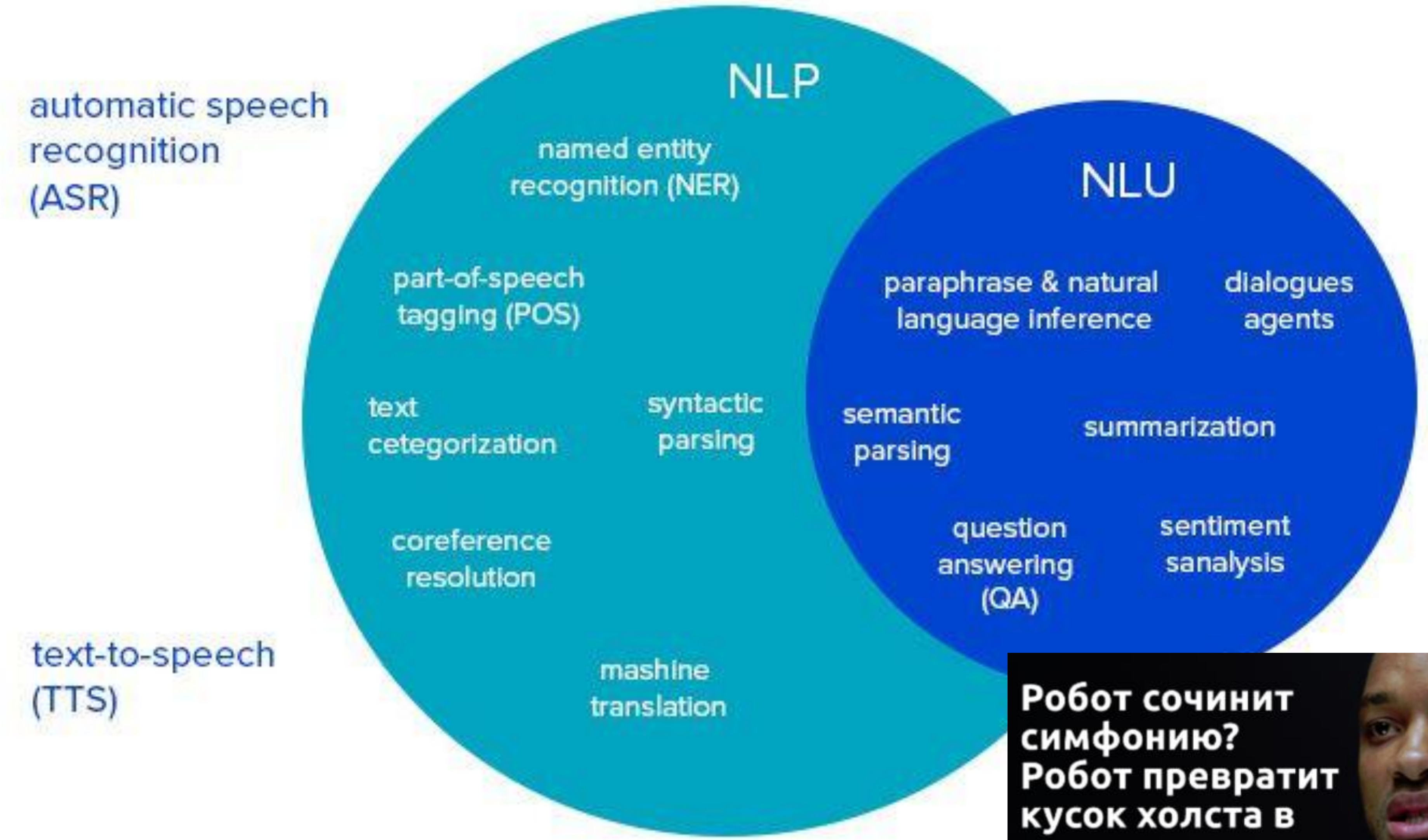
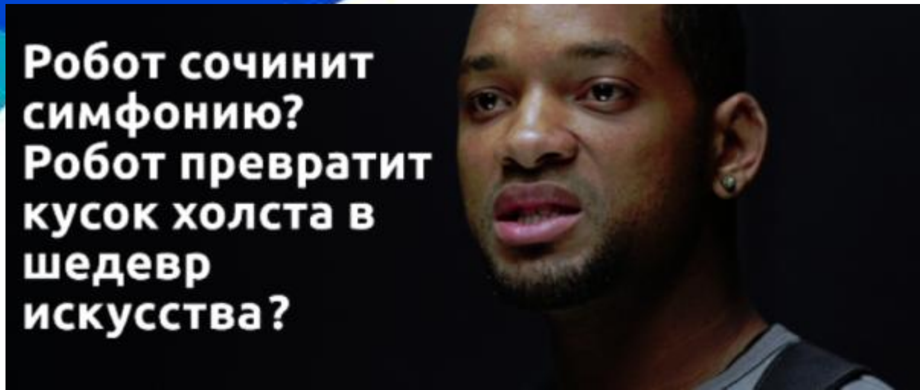
Lecture 4 – NLP. Part 1.
From Words to Vectors

Natural language processing





Machinelearning
Training
Fine-tuning
Glove
Pre-training
Training
Syntax
Sentiment
Intent
Dependency
Chatbot
Generation
Deeplearning
Corpus
Fine-tuning
Stemming
Context
Entities
Lstm
Summarization
Stemming
Entities
Understanding
Stopwords
Semantics
Classification
Grammar
Lemmatization
Algorithm
Coherence
Pos
Pos
Zeroshot
Summarization
Gpt
Ner
Ai
Glove
Translation
Translation
Parse
Lemmatization
Nlg
Rnn
Wordvec
Algorithm
Parse
Sentiment
Lemmatization
Stopwords
Model
Zeroshot
Pipeline
Understanding
Pos
Bert
Lstm
Algorithm
Embeddings
Deeplearning
Inference
Machinelearning
Nlg
Machinelearning
Pipeline
Gru
Ner
Model
Grammar
Cnn
Rnn
Entities
Nlu
Ai
Cnn
Ner
Rnn
Fine-tuning
Dependency
Gpt
Bert
Chatbot
Model
Tokens
Semantics
Nlp
Classification
Transformers
Corpus
Speech
Nlp
Gru
Intent
Wordvec
Understanding
Stopwords
Pre-training
Vocabulary
Pipeline
Coherence
Translation
Inference
Syntax
Classification
Speech
Transformers
Ai
Stemming
Syntax
Training
Context
Attention
Zeroshot
Gru
Corpus
Pre-training
Nlu
Sentiment
Generation
Intent
Cnn
Gpt
Embeddings
Generation
Wordvec
Lstm
Summarization
Deeplearning
Chatbot
Embeddings
Vocabulary
Glove
Inference
Bert
Attention
Context
Attention
Transformers
Coherence
Semantics

natasha slovnet() yargy

nae>al navec nerus.conllu

raz|del corus.tar.gz ipymarkup

https://natasha.github.io/

Possible approaches to make vectors

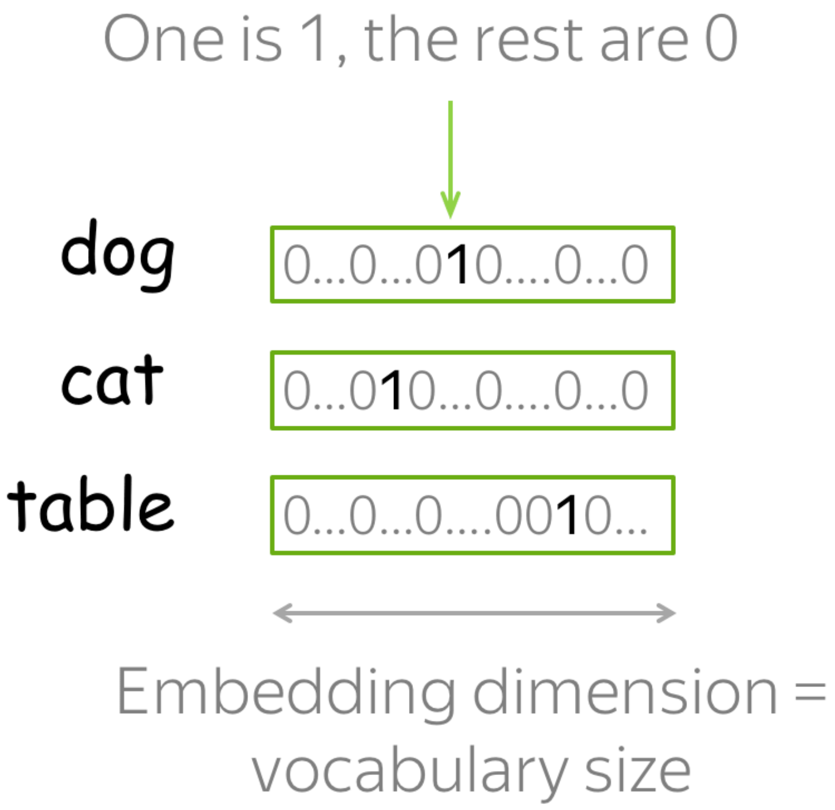
- Naive
  - One-hot
  - BoW
  - TF-IDF

- Count based
  - Co-occurency
  - PPMI & co
  - LSA/LSI

- Prediction-Based
  - Word2Vec
  - GloVe
  - FastText

Transformer-based

One-Hot Encoding



- What if on test we have word that is not in vocabulary?
- How to implement?
- Why not? What a problem?

Bag-of-Words (BoW)

What the difference with One-Hot to sentences?

Bag-of-Words (BoW)

- One-hot is originally for words, if you use it for sentences – it summarize one-hot vectors for words in sentence, but without repetition. BoW – update of One-Hot for sentences, that allow repetitions of words and display it on final vector.

