burgess-busterbrown.txt',
 'carroll-alice.txt',
 'chesterton-ball.txt',
 'chesterton-brown.txt',
 'chesterton-thursday.txt',
 'edgeworth-parents.txt',
 'melville-moby_dick.txt',
 'milton-paradise.txt',
 'shakespeare-caesar.txt',
 'shakespeare-hamlet.txt',
 'shakespeare-macbeth.txt',
 'whitman-leaves.txt']
```

Попробуем обучить LDA на тексте библии и посмотрим какие темы у нас получатся

Тематическое моделирование. Пример



topic_0
wise glory
thyself give
seek wilderness
seed judgment

topic_2
know spirit
power works
jesus
mouth world
christ
faith

topic_4
mother name
daughter jacob wife
jacob abraham
father called

topic_1
long live cast
young written
sword though

topic_3
field round sons
drank told

topic_5
thousand days
five two three
years seven
old four
hundred

Word Embeddings



Эмбеддинг слова - такое отображение из дискретного пространства признаков в непрерывный вектор меньшей (как правило) размерности.

Примеры: word2vec, glove, muse, bert

Зачем:

1. уменьшение размерности
2. учет смысловой близости слов

Word Embeddings (skipgram)



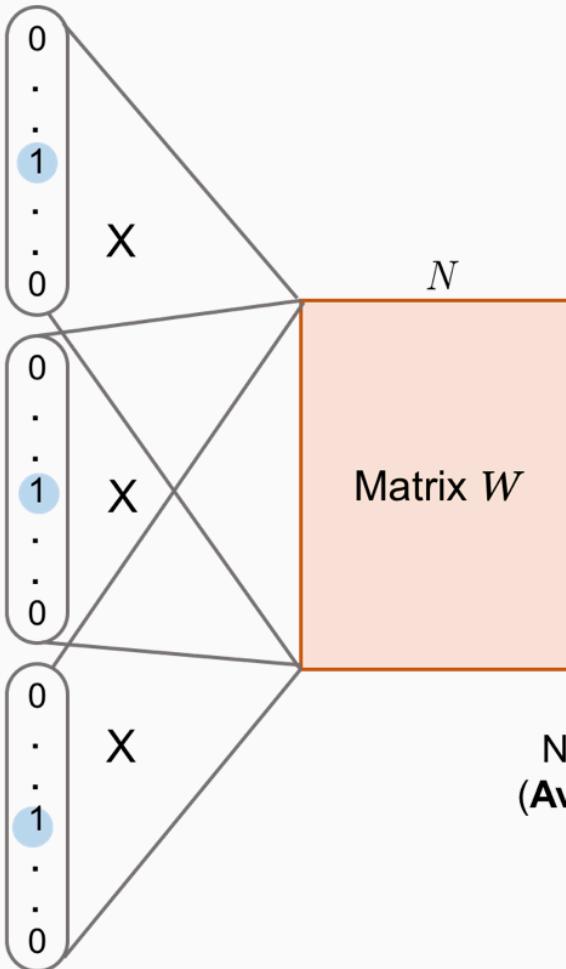
"The man who passes the sentence should swing the sword." – Ned Stark

Sliding window (size = 5)	Target word	Context
[The man who]	the	man, who
[The man who passes]	man	the, who, passes
[The man who passes the]	who	the, man, passes, the
[man who passes the sentence]	passes	man, who, the, sentence
...
