Word Representations

Lecture 2 — Natural language Processing

Kenneth Enevoldsen | 2024

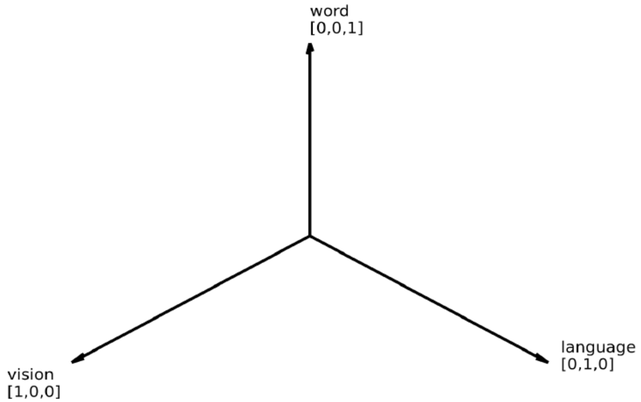
Agenda

- Question: How do we **represent meaning** of a word?
  - How to we **learn** these representation **effectively**
    - Two approaches*
      - Matrix Factorization
      - Window-based approaches
  - How to **compare and use** these embeddings
  - In-depth example:
    - Skip-gram
  - Optional: Perspectives

*GloVe is a mixture of these two approaches

Intuition: What do we want?

- How to **computionally** represent the **meaning** of a word?
  - Immediate intuition: One hot encoding
  - => similar words, dissimilar representations



One-hot encoding

Alternative: Manual construction

- Semantic differential
  - “Rate the genderness of ‘king’ on a scale from 1-10”
- Examples:
  - Affective Norms for English Words (ANEW) dataset
    - Valence, arousal, dominance
  - MRC Psycholinguistic Database
    - imagery, familiarity, concreteness, meaningfulness
- **Problems?**

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)
Gender	−1	1	-0.95	0.97
Royal	0.01	0.02	0.93	0.95
Age	0.03	0.02	0.70	0.69
Food	0.09	0.01	0.02	0.01

Sources & Notes

Hawkins, Del I.; Albaum, Gerald; Best, Roger (1974). "Stapel Scale or Semantic Differential in Marketing Research?". *Journal of Marketing Research*.
Osgood, C. E., May, W. H., and Miron, M. S. (1975). Cross-Cultural Universals of Affective Meaning. Urbana, IL: University of Illinois Press

Co-occurance Patterns

- **Distributional Hypothesis**: “You shall know a word by the company it keeps” — J.R. Firth

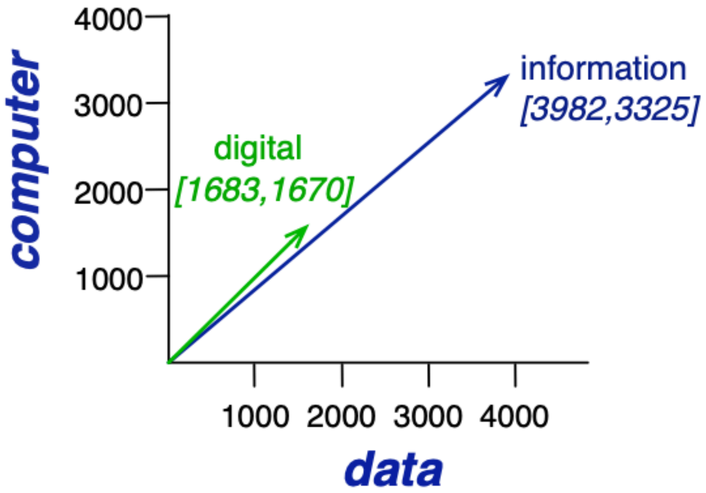
	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Figure 6.6 Co-occurrence vectors for four words in the Wikipedia corpus, showing six of the dimensions (hand-picked for pedagogical purposes). The vector for *digital* is outlined in red. Note that a real vector would have vastly more dimensions and thus be much sparser.

- **What are the problems**

Comparing Vectors

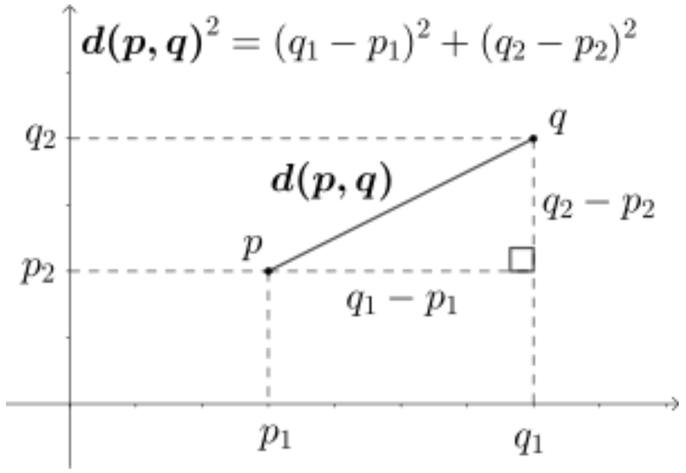
- Euclidian
- Dot product
- Cosine similarity