He is good player,
he is really good!

	he	***good***	***work***	***really***
BoW	**2**	**2**	0	1
One-Hot	**1**	**1**	0	1

Term Frequency-Inverse Document Frequency

TF

Measures how often a term appears in a specific document

- Variants:

— Raw count

— Boolean (presence/absence)

— Log-scale: $1 + log(tf)$

— Double-normalization:

$$0.5 + 0.5 * max(tf \ in \ doc)$$

IDF

Measures how rare/common a term is across the whole document corpus

- Variants:

— Classic: $\log\left(\frac{\mathrm{N}}{\mathrm{df_t}}\right)$

— Smooth: $\log\left(\frac{\mathrm{N+1}}{\mathrm{df_t+1}}\right) + 1$

— Probabilistic: $\log\left(\frac{\mathrm{N-df_t}}{\mathrm{df_t}}\right)$

— Max-IDF..

Overall problem

- How to use connections between words?

Use n-gramms

Benefits:

- captures local context and word order (e.g. “not good” ≠ “good”)
- helps with multi-word expressions / phrases (“New York”, “machine learning”)
- improves discrimination in tasks like sentiment, topic classification, document similarity

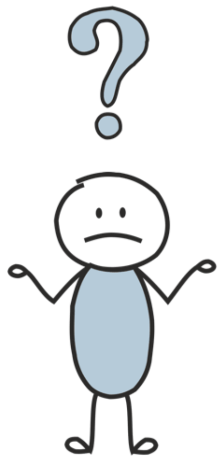
Trade-offs:

- increases feature space dimensionality → more sparse matrices, more memory/time cost
- risk of overfitting on rare n-grams if data small
- need to tune n (2, 3, maybe more) and frequency thresholds

What is meaning?

Do you know what the word tezgüino means ?

(We hope you do not)

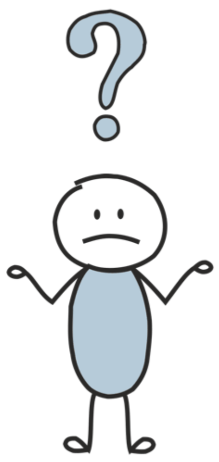


What is meaning?

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 ?



*Words which frequently appear in **similar contexts** have **similar meaning***

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.

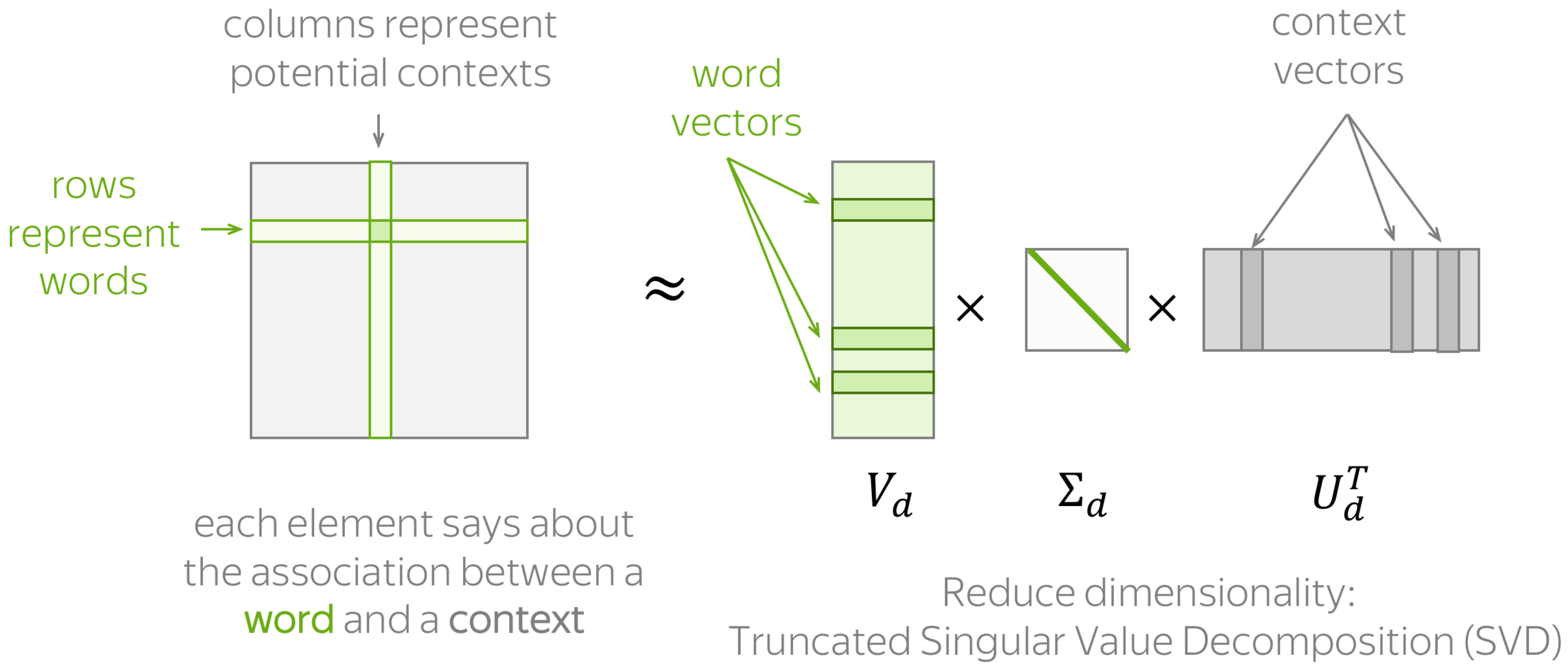
Tezgüino is a kind of alcoholic beverage made from corn.

With context, you can understand the meaning!

https://lena-voita.github.io/nlp_course

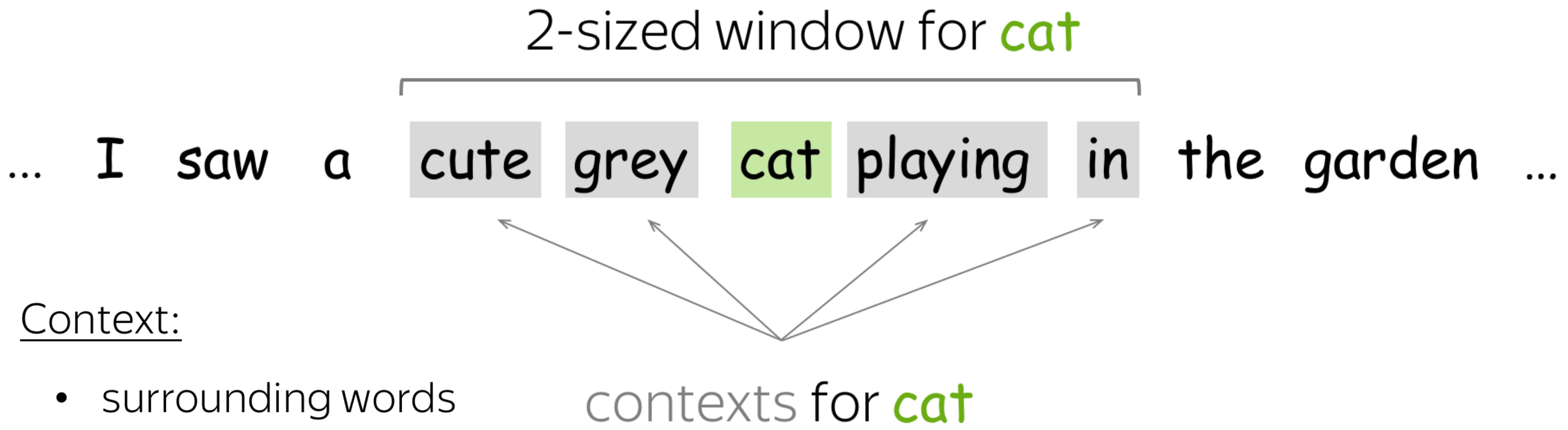
Count-based approaches

Put this information **manually,** based on global corpus statistics.



https://lena-voita.github.io/nlp_course

What is context and matrix element



Context:

- surrounding words in a L-sized window

Matrix element:

- N(w, c) – number of times word w appears in context c

https://lena-voita.github.io/nlp_course

About context

- Are context words at different distances equally important? If not, how can we modify co-occurrence counts?
  HAL paper

- Context before interesting than after?
  In HAL paper there are two context – left + right

- How to choose length of window?
  Redefining Context Windows for Word Embedding Models

About matrix element

Context:

- surrounding words in a L-sized window

Matrix element:

- PPMI(w, c) = max(0, PMI(w, c)),

where

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

- Raw count / conditional probability
- Positive Pointwise Mutual Information
- PPMI with Context Distribution Smoothing) – $P(c)^{\alpha}, \alpha < 1$ - increase probability rare context
- t-statistic - $\frac{P(w,c)-P(w)P(c)}{\sqrt{P(w,c)/N}}$, N – corpus size
- Shifted PPMI (SPPMI) – when PPMI is huge and sparse
- * SVD Weighting – additional weights for not null elements after PPMI

https://lena-voita.github.io/nlp_course

Latent Semantic Analysis

- While in the previous approaches contexts served only to get word vectors and were thrown away afterward, here we are also interested in context, or, in this case, document vectors.