[sentence should swing the sword]	swing	sentence, should, the, sword
[should swing the sword]	the	should, swing, sword
[swing the sword]	sword	swing, the

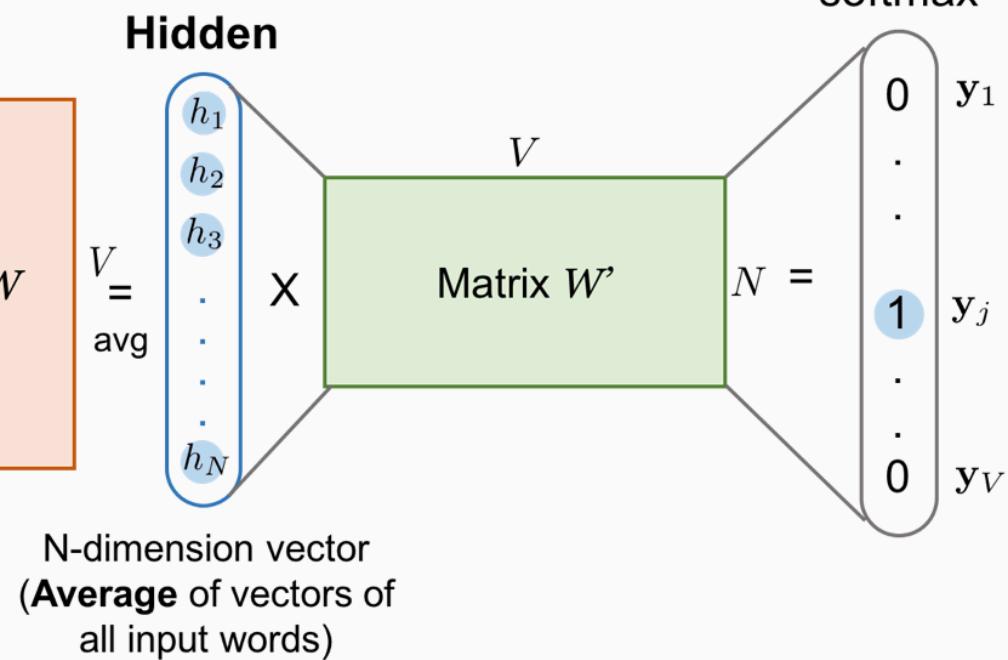
