Natural Language Processing: Word Embeddings

HSE Faculty of Computer Science

Machine Learning and Data-Intensive Systems

Murat Khazhgeriev

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Table of Content

- **Organizational matters**
  - Homework & grade policy
  - Resources
- Preprocessing pipeline
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Table of Content

- Organizational matters
  - **Homework & grade policy**
  - Resources
- Preprocessing pipeline
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Grade policy

60% (homework) + 10% (interim testing) + 30% (exam)

Homework

- (20%) Week 2. Training embeddings using the fasttext library, implementation of a real search engine for embedding-response upon request in a vector database.
- (20%) Week 4. Fine-tuning BERT on your own data, training GPT from scratch
- (20%) Week 5: Fine tuning LLM using PEFT.

Table of Content

- Organizational matters
  - Homework & grade policy
  - **Resources**
- Preprocessing pipeline
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Course materials

- Syllabus (Notion)
- Github
- HSE Wiki

Useful sources

- NLP Course For You
- YSDA NLP Course
- CS224n

Table of Content

- Organizational matters
- **Preprocessing pipeline**
  - Tokenization
  - Lowering, Punctuation, Stop Words, Filtration
  - Normalization
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Table of Content

- Organizational matters
- Preprocessing pipeline
  - **Tokenization**
  - Lowering, Punctuation, Stop Words, Filtration
  - Normalization
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Word-Level Tokenization

“ChatGPT is a powerful AI tool.” ⟶ ["ChatGPT", "is", "a", "powerful", "AI", "tool", "."]

Character-Level Tokenization-Level Tokenization

“ChatGPT is a powerful AI tool.” ⟶ ["C", "h", "a", "t", "G", "P", "T", " ", "i", "s", " ", "a", " ", "p", "o", "w", "e", "r", "f", "u", "l", " ", "A", "I", " ", "t", "o", "o", "l", "."]

Byte-Pair Encoding (BPE) Tokenization

“ChatGPT is a powerful AI tool.” ⟶ ["Chat", "GP", "T", "is", "a", "power", "ful", "AI", "tool", "."]

abacaba

dict: a
b
c
ab
aba

ab - 2
ba - 2
ac - 1
...

aba - 2
...

Table of Content

- Organizational matters
- Preprocessing pipeline
  - Tokenization
  - **Lowering, Punctuation, Stop Words, Filtration**
  - Normalization
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

"The quick brown fox jumps over the lazy dog!"

↓

✓ Lowering: "the quick brown fox jumps over the lazy dog!"

↓

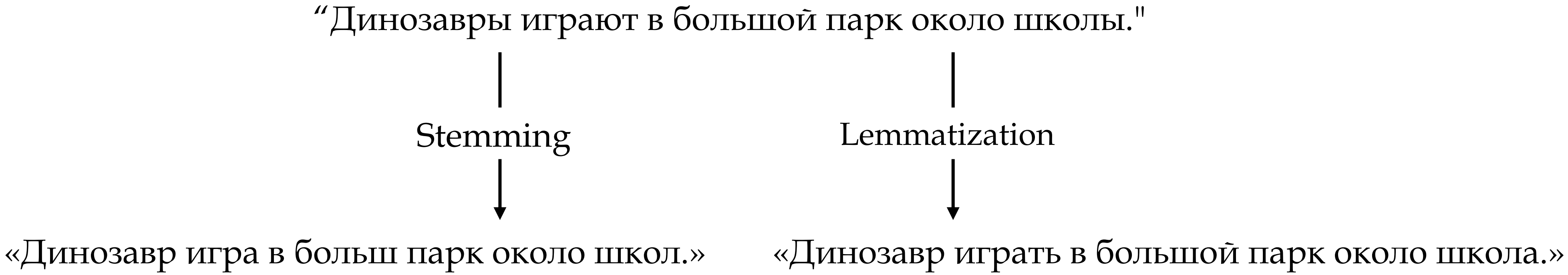
✓ Punctuation removal: "the quick brown fox jumps over the lazy dog"

↓

✓ Stop Words Removal: "quick brown fox jumps lazy dog"

Table of Content

- Organizational matters
- Preprocessing pipeline
  - Tokenization
  - Lowering, Punctuation, Stop Words, Filtration
  - **Normalization**
- But what is a Word Embedding?
- Statistics-based approaches
- Deep Learning approaches
- Useful facts



“Динозавры играют в большой парк около школы."

Stemming

Lemmatization

«Динозавр игра в больш парк около школ.»

«Динозавр играть в большой парк около школа.»

(English)

(Русский)

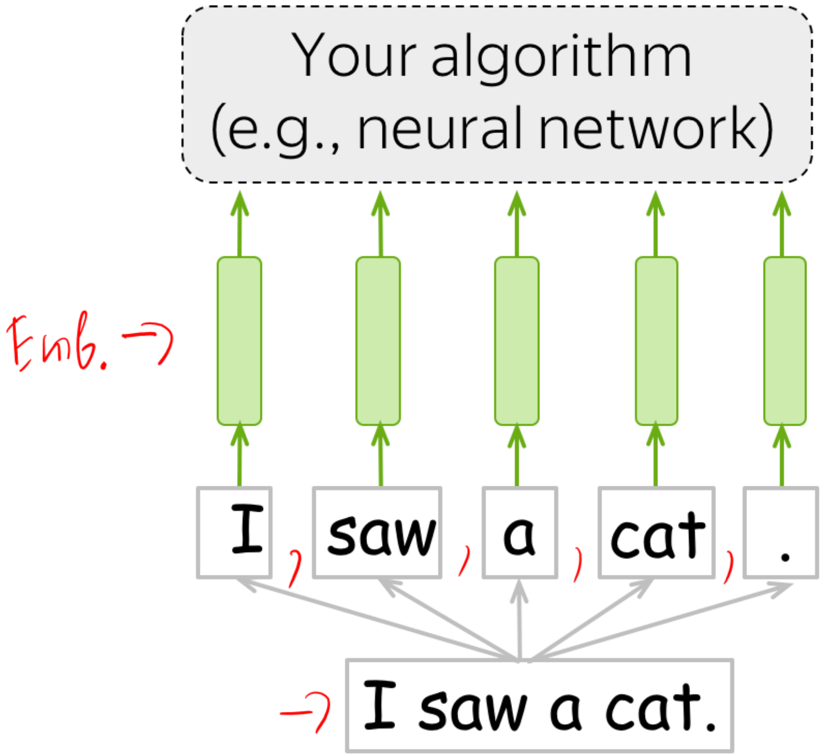
Table of Content

- Organizational matters
- Preprocessing pipeline
- **But what is a Word Embedding?**
    - Intuition behind
    - One-Hot Vectors
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
  - **Intuition behind**
  - One-Hot Vectors
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

Tokenize an input text for further processing



Any algorithm for solving a task

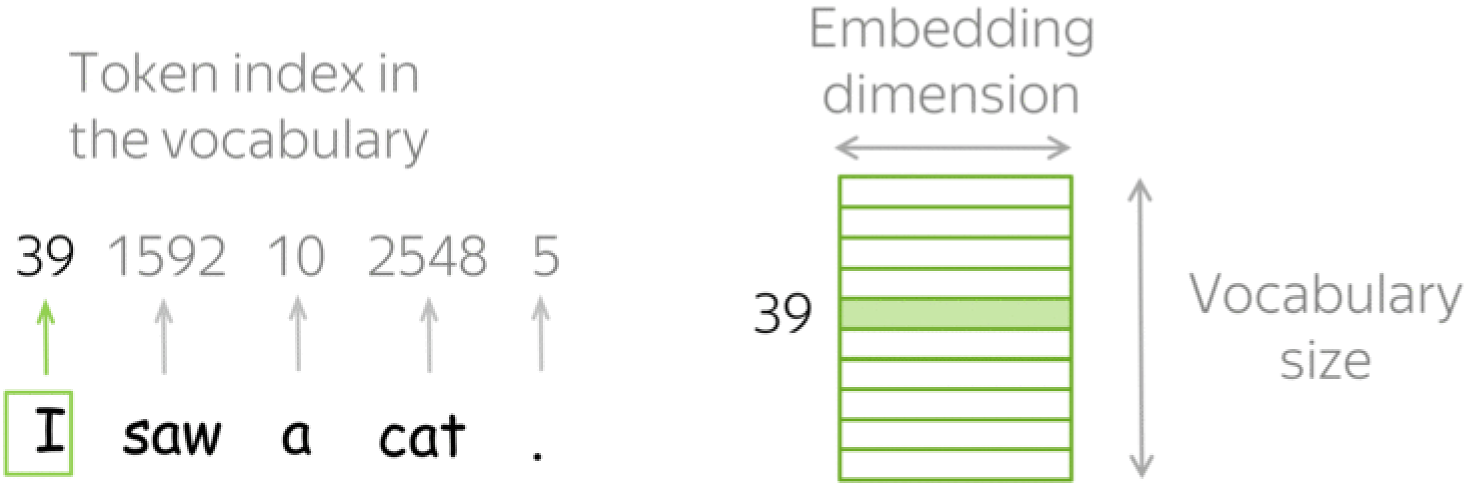
Word representation - vector
(input for your model/algorithm)