Context:

- document d (from a collection D)

Matrix element:

- $\text{tf-idf}(w, d, D) = \text{tf}(w, d) \cdot \text{idf}(w, D)$

$$N(w, d)$$

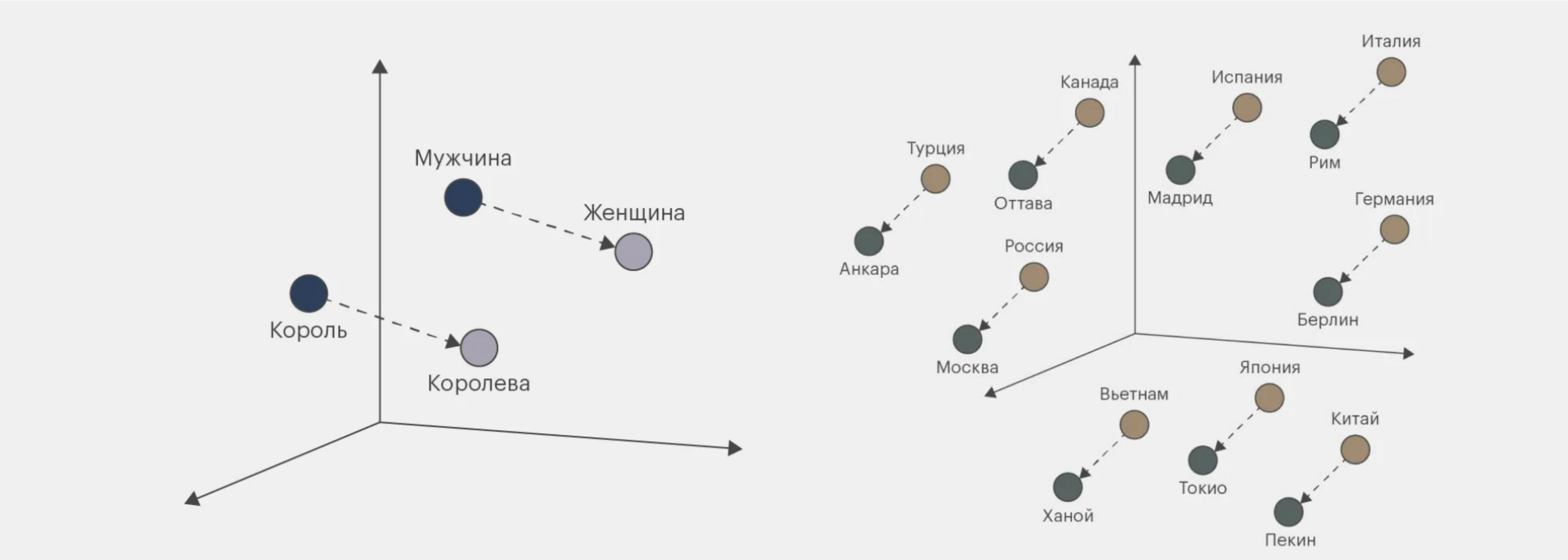
term frequency

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

inverse document frequency

https://lena-voita.github.io/nlp_course

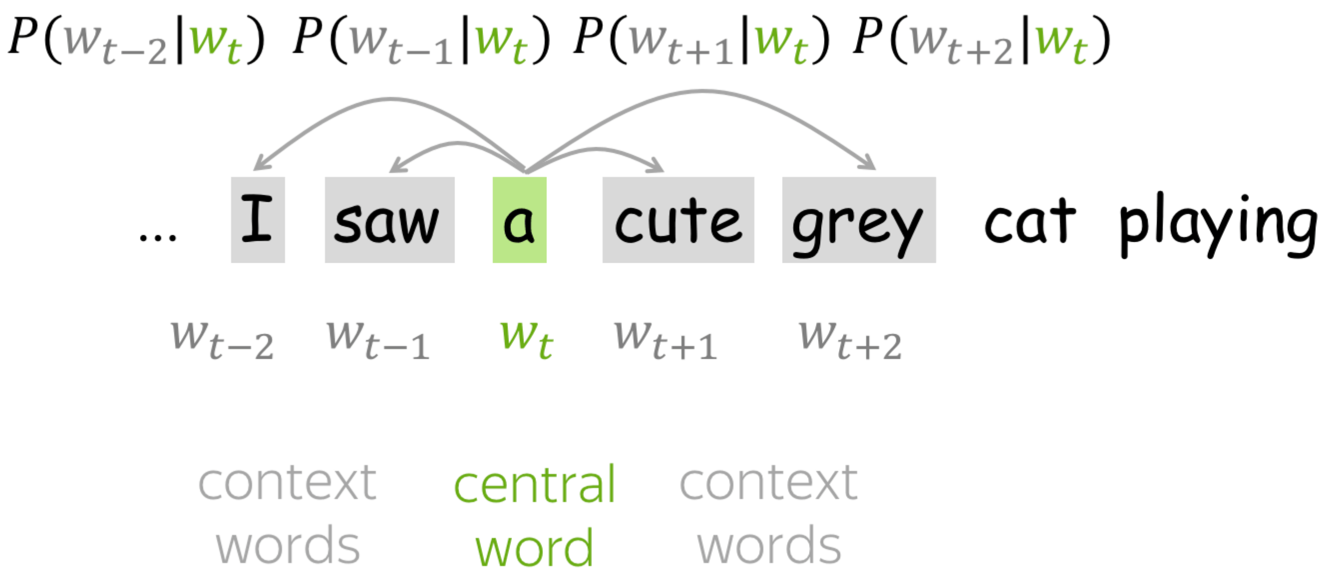
Word2Vec



https://blog.skillbox.by/

Word2Vec

- **Goal:** Learn word vectors by teaching them to predict contexts
  - take a huge text corpus
  - go over the text with a sliding window, moving one word at a time.
  - for the central word, compute probabilities of context words;
  - adjust the vectors to increase these probabilities.



https://lena-voita.github.io/nlp_course

Word2Vec

$P(w_{t-2}|w_t)$ $P(w_{t-1}|w_t)$ $P(w_{t+1}|w_t)$ $P(w_{t+2}|w_t)$

… 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

https://lena-voita.github.io/nlp_course

Objective Function: Negative Log-Likelihood

Word2Vec tries to find the parameters that maximize the data likelihood:

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

We want our model to think that the training data is "likely"

To do this, it uses negative (log-)likelihood as its loss function:

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

How to compute this?

agrees with our plan above ⟼ go over text

with a sliding window

compute probability of the context word given the central

https://lena-voita.github.io/nlp_course

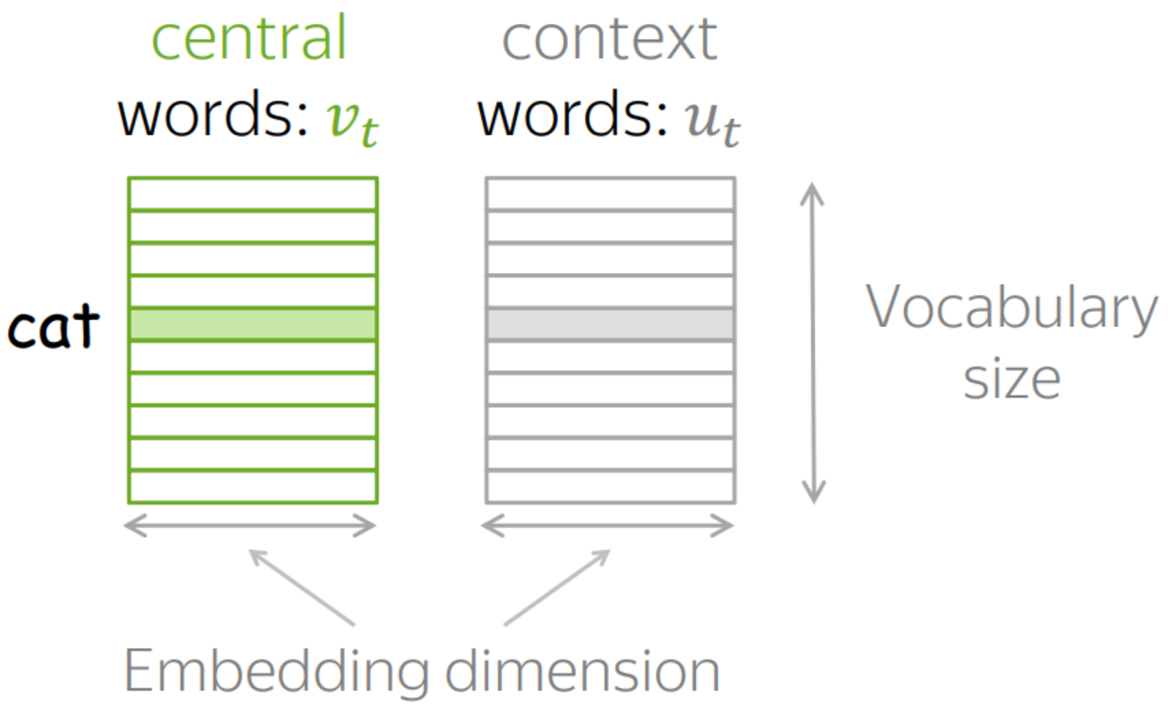
How to compute $P(w_{t+j}|w_t, \theta)$?

For each word w, we will have two vectors:

- v_w when it is a central word
- u_w when it is a context word

Once the vectors are trained, usually we throw away context vectors and use only word vectors.

This is our θ!
All v_t and u_t together



https://lena-voita.github.io/nlp_course

How to compute $P(w_{t+j}|w_t, \theta)$?

