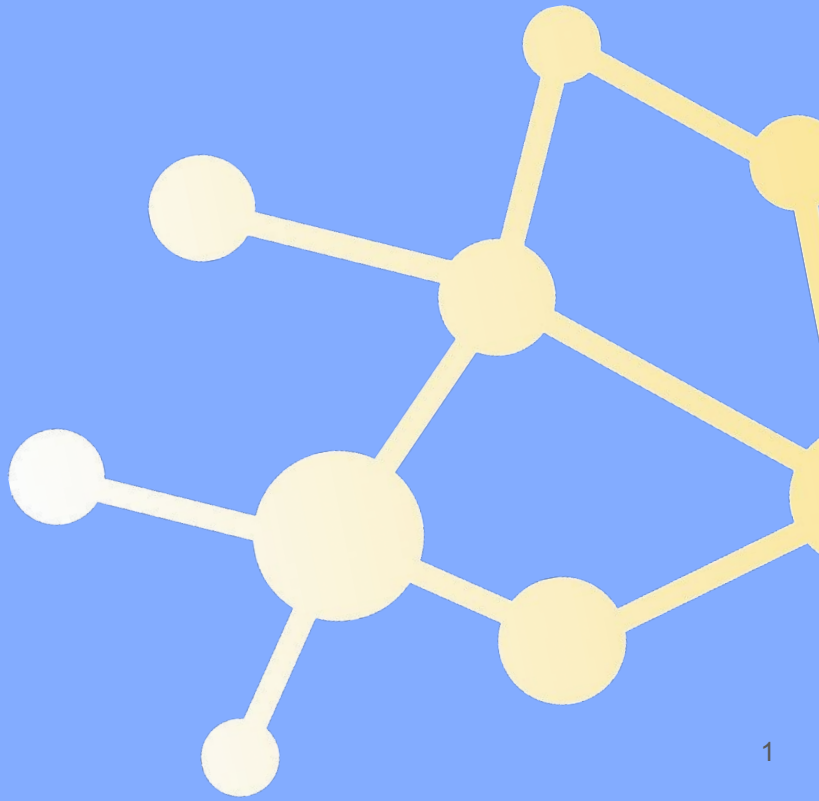
Computational Materials Design Lab

MIPT

Введение в NLP. LLM в химии и материаловедении.

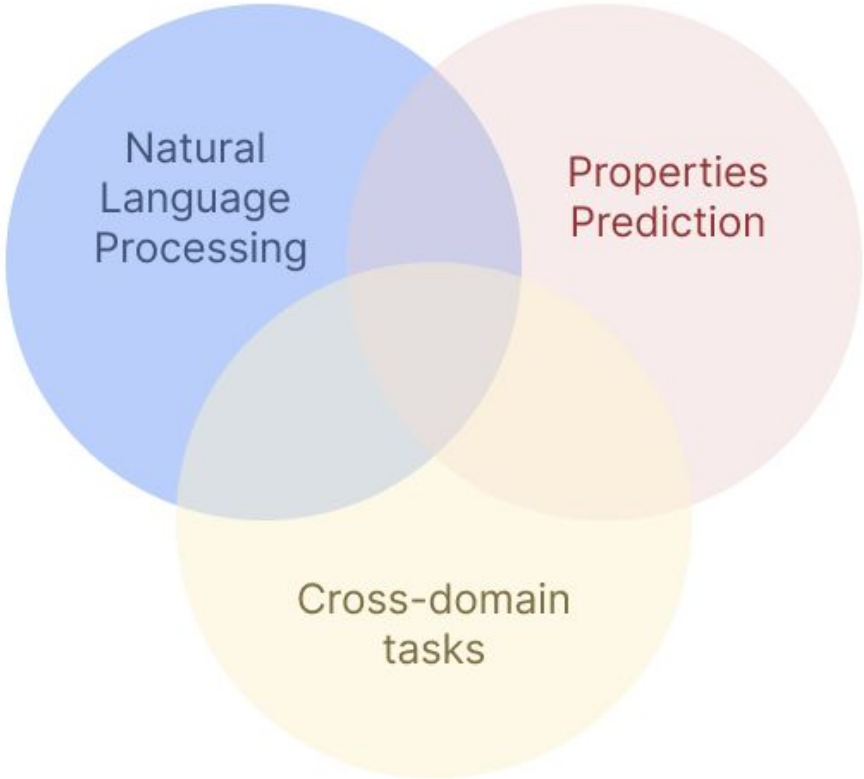
Яна Пропад
научный сотрудник
лаборатории компьютерного дизайна материалов

tg: @yanapropad



1. Какие задачи решают NLP и LLM?
2. Word embeddings
3. word2vec

Введение


Natural
Language
Processing
Properties
Prediction
Cross-domain
tasks

1. Какие задачи решают NLP и LLM?

Задачи, относящиеся к NLP и решаемые LLM

Поиск ключевых слов,
синонимов/антонимов в речи

Автоматический перевод
с одного языка на другой

Классификация документов
(по темам, жанрам и тд)

Распознавание именованных сущностей
(дат, формул, физических величин...)

Распознавание речи

Определение спама

Поиск релевантных документов
по запросу и их ранжирование

Диалоговые системы, чат-боты

Задача суммаризации
(автоматическое составление краткого пересказа)

Определение частей речи

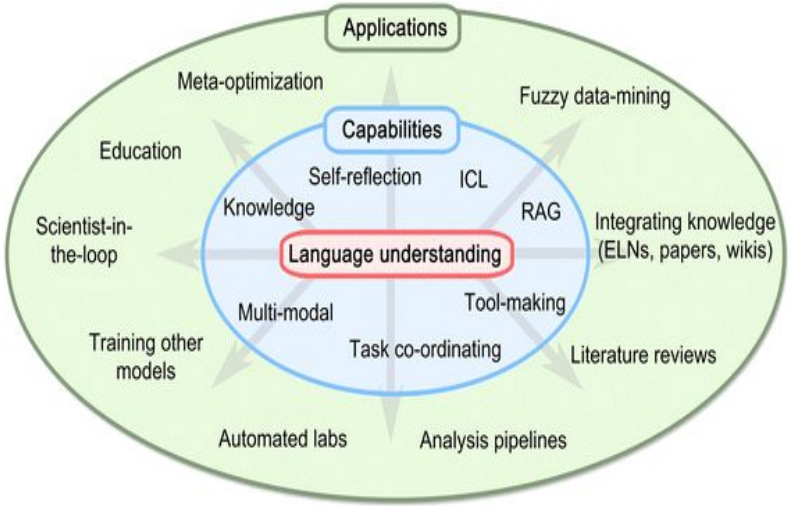
Определение эмоциональной окраски текста

