Natural Language Processing: Word Embeddings

HSE Faculty of Computer Science

Machine Learning and Data-Intensive Systems

Murat Khazhgeriev

Table of Content

Table of Content

Table of Content

Grade policy

70% (homework) + 30% (exam)

Homeworks

Mandatory:

- (30%) 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.
- (20%) Week 5: Fine tuning LLM using PEFT.

Optional:

- (15%) Week 6. Fine-tuning your own model using the TRL library.
- (15%) Week 7. Implementation of Round-to-Nearest (RTN), Generalized Post-Training Quantization (GPTQ)

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

Table of Content

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", "."]

Table of Content

"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

«Динозавр игра в больш парк около школ."

Lemmatization

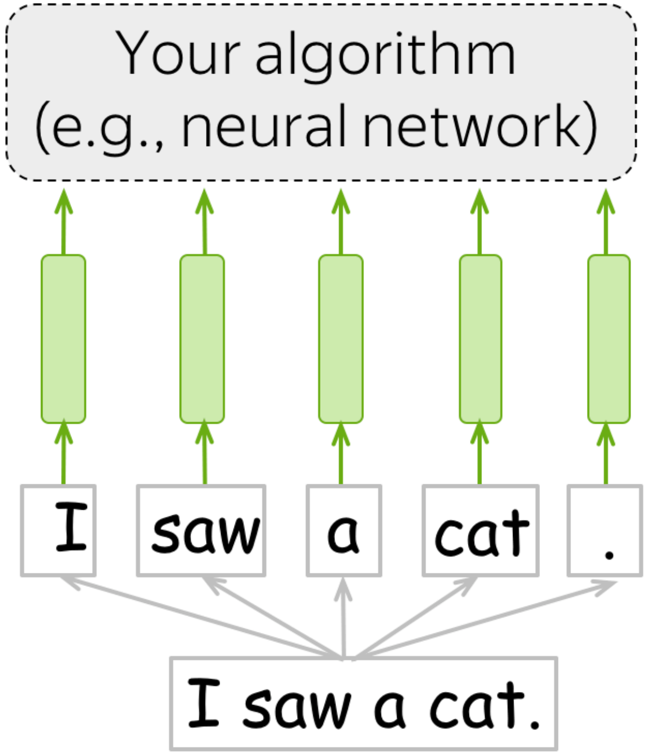
«Динозавр играть в большой парк около школа."

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

Tokenize an input text for further processing



Any algorithm for solving a task

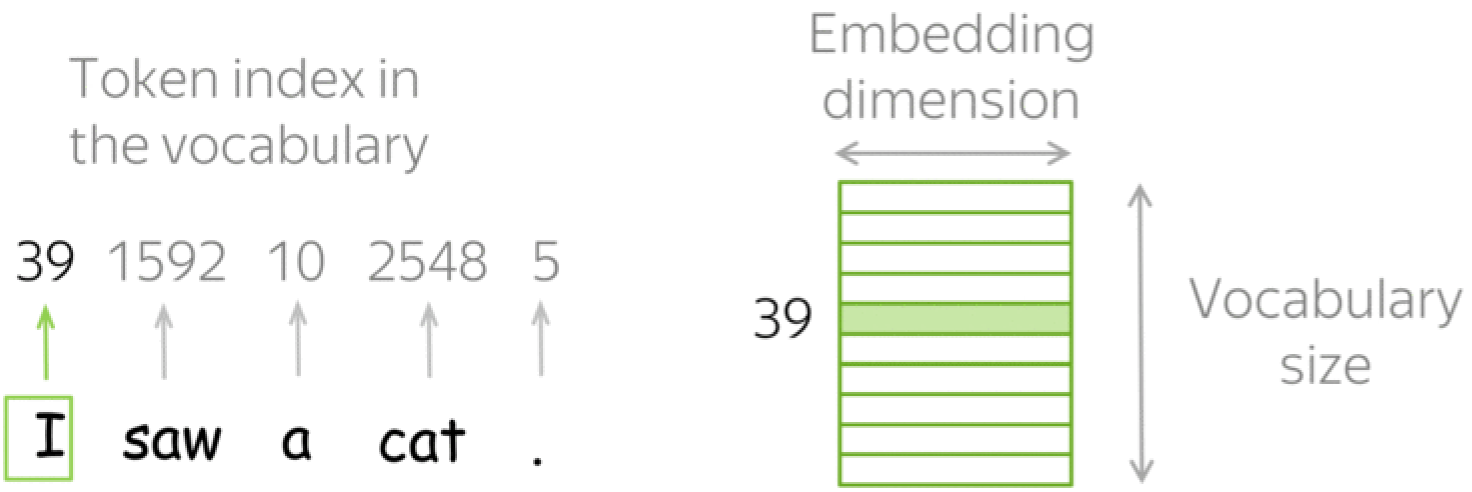
Word representation - vector (input for your model/algorithm)