Word Embeddings (cbow)



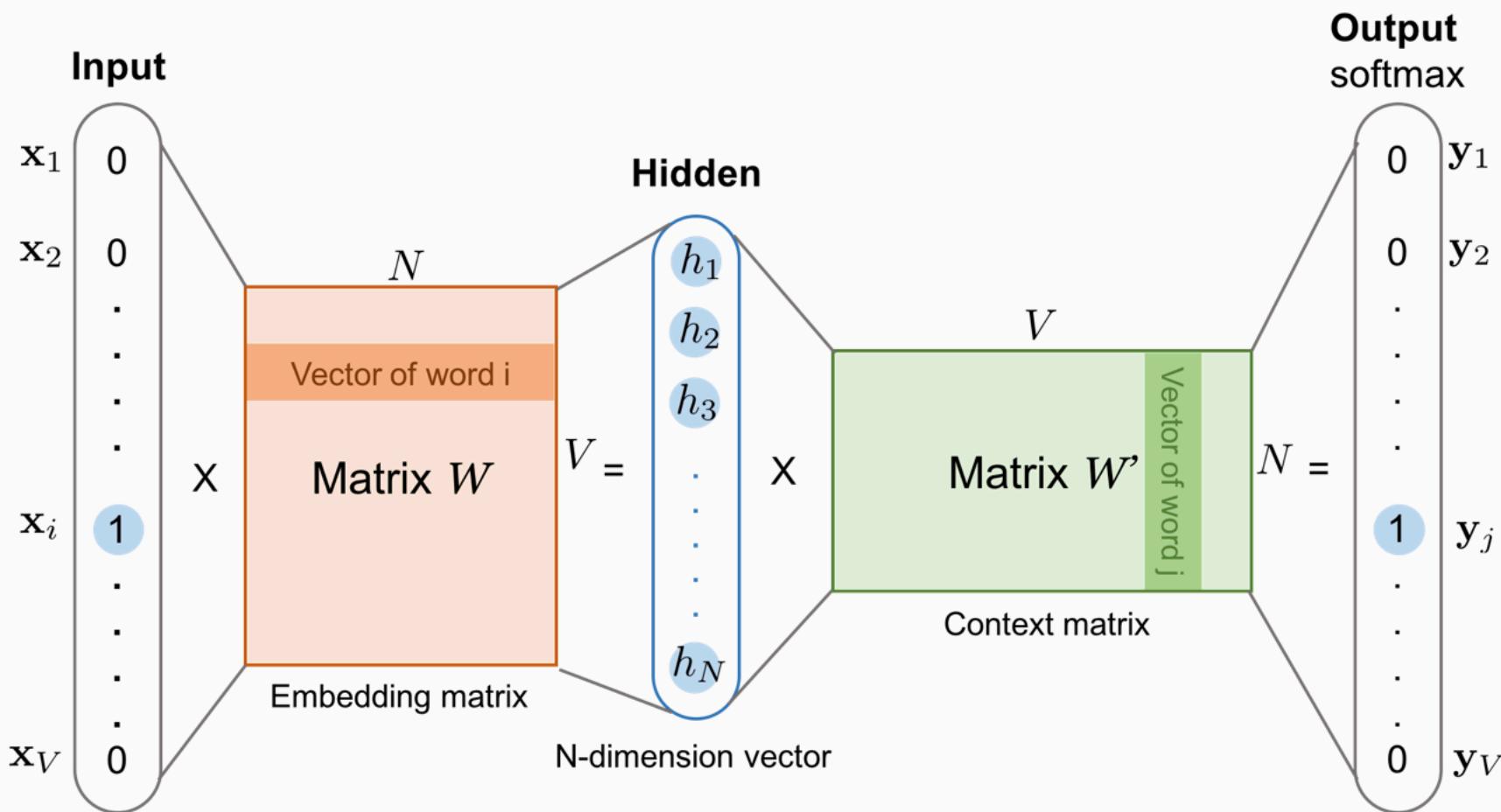
Input



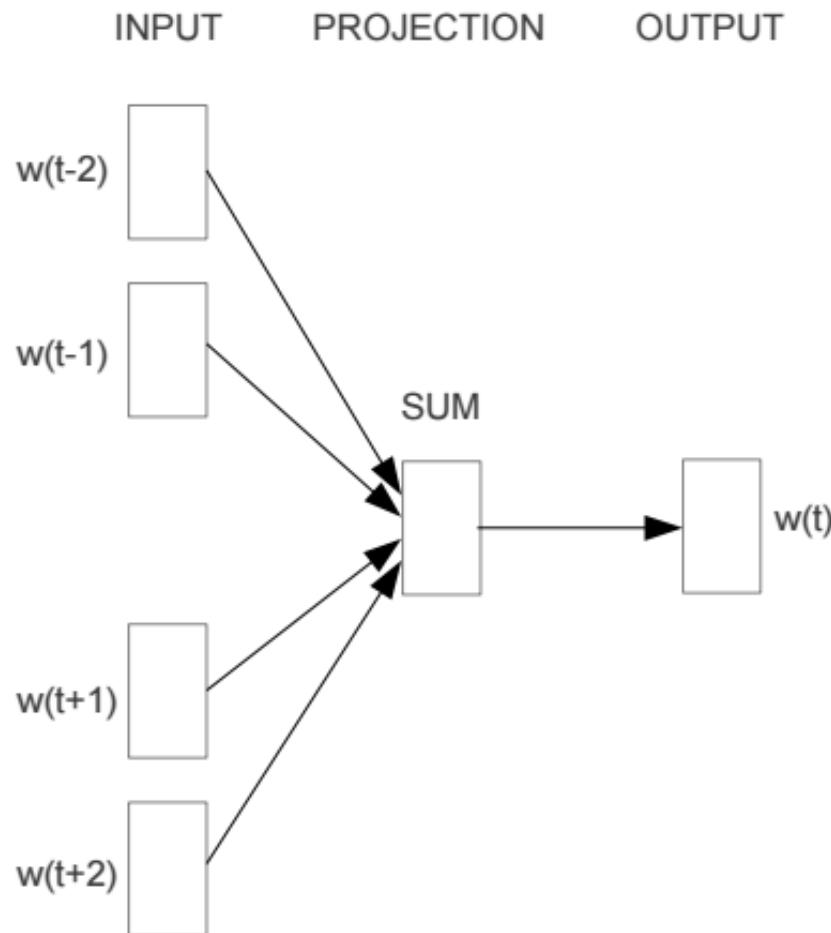
Hidden



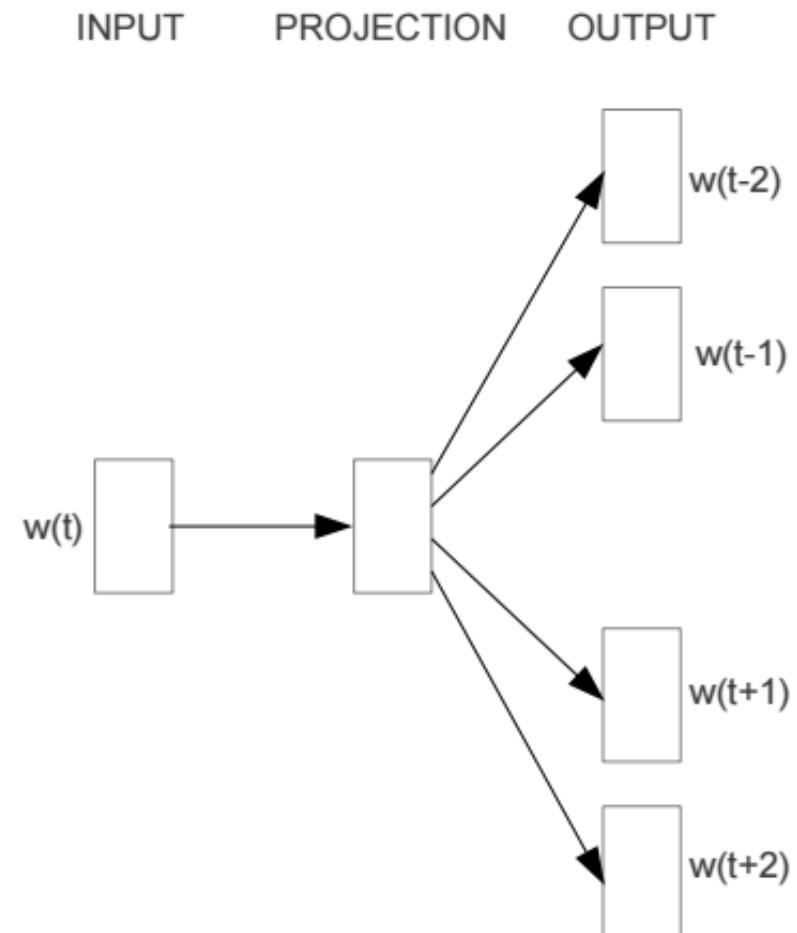