Распознавание и исправление ошибок

Вопросно-ответные системы
(выбор ответа из нескольких предложенных вариантов или вопросы с открытым ответом)

https://cobusgreyling.medium.com/the-large-language-model-landscape-version-5-82785282672f

LLM в материаловедении



Predictive Modeling	**Automatization & Novel Interfaces**	**Knowledge Extraction**
use LLMs for classification and regression tasks	*use natural language descriptions to connect existing tools*	*extract structured information from unstructured text*

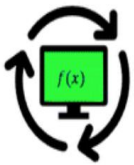
https://pubs.rsc.org/en/content/articlelanding/2023/dd/d3dd00113j

LLM в материаловедении

Training a Prediction Model between vectorized Concrete Formulations (X) and Labels (Y)

X

F1	F2	F3
0.5	1	60
0.3	0	65
	...	
0.4	1	40



Y

T1
22
50
...
36

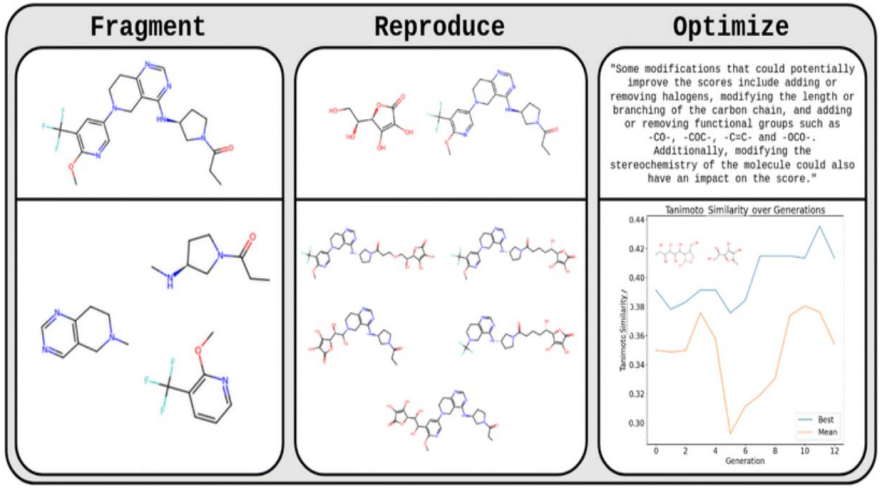
In-Context Learning

User

{ "context": " high water-cement ratio (F1) reduces strength / heat curing (F3) increases FA-based binders (F2) strength / ... ",
"examples":
[{ "input": "F1=0.5 / F2=1 / F3=60", "output": "T1=22 MPa" },
{ "input": "F1=0.3 / F2=0 / F3=65", "output": "T1=50 MPa" },
{ "input": "F1=0.4 / F2=1 / F3=40", "output": "T1=36 MPa" }],
"prompt": "What is the output for F1=0.3 / F2=1 / F3=30?" }

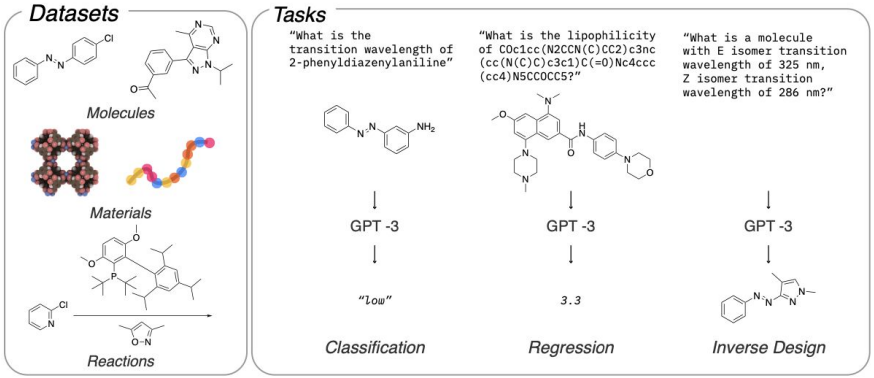
GPT

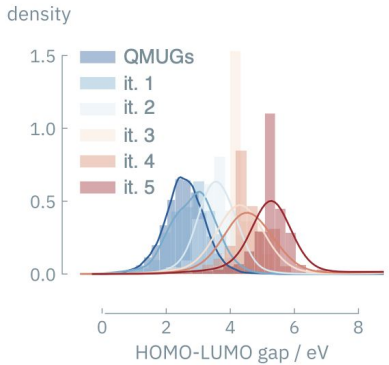
{ "response": "T1=45 MPa" }



https://pubs.rsc.org/en/content/articlelanding/2023/dd/d3dd00113j

LLM в материаловедении



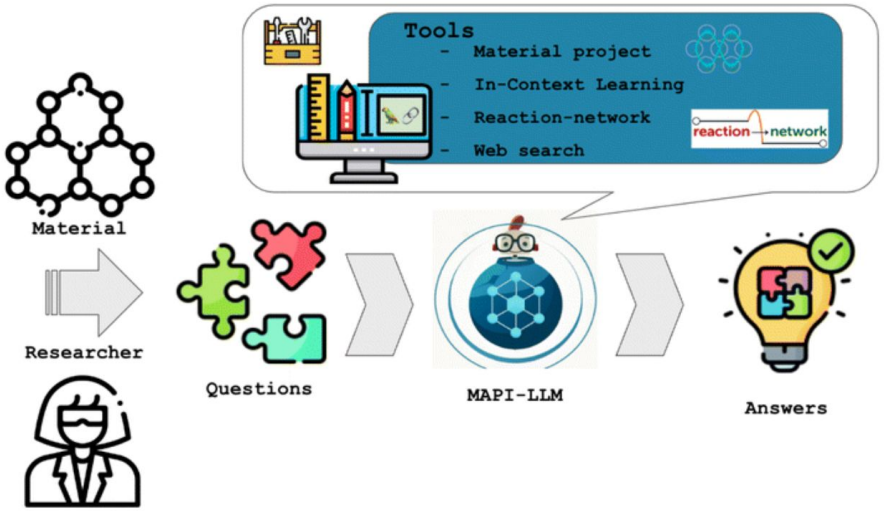


https://pubs.rsc.org/en/content/articlelanding/2023/dd/d3dd00113j

Predictive Modeling	**Automatization & Novel Interfaces**	**Knowledge Extraction**
use LLMs for classification and regression tasks	*use natural language descriptions to connect existing tools*	*extract structured information from unstructured text*