Sequence of tokens

Text (your input)

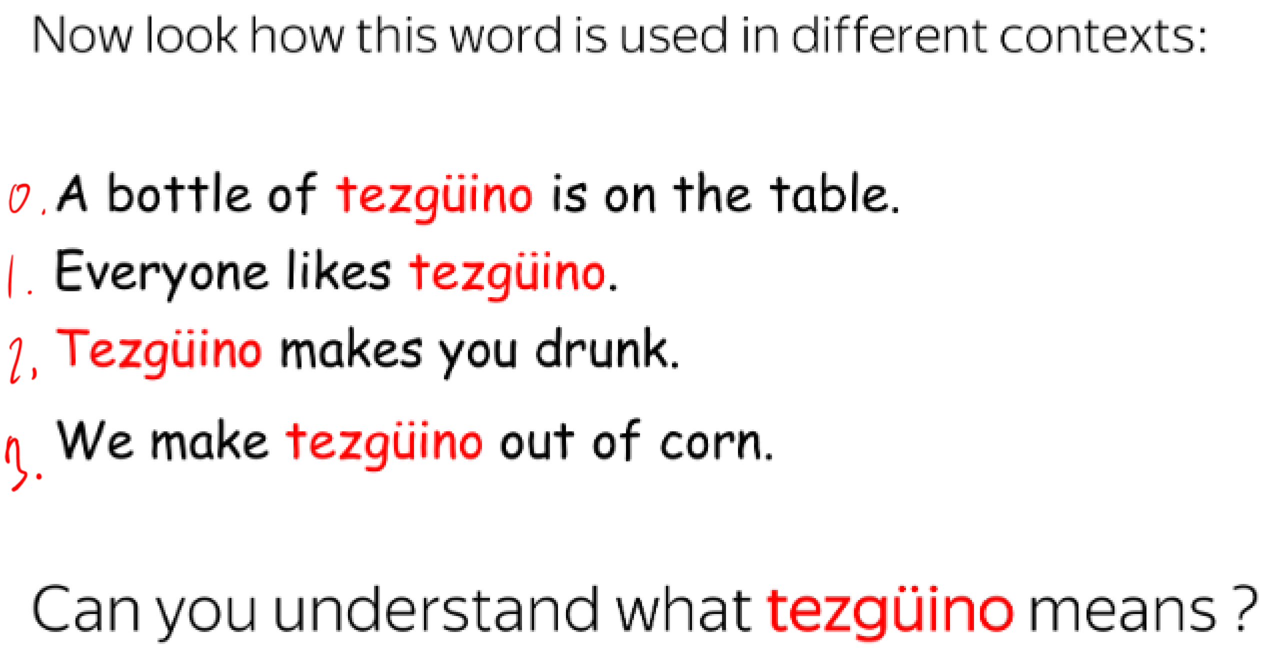
Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

Match each token to a vector



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

A word’s meaning is defined by its context



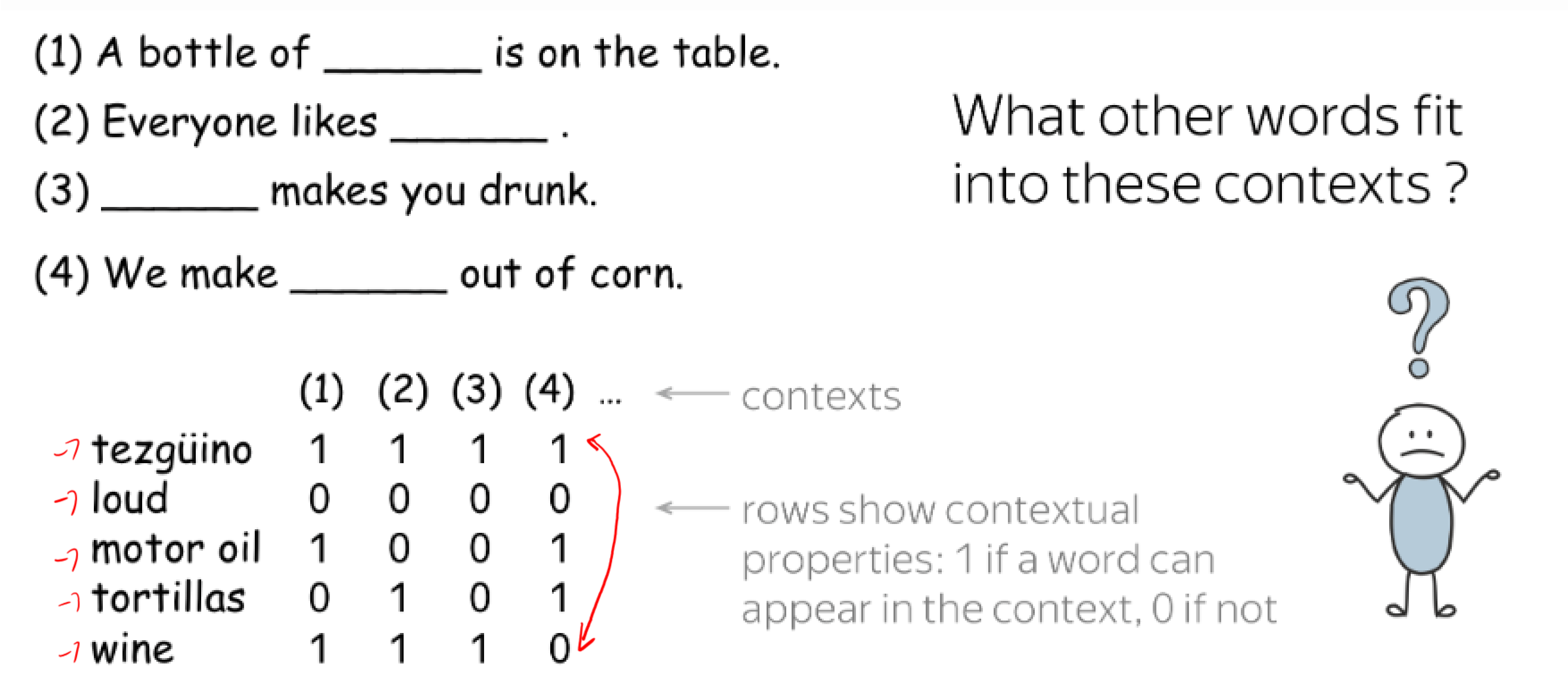
Now look how this word is used in different contexts:

0. A bottle of tezgüino is on the table.
1. Everyone likes tezgüino.
2, Tezgüino makes you drunk.
3. We make tezgüino out of corn.

Can you understand what tezgüino means ?

Source: NLP Course for You, Jacob Eisenstein's NLP notes

A word's meaning is defined by its context



(1) A bottle of _______ is on the table.

(2) Everyone likes _______ .

(3) _______ makes you drunk.

(4) We make _______ out of corn.

What other words fit into these contexts ?