Word Embeddings



Word Embeddings

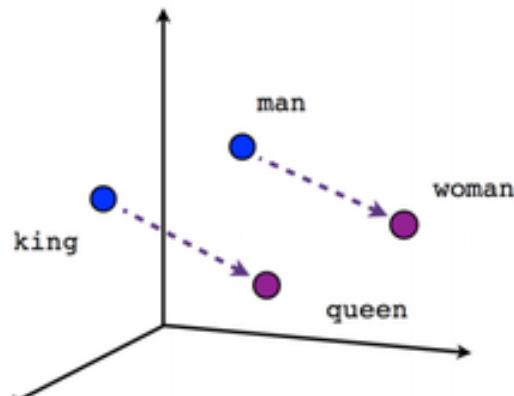


CBOW

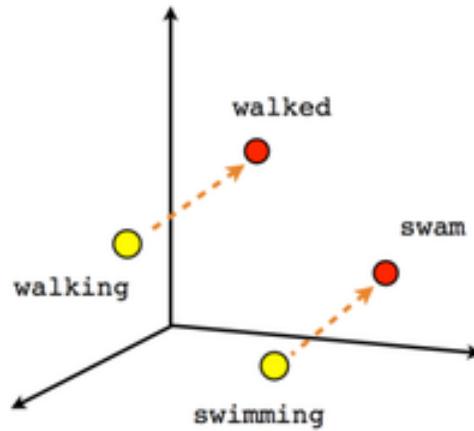


Skip-gram

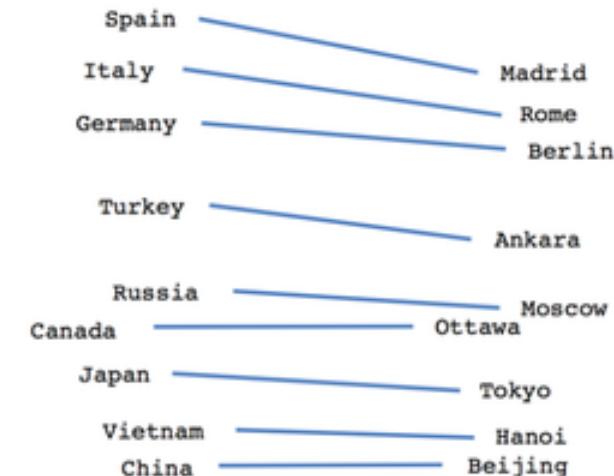
