# AOS2– Deep learning

## Lecture 03: Word embeddings

Sylvain Rousseau

# Natural language processing

- What is natural language processing (NLP)?
  - Field of artificial intelligence
  - Read and understand human languages
- NLP is hard
  - Homographs like the word “set”
  - “I made her duck” (at least 5 different meanings)
  - No strict rules: “How R U?”

# NLP tasks

Some NLP subtasks:

- Part-of-speech (POS) tagging
  label tokens in a text based on their role: nouns, verbs, adjectives, adverbs, punctuation…
- Named Entity Recognition (NER)
  identifying specific words or phrases (“entities”) and categorizing them as persons, locations…
- Topic modelling
  identify topics in a text
- Sentiment analysis
  interpret and classify emotions in subjective data.
- …

Most of them rely on good representation of words !

# Word vector representation

# Notations

- A **word** as a symbol is denoted $w$ or $w_t$ if it is part of a sequence
- A **word vector** is in bold, $\boldsymbol{w} \in \mathbb{R}^N$

$$\boldsymbol{w} = (-0.3320, 0.3401, \ldots, -1.6941)$$

- **Vocabulary:** list of known words of length $V$

$$\mathcal{V} = \{w_1, \ldots, w_V\}$$

- **Embedding:** a vector for each word in the vocabulary

# One-hot representation

- To each word $w_k$ of the vocabulary $\mathcal{V}$, we associate the vector $\boldsymbol{w}_k \in \mathbb{R}^V$ such that

$$\boldsymbol{w}_k = (0, 0, \ldots, 0, \overset{i(k)}{1}, 0, \ldots, 0) \qquad \boldsymbol{w}_{kj} = \begin{cases} 0 & \text{if } j \neq i(k) \\ 1 & \text{if } j = i(k) \end{cases}$$

- Properties
  - A text can be represented of a matrix of size $V \times L$
  - Extremely sparse representation
- Drawbacks:
  - Very inefficient representation as $V$ is typically large $V \sim 10^{5-7}$
  - All words are equidistant ($\left\|\boldsymbol{w}_i - \boldsymbol{w}_j\right\| = \delta_{ij}\sqrt{2}$)
  - Impossible to add a new word without changing the dimension of the representation

# Word embedding

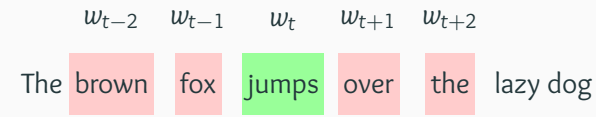
- Starts with the **distributional hypothesis** stating that words in similar contexts have similar meanings. Captured in a famous quote by Firth:

  *"You shall know a word by the company it keeps", Firth 1957*

- Count-based approaches (counting co-occurrences, frequencies of words)
  - Co-occurrence + SVD
  - GloVe — Pennington, Socher, and Manning 2014
- Prediction-based approaches (optimizating a loss function)
  - Word2vec — Mikolov et al. 2013
  - FastText — Bojanowski et al. 2017
  - ELMO — Peters et al. 2018
  - Bert — Devlin et al. 2019
  - …

# Window-based context

- The context of a word is defined as a window of specific size around that word
- Window of size 5 center at “jumps”




- Notations (window of size $2m + 1$)
  - **Center word:** $w_t$
  - **Context:** $w_{t-m}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+m}$

# Co-occurrence matrix

- Co-occurrence of center word and words in its context
- Example corpus composed of 3 sentences:
  - I like deep learning .
  - I like NLP .
  - I enjoy flying .
- Vocabulary (8 tokens)

$$\mathcal{V} = \left\{ \text{I}, \text{like}, \text{enjoy}, \text{deep}, \text{learning}, \text{NLP}, \text{flying}, . \right\}$$

# Example

Window of size 3: $m = 1$

| | I | like | enjoy | deep | learning | NLP | flying | . |
|---|---|---|---|---|---|---|---|---|
| I | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 |
| like | 2 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| enjoy | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| deep | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| learning | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| NLP | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| flying | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| . | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

# Properties

- Properties
  - Similar words have similar representation
  - Co-occurrence matrix is symmetric
  - Co-occurrence matrix is sparse
- Drawbacks:
  - Static representation
  - Co-occurrence matrix is huge: size of $C$ is $V \times V$

# Dimension reduction

We want to reduce the number of columns of $C$ and retain most of the information: PCA!