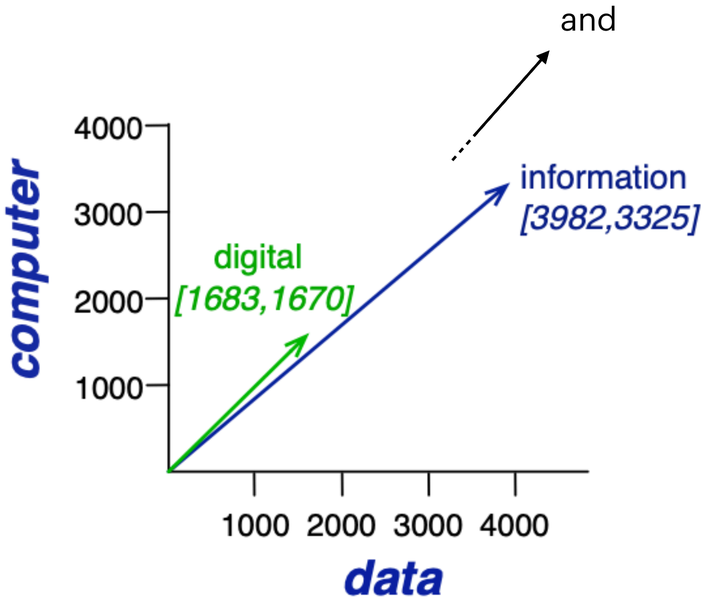


Comparing Vectors

- **Euclidian**
- Dot product
- Cosine similarity

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$



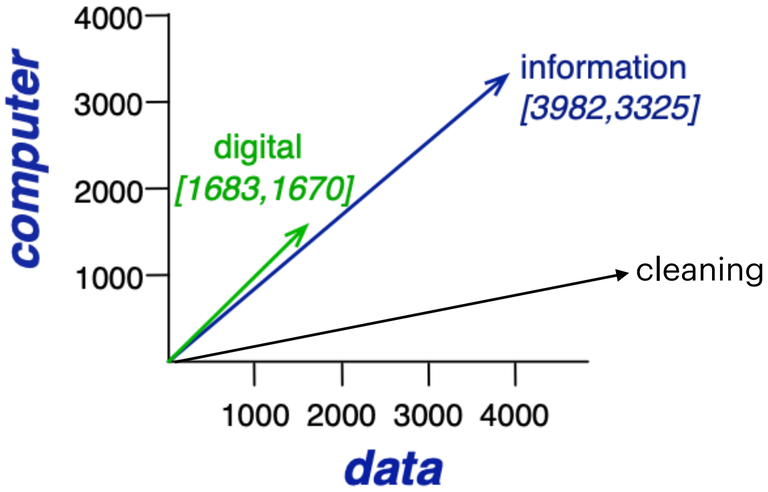


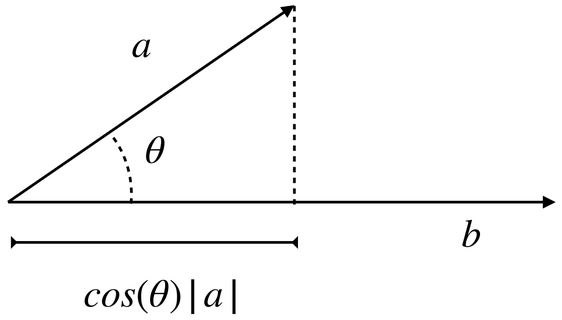
Comparing Vectors

- Euclidian
- **Dot product**
- Cosine similarity

$$a \cdot b = [a_1, a_2][b_1, b_2]^T = a_1 b_1 + a_2 b_2$$

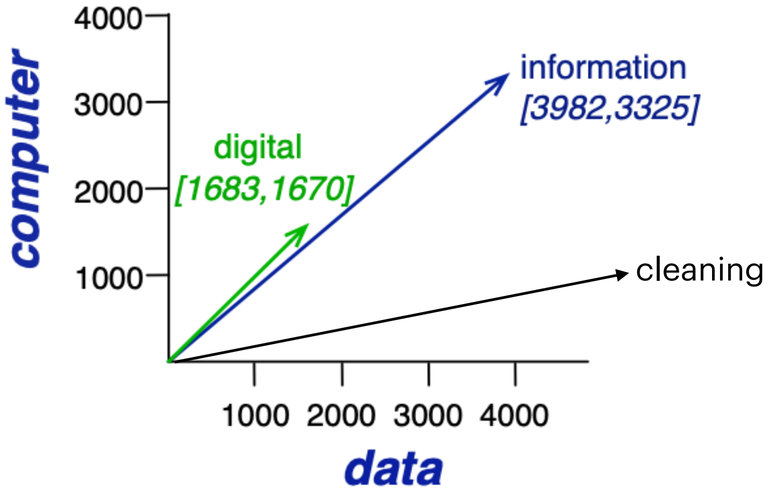
$$= |a||b| cos(\theta)$$





Comparing Vectors

- Euclidian
- Dot product
- **Cosine similarity**
  - Computational trick
    - Attention in transformers
- Information Retrieval and search



Co-occurance Patterns

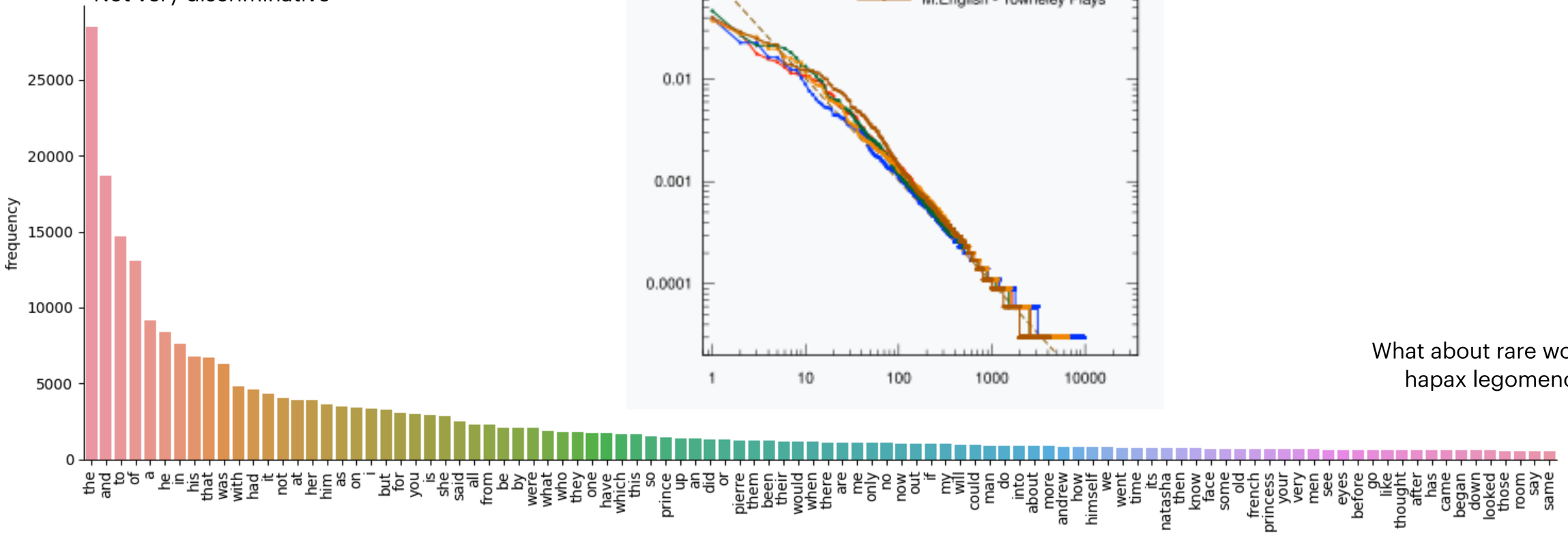
- Global vs Local context

is traditionally followed by	**cherry**	pie, a traditional dessert
often mixed, such as	**strawberry**	rhubarb pie. Apple pie
computer peripherals and personal	**digital**	assistants. These devices usually
a computer. This includes	**information**	available on the internet

- **What are the benefits of each?**

Zipf's Law

Not very discriminative



What about rare words?
hapax legomenons

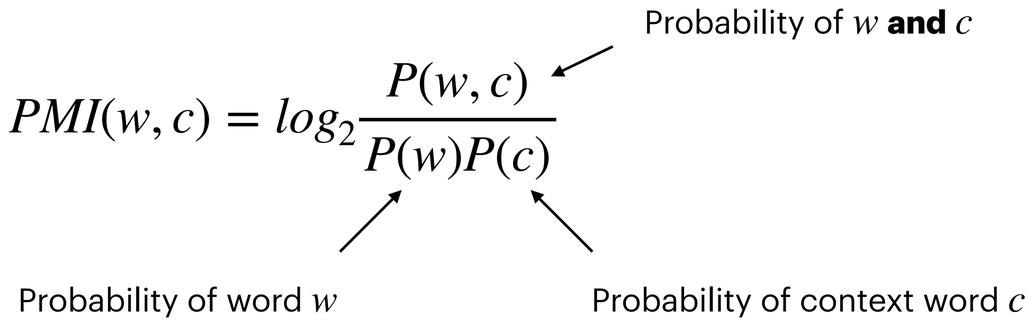
Sources & Notes

Hapax Legomenons is words which only appear once
https://en.wikipedia.org/wiki/Zipf's_law

CENTER FOR HUMANITIES COMPUTING

Positive Pointwise Mutual Information (PPMI)

- "How much more does two words ***co-occur*** rather than what we would have expected them to by ***chance***"



- What does this mean?
  - What happens if they are **independent**?
  - What happens if they are **perfectly dependent**?