Word Embeddings



Male-Female



Verb tense



Country-Capital

Проблемы word2vec



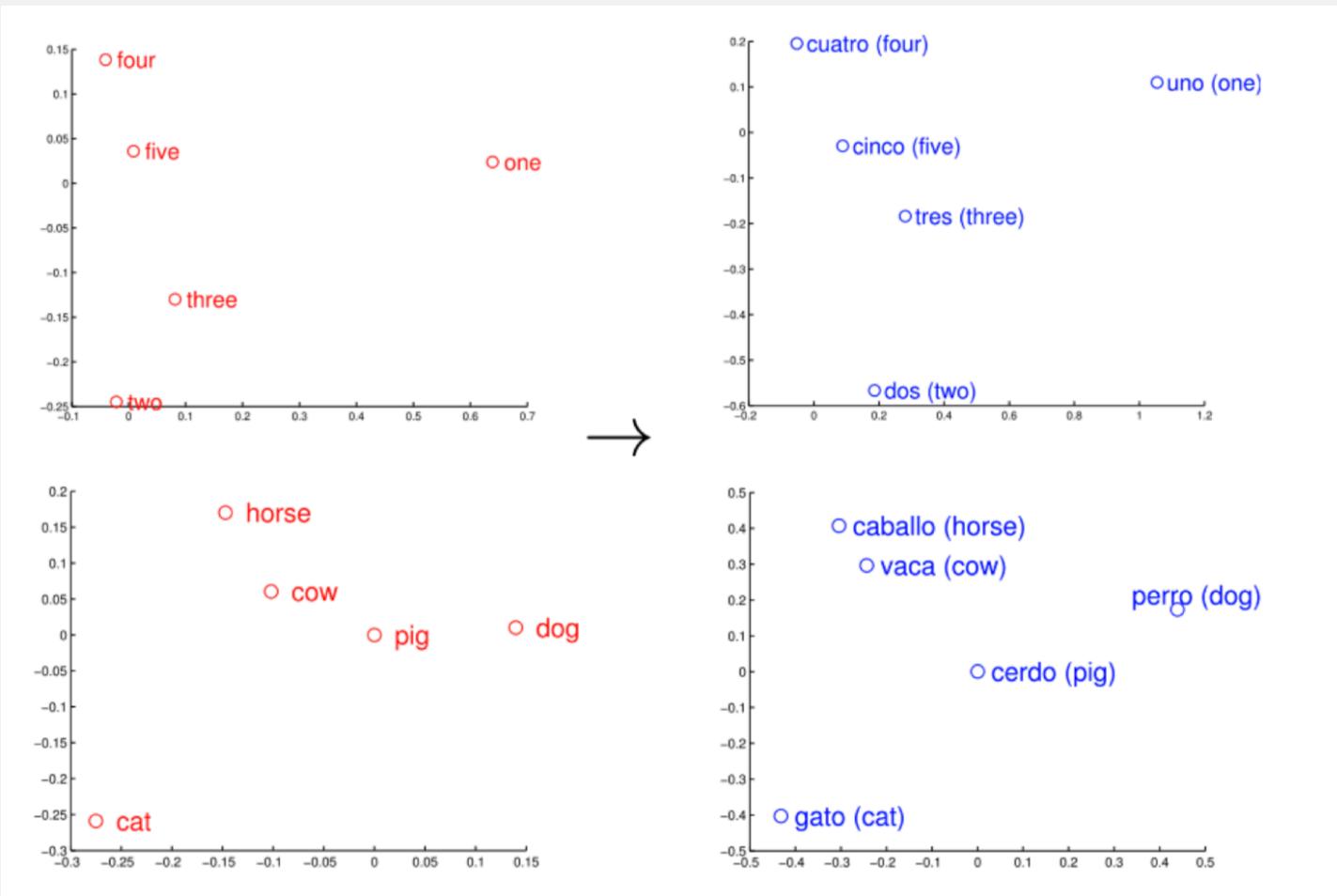
Леброн Джеймс набрал +10 очков в 