- To perform a PCA we do a SVD decomposition of $C$

$$C = USU^T$$

  where $U^T U = I$, ($U = V$ because $C$ is symmetric)
- Instead of $C$ we choose $N$ first columns of $US$

$$\tilde{C} = (US)_{[1:N]}$$

  where $N$ is typically 25–1000.
- $\tilde{C}$ is of size $V \times N$ instead of $V \times V$
- $\tilde{C}$ is the best rank-$N$ matrix preserving correlations ($CC^T$)

# word2vec

# Prediction-based approach

Principle: learn to model words through a prediction task

- **Word2vec** from Mikolov et al. 2013
  - Simplest prediction-based approach
  - Two different architectures
    - Continuous Bag Of Words (CBOW)
    - Skip-gram
  - Variants:
    - Negative sampling
    - Hierarchical softmax

# Continuous Bag of Words (CBOW)

- Principle: model the distribution of **center word conditional to its context**

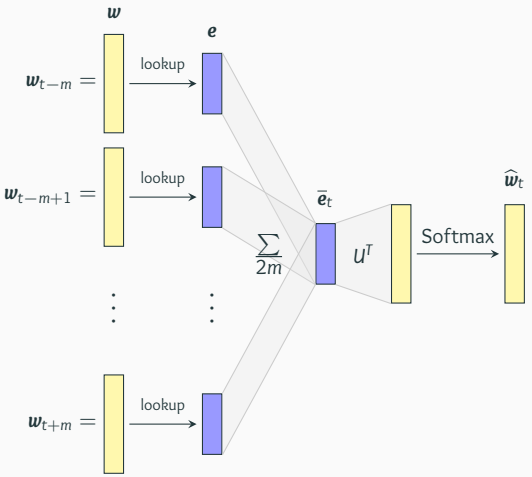
$$p(w \mid w_{t-m}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+m})$$

  Or $p(w \mid \mathcal{C})$ with $\mathcal{C} = \{w_{t-m}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+m}\}$

- Idea: use a neural network to learn that conditional distribution given given one-hot encoded context words

# CBOW model

- Use a neural network to model $p(w \mid C)$:
  - Inputs are one-hot encoded context words: $\boldsymbol{w}_{t+k}$, $k \in \{-m, \ldots, m\}$, $k \neq 0$
  - First (shallow) layer: simple lookup
  - First real layer: averaging
  - Second layer: linear transform
  - Output layer: softmax to get a distribution on the vocabulary: $\widehat{\boldsymbol{w}}_t$
- Parameters
  - Embeddings: $\boldsymbol{e}_{t+k}$ for $k \in \{-m, \ldots, m\}$, $k \neq 0$
  - Matrix: $U$



# Layers

- First (shallow) layer: lookup table $E = [\boldsymbol{e}_1, \ldots, \boldsymbol{e}_V]$, (size of $E$ is $N \times V$)

$$\boldsymbol{e}_{t-k} = E\boldsymbol{w}_{t-k}$$

- First real layer: averaging

$$\overline{\boldsymbol{e}}_t = \frac{1}{2m} \sum_{\substack{k=-m \\ k \neq 0}}^{m} \boldsymbol{e}_{t+k}$$

- Second layer: linear transform

$$U^T \overline{\boldsymbol{e}}_t$$

- Third layer: softmax

$$\widehat{\boldsymbol{w}}_t = \text{softmax}\left(U^T \overline{\boldsymbol{e}}_t\right)$$