Sources & Notes

Very much related to TF-iDF, which we will get more into next time

Positive Pointwise Mutual Information (PPMI)

- "How much more does two words ***co-occur*** rather than what we would have expected them to by ***chance***"
- If they are **independent**?

$$PMI(w,c) = log_2 \frac{P(w,c)}{P(w)P(c)} = log_2 \frac{P(w)P(c)}{P(w)P(c)} = log_2 1 = 0$$

- If they are **always co-occur**?

$$PMI(w,c) = log_2 \frac{P(w,c)}{P(w)P(c)} = log_2 \frac{P(w)}{P(w)P(c)} = log_2 \frac{1}{P(c)} = log_2 \frac{1}{0.001} = \approx 10$$

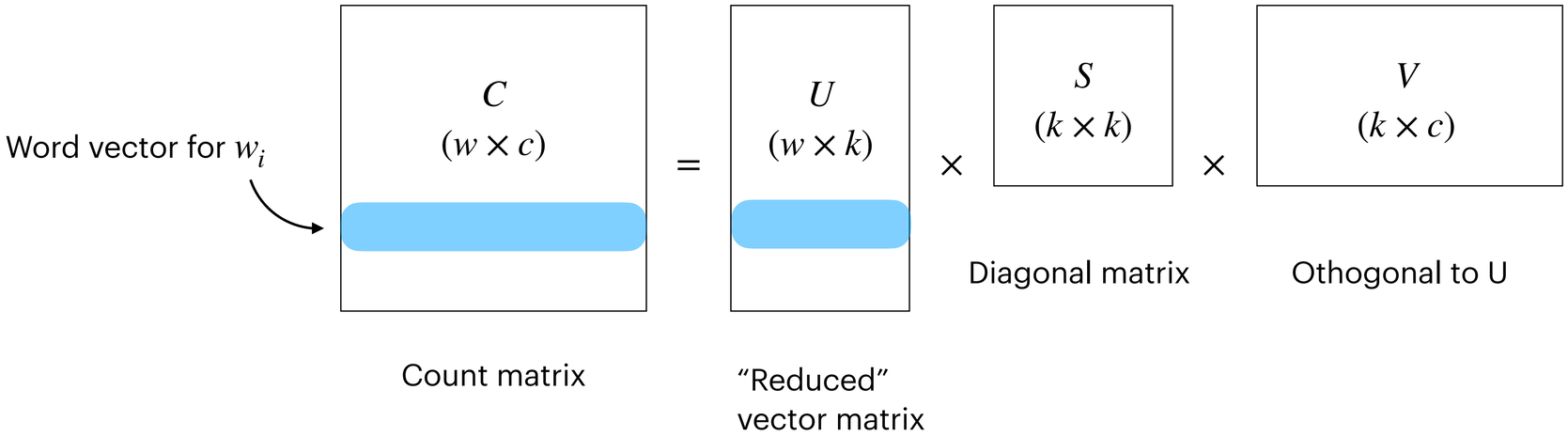
Assumed P(c) = 0.001

Sources & Notes

Very much related to TF-iDF, which we will get more into next time

Matrix Factorization Methods

- Example:
  - **Singular value decomposition** (SVD): A matrix can be decomposed as follows:



- **What do we lose?**
- **What do we gain?**

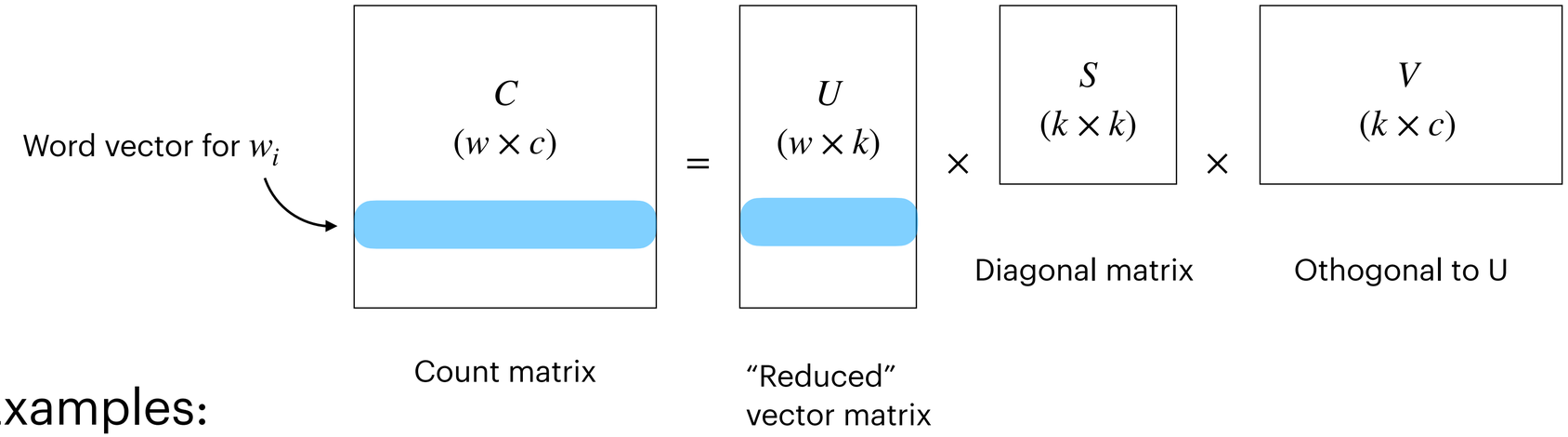
Sources & Notes

Other notable methods include LSA

You can play around with a similar process here: https://projector.tensorflow.org/

Generalizable Concept!

- Efficiency-performance trade-off
  - Methods where we can dymanically determine this tend to perform well



- Examples:
  - Low-precision compute (quantization)
  - Dymanic Embeddings sizes

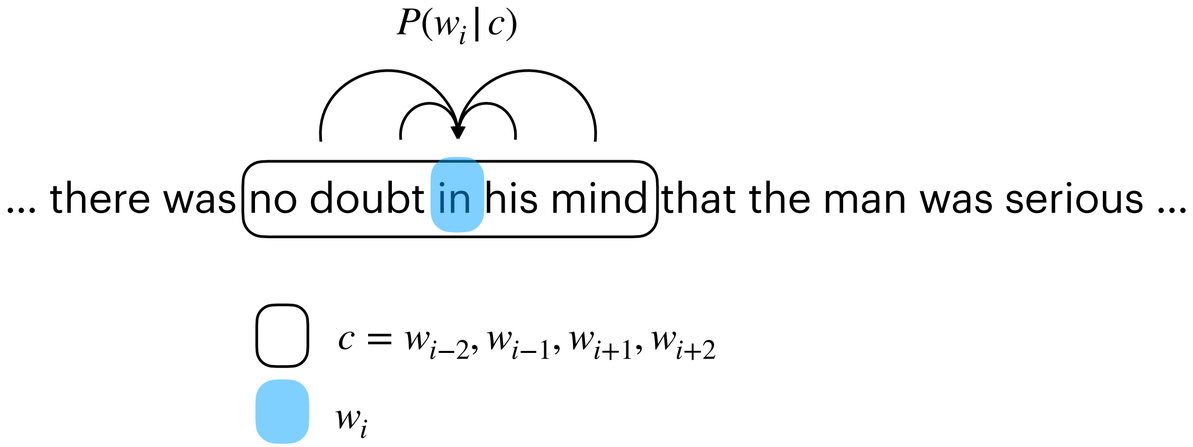
Sources & Notes

Kusupati, A., Bhatt, G., Rege, A., Wallingford, M., Sinha, A., Ramanujan, V., ... & Farhadi, A. (2022). Matryoshka representation learning. *Advances in Neural Information Processing Systems*, *35*, 30233-30249.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., ... & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2704-2713).