Jablonka et al. have shown that LLMs can be employed to predict various chemical properties, such as solubility or HOMO–LUMO gaps based on line representations of molecules such as self-referencing embedded strings (SELFIES) and SMILES.

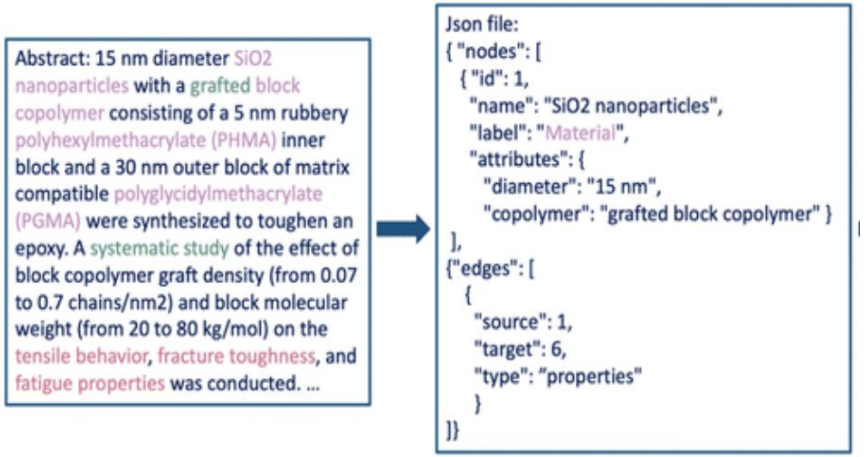
LLM в материаловедении



Predictive Modeling	Automatization & Novel Interfaces	Knowledge Extraction
use LLMs for classification and regression tasks	*use natural language descriptions to connect existing tools*	*extract structured information from unstructured text*
	Yao et al and *Schick et al.* have shown that LLMs can be used as agents that can autonomously make use of external tools such as Web-APIs—a paradigm that some call MRKL (pronounced “miracle”) systems—modular reasoning, knowledge, and language systems.	

https://pubs.rsc.org/en/content/articlelanding/2023/dd/d3dd00113j

LLM в материаловедении

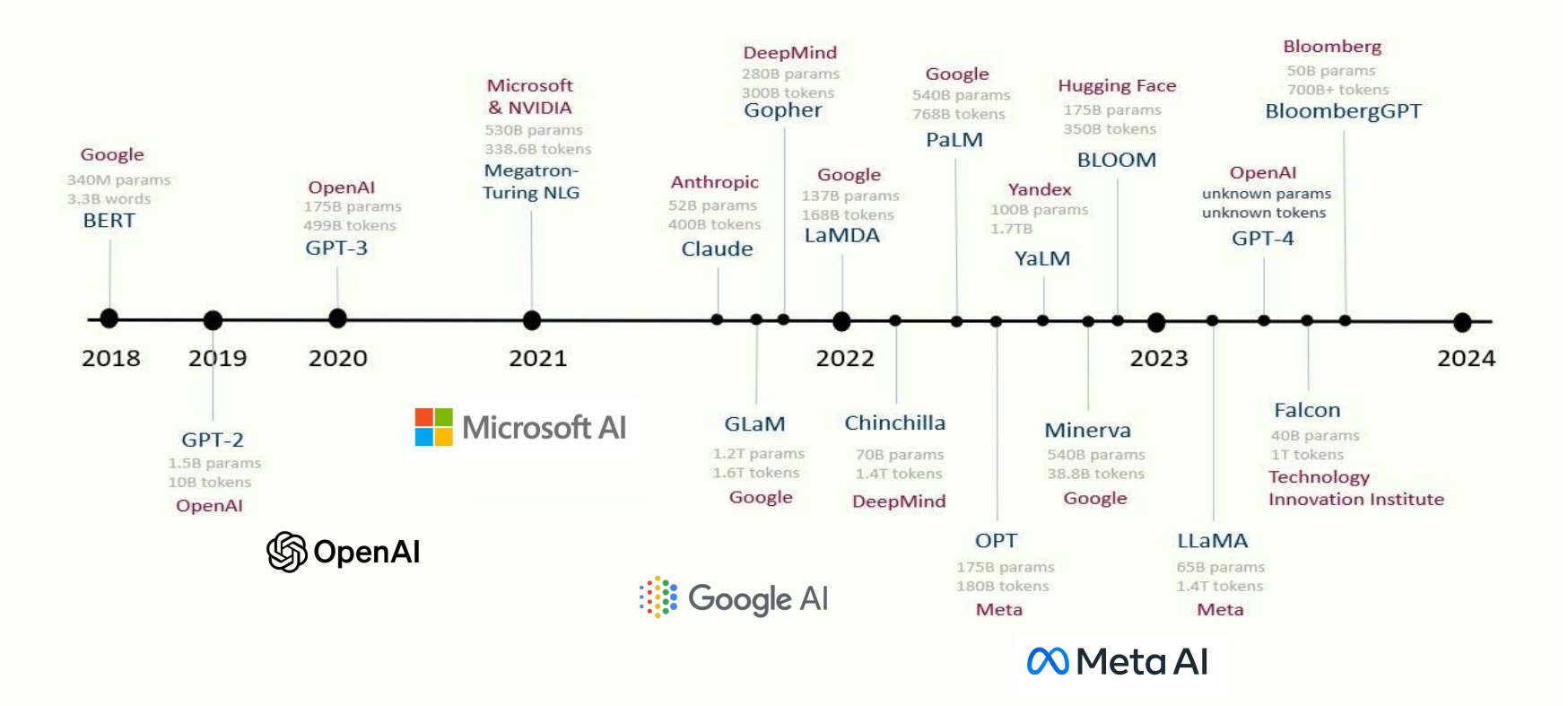


Predictive Modeling	Automatization & Novel Interfaces	Knowledge Extraction
use LLMs for classification and regression tasks	*use natural language descriptions to connect existing tools*	*extract structured information from unstructured text*