# Expression

- Relationship between one-hot encoded context words and "one-hot" prediction of expected word

$$\widehat{\boldsymbol{w}}_t = \text{softmax}\left(U^T \cdot \left(\frac{1}{2m}\sum_{k=-m}^{m} E\boldsymbol{w}_{t+k}\right)\right)$$

- Matrix $E$ is the embedding matrix: $\boldsymbol{e}_i = E\boldsymbol{w}_i$

$$\widehat{\boldsymbol{w}}_t = \text{softmax}\left(U^T \cdot \left(\frac{1}{2m}\sum_{k=-m}^{m} \boldsymbol{e}_{t+k}\right)\right), \qquad \widehat{\boldsymbol{w}}_t = \text{softmax}\left(U^T\overline{\boldsymbol{e}}_t\right)$$

# Expanded form

- Softmax is done on values:

$$\langle \mathbf{u}_i, \overline{\boldsymbol{e}_t} \rangle \quad \text{for} \quad i = 1, \ldots, V$$

  where $U = [\mathbf{u}_1, \ldots, \mathbf{u}_n]$

- Per-word expanded version (probability that expected word $w_t$ is $w_i$ )

$$\widehat{\boldsymbol{w}}_{ti} = \frac{\exp\left(\langle \mathbf{u}_i, \overline{\boldsymbol{e}}_t \rangle\right)}{\sum_{k=1}^{V} \exp\left(\langle \mathbf{u}_k, \overline{\boldsymbol{e}}_t \rangle\right)}$$

# Loss

- Compare predicted distribution $\widehat{\boldsymbol{w}}_t$ over vocabulary with one-hot encoded expected center word $\boldsymbol{w}_t$ with cross-entropy:

$$H(\boldsymbol{w}_t, \widehat{\boldsymbol{w}}_t) = -\sum_{j=1}^{V} \boldsymbol{w}_{tj} \log \widehat{\boldsymbol{w}}_{tj}$$

- Since $\boldsymbol{w}_t$ is a one-hot vector, it reduces to

$$H(\boldsymbol{w}_t, \widehat{\boldsymbol{w}}_t) = -\log \widehat{\boldsymbol{w}}_{t,i(t)} \qquad (i(t) \text{ is the one-hot index of token } w_t)$$

- Final loss is

$$H(\boldsymbol{w}_t, \widehat{\boldsymbol{w}}_t) = -\left\langle \mathbf{u}_{i(t)}, \overline{\boldsymbol{e}}_t \right\rangle + \log \left( \sum_{k=1}^{V} \exp\left( \langle \mathbf{u}_k, \overline{\boldsymbol{e}}_t \rangle \right) \right)$$

- Parameters to optimize are: $\boldsymbol{e}_{t+k}$ for $k \in \{-m, \ldots, m\}$, $k \neq 0$, $\mathbf{u}_k$ for $k \in V$

# Skip-gram

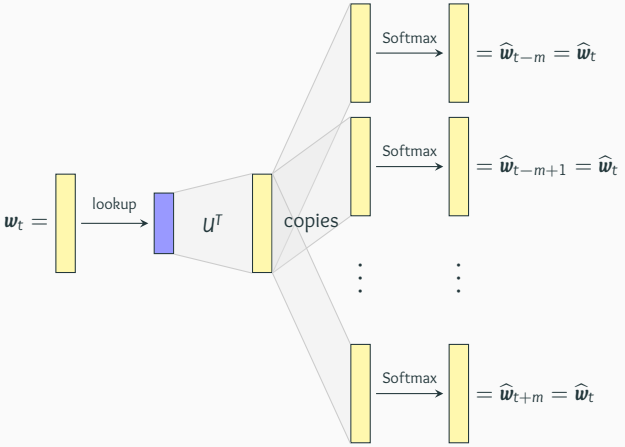
- Principle: model the distribution of **context words conditional to its center**

$$p(w_{t-m}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+m} \mid w_t)$$

- Idea: use a neural network to learn that conditional distribution given given one-hot encoded context words