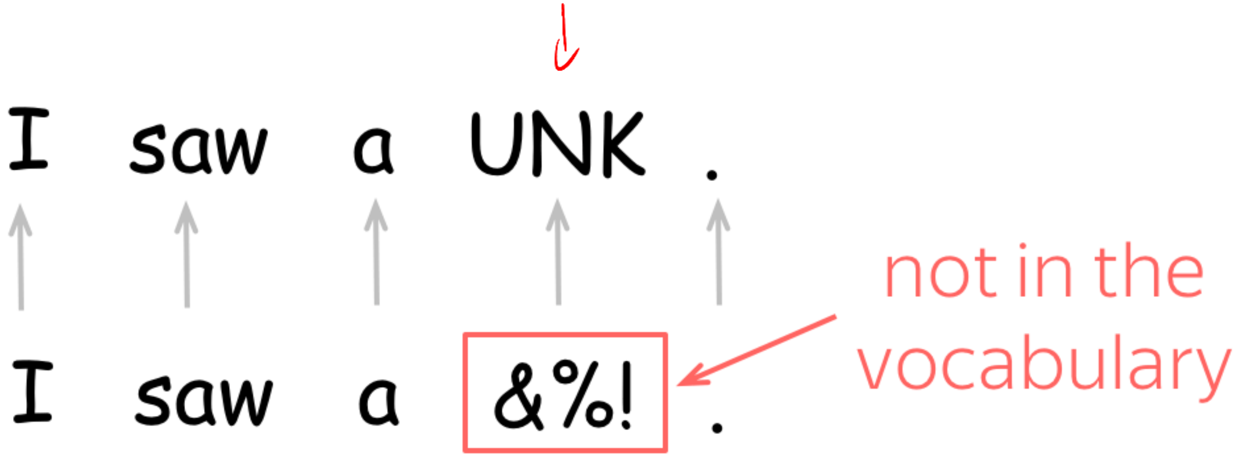
	(1)	(2)	(3)	(4)	...
tezgüino	1	1	1	1	
loud	0	0	0	0	
motor oil	1	0	0	1	
tortillas	0	1	0	1	
wine	1	1	1	0	

← contexts

← rows show contextual properties: 1 if a word can appear in the context, 0 if not

Source: NLP Course for You, Jacob Eisenstein's NLP notes

Reserve a token for special cases e.g. unknown words

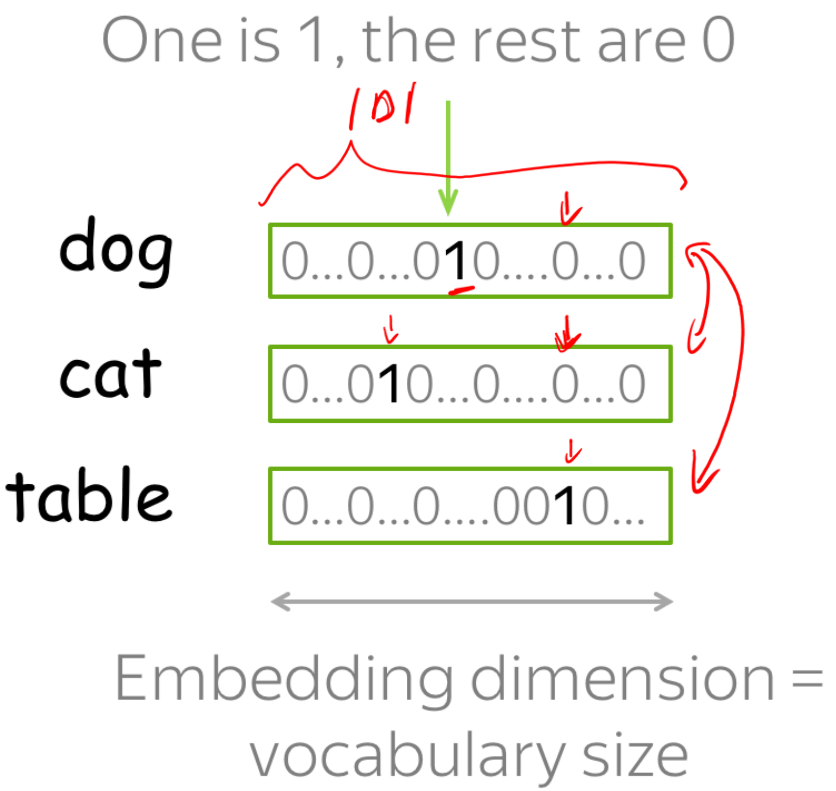


Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
  - Intuition behind
  - **One-Hot Vectors**
- Statistics-based approaches
- Deep Learning approaches
- Useful facts

The easiest way to go is One-Hot Encoding



One is 1, the rest are 0

dog 0...0...010....0...0

cat 0...010...0....0...0

table 0...0...0....0010...

Embedding dimension = vocabulary size

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

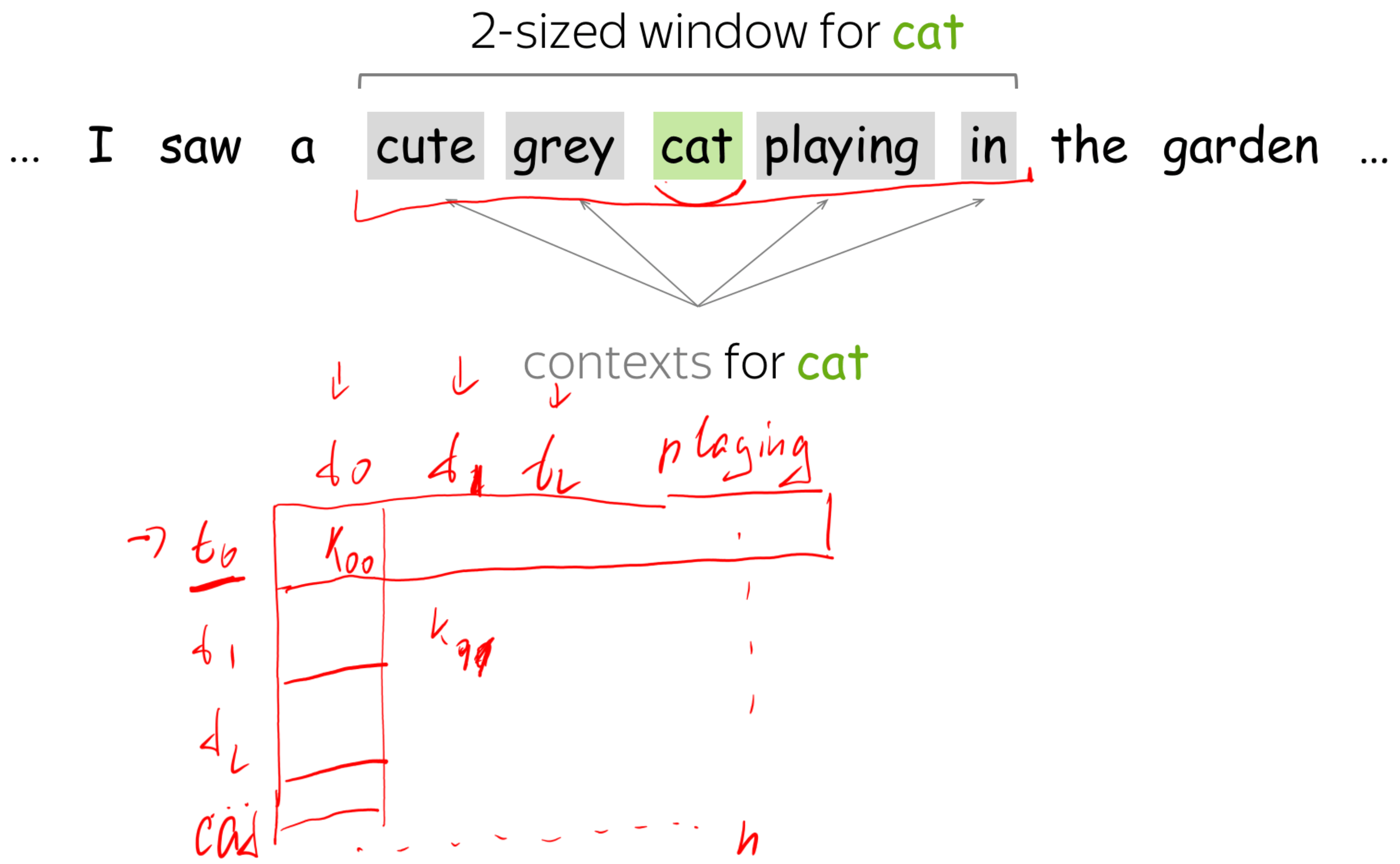
Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- **Count-based (pre-neural) approaches**
    - Co-occurrence count
    - Bag-of-Words (BOW)
    - PPMI
    - TF-IDF
    - Latent Semantic Analysis
- Word2vec
- Useful facts