To facilitate downstream use of the information, LLMs can also convert unstructured data—the typical form of these literature reports—into structured data. The use of GPT for this application has been reported by Dunn et al. and Walker et al., who used an iterative fine-tuning approach to extract data structured in JSON from papers.

https://pubs.rsc.org/en/content/articlelanding/2023/dd/d3dd00113j

Таймлайн развития LLM



https://towardsdatascience.com/emergent-abilities

Режимы работы с данными

Many-to-one

На вход подается последовательность объектов, на выходе один объект

Ex.: классификация текстов и видео
Ex.: тематическая классификация

One-to-many

На вход подается один объект, на выходе последовательность объектов

Ex.: генерация описания или заголовка к тексту или изображению

Many-to-many

На входе и выходе последовательности нефиксированной длины

Ex.: машинный перевод
Ex.: суммаризация текста
Ex.: генерация заголовка к статье

Many-to-many, синхронизированный вариант

На входе и выходе последовательности одинаковой длины, токены одной явно сопоставлены токенам выходной

Ex.: генерация покадровых субтитров к видео

Режимы работы с данными

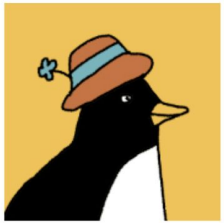
Many-to-one

Было сложно, мне понравилось

↓

Позитивный

One-to-many



⟶ Пингвин в шляпе

Many-to-many

I seem to have overfitted

↓

Кажется, я переобучился

Many-to-many, синхронизированный вариант

Будет	сложно,	вам	понравится
↓	↓	↓	↓
глагол	наречие	местоимение	глагол

2. Word embeddings

Word Embeddings

К векторизации текста есть два базовых подхода:

Векторизовать текст целиком, превращая его в один вектор

↓

Bag-of-Words

Векторизовать отдельные структурные единицы, превращая текст в последовательность векторов

↓

CBOW
(Continuous Bag-of-Words)
Skip-gram

Токенизация

Токен – атомарный элемент последовательности

Я хочу применять методы машинного обучения в кристаллографии.

Я	хочу	применять	методы	машинного	обучения	в	кристаллографии

Токеном может быть слово, морфема, символ – это вопрос договорённости в каждой задаче

Чем предобрабатывать тексты?

NLTK
nltk.stem.SnowballStemmer
nltk.stem.PorterStemmer
nltk.stem.WordNetLemmatizer
nltk.corpus.stopwords

Bag-of-Words

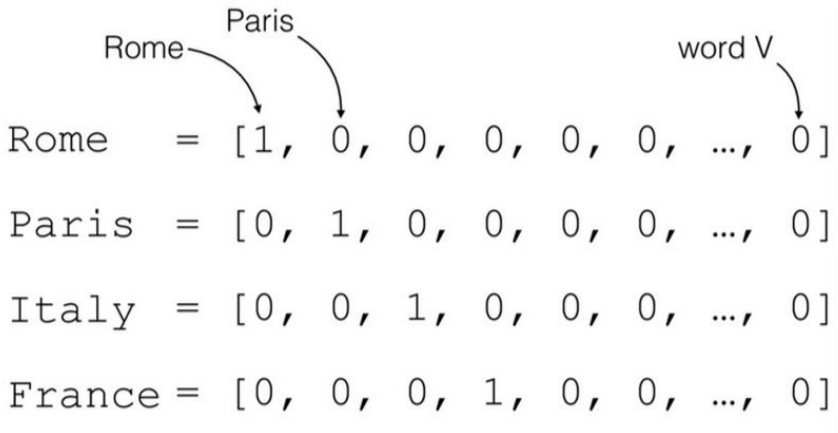
Я хочу применять методы машинного обучения в кристаллографии.

Я	хочу	применять	методы	машинного	обучения	в	кристаллографии	химии
1	1	1	1	1	1	1	1	0

Минусы:

– Теряем информацию о порядке слов
– Векторы высокой размерности
– Векторы крайне разреженные
– Разные формы слов воспринимаются как разные слова

Word Embeddings: one-hot вектор



Минусы:

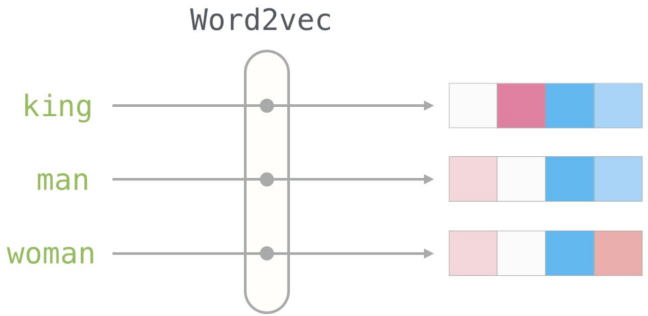
– Векторы высокой размерности
– Векторы крайне разреженные
– Все векторы взаимноортагональны

3. word2vec

Word Embeddings: word2vec

word2vec – метод построения информативных векторных представлений слов, представлен в работе 2013 года