For the central word c and context word o (o - outside):

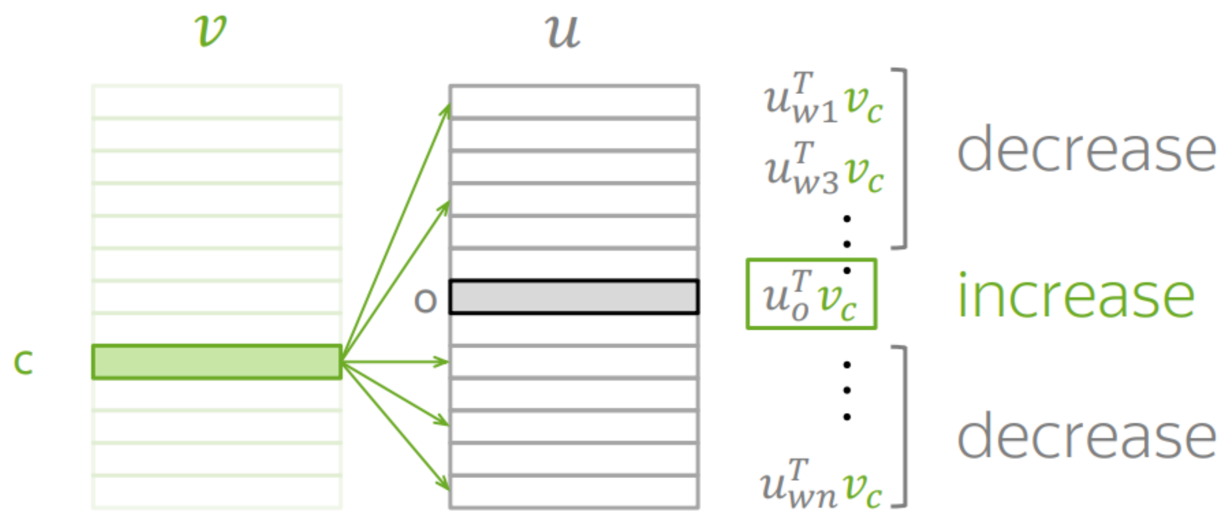
$$P(o|c) = \frac{\exp(\boxed{u_o^T v_c})}{\sum_{w \in V} \exp(u_w^T v_c)}$$

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

Normalize over entire vocabulary
to get probability distribution

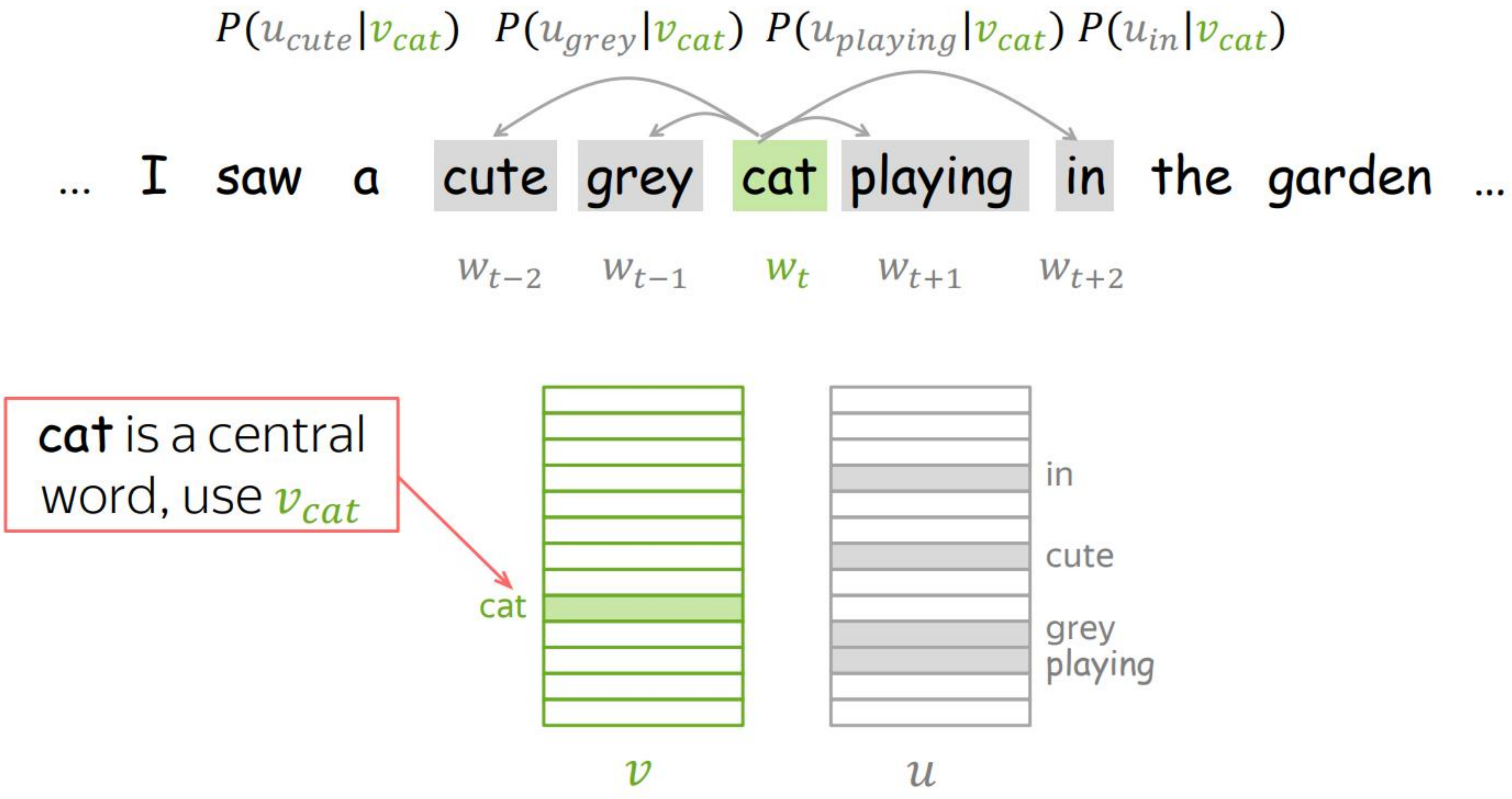
Let us recall our plan:

- ...
- adjust the vectors to increase these probabilities.



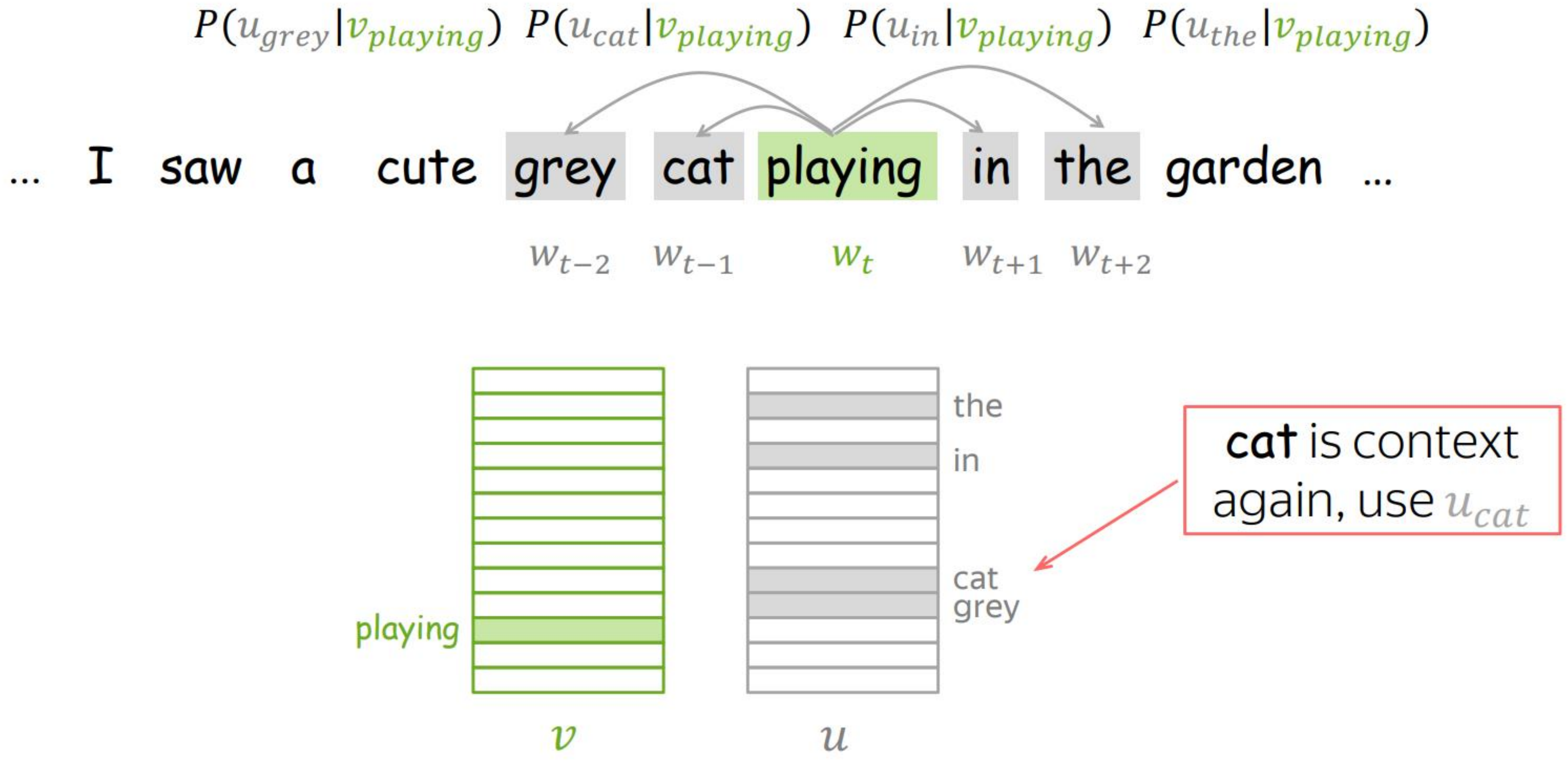
https://lena-voita.github.io/nlp_course

Two vectors for each word



https://lena-voita.github.io/nlp_course

Two vectors for each word



https://lena-voita.github.io/nlp_course

One training step in detail

Loss for word j in window t

$$\text{Loss} = J(\theta) = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m,\\ j\neq 0}} \log P\left(w_{t+j}\middle|w_t,\theta\right) = -\frac{1}{T}\sum_{t=1}^{T}\underbrace{\sum_{\substack{-m\le j\le m,\\ j\neq 0}} J_{t,j}(\theta)}_{\text{pick one window}}$$