# Skip-gram model

- Reverse of CBOW model
  - Reverse arrows
  - Replace averaging by copies
  - Swap softmax/lookup
- First layer: center word lookup
- Second layer: linear transform
- Third (shallow layer): copies
- Third real layer: softmax to get distributions over $\mathcal{V}$
- Parameters
  - Embeddings: $\boldsymbol{e}_{t+k}$ for $k \in \{-m, \ldots, m\}, k \neq 0$
  - Matrix: $U$



# Layers

- First layer: center word lookup

$$\boldsymbol{e}_t = E\boldsymbol{w}_t$$

- Second layer: linear transform

$$U^T\boldsymbol{e}_t$$

- Third (shallow layer): copies
- Third real layer: softmax to get distributions over $\mathcal{V}$ (which are all the same)

$$\widehat{\boldsymbol{w}}_t = \text{softmax}\left(U^T\boldsymbol{e}_t\right)$$

# Expression

- Probability of word of index $i$ in $V$ to appear in the context of $w_t$

$$\widehat{\boldsymbol{w}}_{ti} = \frac{\exp\left(\langle \mathbf{u}_i, \boldsymbol{e}_t \rangle\right)}{\sum_{j=1}^{V} \exp\left(\langle \mathbf{u}_j, \boldsymbol{e}_t \rangle\right)}$$

- For context word $w_k$, corresponding probability is $\widehat{\boldsymbol{w}}_{t,i(k)}$

$$\widehat{\boldsymbol{w}}_{t,i(k)} = \frac{\exp\left(\langle \mathbf{u}_{i(k)}, \boldsymbol{e}_t \rangle\right)}{\sum_{j=1}^{V} \exp\left(\langle \mathbf{u}_j, \boldsymbol{e}_t \rangle\right)}$$

  where $i(k)$ is the index of token $w_k$ in vocabulary $\mathcal{V}$

# Loss

- Sum of cross-entropies on each output

$$\mathcal{L} = \sum_{\substack{k=t-m \\ k \neq t}}^{t+m} H(\boldsymbol{w}_{i(k)}, \widehat{\boldsymbol{w}}_t)$$

- For word $w_k$, we have

$$\begin{aligned} H(\boldsymbol{w}_{i(k)}, \widehat{\boldsymbol{w}}_t) &= \sum_{j=1}^{V} \boldsymbol{w}_{i(k)j} \log \widehat{\boldsymbol{w}}_{tj} \\ &= \log \widehat{\boldsymbol{w}}_{ti(k)} \qquad (\boldsymbol{w}_{i(k)} \text{ is the one-hot vector}) \end{aligned}$$

- Final loss is

$$\mathcal{L} = \sum_{\substack{k=t-m \\ k \neq t}}^{t+m} H(\boldsymbol{w}_k, \widehat{\boldsymbol{w}}_k) = -\sum_{\substack{k=t-m \\ k \neq t}}^{t+m} \log \widehat{\boldsymbol{w}}_{ti(k)}$$

# Loss

- Expanded loss is

$$\mathcal{L} = -\sum_{\substack{k=t-m \\ k\neq t}}^{t+m} \left\langle \mathbf{u}_{i(k)}, \boldsymbol{e}_t \right\rangle + 2m \log \sum_{j=1}^{V} \exp\left(\left\langle \mathbf{u}_j, \boldsymbol{e}_t \right\rangle\right)$$

- Parameters to optimize are: $\boldsymbol{e}_t$ and $\mathbf{u}_j$ for $j = 1, \ldots, V$