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
    - **Co-occurrence count**
    - Bag-of-Words (BOW)
    - PPMI
    - TF-IDF
    - Latent Semantic Analysis
- Word2vec
- Useful facts

Define context via a window in a text



2-sized window for cat

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

contexts for cat

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

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
    - Co-occurrence count
    - **Bag-of-Words (BOW)**
    - PPMI
    - TF-IDF
    - BM25
    - Latent Semantic Analysis
- Word2vec
- Useful facts

We can also treat the whole document as a context

D1: a cat sat on a mat

D2: a mat for a dog

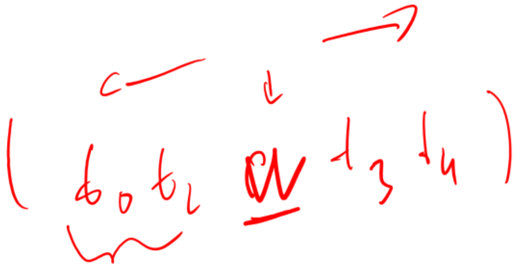
dict

	D1	D2
a	2	1
cat	1	0
sat	1	0
on	1	0
mat	1	1
for	0	1
dog	0	1

Table of Content

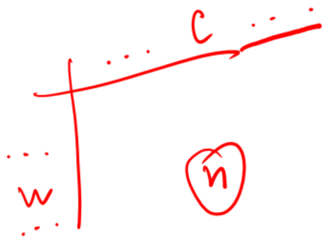
- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
  - Co-occurrence count
  - Bag-of-Words (BOW)
  - **PPMI**
  - TF-IDF
  - BM25
  - Latent Semantic Analysis
- Word2vec
- Useful facts

Positive Pointwise Mutual Information



Context:

- surrounding words in a L-sized window



Matrix element:

- $\mathrm{PPMI}(w, c) = \max(0, \mathrm{PMI}(w, c)),$

where

$$\to \mathrm{PMI}(w, c) = \log \frac{P(w, c)}{P(w)P(c)} = \log \frac{N(w, c)|(w, c)|}{N(w)N(c)}$$

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

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
  - Co-occurrence count
  - Bag-of-Words (BOW)
  - PPMI
  - **TF-IDF**
  - BM25
  - Latent Semantic Analysis
- Word2vec
- Useful facts

We can also account for a term being widespread

Context:

- document d (from a collection D)

Matrix element:

- tf-idf($\underline{w}$, $\underline{d}$, $\underline{D}$) = tf(w, d) · idf(w, D)

$$\rightarrow \frac{N(w, d)}{\sum_{t' \in d} N(d')}$$

term frequency

$$\log \frac{|D|}{|\{d \in D : \underline{w} \in \underline{d}\}|}$$

inverse document frequency

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

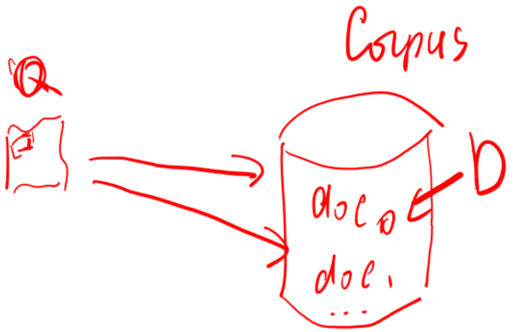
Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
  - Co-occurrence count
  - Bag-of-Words (BOW)
  - PPMI
  - TF-IDF
  - **BM25**
  - Latent Semantic Analysis
- Word2vec
- Useful facts

BM25 (best matching 25)

$$\mathrm{score}(D,Q)=\sum_{i=1}^{n}\mathrm{IDF}(q_i)\cdot\frac{\overset{\mathrm{TF}(q_i,D)}{\overset{\downarrow}{f(q_i,D)}}\cdot\overbrace{(k_1+1)}^{,D}}{f(q_i,D)+k_1\cdot\underbrace{\left(1-b+b\cdot\frac{|D|}{\text{avgdl}}\right)}_{D}}$$

$\forall q_i \in Q : \mathrm{score}(q_i, D)$



$$\mathrm{IDF}(q_i)=\ln\left(\frac{N-n(q_i)+0.5}{n(q_i)+0.5}+1\right)$$

$D \in \mathrm{Corpus}$

$\forall D \in \mathrm{Corpus}$

Source: https://en.wikipedia.org/wiki/Okapi_BM25

Table of Content

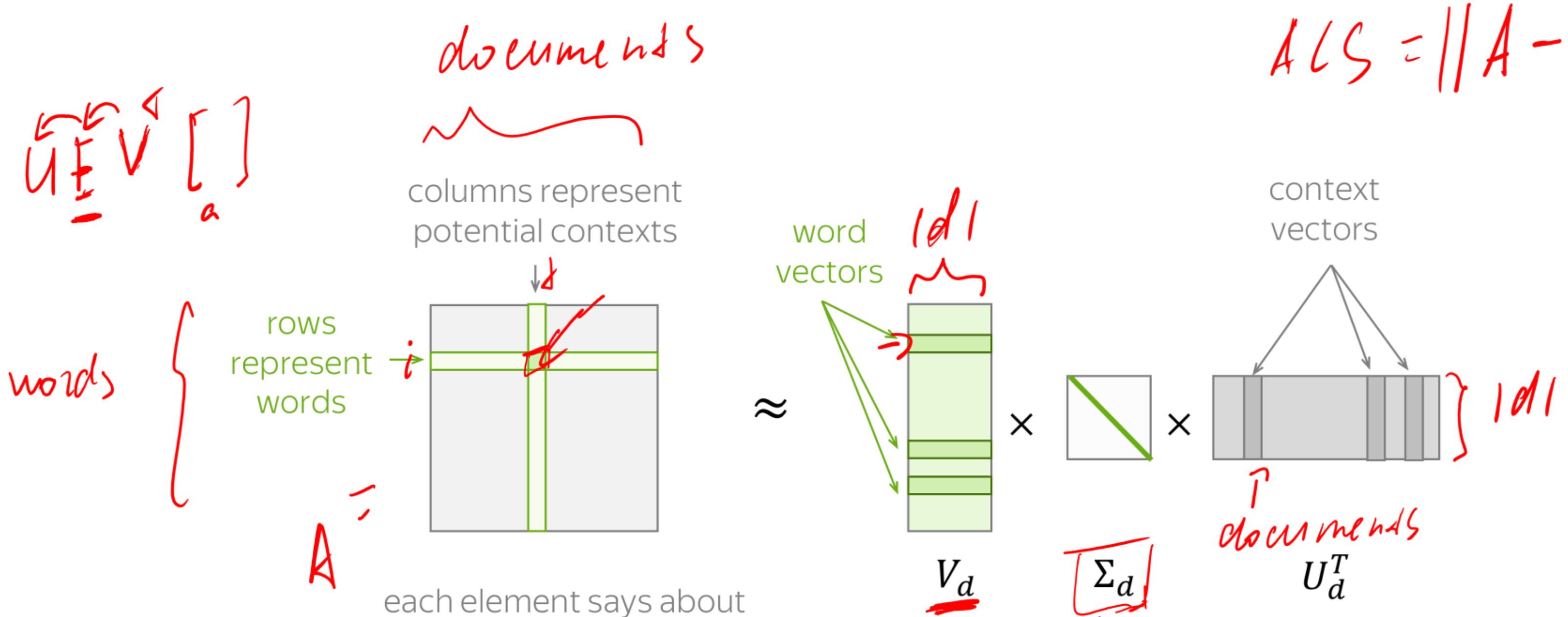
- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
    - Co-occurrence count
    - Bag-of-Words (BOW)
    - PPMI
    - TF-IDF
    - BM25
    - **Latent Semantic Analysis**
- Word2vec
- Useful facts

Matrix factorization is a way to get dense embeddings

$M = U \Sigma V^T$

$M[\ ]_{d} = U \underline{\Sigma} V^T [\ ]_a$

$ALS = \|A - UV^T\|_2^2 \to \min$



each element says about the association between a word and a context

Reduce dimensionality: Truncated Singular Value Decomposition (SVD)