pick one window

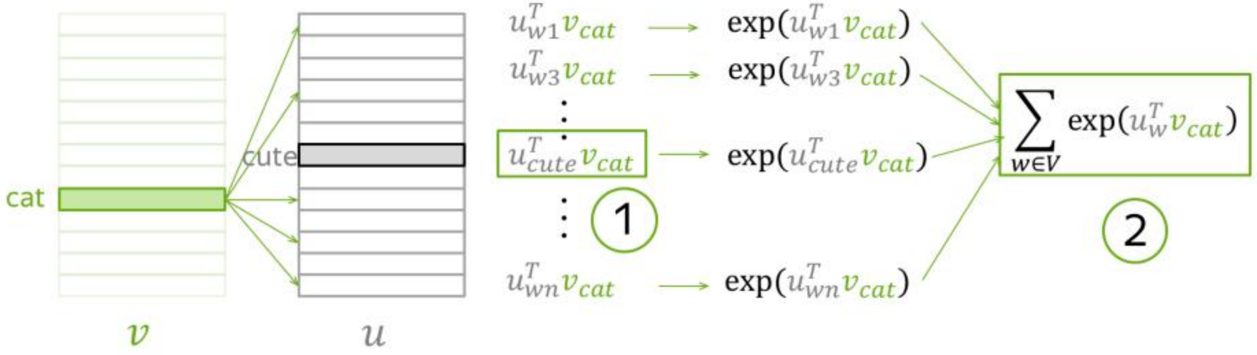
… I saw a cute grey cat playing in the garden …

w_{t-2} w_{t-1} w_t w_{t+1} w_{t+2}

One training step in detail

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

$$-\log P(cute|cat)$$
$$= -u_{cute}^T v_{cat} + \log \sum_{w \in V} \exp(u_w^T v_{cat})$$



4. get loss (for this one step)

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

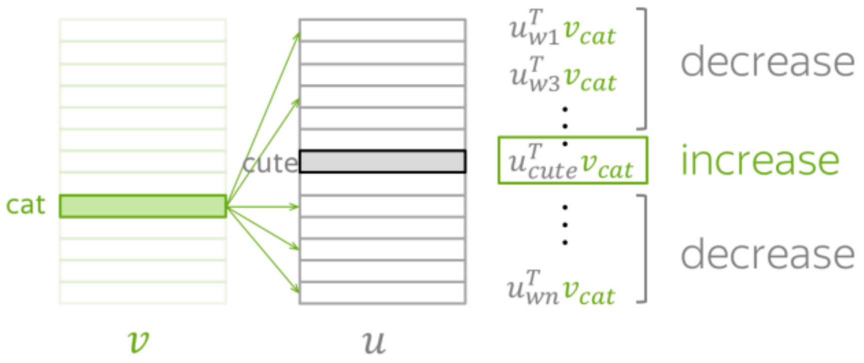
5. evaluate the gradient, make an update

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

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

Let's do better: Negative Sampling

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

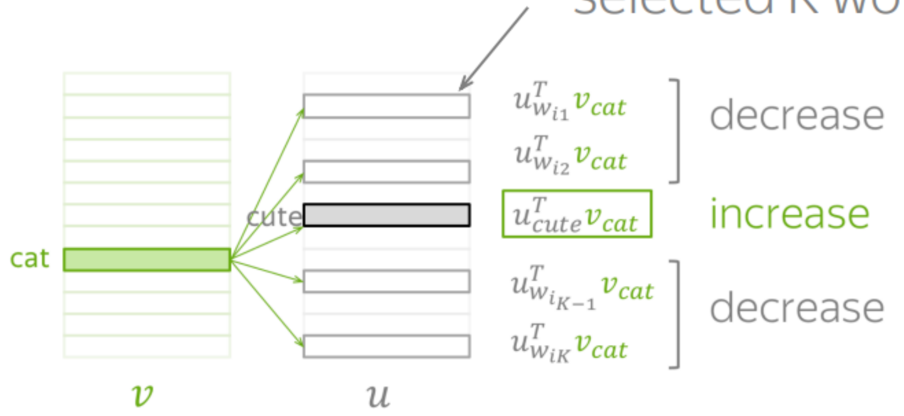


Parameters to be updated: bad
- v_{cat}
- u_w for all w in the vocabulary

|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: good
- v_{cat}
- u_{cute} and u_w for w in K negative examples

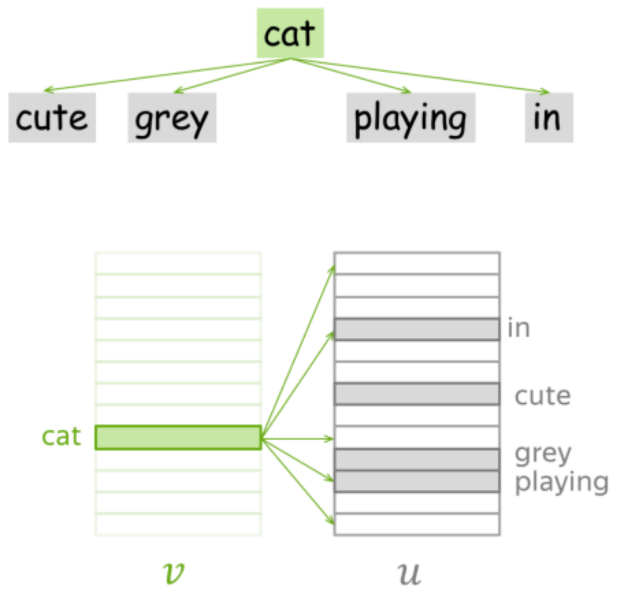
K + 2 vectors

https://lena-voita.github.io/nlp_course

Word2Vec Variants: Skip-Gram and CBOW

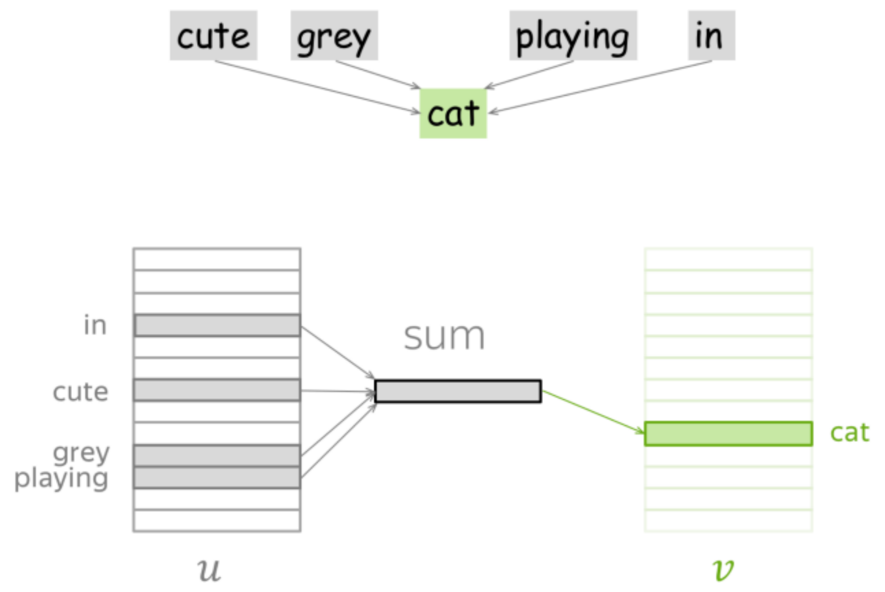
… I saw a cute grey cat playing in the garden …

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



(this is what we did so far)

CBOW: from sum of context predict central



(Continuous Bag of Words)

https://lena-voita.github.io/nlp_course

To reed:

- Distributed Representations of Words and Phrases and their Compositionality. Paper
- Efficient Estimation of Word Representations in Vector Space. Paper
- word2vec Parameter Learning Explained. Paper
- Habr: Russian translation of Jay Alammar blogpost

Relation to PMI Matrix Factorization

Related Papers

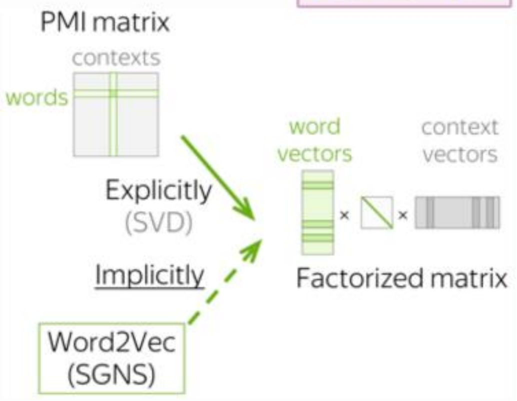
Good Old Classics

NeurIPS 2014

Neural Word Embedding as Implicit Matrix Factorization

Omer Levy, Yoav Goldberg

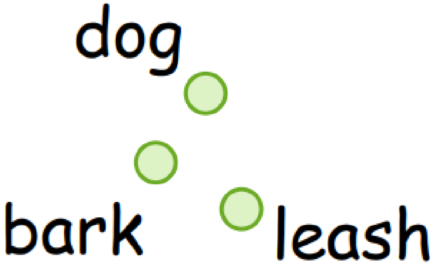
Theoretically, Word2Vec is not so different from matrix factorization approaches! Skip-gram with negative-sampling (SGNS) implicitly factorizes the shifted pointwise mutual information (PMI) matrix: $PMI(w, c) - \log k$, where k is the number of negative examples in negative sampling.