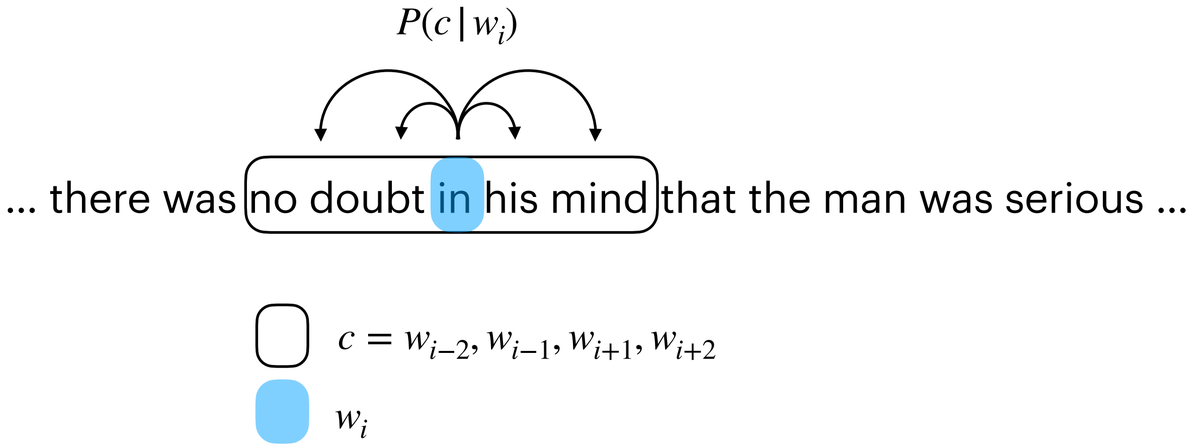
Shallow Window-based methods

- Countinous bag-of-words (**CBOW**):
  - “Given context *c* what is the most likely word *w*?”



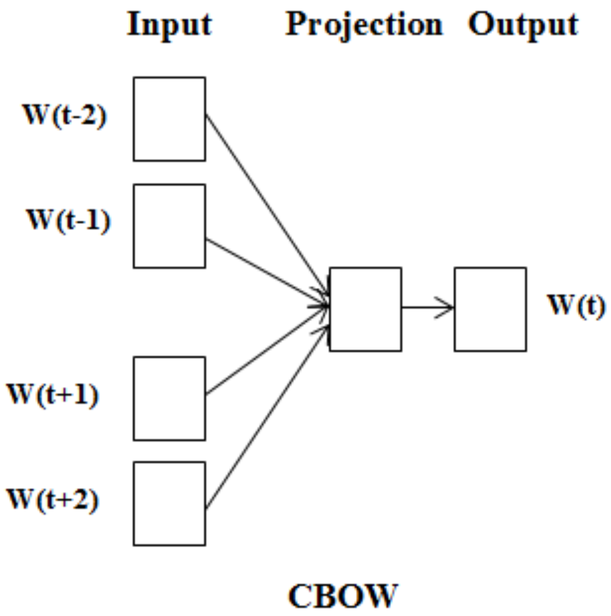
Shallow Window-based methods

- **Skip-gram**:
  - "Given word *w* what is the most likely context *c*?"

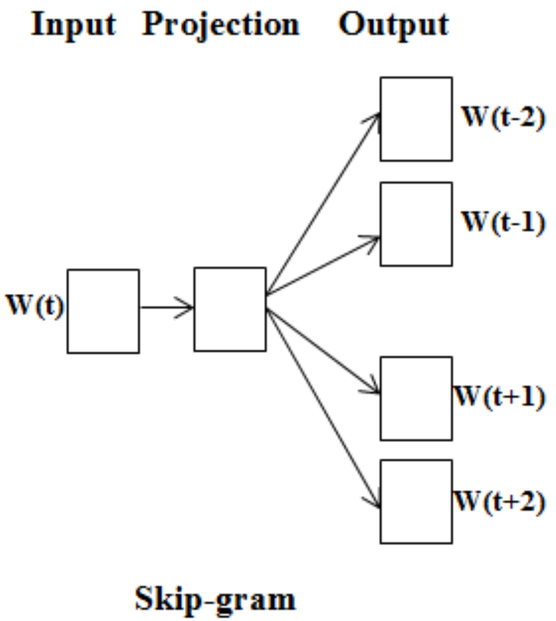


Shallow Window-based methods

- Word2Vec, FastText
- **Countinous BoW (CBOW)**:
  - “Given context *c* what is the most likely word *w*?”
  - Better for frequent words on large data
- **Skip-gram**:
  - “Given word *w* what is the most likely context *c*?”
  - Better for rarer words and small data



CBOW



Skip-gram

Sources & Notes

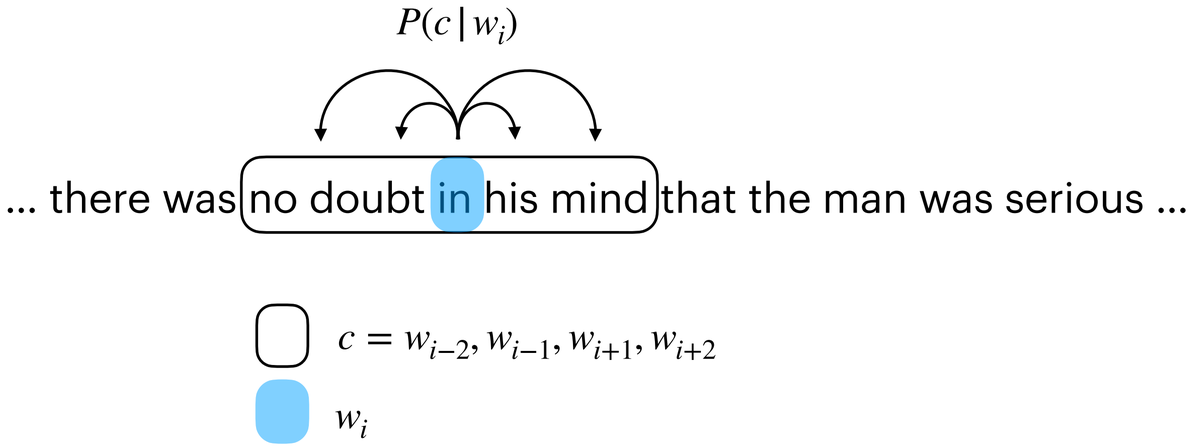
Mikolov, T. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Learn more:

- https://www.youtube.com/watch?v=viZrOnJclY0
- https://jalammar.github.io/illustrated-word2vec/

Generalizable concept: Self-supervision

- Statistical patterns can be learned directly from data
- No labels required
- Decrease the need for labelled data

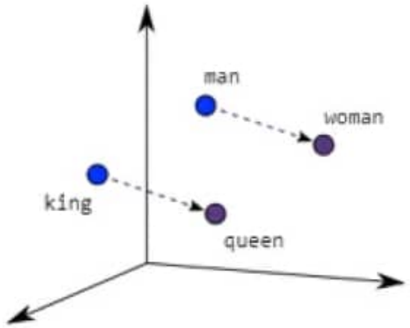


Exploring a word embedding space

- https://lena-voita.github.io/resources/lectures/word_emb/analysis/glove100_twitter_top3k.html
- **What can you find in this embedding space?**