$\underset{n \times m}{A} = \underset{n \times d}{U} \underset{d \times m}{V^T}$

$\underset{words \times d}{A} = UV^T$; $U = \underset{words}{\left[\ \overset{d}{\boxed{\ }}\ \right]}$; $V^T = \overset{contexts}{\left[\ \boxed{\ }\ \right]}\}d$ $d \times context$

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

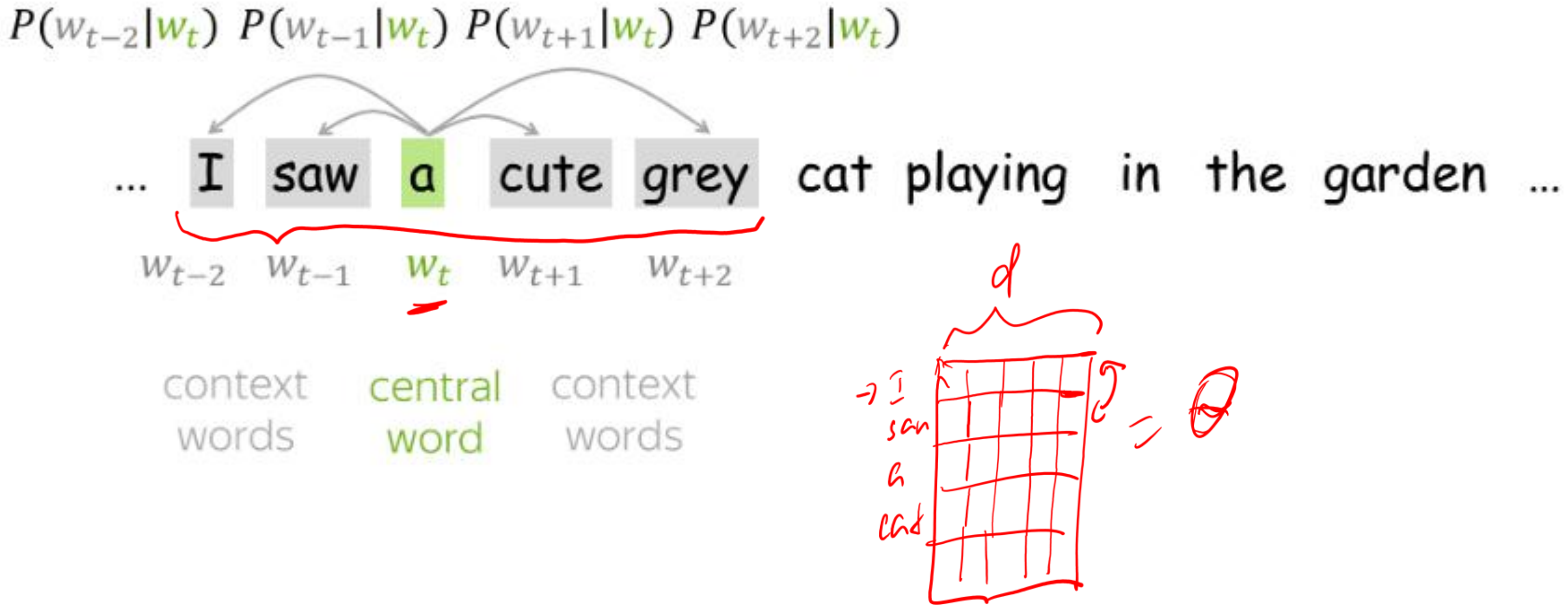
Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- **Word2vec**
  - Idea behind
  - Objective function
  - Training Procedure
  - Negative sampling
  - Skip-Gram vs. CBOW
  - GloVe
- Useful facts

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
  - **Idea behind**
  - Objective function
  - Training procedure
  - Negative sampling
  - Skip-Gram vs. CBOW
  - GloVe
- Useful facts

Slide one word at a time

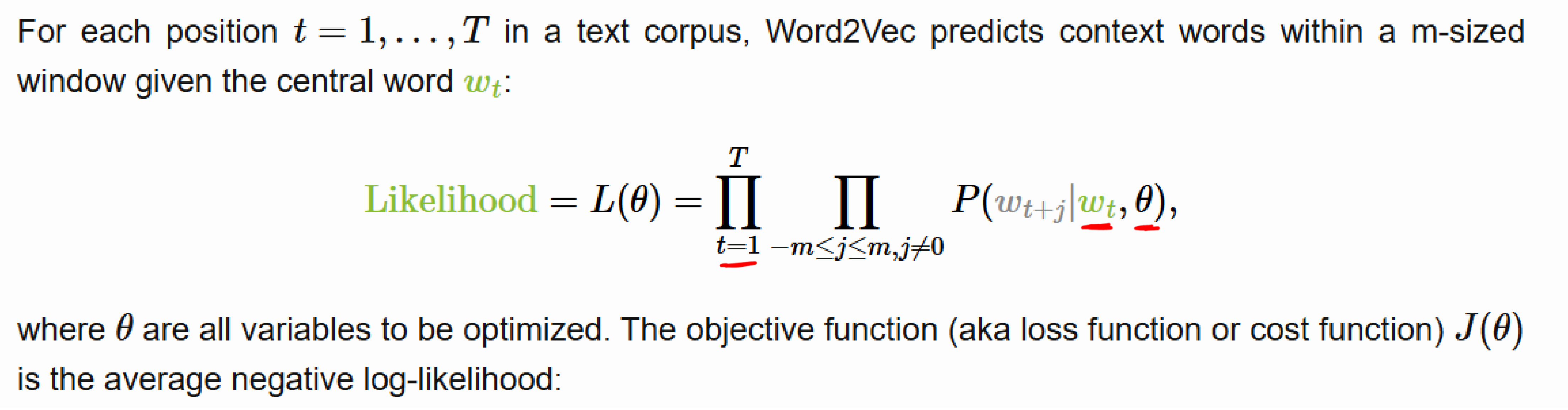

$P(w_{t-2}|w_t)$ $P(w_{t-1}|w_t)$ $P(w_{t+1}|w_t)$ $P(w_{t+2}|w_t)$
... I saw a cute grey cat playing in the garden ...
w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}
context words
central word
context words
d
I
saw
a
cat
= θ

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

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
  - Idea behind
  - **Objective function**
  - Training procedure
  - Negative Sampling
  - Skip-Gram vs. CBOW
  - GloVe
- Useful facts

Maximize the probability of encountering a target word given context



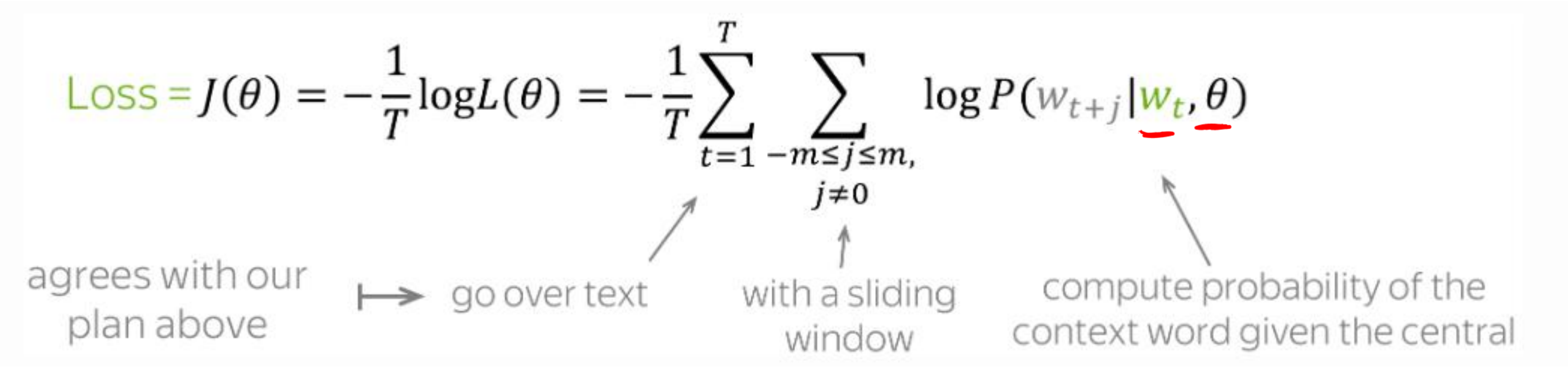
For each position $t = 1, \dots, T$ in a text corpus, Word2Vec predicts context words within a m-sized window given the central word w_t:

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

where θ are all variables to be optimized. The objective function (aka loss function or cost function) $J(\theta)$ is the average negative log-likelihood:

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

Loglikelihood for computational efficiency



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

agrees with our plan above ⟼ go over text

with a sliding window

compute probability of the context word given the central

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

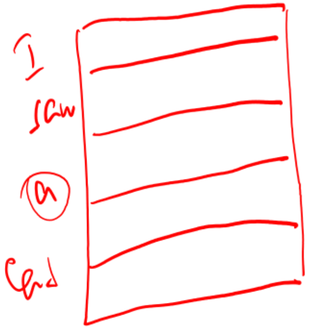
Loglikelihood for computational efficiency

$$\left\langle \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}}_{=\vec{a}}, \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}}_{=\vec{b}} \right\rangle = a_1 b_1 + a_2 b_2 + a_3 b_3 = (-\vec{a}-)\begin{bmatrix} | \\ \vec{b} \\ | \end{bmatrix}$$

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

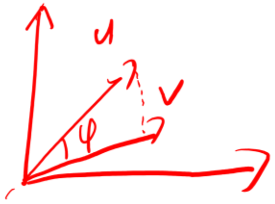
Dot product: measures similarity of o and c
Larger dot product = larger probability

Normalize over entire vocabulary
to get probability distribution



$P(I|a) = \langle I, a \rangle$

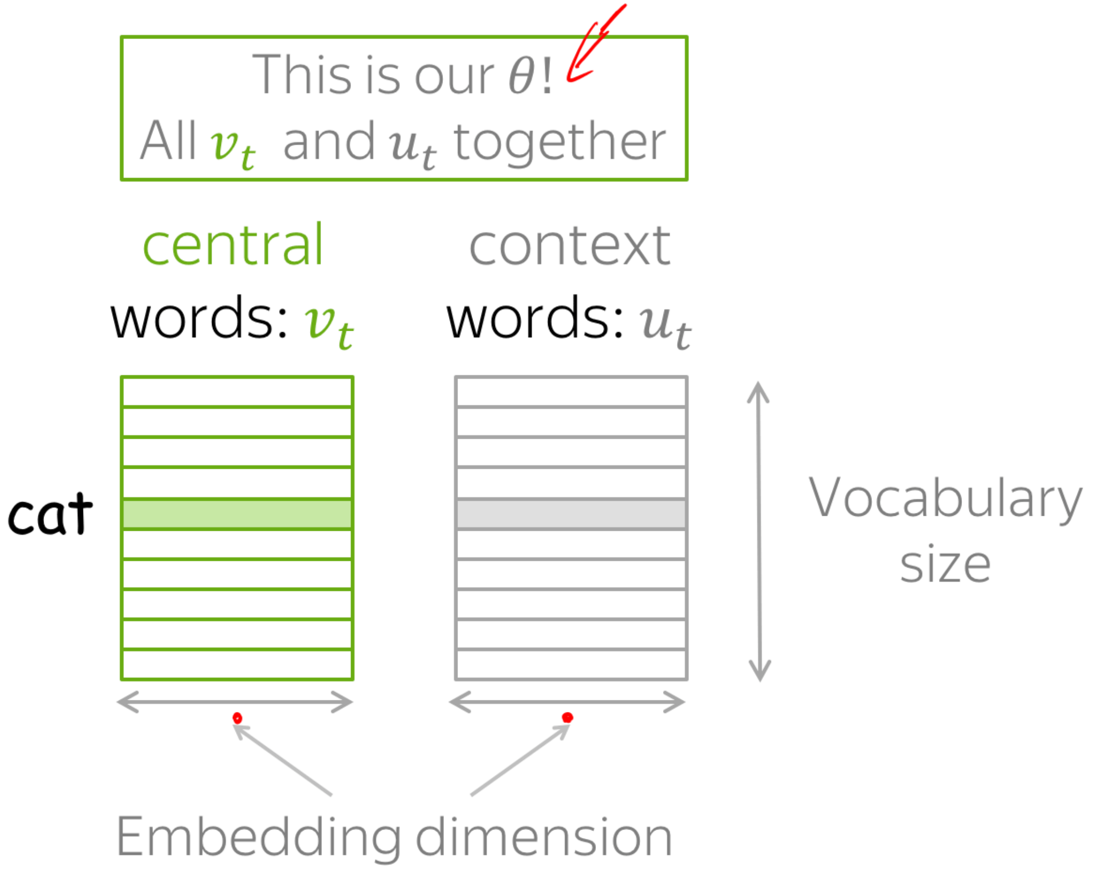
$P(saw|a) = \langle I, a \rangle$



$\langle u, v \rangle = |u| \cos\varphi \cdot |v|$

$\langle u, v \rangle = |u||v| \cos\varphi$

Note that we have distinct embeddings for context and target cases



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

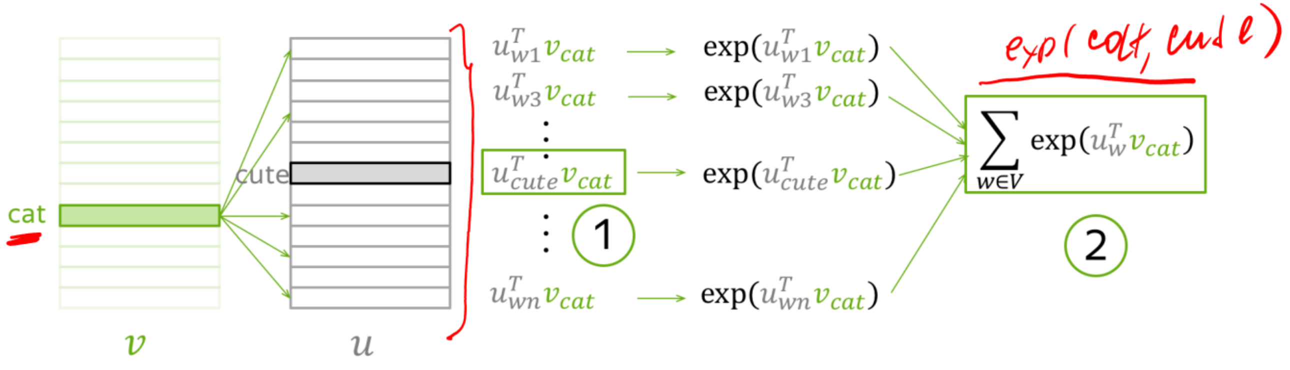
Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
  - Idea behind
  - Objective function
  - **Training procedure**
  - Negative Sampling
  - Skip-Gram vs. CBOW
  - GloVe
- Useful facts

A schematic overview on the training procedure

a cute cat is on a mat

1. Take dot product of v_{cat} with **all** u 2. exp 3. sum all



4. get loss (for this one step)

$$J_{t,j}(\theta) = \underbrace{-u_{cute}^T v_{cat}}_{1} + \underbrace{\log \sum_{w \in V} \exp(u_w^T v_{cat})}_{2}$$

5. evaluate the gradient, make an update

$$v_{cat} := v_{cat} - \alpha \frac{\partial J_{t,j}(\theta)}{\partial v_{cat}}$$

$$u_w := u_w - \alpha \frac{\partial J_{t,j}(\theta)}{\partial u_w} \; \forall\, w \in V$$

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

Table of Content

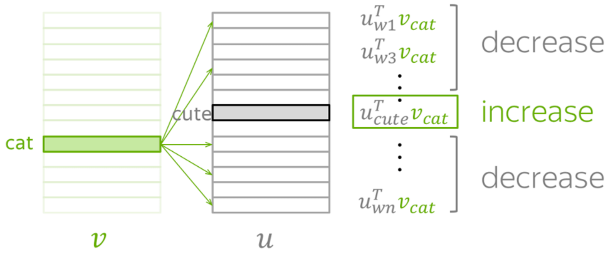
- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
  - Idea behind
  - Objective function
  - Training procedure
  - **Negative Sampling**
  - Skip-Gram vs. CBOW
  - GloVe
- Useful facts