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:

A bottle of tezgüino is on the table.

Everyone likes tezgüino.

Tezgüino makes you drunk.

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.

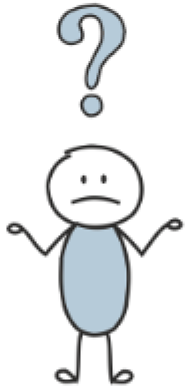
(4) We make _______ out of corn.

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	

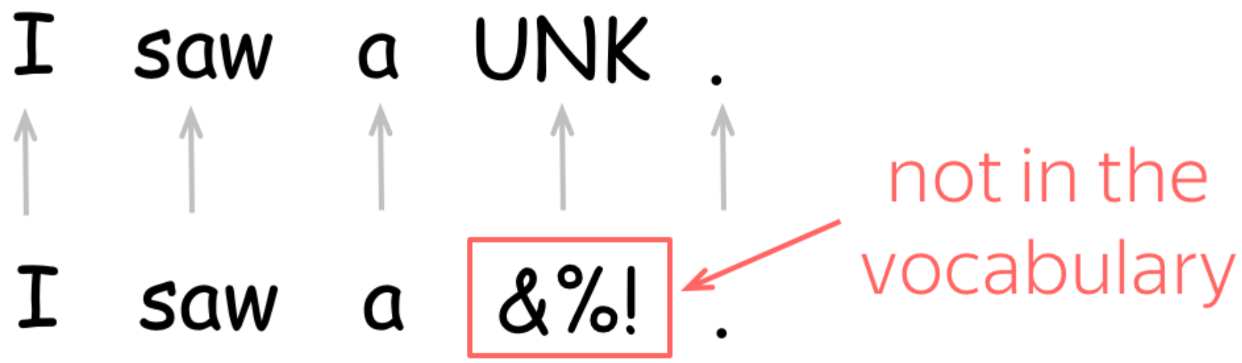
← contexts

← 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

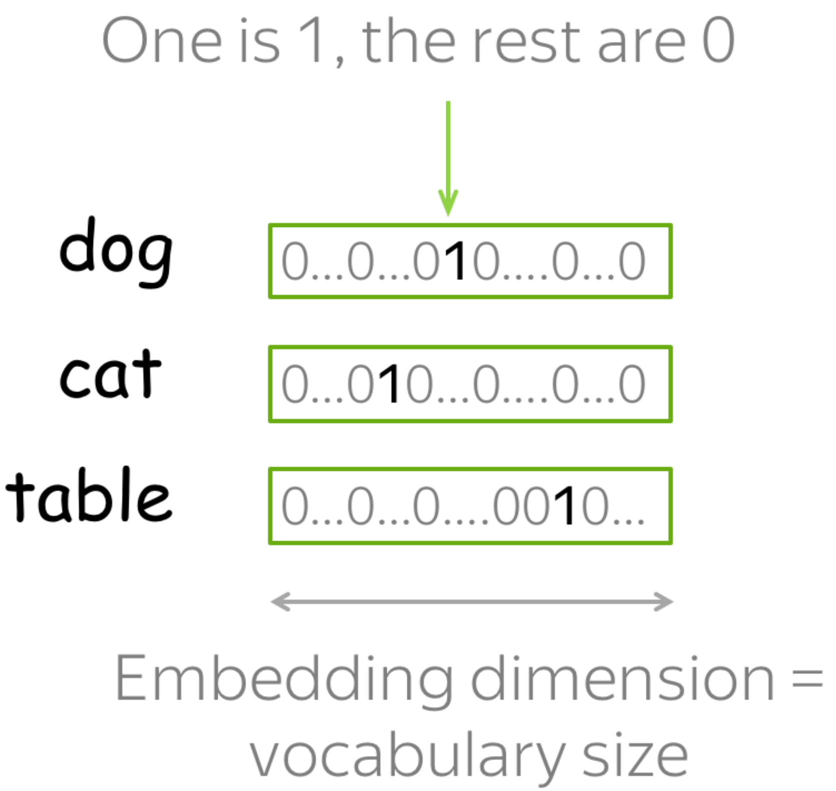
Reserve a token for special cases e.g. unknown words