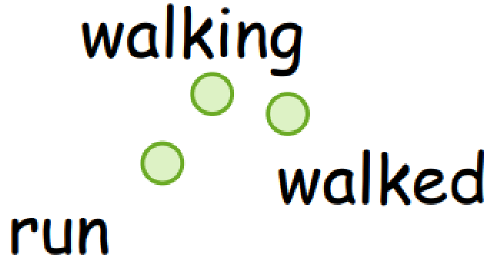
▶ More details

The Effect of Context Window Size

- Larger windows – more topical similarities

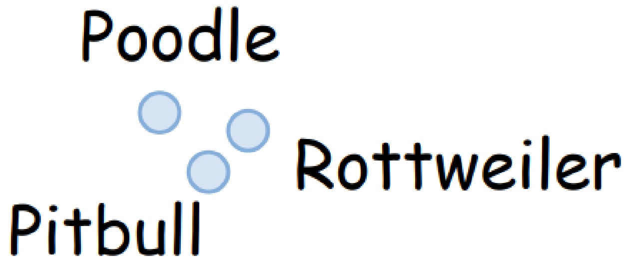


(grouped together)

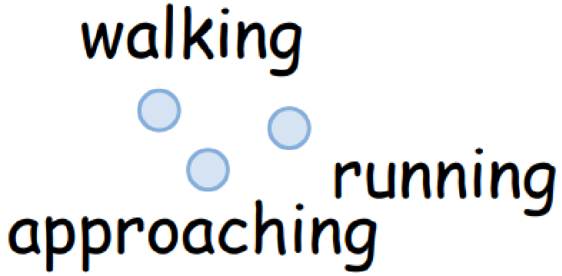


(grouped together)

- Smaller windows – more functional and syntactic similarities



(grouped together)



(grouped together)

https://lena-voita.github.io/nlp_course

GloVe: Global Vectors for Word Representation

	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

GloVe: Global Vectors for Word Representation

context vector

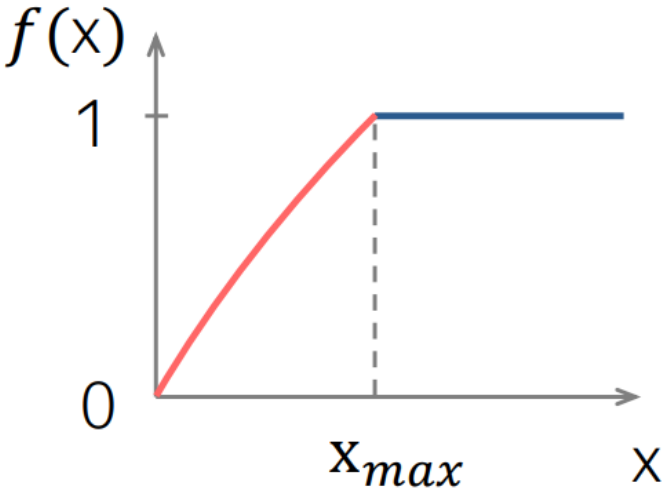
word vector

bias terms (also learned)

$$J(\theta) = \sum_{w,c \in V} \underbrace{f(\mathrm{N}(w, c))} \cdot (u_c^T v_w + b_c + \overline{b_w} - \log \mathrm{N}(w, c))^2$$

Weighting function to:

- penalize rare events
- not to over-weight frequent events



$$\begin{cases} (x/x_{max})^{\alpha} & \text{if } x < x_{max}, \\ 1 & \text{otherwise.} \end{cases}$$

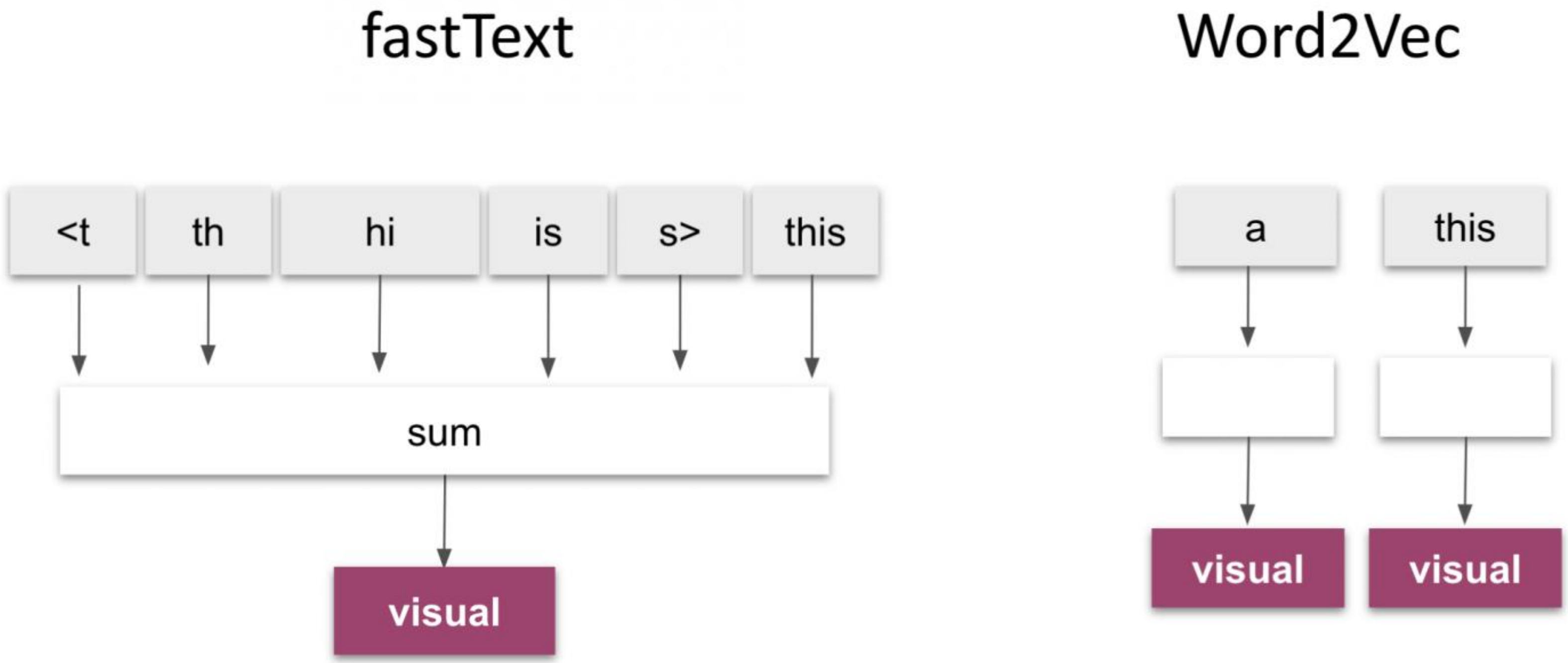
$\alpha = 0.75$, $x_{max} = 100$

https://lena-voita.github.io/nlp_course

FastText

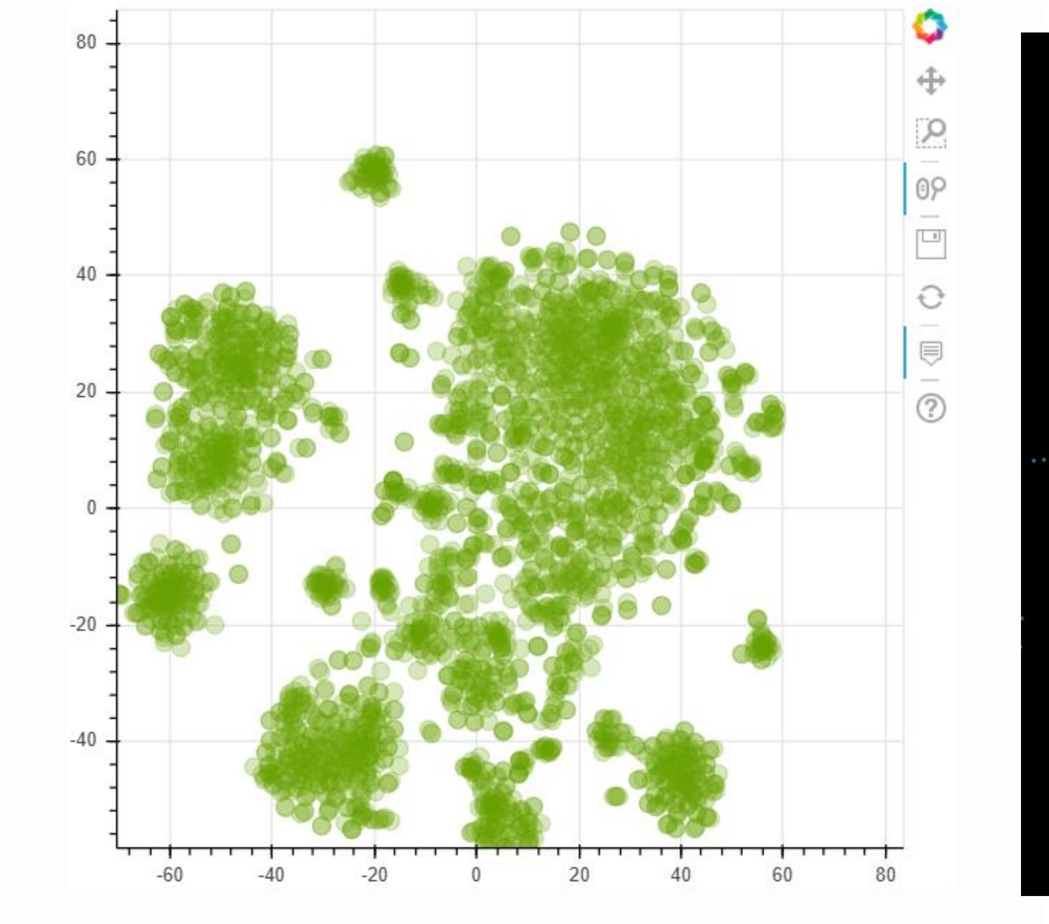
Let’s analyse word as a set of symbolic n-grams

- Effective work with rare words and OOV
- Consideration of morphology and internal structure of words