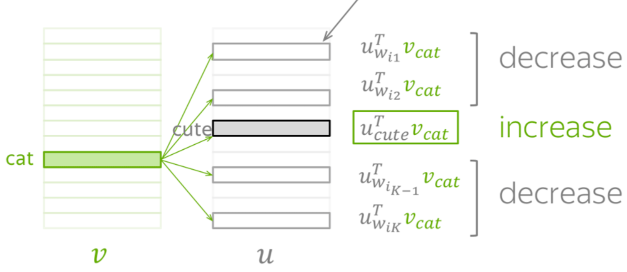
Negative sampling to speed up the computations

Dot product of v_{cat}:
- with u_{cute} - increase,
- with **all other** u - decrease

Dot product of v_{cat}:
- with u_{cute} - increase,
- with **a subset of other** u - decrease

Negative samples: randomly selected K words





Parameters to be updated:
- v_{cat}
- u_w for all w in the vocabulary

|V| + 1 vectors

Parameters to be updated:
- v_{cat}
- u_{cute} and u_w for w in K negative examples

K + 2 vectors

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

A loss function given negative sampling

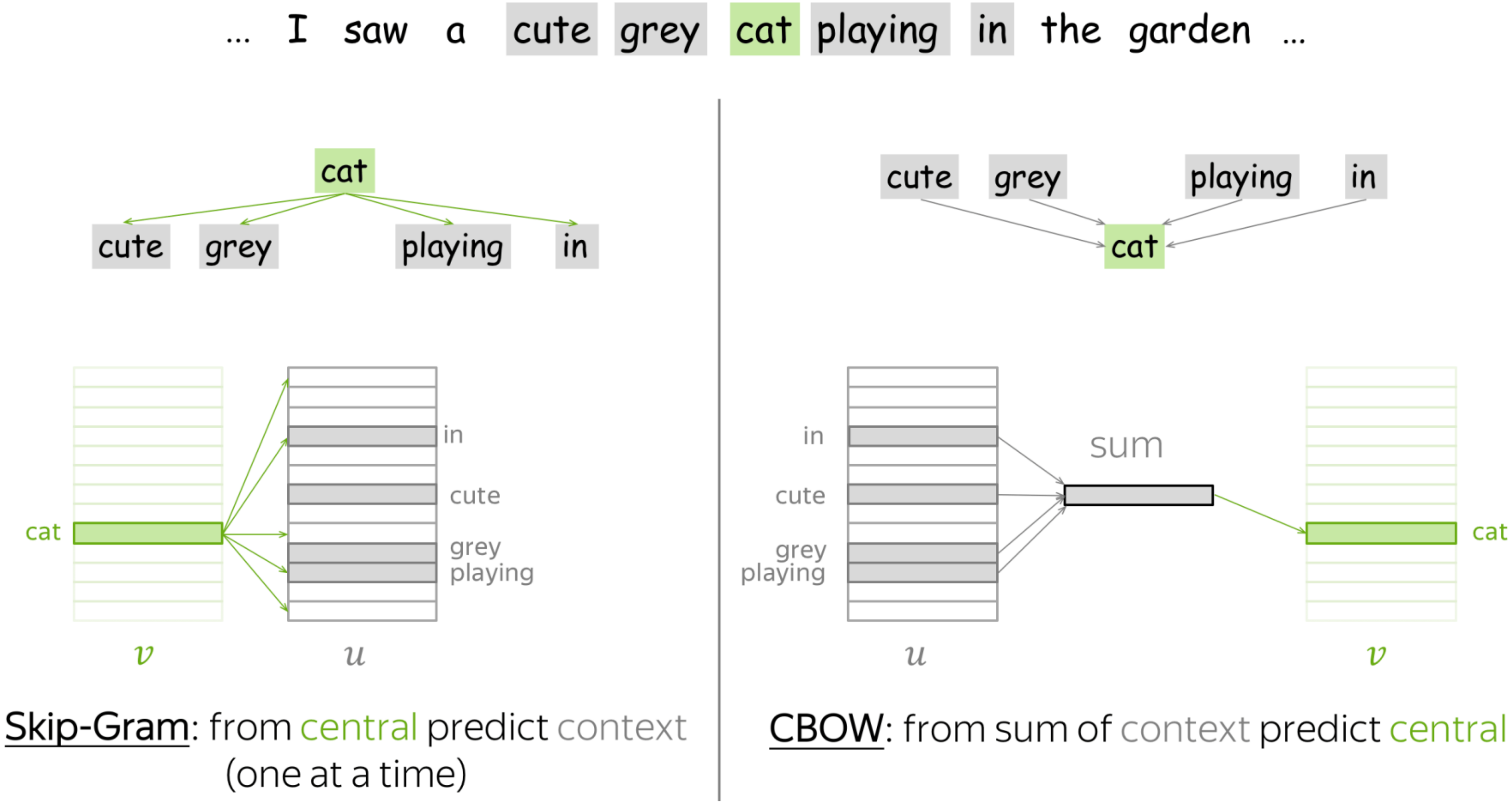
$$J_{\underline{t,j}}(\theta) = -\log\sigma(\underline{u_{cute}^{T}v_{cat}}) - \underbrace{\sum_{w\in\{w_{i_1},\ldots,w_{i_K}\}}}_{K}\log\sigma(-u_{w}^{T}v_{cat})$$

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

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
  - Idea behind
  - Objective function
  - Training procedure
  - Negative Sampling
  - **Skip-Gram vs. CBOW**
  - GloVe
- Useful facts

There are two ways to train the model



Skip-Gram: from central predict context (one at a time)

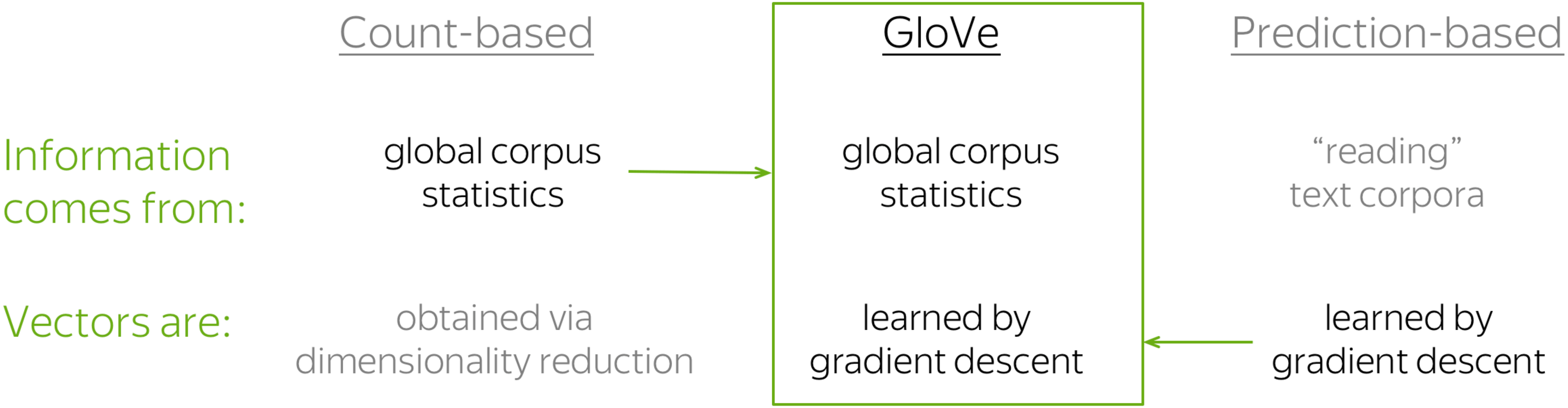
CBOW: from sum of context predict central

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

Table of Content

- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
  - Idea behind
  - Objective function
  - Training procedure
  - Negative Sampling
  - Skip-Gram vs. CBOW
  - **GloVe**
- Useful facts

We can merge the two world views



	Count-based	GloVe	Prediction-based
Information comes from:	global corpus statistics	global corpus statistics	“reading” text corpora
Vectors are:	obtained via dimensionality reduction	learned by gradient descent	learned by gradient descent