867-й игре подряд и побил рекорд Майкла Джордана



Huawei представила VR-очки с доступом к hd-контенту

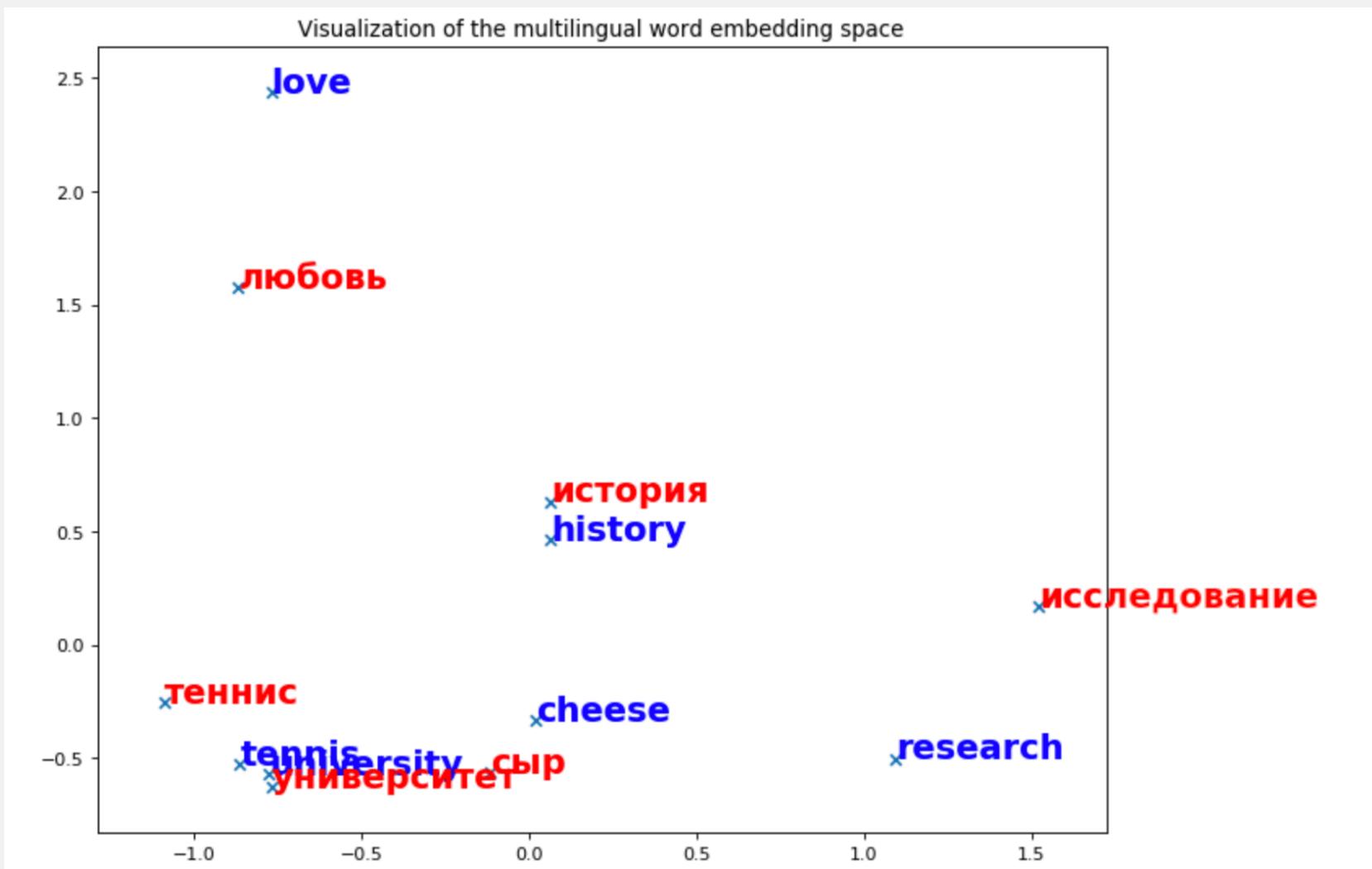
Как называется такая проблема?