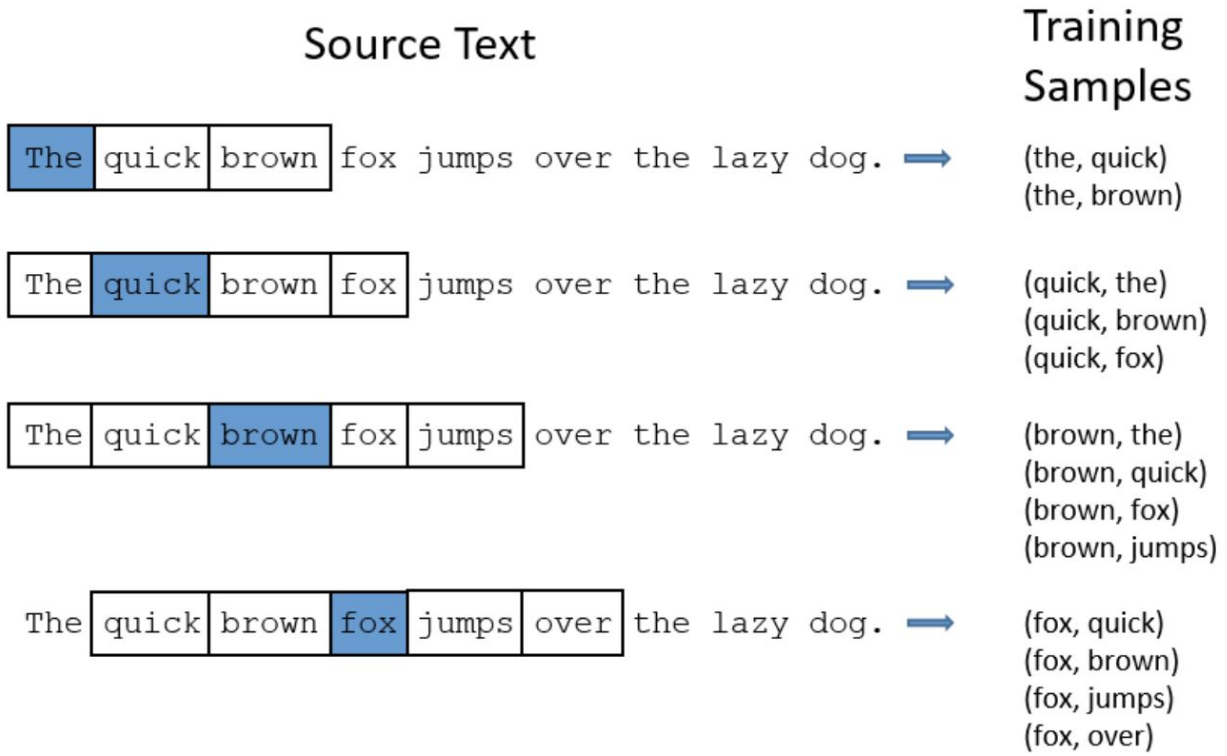
– У нас есть большой корпус («тело») текста: длинный список слов
– Каждое слово в фиксированном словаре представлено вектором
– Проходимся по каждой позиции t в тексте, которая имеет центральное слово A и контекстные («внешние») слова B
– Используем сходство векторов слов для A и B, чтобы вычислить вероятность A при заданном B (или наоборот)
– Продолжаем корректировать векторы слов, чтобы максимизировать эту вероятность



https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec

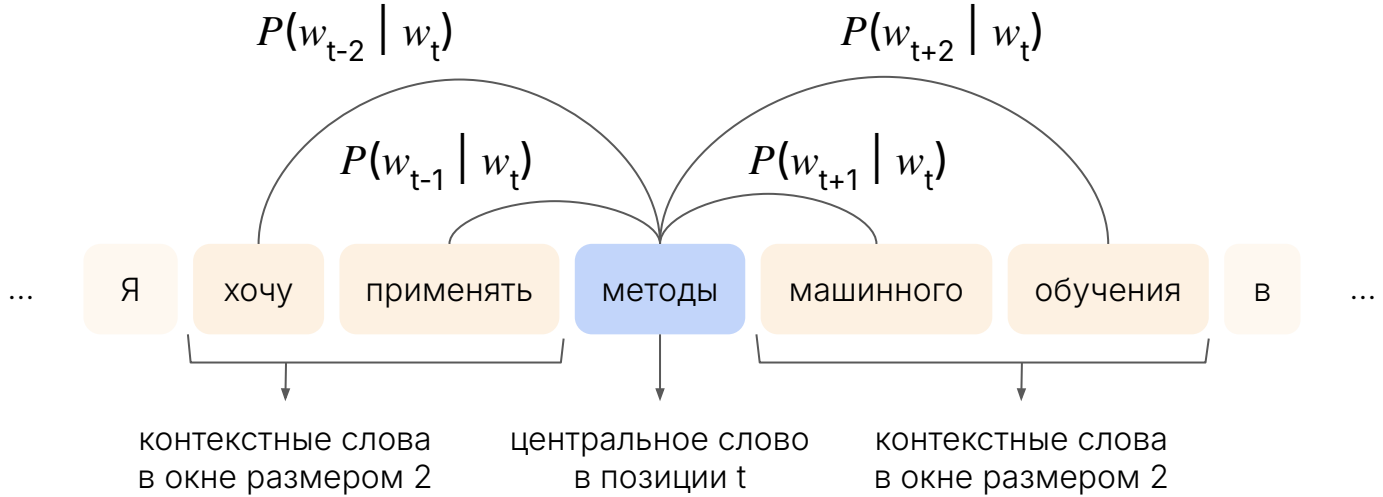
Скипграмма –



https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec

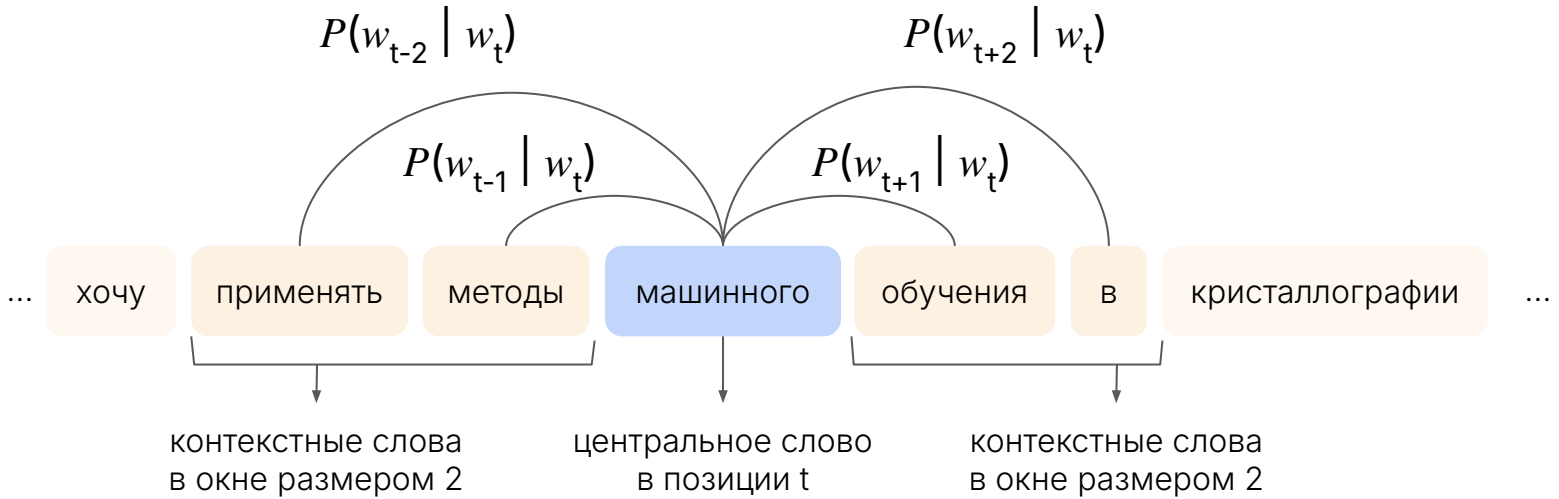
Пример окна и вычисления $P(w_{t+j} \mid w_t)$