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

Table of Content

The easiest way to go is One-Hot Encoding



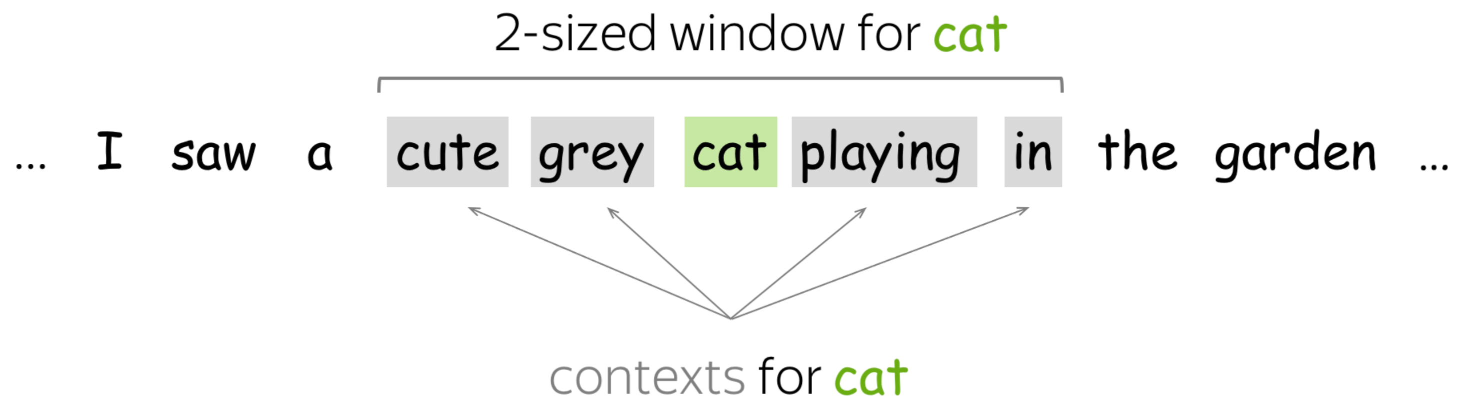
Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

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

Define context via a window in a text



$$
\begin{array}{c|cccc}
 & w_1 & w_2 & \dots & w_d \\
\hline
w_1 & n_{11} & n_{12} & \dots & n_{1d} \\
w_2 & & \circ\ \circ\ \circ & & \\
\vdots & & & & \\
w_d & & & & n_{dd}
\end{array}
$$

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

Table of Content

We can also treat the whole document as a context

D1: a cat sat on a mat

D2: a mat for a dog

→

	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

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

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

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

Table of Content

We can also account for a term being widespread

<u>Context:</u>

- document d (from a collection D)

<u>Matrix element:</u>

- tf-idf(w, d, D) = tf(w, d) · idf(w, D)