# Cosine similarity

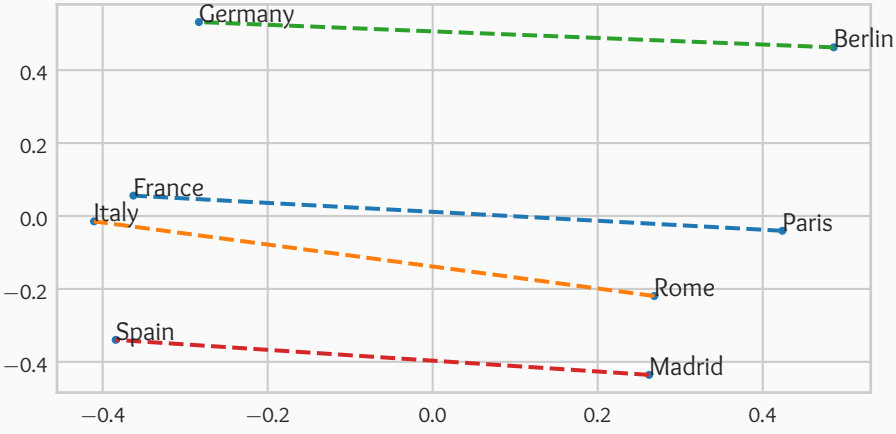
- Cosine similarity
$$S = \frac{\langle \boldsymbol{w}_i, \boldsymbol{w}_j \rangle}{\|\boldsymbol{w}_i\| \, \|\boldsymbol{w}_j\|} = \cos\theta$$
- $S \in (-1, 1)$
- Orthogonal if $S = 0$
- Same orientation if $S = 1$ ($\boldsymbol{w}_i = \lambda \boldsymbol{w}_j$, $\lambda \geqslant 0$)
- Two random tokens are $\approx$ orthogonal
$$\mathbb{E}(S) = 0$$
- Non-zero is significant
$$\mathrm{Var}(S) = \mathcal{O}(1/N)$$

- Maximum cosine similarity

| car | score | cat | score |
|---|---|---|---|
| cars | 0.8139 | cats | 0.8064 |
| automobile | 0.7040 | dog | 0.7927 |
| truck | 0.6975 | kitten | 0.7527 |
| vehicle | 0.6854 | rabbit | 0.7517 |
| roadster | 0.6779 | pet | 0.7104 |
| SUV | 0.6734 | fox | 0.7061 |
| motorbike | 0.6674 | feline | 0.6898 |
| motorcycle | 0.6509 | kittens | 0.6837 |
| motorcar | 0.6486 | puppy | 0.6721 |
| driver | 0.6452 | tomcat | 0.6718 |

# Capitals and countries

PCA in 2D



# Analogies

Concept of “capital of” as “Madrid” - “Spain”

- Most similar tokens to:

  (“Madrid” - “Spain”) + “France”

  which translates to

  <capital of> “France”

- Same thing for

  (“Madrid” - “Spain”) + “Germany”

| (“Madrid” - “Spain”) + “France” | score |
|---|---|
| Paris | 0.7620 |
| Lille | 0.7085 |
| Marseille | 0.6987 |
| Strasbourg | 0.6825 |
| France | 0.6809 |

| (“Madrid” - “Spain”) + “Germany” | score |
|---|---|
| Munich | 0.7471 |
| Stuttgart | 0.7351 |
| Berlin | 0.7149 |
| Dresden | 0.7014 |
| Germany | 0.6990 |

# Additive compositionality

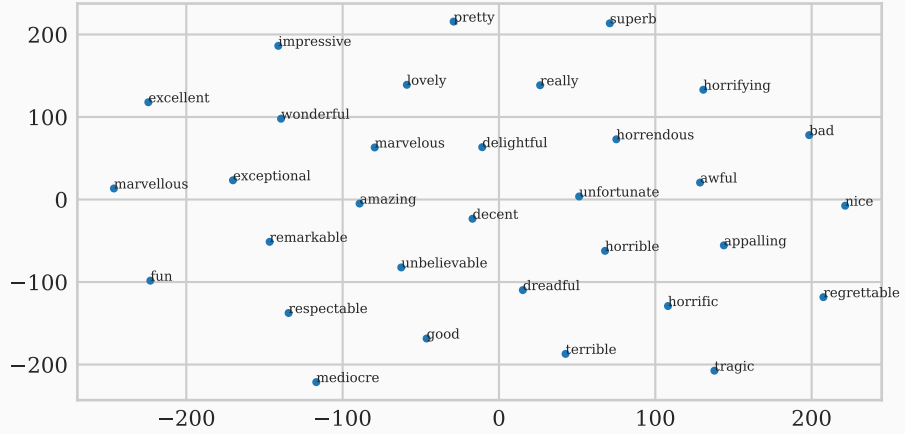
- 10 closest token of a sum of embeddings