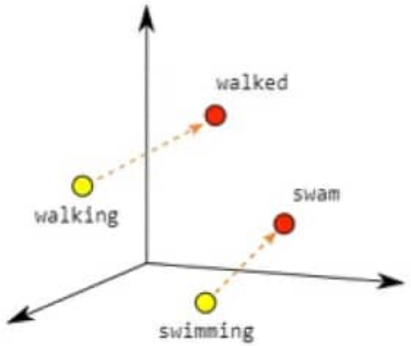
Linear Structure

Semantic



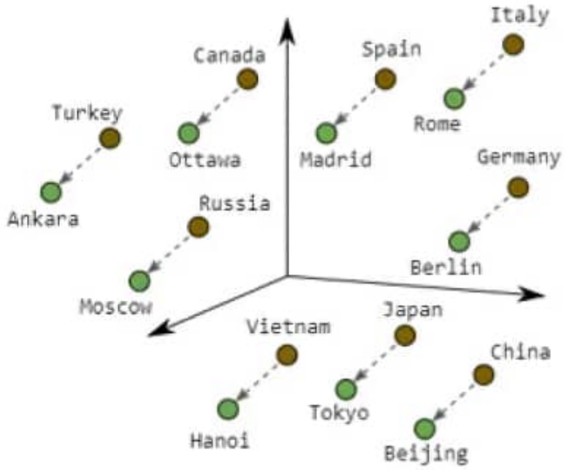
Male-Female

Syntactic



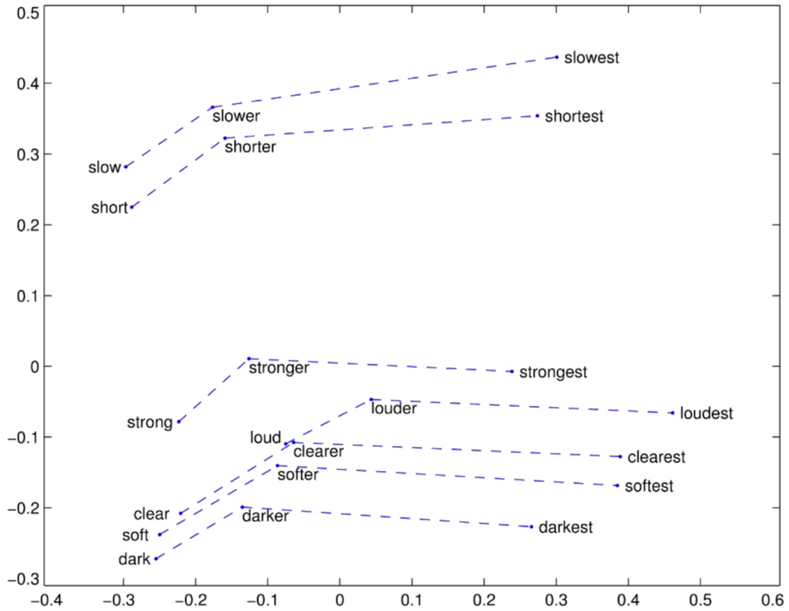
Verb Tense

Knowledge



Country-Capital

Syntactic



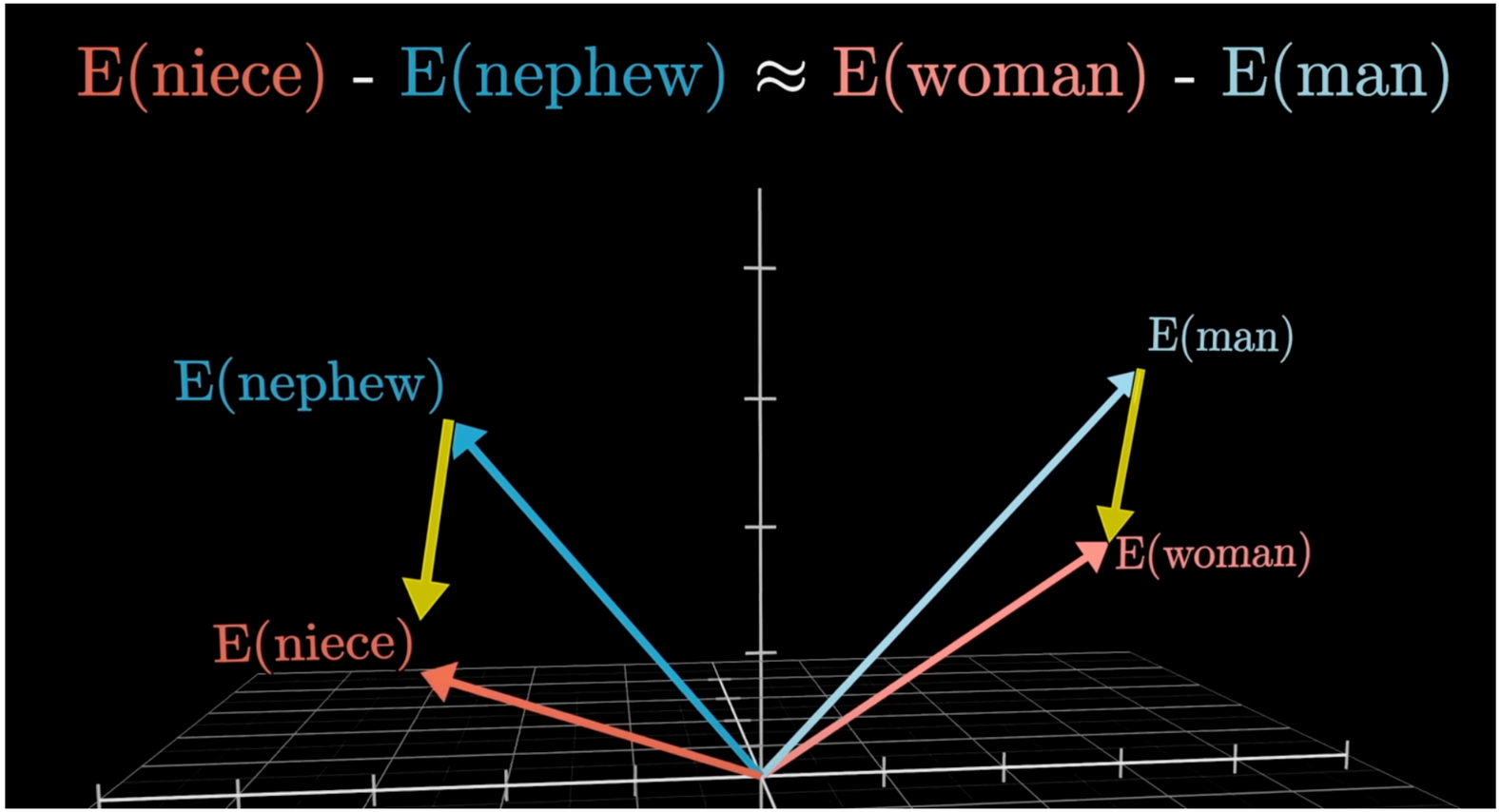
Morphological relationships

Sources & Notes

https://arxiv.org/pdf/1901.09813

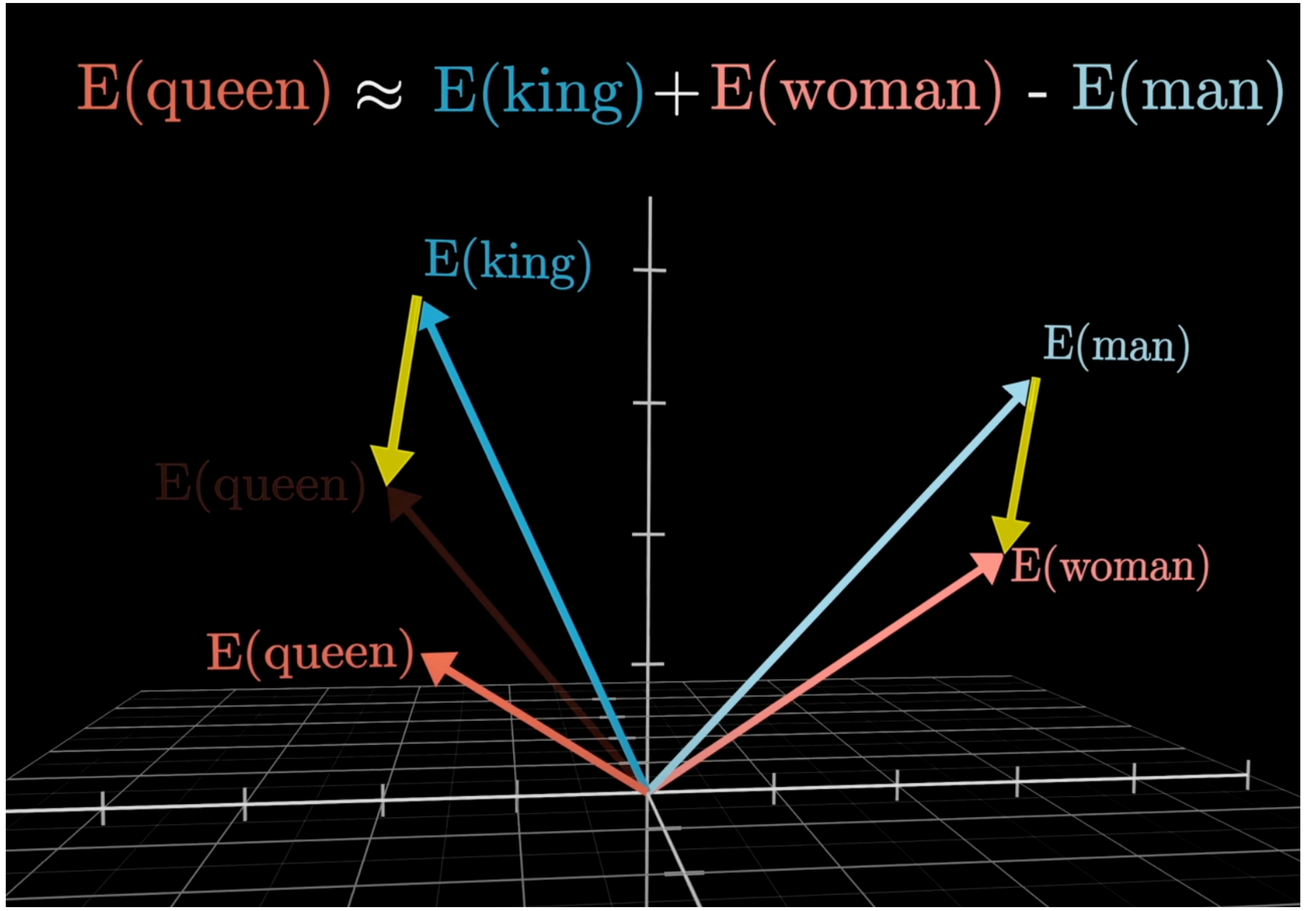
CENTER FOR HUMANITIES COMPUTING