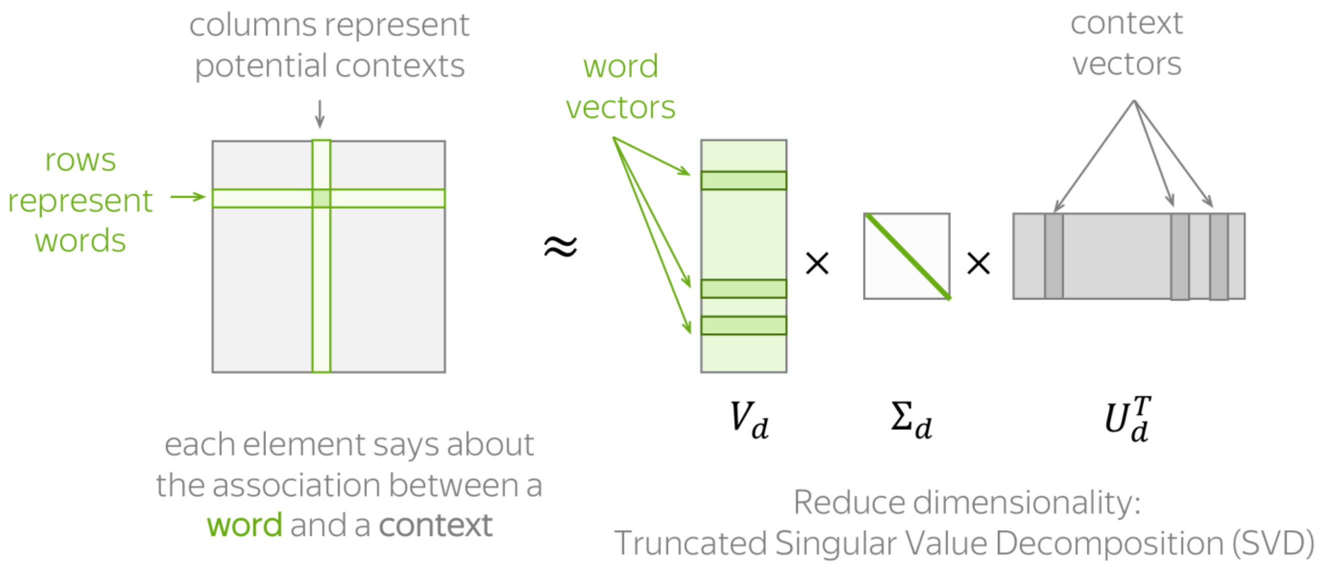
$$N(w, d) \qquad \log \frac{|D|}{|\{d \in D: w \in d\}|}$$

term frequency — inverse document frequency

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

Table of Content

Matrix factorization is a way to get dense embeddings

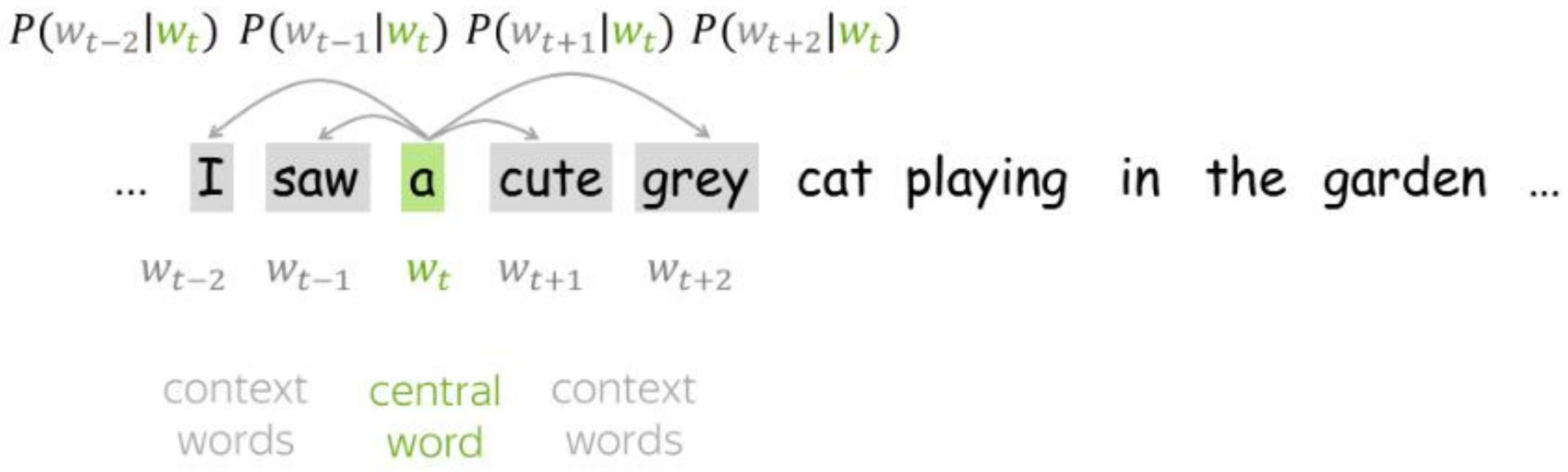


Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

Table of Content

Table of Content

Slide one word at a time



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:

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

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

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

agrees with our plan above $\longmapsto$

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

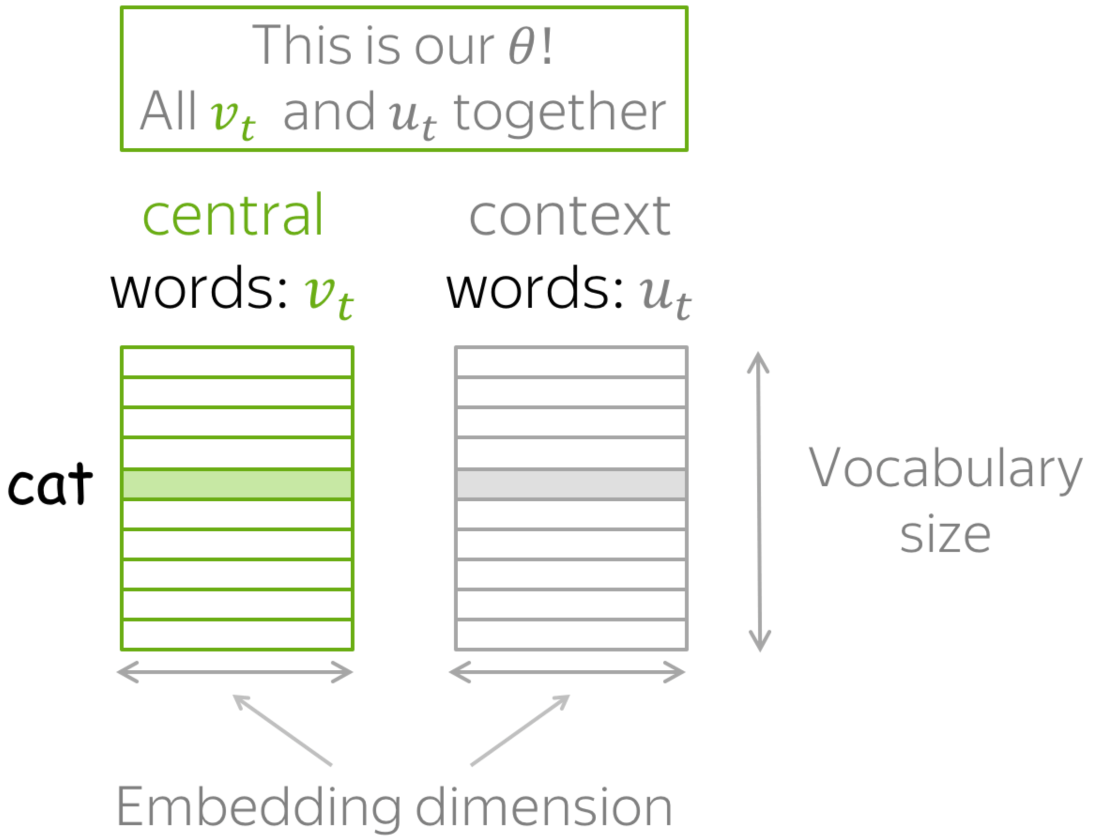
Loglikelihood for computational efficiency