| Czech + currency | score |
|---|---|
| Czech | 0.7478 |
| currency | 0.7478 |
| koruna | 0.6906 |
| currencies | 0.6406 |
| Slovak | 0.6321 |
| Slovakia | 0.6178 |
| forint | 0.6016 |
| kroon | 0.5836 |
| króna | 0.5812 |
| markka | 0.5798 |

| German + airlines | score |
|---|---|
| airlines | 0.7295 |
| German | 0.7295 |
| Interflug | 0.7184 |
| airline | 0.6487 |
| TAROM | 0.6281 |
| TUIfly | 0.6279 |
| Eurowings | 0.6246 |
| Crossair | 0.6223 |
| Cargolux | 0.6222 |
| Germanwings | 0.6167 |

# Word2vec: illustrations

Depth 2 and 5—neighborhood of “good” and “bad” cosine similarity, t–SNE visualization



# Properties

- Semantic information is encoded in a linear way (synonyms, analogy, additive compositionality)
  - Synonyms are close
  - Additive compositionality to simple question answering
  - Concepts as vector for analogy
- Shortcomings
  - Softmax operation is costly (size $V \sim 10^{5-7}$)
  - Don't use token frequencies: very frequent words carry less information than rare ones
  - Same embedding given to opposite words
- Improvements
  - Negative sampling
  - Hierarchical softmax

# Hierarchical softmax

- Skip-gram model: $\widehat{\boldsymbol{w}}_k = \text{softmax}\left(U^T \boldsymbol{e}_t\right)$
- Replace linear transform + softmax layers by a **binary tree structure**

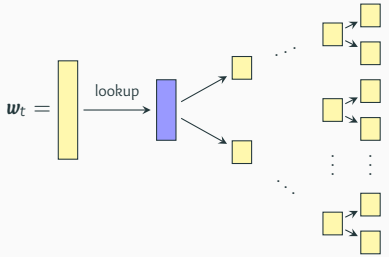
- If $\mathcal{E}$ is the set of pairs of nodes from root to index $i$

$$\widehat{\boldsymbol{w}}_{ki} = \prod_{(n,m)\in\mathcal{E}} \sigma(\varepsilon(m)\,\langle \mathbf{v}_n, \boldsymbol{e}_t \rangle)$$

- $\varepsilon(m)$ encodes "is node left/right"

$$\varepsilon(m) = \begin{cases} -1 & \text{if node } m \text{ is a left child} \\ 1 & \text{otherwise} \end{cases}$$

- $\mathbf{v}_n$ attached to each (non-leaf) node including root



# Negative sampling

- Based on the Skip-gram model but different loss function
- Skip-gram: model $\mathcal{C}$ conditional to $w_t$
- Negative sampling: model the probability that for a token $w$, the token $c$ is in the context ($D = 1$)

$$\Pr\left((w, c) \text{ in corpus}\right) = \Pr\left(D = 1 \mid w, c\right)$$

- Skip-gram model

$$\Pr\left(D = 1 \mid w, c\right) = \sigma(\langle \boldsymbol{e}_w, \mathbf{u}_c \rangle), \quad \Pr\left(D = 0 \mid w, c\right) = \sigma(-\langle \boldsymbol{e}_w, \mathbf{u}_c \rangle)$$