Math with word embeddings



https://youtu.be/wjZofJX0v4M?si=4QX3KkBTpdbNjSaw

Math with word embeddings



Source: https://youtu.be/wjZofJX0v4M?si=4QX3KkBTpdbNjSaw

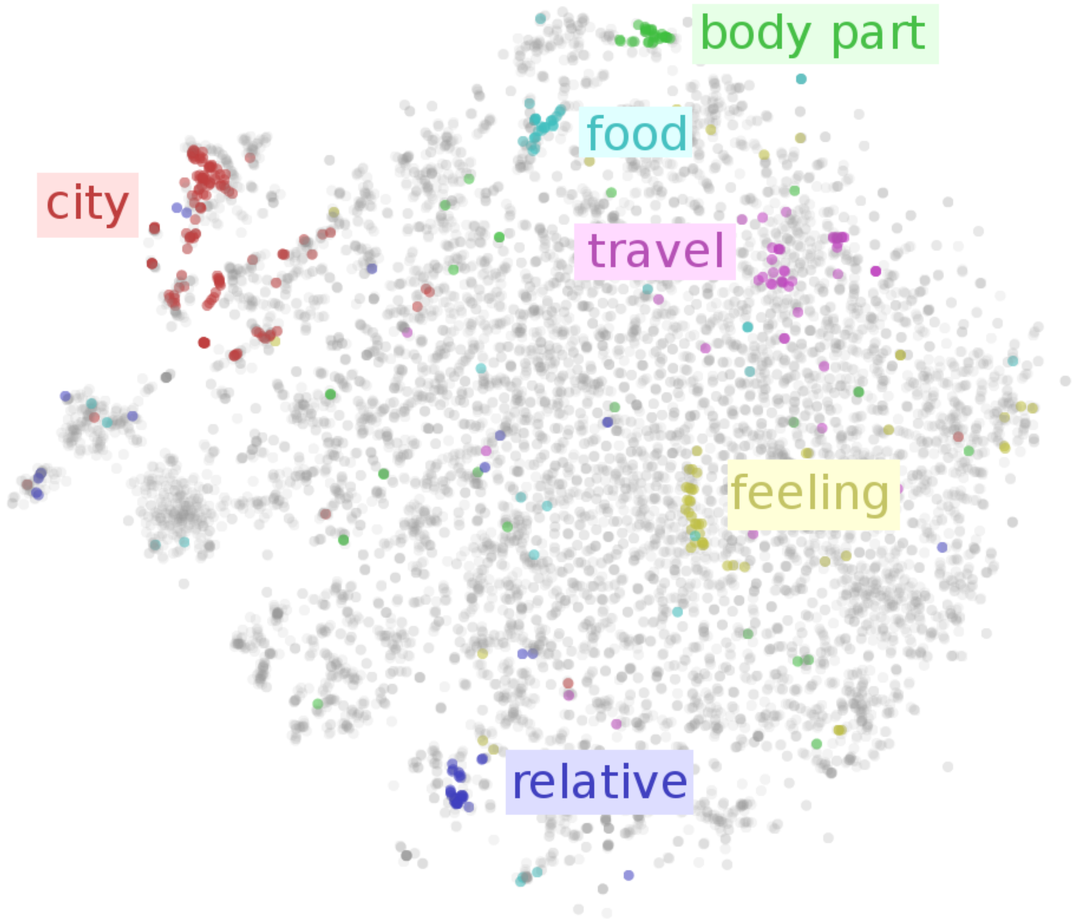
Synonymy

- Couch/sofa
- Car/automobile
- **Principles of contrast**: Always a difference in meaning
  - Water/H20

Sources & Notes

principle of contrast (Girard 1718, Bréal 1897, Clark 1987)