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

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

Normalize over entire vocabulary
to get probability distribution

Note that we have distinct embeddings for context and target cases

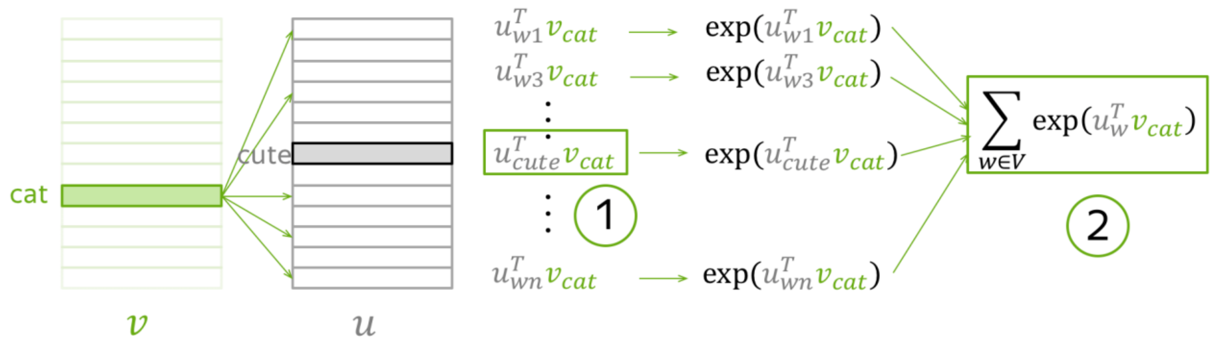


Source: https://lena-voita.github.io/nlp_course/word_embeddings.html

Table of Content

A schematic overview on the training procedure

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

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

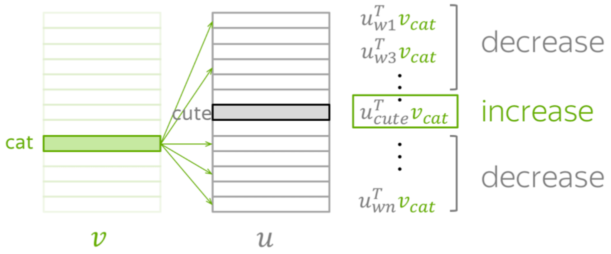
$$u_w := u_w - \alpha \frac{\partial J_{t,j}(\theta)}{\partial u_w} \ \forall\, w \in V$$