- Loss

$$\mathcal{L} = -\log \sigma(\langle \boldsymbol{e}_w, \mathbf{u}_c \rangle) - \sum_{r \in \mathcal{N}} \log \sigma(-\langle \boldsymbol{e}_w, \tilde{\mathbf{u}}_r \rangle)$$

where $\mathcal{N}$ is a random sample of tokens most likely not part of context

# GloVe

# GloVe

- Principle: Model co-occurence matrix from words representation

$$\log C(w_i, w_j) = \langle \boldsymbol{w}_i, \tilde{\boldsymbol{w}}_j \rangle + b_i + \tilde{b}_j$$

  where $\boldsymbol{w}_i$ and $\tilde{\boldsymbol{w}}_j$ are two distinct word representations, $b_i$ and $\tilde{b}_i$ are biases

- Squared loss that weights all co-occurrences equally

$$J = \sum_{i,j=1}^{V} \left( \langle \boldsymbol{w}_i, \tilde{\boldsymbol{w}}_j \rangle + b_i + \tilde{b}_j - \log C(w_i, w_j) \right)^2$$

  - Rare co-occurrence are given same importance as more frequent one

# Loss

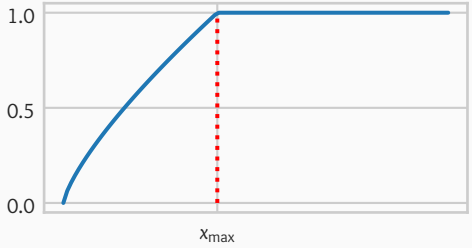
- Weighted squared loss

$$J = \sum_{i,j=1}^{V} f\big(C(w_i, w_j)\big)\Big(\langle \boldsymbol{w}_i, \tilde{\boldsymbol{w}}_j\rangle + b_i + \tilde{b}_j - \log C(w_i, w_j)\Big)^2$$

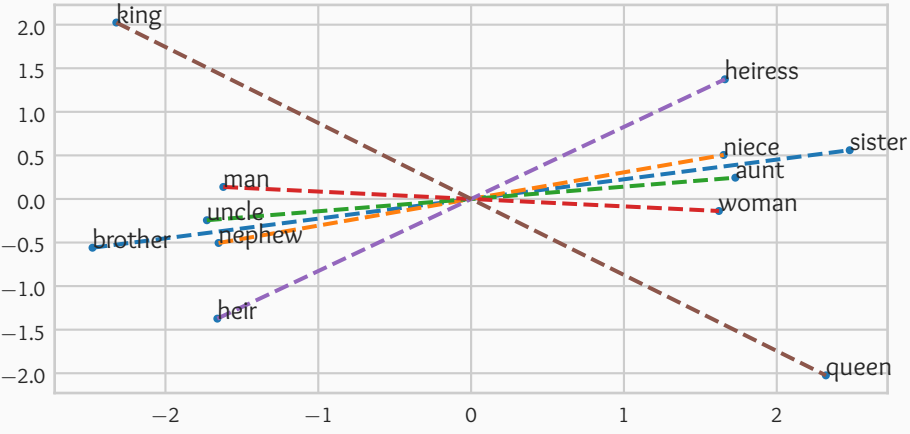
- Weighting function $f$ with $\alpha = 3/4$

- No weight on non-existent words
- Weight rapidly increasing
- Don't put to much weight on frequent words

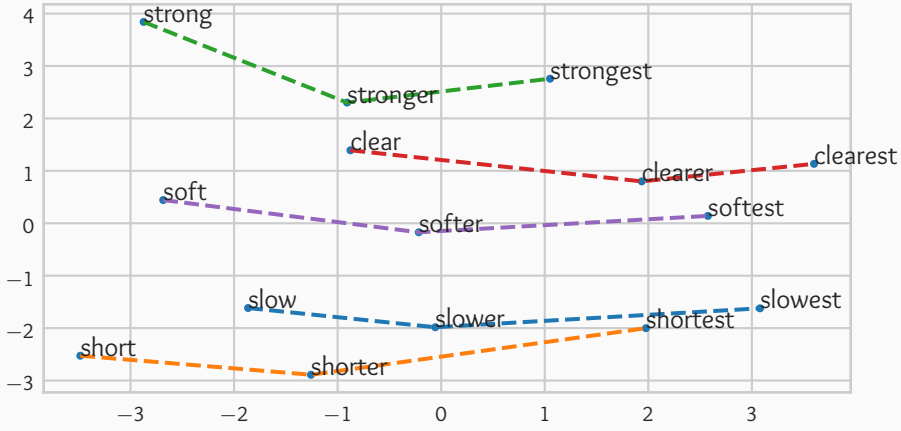
$$f(x) = \begin{cases} \left(\frac{x}{x_{max}}\right)^{\alpha} & \text{if } x < x_{max} \\ 1 & \text{otherwise} \end{cases}$$