https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec

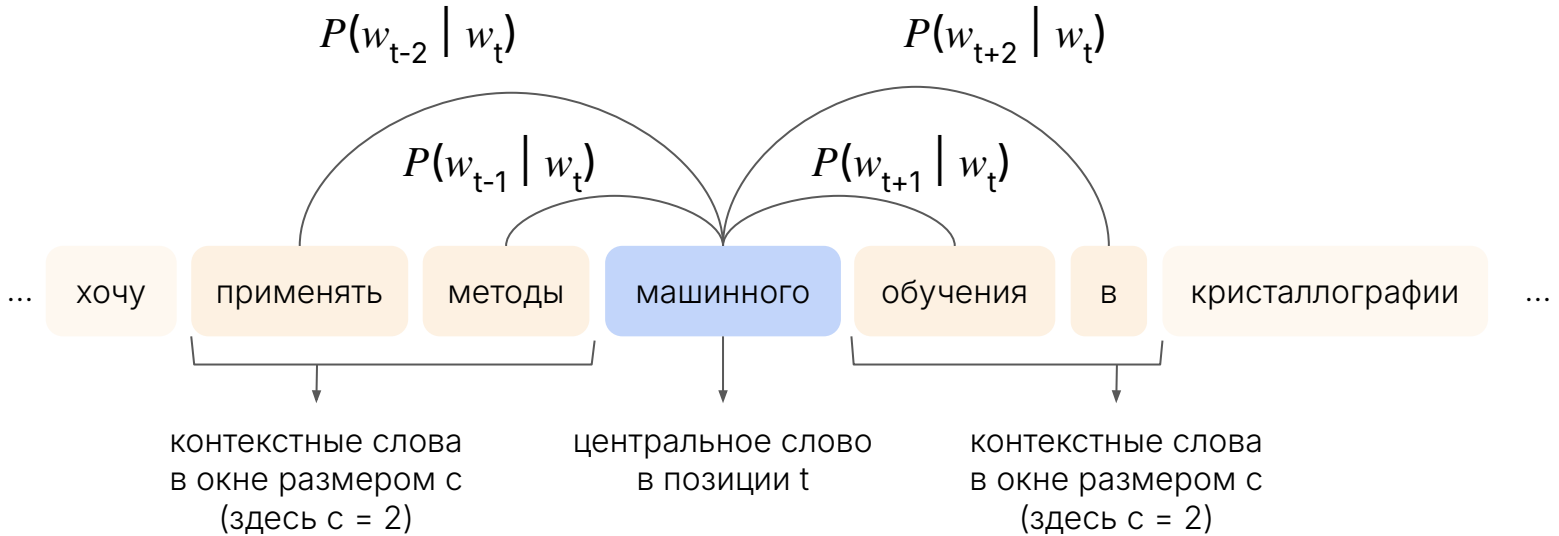
Пример окна и вычисления $P(w_{t+j} \mid w_t)$



https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec

Пример окна и вычисления $P(w_{t+j} \mid w_t)$



Максимизируемый функционал:
(логарифм правдоподобия)

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0}\log p(w_{t+j}|w_t)$$

https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec

Максимизируемый функционал:
(логарифм правдоподобия)