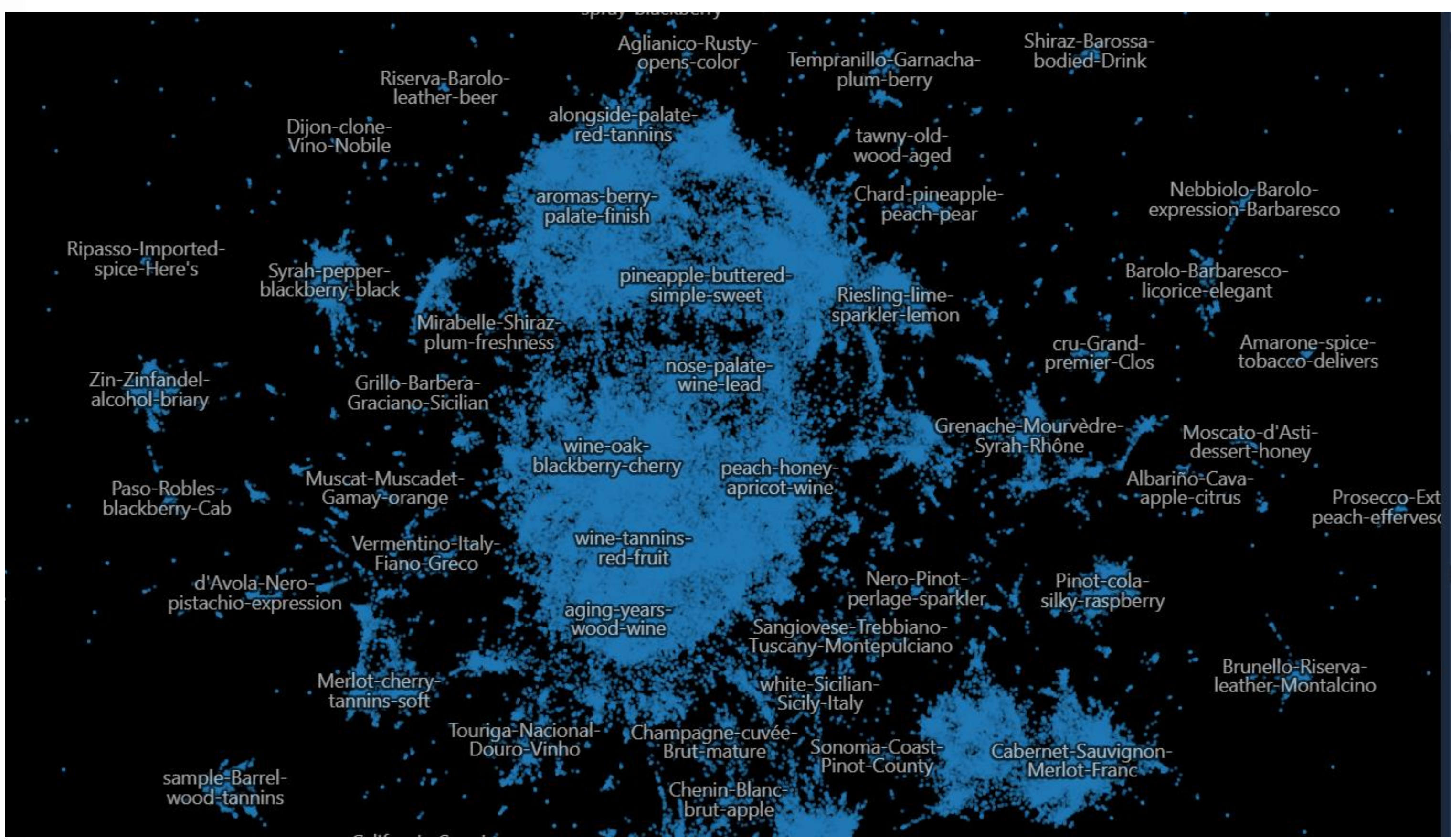


That's all! Thank you for your attention!

Visualization1



Visualization2



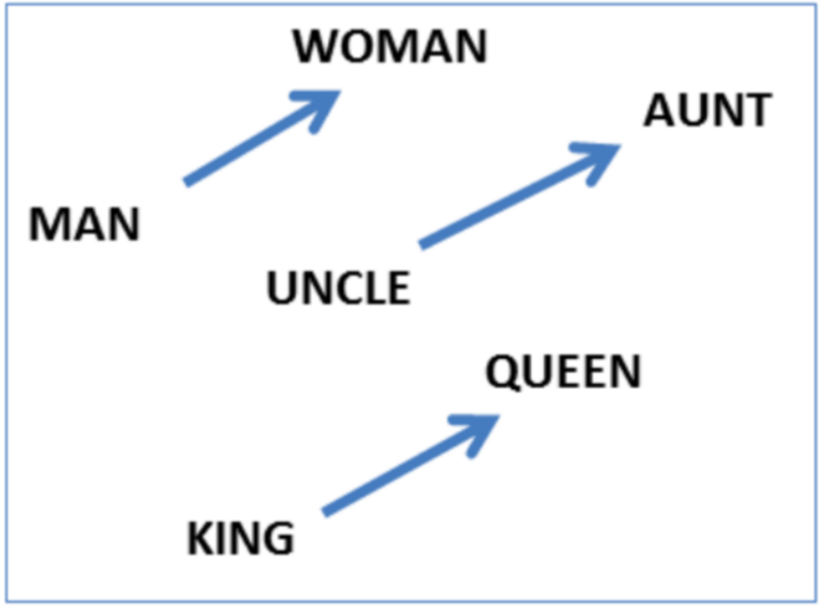
Word Similarity Benchmarks

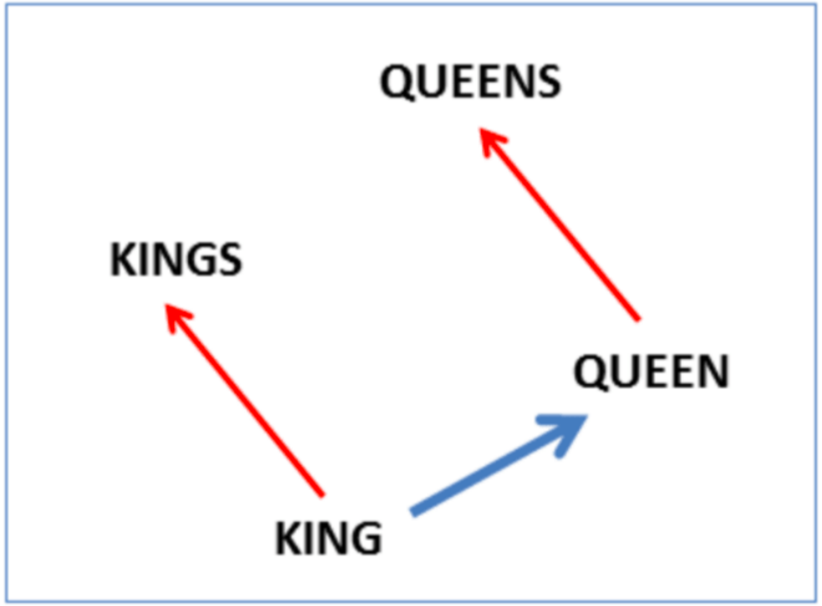
word pair		score
vulgarism	profanity	9.62
subdividing	separate	8.67
friendships	brotherhood	7.5
exceedance	probability	5.0
assigned	allow	3.5
marginalize	interact	2.5
misleading	beat	1.25
radiators	beginning	0

Linear Structure

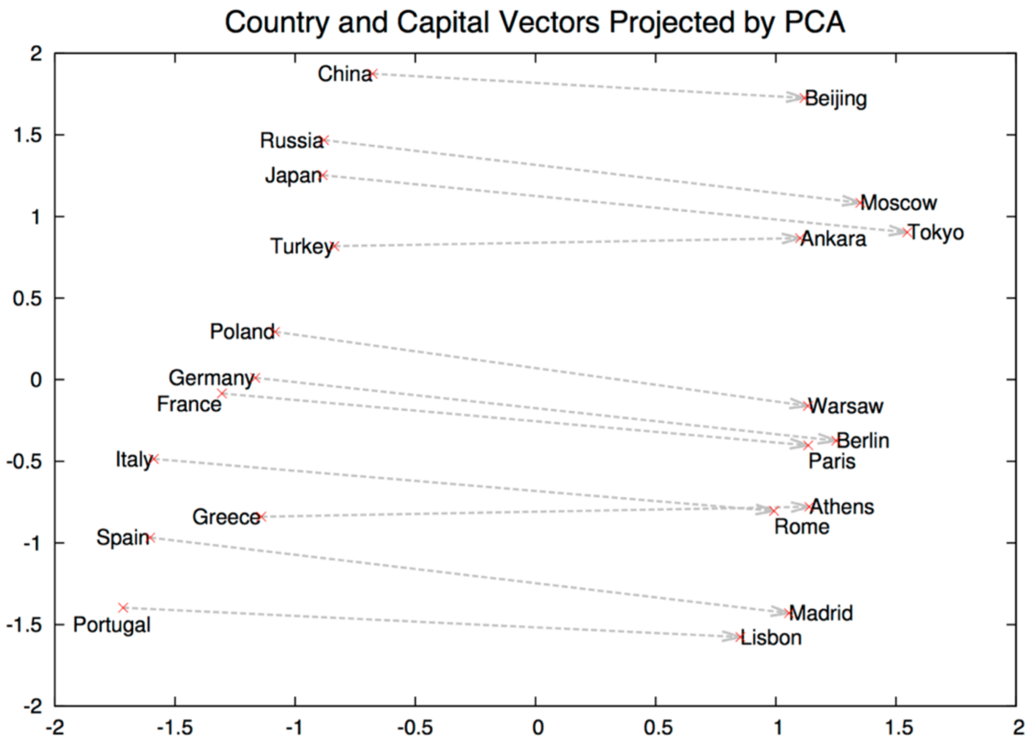
semantic: v(**king**) – v(**man**) + v(**woman**) ≈ v(**queen**)