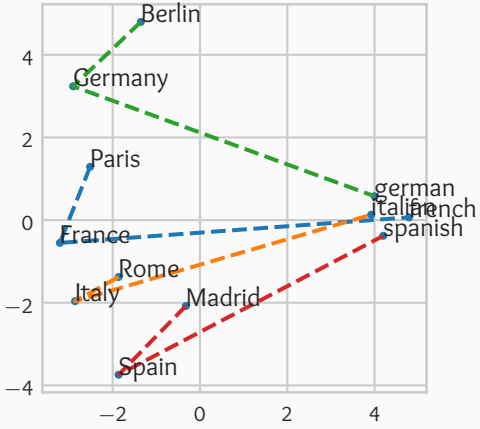
GloVe (recentered)



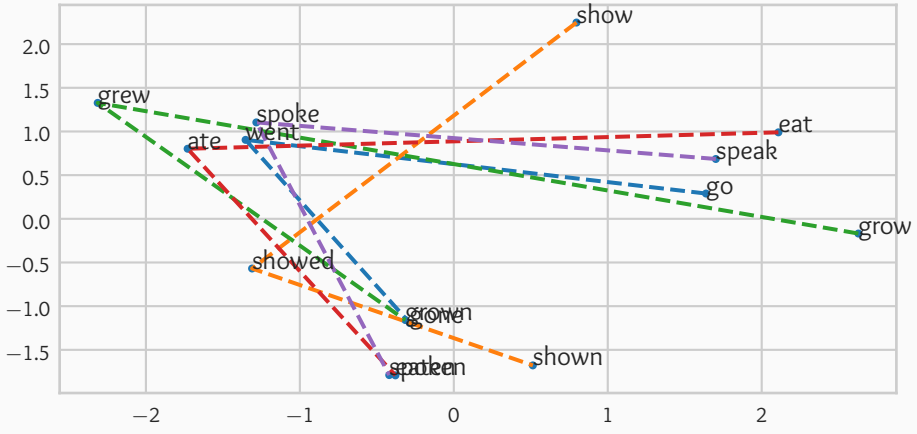
GloVe


strong
stronger
strongest
clear
clearer
clearest
soft
softer
softest
slow
slower
slowest
short
shorter
shortest
4
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3

GloVe


Berlin
Germany
Paris
France
Rome
Italy
Madrid
Spain
german
italian
french
spanish
4
2
0
−2
−4
−2
0
2
4

GloVe (recentered)



# References i


[1] John R. Firth. **“A Synopsis of Linguistic Theory, 1930-1955”.** In: *Studies in linguistic analysis* (1957).

[2] Tomas Mikolov et al. **“Distributed Representations of Words and Phrases and Their Compositionality”.** In: *Advances in Neural Information Processing Systems*. 2013, pp. 3111–3119.

[3] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. **“Glove: Global Vectors for Word Representation”.** In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543.

[4] Piotr Bojanowski et al. **“Enriching Word Vectors with Subword Information”.** In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

# References ii


[5] Matthew E. Peters et al. **“Deep Contextualized Word Representations”.** 2018. arXiv: 1802.05365.

[6] Jacob Devlin et al. **“BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”.** May 24, 2019. arXiv: 1810.04805 [cs]. URL: http://arxiv.org/abs/1810.04805 (visited on 11/30/2021).