Проблемы word2vec

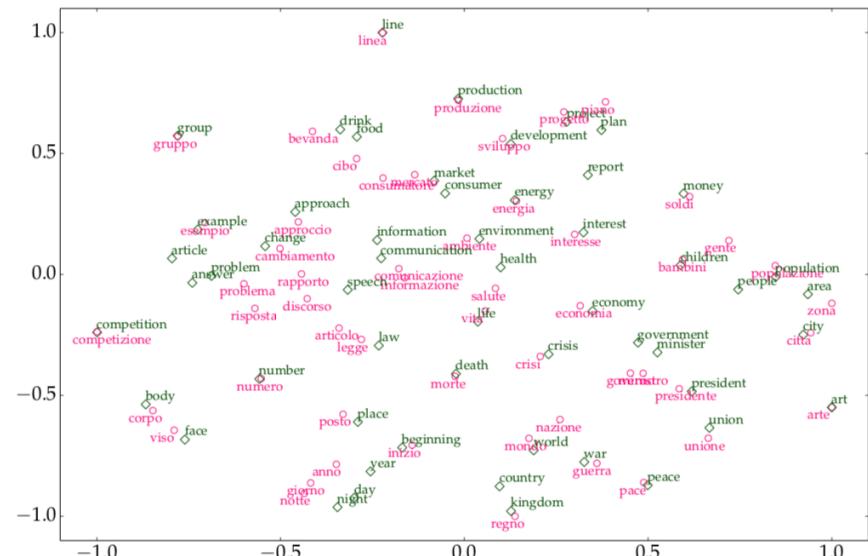
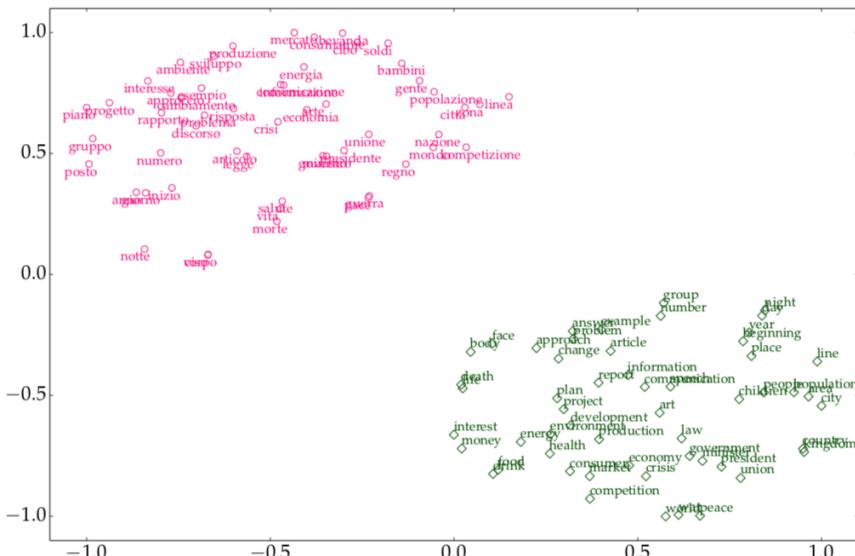


W2v не умеет в многоязычность

Проблемы word2vec



Проблемы word2vec



Что бы нам хотелось получить

Практика



Дано: датасет заголовков англоязычных новостей

Что будем делать: обучим ARTM модель и попробуем поискать ближайших соседей к заголовкам.