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

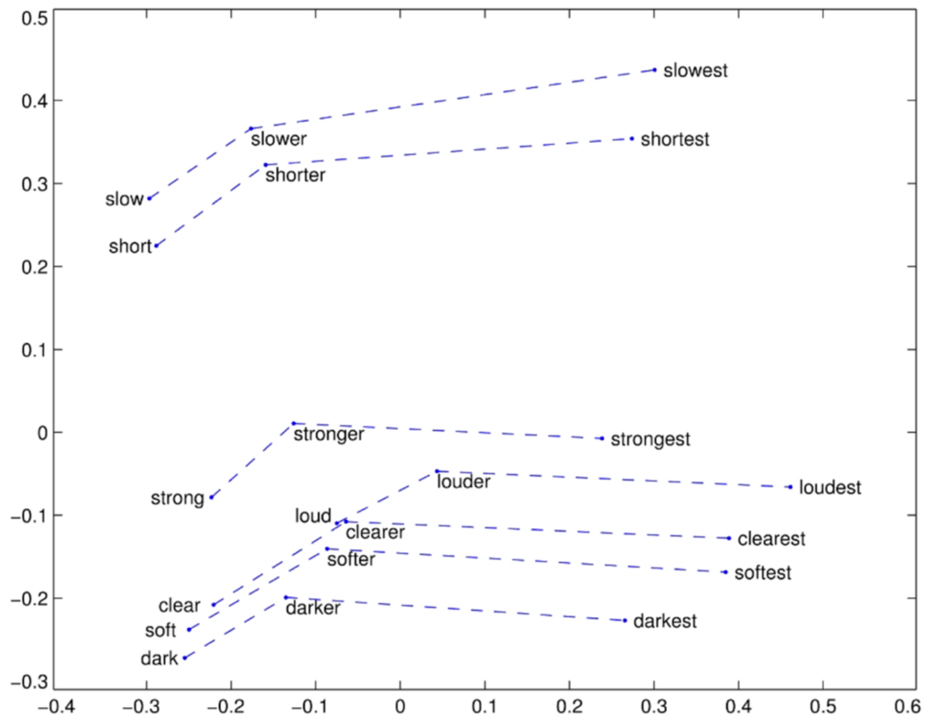




Linear Structure



Word2Vec



GloVe

Similarities Across Languages

The recipe for building large dictionaries from small ones

Ingredients:

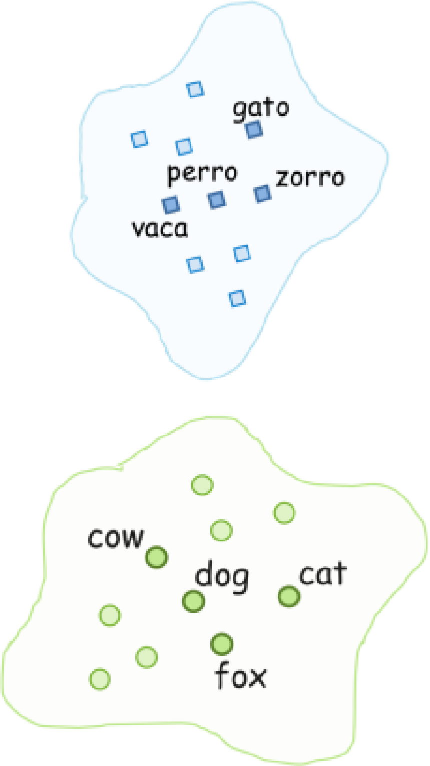
- corpus in one language (e.g., English)
- corpus in another language (e.g., Spanish)

- very small dictionary

  cat ↔ gato
  cow ↔ vaca
  dog ↔ perro
  fox ↔ zorro
  ...

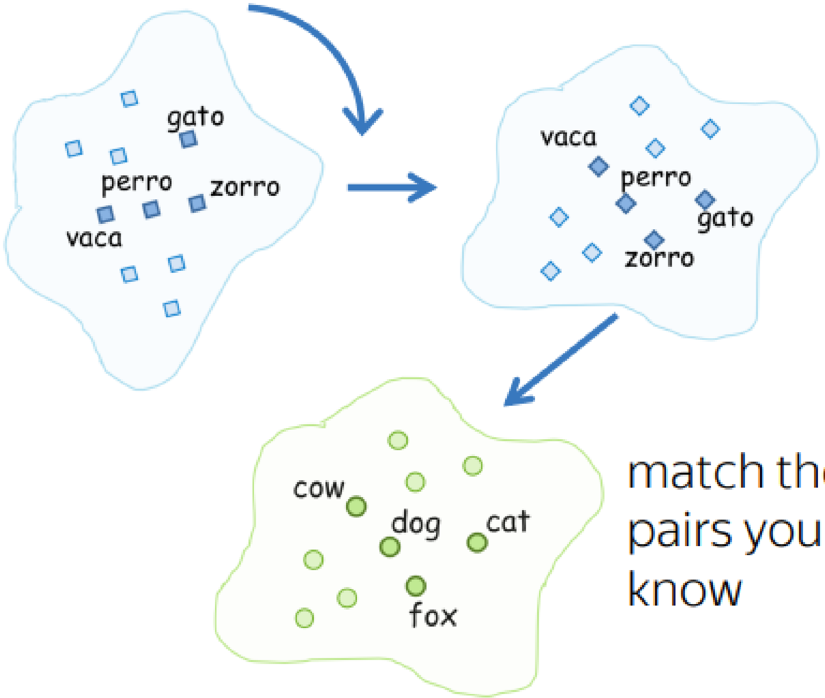
Step 1:

- train embeddings for each language



Step 2:

- linearly map one embeddings to the other to match words from the dictionary



match the pairs you know

Similarities Across Languages

The recipe for building large dictionaries from small ones

Ingredients:

- corpus in one language (e.g., English)
- corpus in another language (e.g., Spanish)
- very small dictionary

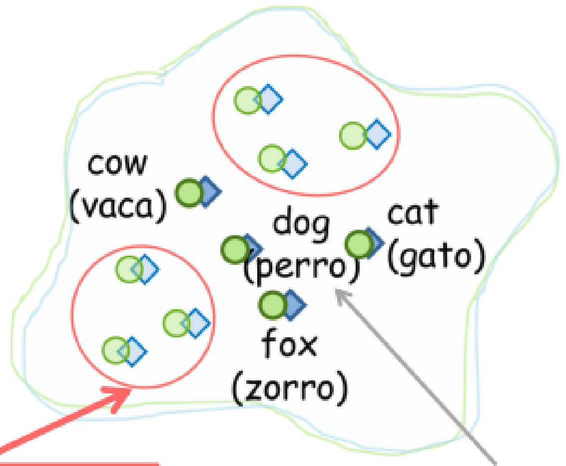
cat ↔ gato
cow ↔ vaca
dog ↔ perro
fox ↔ zorro
...

Steps1-2:

- match words from the vocabulary

Step 3:

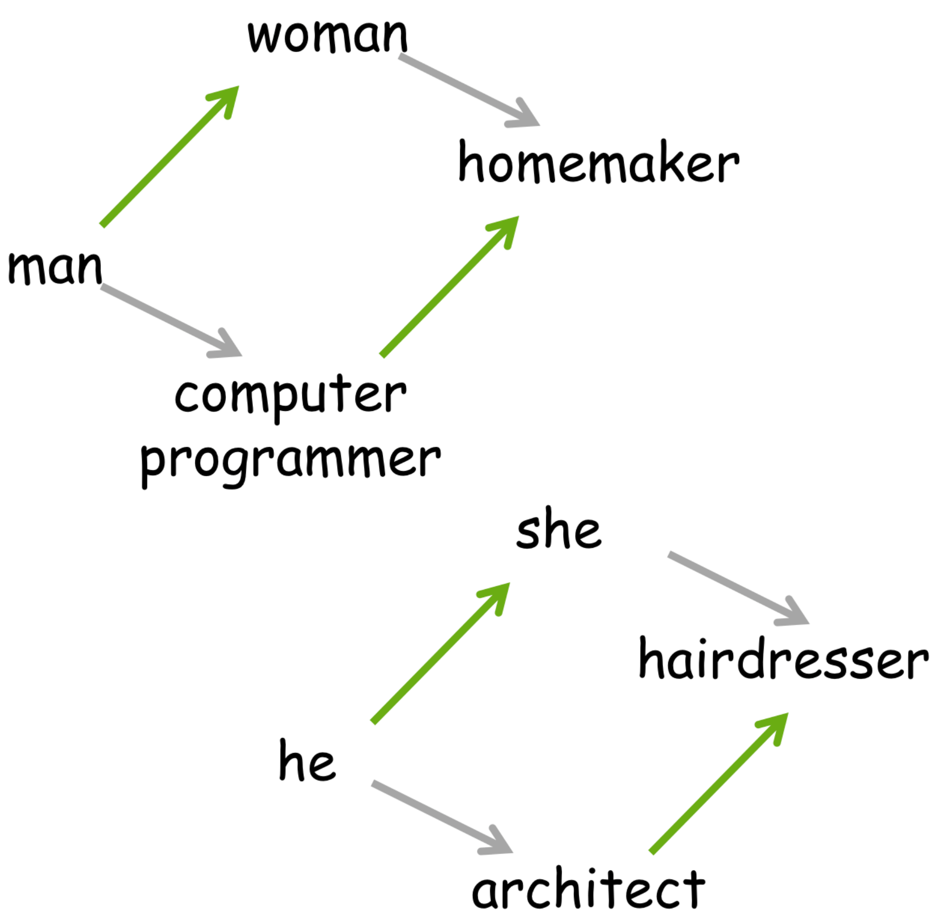
- after matching the two spaces, get new pairs from the new matches



new translations – you learned them!

old translations – the ones you knew

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings



Gender stereotype **she-he** analogies

sewing-carpentry	registered nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	lovely-brilliant

Gender appropriate **she-he** analogies

queen-king	ovarian cancer-prostate cancer	sister-brother
waitress-waiter	convent-monastery	mother-father