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0}\log p(w_{t+j}|w_t)$$

– на каждой позиции t = 1, ... ,T предсказываем контекст в окне шириной c

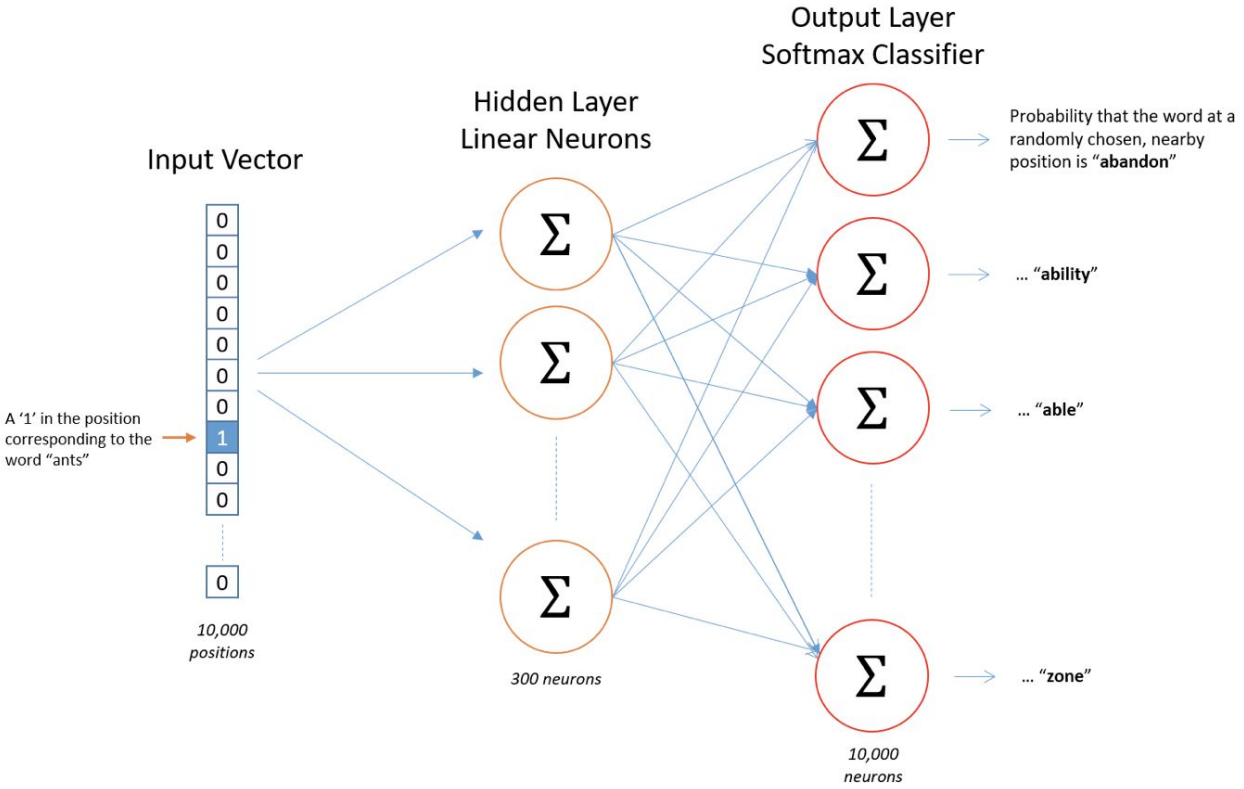
Оцениваем вероятности
через Softmax

$$p(w_j|w_I)=\frac{\exp\left({\mathbf{v}'_{w_j}}^T\mathbf{v}_{w_I}\right)}{\sum_{j'=1}^{V}\exp\left({\mathbf{v}'_{w_{j'}}}^T\mathbf{v}_{w_I}\right)}$$

Note that $\mathbf{v}_w$ and $\mathbf{v}'_w$ are two representations of the word w. $\mathbf{v}_w$ comes from rows of $\mathbf{W}$, which is the input→hidden weight matrix, and $\mathbf{v}'_w$ comes from columns of $\mathbf{W}'$, which is the hidden→output matrix. In subsequent analysis, we call $\mathbf{v}_w$ as the "**input vector**", and $\mathbf{v}'_w$ as the "**output vector**" of the word w.

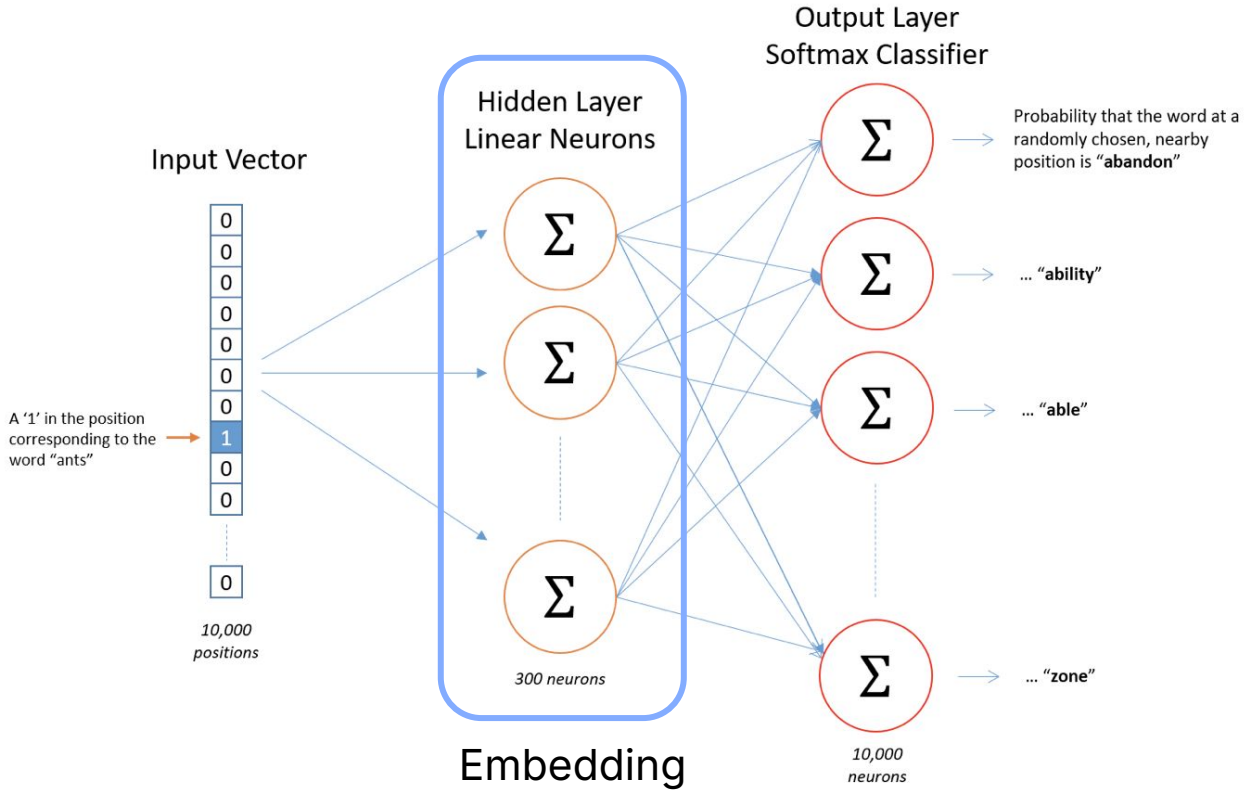
https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec