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

Table of Content

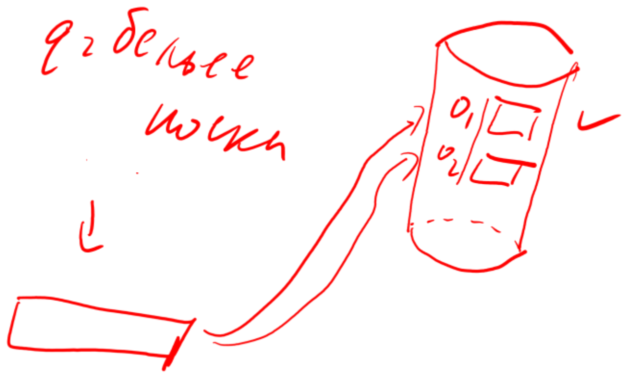
- Organizational matters
- Preprocessing pipeline
- But what is a Word Embedding?
- Count-based (pre-neural) approaches
- Word2vec
- **Useful facts**

- Normalize vectors due to cosine similarities nuances before moving embeddings to memory

$$\langle u, v \rangle = \underline{|u|}\,\underline{|v|}\cos\varphi$$

$$u := \frac{u}{|u|}$$

$$v := \frac{v}{|v|}$$

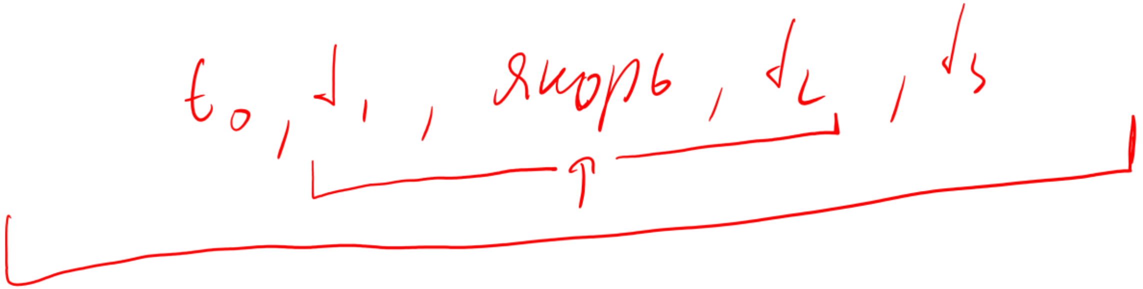


Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

- Normalize vectors due to cosine similarities nuances before moving embeddings to memory
- The context for antonyms is very similar, hence embeddings for them are close

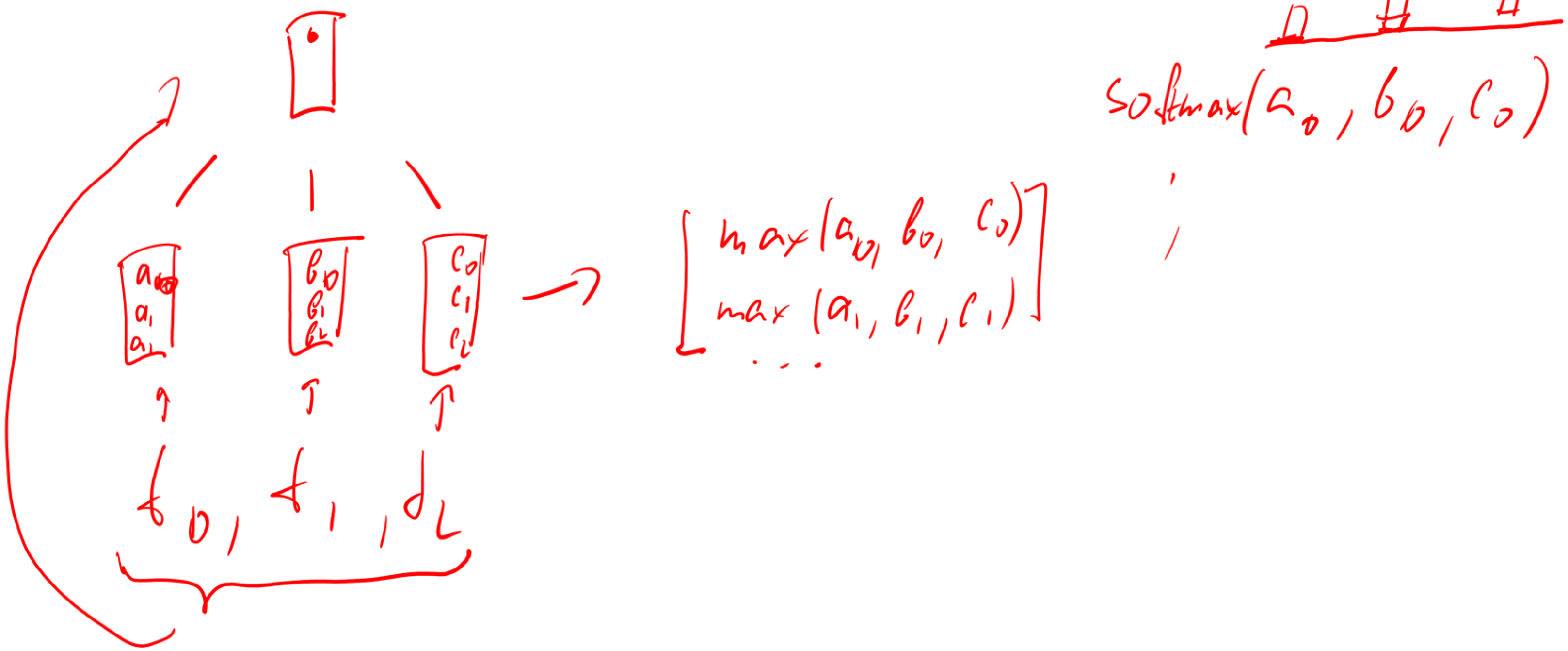
Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

- Normalize vectors due to cosine similarities nuances before moving embeddings to memory
- The context for antonyms is very similar, hence embeddings for them are close
- Window size of a context determines the kind of relations one captures



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

- Normalize vectors due to cosine similarities nuances before moving embeddings to memory
- The context for antonyms is very similar, hence embeddings for them are close
- Window size of a context determines the kind of relations one captures
- We can aggregate individual word-level embeddings to represent a semantic of a collection of words

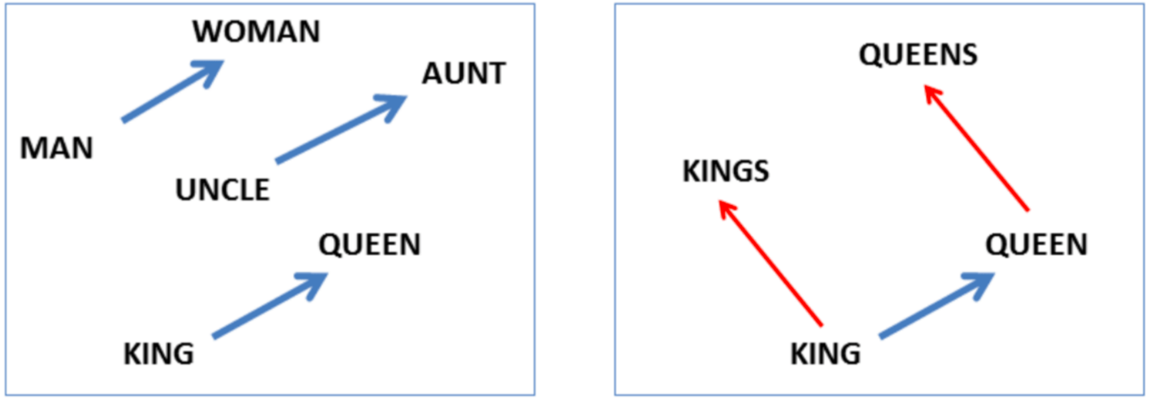


Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

- Embeddings learned with word2vec lie in a linear well-explainable space
- Similar languages preserve the form of the space accurate to linear transformations

semantic: v(**king**) – v(**man**) + v(**woman**) ≈ v(**queen**)

syntactic: v(**kings**) – v(**king**) + v(**queen**) ≈ v(**queens**)



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

HSE UNIVERSITY • EST 1992 •