Meaningful Global structures



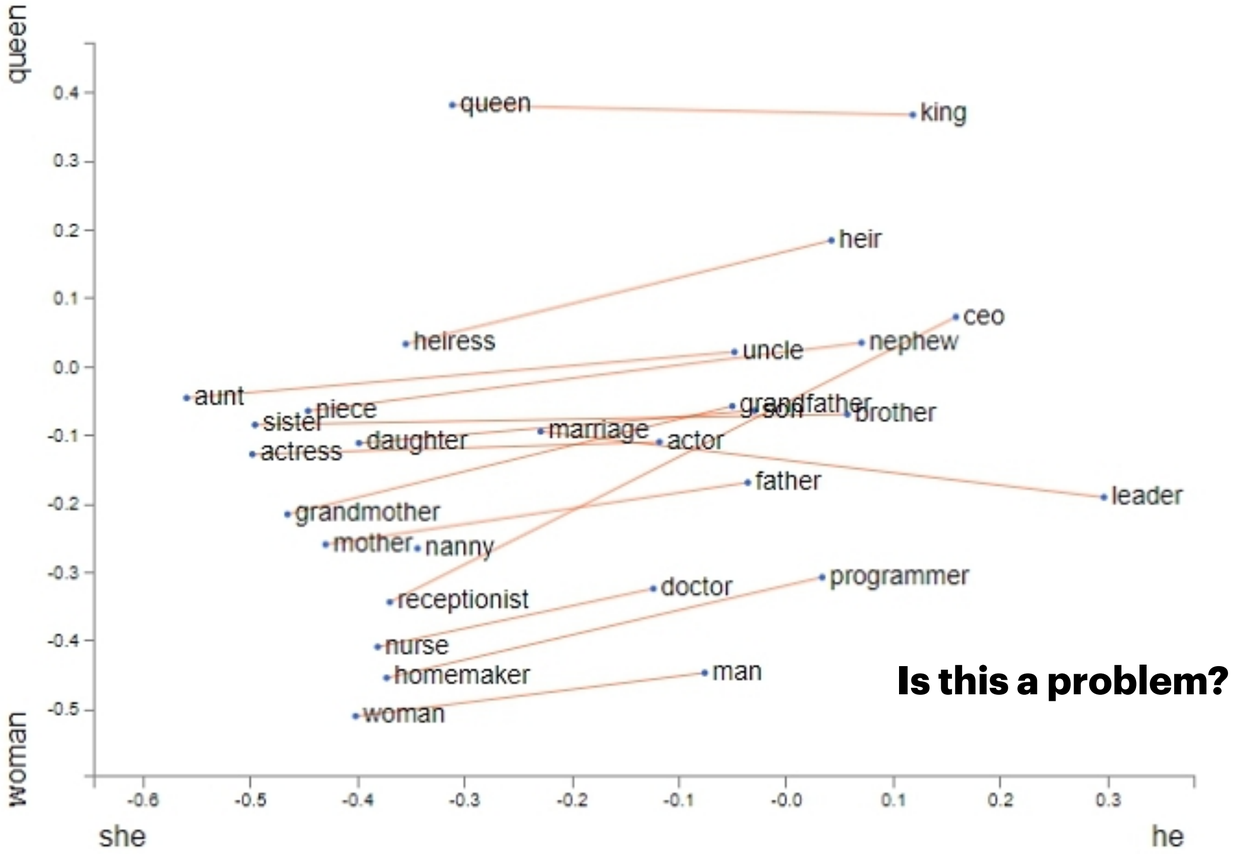
Sources & Notes

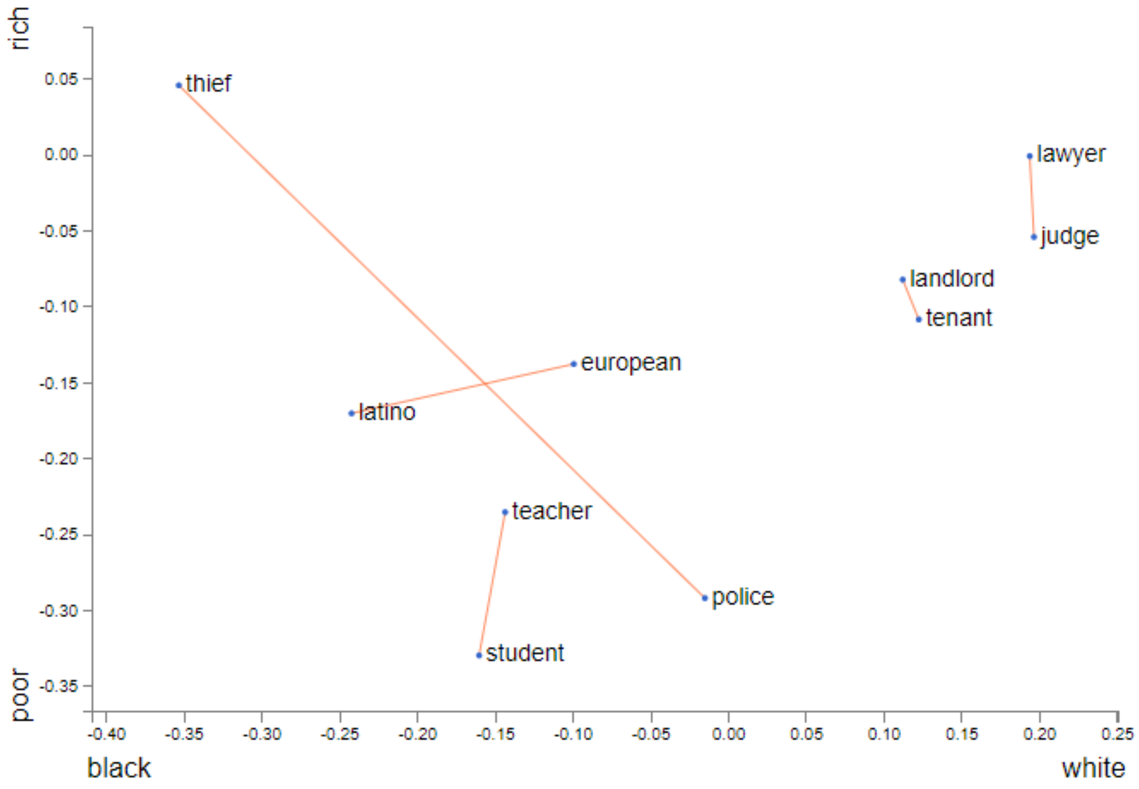
For more on good representation learning and especially on what a representation should do, check out:

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence, 35(8), 1798-1828.

CENTER FOR HUMANITIES COMPUTING

Embedding Bias





https://dl.acm.org/doi/fullHtml/10.1145/3582768.3582804

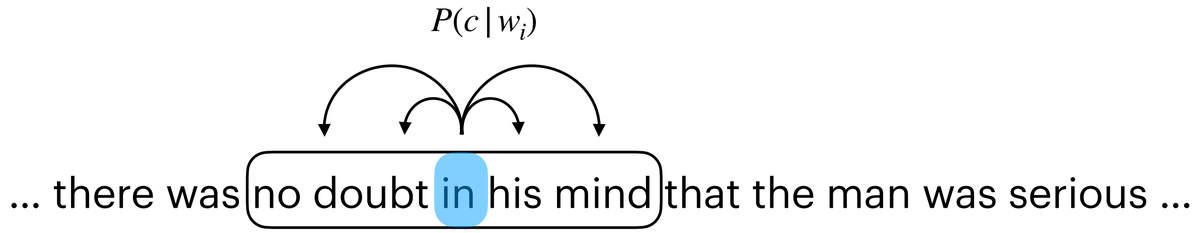
Sources & Notes

CENTER FOR HUMANITIES COMPUTING

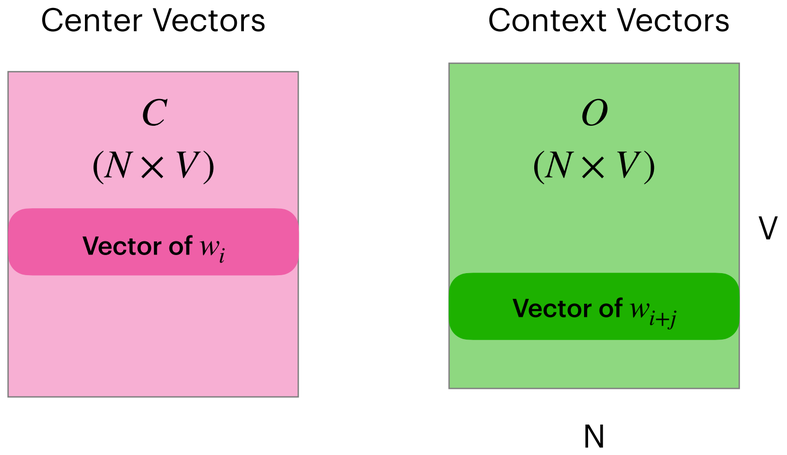
How do we train it?