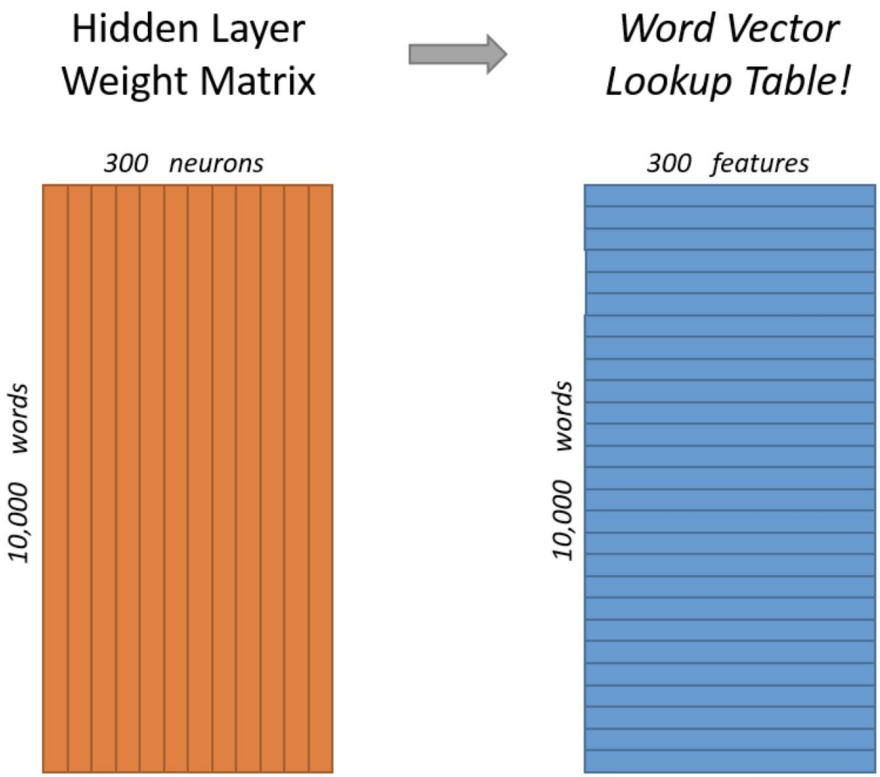
https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec



https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec



https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec – Subsampling

Векторные представления частых слов существенно не меняются после обучения на нескольких миллионах примеров.

Чтобы устранить дисбаланс между редкими и частыми словами, используется простой подход подвыборки: каждое слово w_i в обучающем наборе отбрасывается с вероятностью, вычисленной по формуле:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

где f(w_i) — частота слова w_i, а t — выбранный порог, обычно около 10^{-5}.

https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec – Negative sampling

Имеет смысл не только “сближать” похожие (близкие по контексту) слова, но и “отдалять” непохожие.
Для этого воспользуемся механизмом negative sampling.

Чем чаще встречается слово в обучающем корпусе, тем больше вероятность использовать его в качестве negative sample.

$$P(w_i) = \frac{f(w_i)}{\sum_{j=0}^{n}\left(f(w_j)\right)} \Longrightarrow P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n}\left(f(w_j)^{3/4}\right)}$$

Задача состоит в том, чтобы отличить целевое слово от слов из распределения шума с помощью логистической регрессии, где есть k отрицательных выборок для каждой выборки данных.
Эксперименты показывают, что значения k в диапазоне 5–20 полезны для небольших обучающих наборов данных, тогда как для больших наборов данных k может быть всего 2–5.

https://jalammar.github.io/illustrated-word2vec/

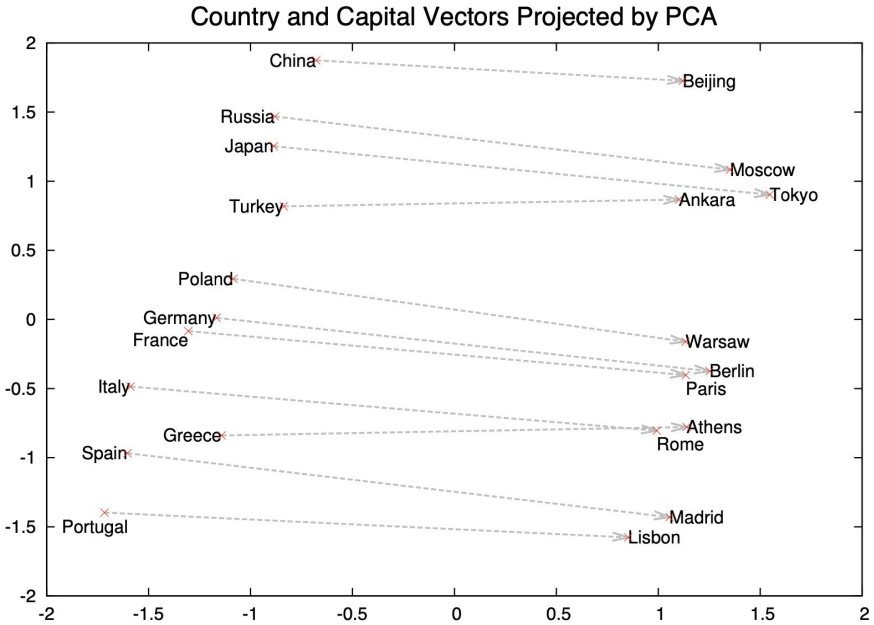
Word Embeddings: word2vec – Negative sampling

Обновленный оптимизируемый функционал:
рассматриваем лишь положительный пример и несколько отрицательных:

$$\log \sigma({v'_{w_O}}^\top v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^\top v_{w_I}) \right]$$

https://jalammar.github.io/illustrated-word2vec/

Word Embeddings: word2vec



Двумерная проекция PCA 1000-мерных векторов Skip-gram стран и их столиц.
Рисунок иллюстрирует способность модели автоматически организовывать концепции и неявно изучать связи между ними, поскольку во время обучения мы не предоставляли никакой контролируемой информации о том, что означает столица.

https://jalammar.github.io/illustrated-word2vec/

Практика:

написать Skipgram-модель с Negative sampling

Идея:
Схожие структуры будут иметь схожие векторы в латентном пространстве.
Изучить визуализацию латентного пространства, выделяет ли модель классы структур.

Корпус:
текстовые описания экспериментальных кристаллических структур из базы данных Materials Project. Описания сгенерированы с помощью Robocrystallographer и хранятся в виде списка строк в текстовом формате.

Задание: выполнить все задания из ноутбука

Мы получили Word Embeddings для кристаллов. Что теперь с ними делать?...

– классифицировать кристаллические структуры

– предсказывать физико-химические свойства

Pipeline for chem-phys properties prediction using LLM's embeddings