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

Table of Content

Negative sampling to speed up the computations

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

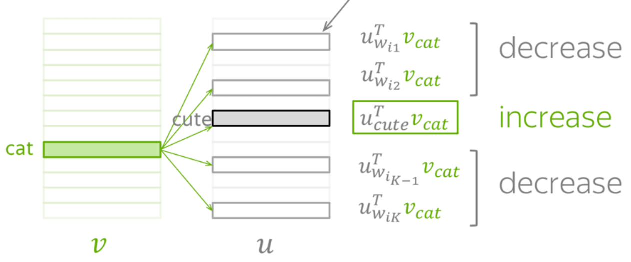


Parameters to be updated:
- v_{cat}
- u_w for all w in the vocabulary

|V| + 1 vectors

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

<u>Negative samples</u>: randomly selected K words



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

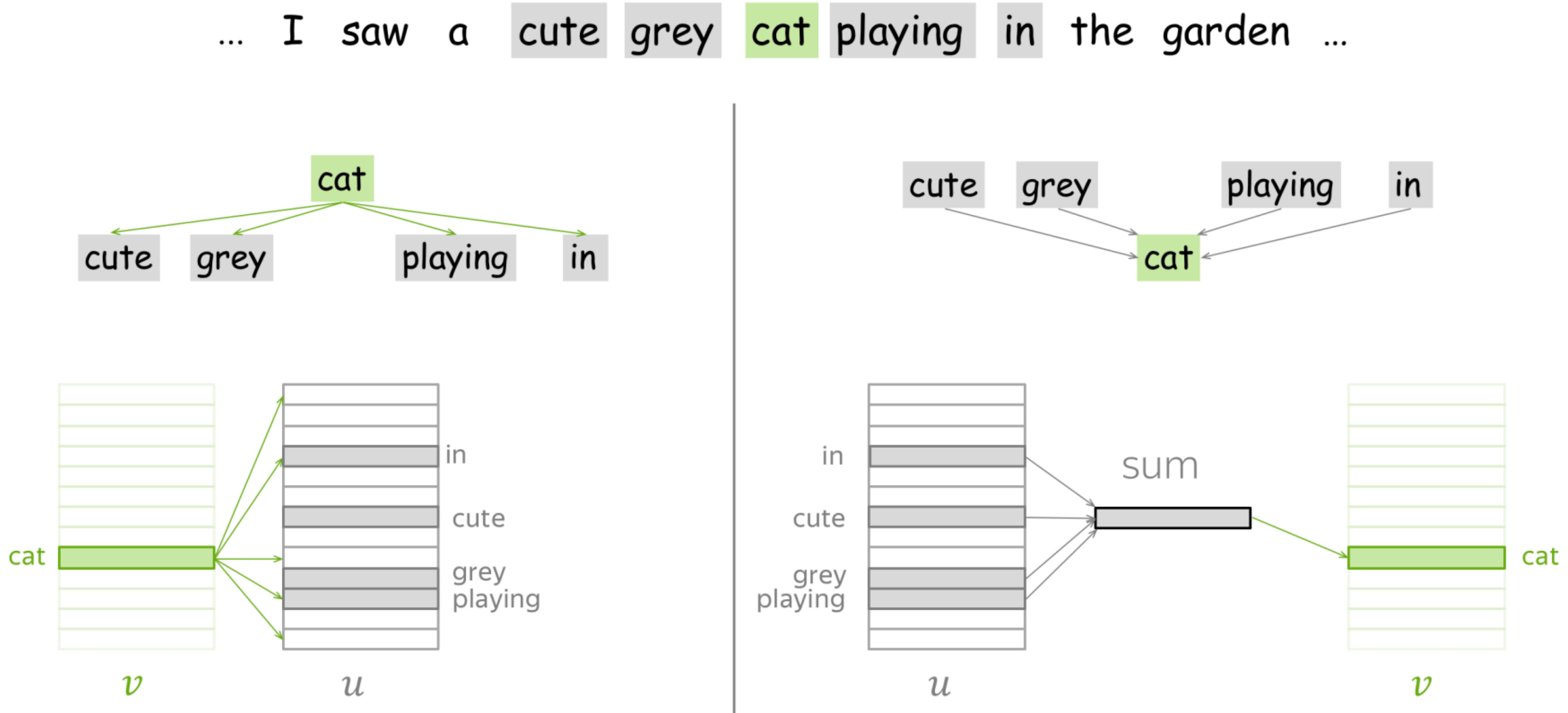
A loss function given negative sampling

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

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

Table of Content

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

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**

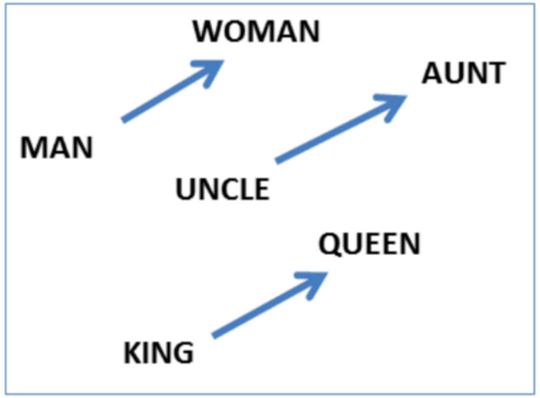
- Normalize vectors due to cosine similarities nuances before moving embeddings to memory
- The context for antonyms is very similar, hence embeddings for them are close

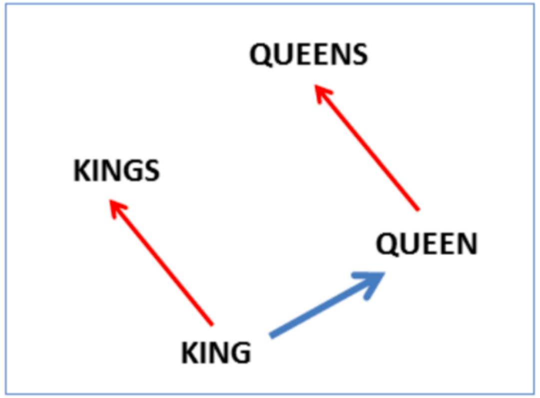
$$\langle u, v \rangle = |u||v| \cos\varphi$$

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

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

syntactic: $v(\text{kings}) - v(\text{king}) + v(\text{queen}) \approx v(\text{queens})$





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

HSE UNIVERSITY • EST 1992