Далее сравним с предобученными эмбеддингами BERT (усреднение контексто-зависимых эмбеддингов токенов)

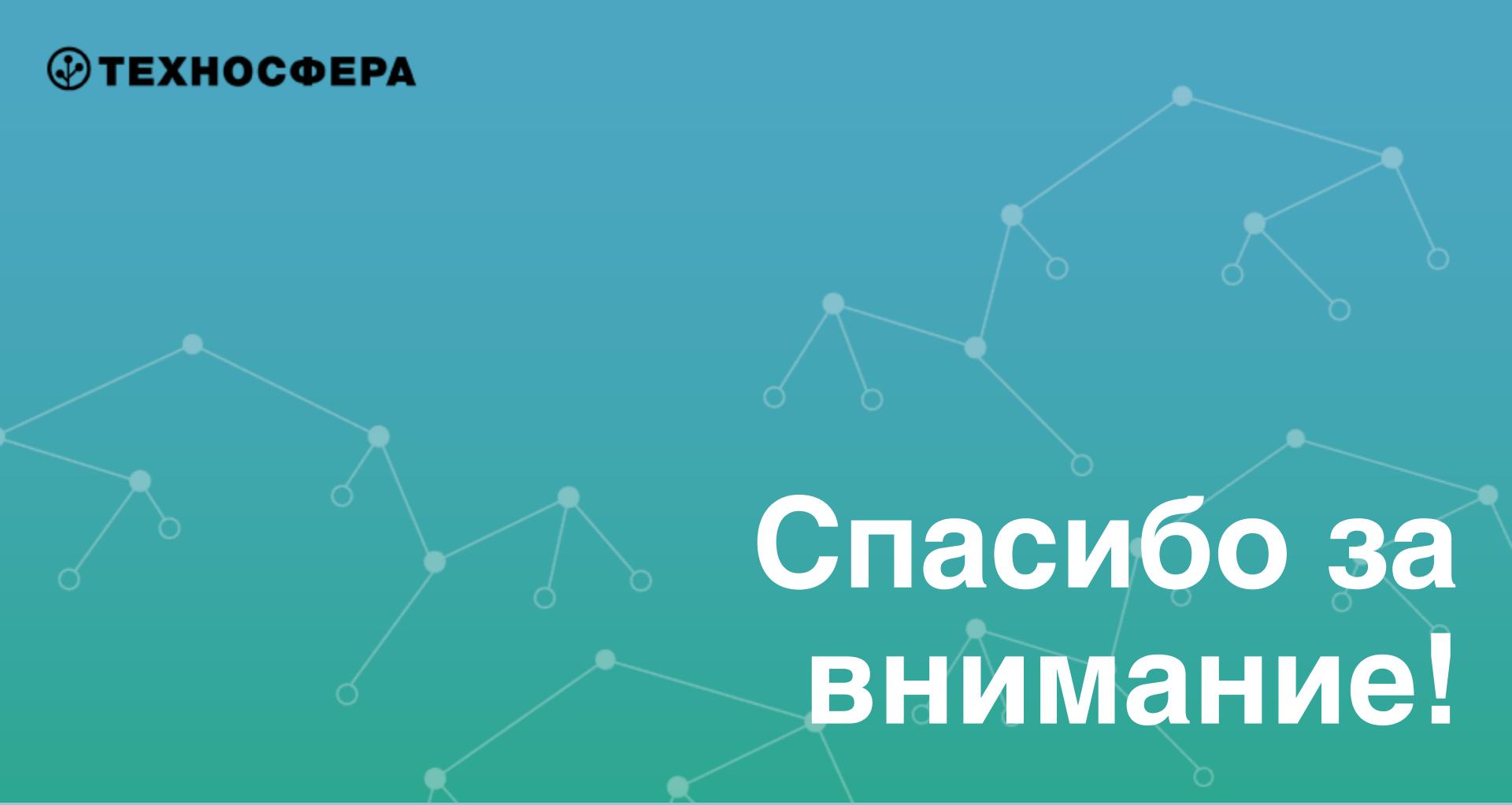
Домашнее задание № 1



- Предлагается скачать датасет Blog Authorship (<http://u.cs.biu.ac.il/~koppel/BlogCorpus.htm>) и сделать exploratory data analysis, обязательно включающий в себя визуализацию эмбеддингов слов в 2-х или 3-х мерном пространстве*
- * необязательно визуализировать все слова
- ** для эмбеддингов слов необязательно использовать word2vec (можно также посмотреть в сторону glove, muse, etc)

Срок сдачи

2019-11-19



Спасибо за
внимание!

Зраев Артем