An example using Skip-gram

Skip-gram: The Idea



- Extract vector for **center word** w_i from C
- Extract vector for **context words** w_{i+j} from O
- We want
  - to produce the probability of each word using these vectors
  - **High probability** for context words, **low probability** for non-context words
- **Optimization problem**: Optimize the weights of C and O such that this is the case



Skip-gram: Likelihood

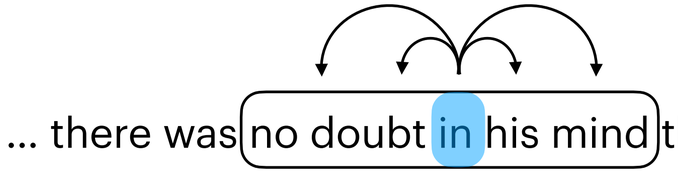
Goal: Learn the weights θ in O and C that maximizes the *likelihood* of data

The weight in the matrices we want to update

$$\prod_{-m \leq j \leq m, j \neq 0} P(w_{i+j} \mid w_i, \theta)$$

Specified the window size

$$P(no \mid in) \cdot P(doubt \mid in) \cdot P(his \mid in) \cdot P(mind \mid in)$$



... there was no doubt in his mind that the man was serious

Skip-gram: Likelihood

Goal: Learn the weights θ in O and C that maximizes the *likelihood* of data

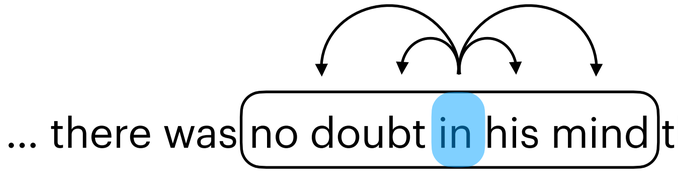
The weight in the matrices we want to update

$$\prod_{i=1}^{L} \prod_{-m \le j \le m, j \neq 0} P(w_{i+j} \mid w_i, \theta)$$

Specified the window size

Slide it across the sequence of length L

$$P(no \mid in) \cdot P(doubt \mid in) \cdot P(his \mid in) \cdot P(mind \mid in)$$



... there was no doubt in his mind that the man was serious

Skip-gram: Likelihood

Goal: Learn the weights θ in O and C that maximizes the *likelihood* of data

The weight in the matrices we want to update

$$Likelikhood = L(\theta) = \prod_{i=1}^{L} \prod_{-m \leq j \leq m, j \neq 0} P(w_{i+j} \mid w_i, \theta)$$

$$P(no \mid in) \cdot P(doubt \mid in) \cdot P(his \mid in) \cdot P(mind \mid in)$$

Specified the window size

Slide it across the sequence of length L

Skip-gram: Likelihood

Goal: Learn the weights θ in O and C that maximizes the *likelihood* of data

The weight in the matrices we want to update

$$Likelikhood = L(\theta) = \prod_{i=1}^{L} \prod_{-m \leq j \leq m, j \neq 0} P(w_{i+j} \mid w_i, \theta)$$

$$P(no \mid in) \cdot P(doubt \mid in) \cdot P(his \mid in) \cdot P(mind \mid in)$$

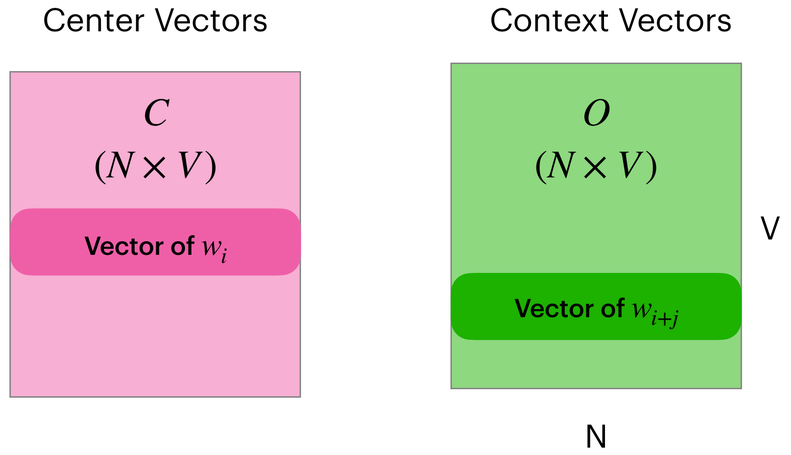
How is this computed?

Specified the window size

Slide it across the sequence of length L

Skip-gram: Softmax

$$P(w_{i+j} \mid w_i) = \frac{exp(o_{i+j}^T c_i)}{\sum_{v \in V} exp(o_v^T c_i)}$$



Sources & Notes

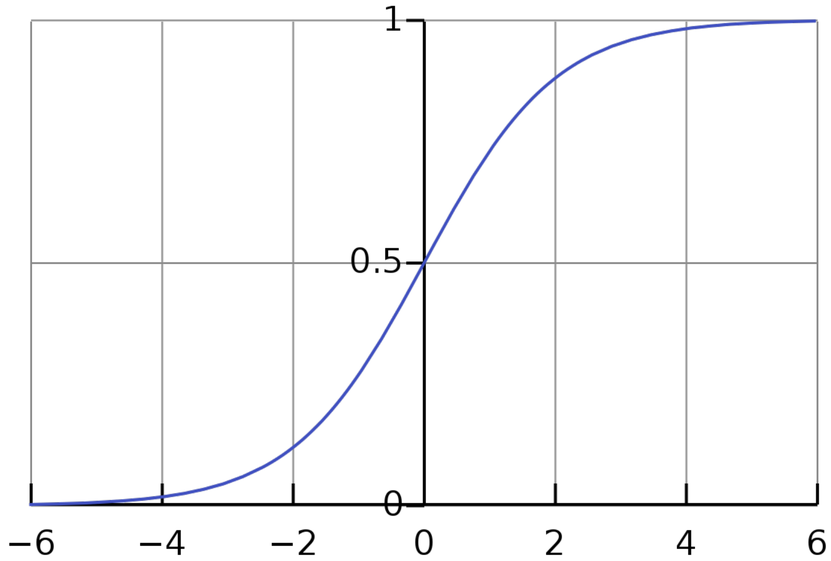
You can formalize this probability differently.
You can also note that the denominator it quite expensive to compute. Turns out that we can simply sample from the vocabulary instead. This leads us to negative sampling.

Interlude: The sigmoid

- Softmax for 2 classes => sigmoid*
- Sigmoid = Logistic function
- You know this from regression

Linear Regression: $y = X\beta$

Logistic Regression: $y = \sigma(X\beta)$



$$\sigma(X\beta) = \frac{1}{1 + exp(X\beta)}$$

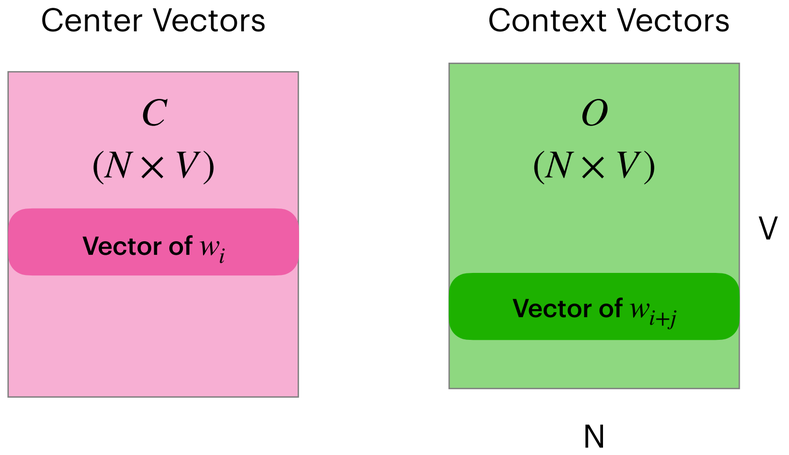
See explanation as to why here: *https://stats.stackexchange.com/questions/233658/softmax-vs-sigmoid-function-in-logistic-classifier

Skip-gram: Softmax

$$P(w_{i+j} \mid w_i) = \frac{exp(o_{i+j}^T c_i)}{\sum_{v \in V} exp(o_v^T c_i)}$$

Dot Product: Why is that important?

Normalization: Similarity across the vocabulary



This function is used for turning numerical scores into probabilities. Very common in **multiclass** classifiers

Sources & Notes

You can formalize this probability differently.
You can also note that the denominator it quite expensive to compute. Turns out that we can simply sample from the vocabulary instead. This leads us to negative sampling.

CENTER FOR HUMANITIES COMPUTING

Skip-gram: Optimization

$$Likelikhood = L(\theta) = \prod_{i=1}^{L} \prod_{-m \leq j \leq m, j \neq 0} P(w_{i+j} \mid w_i, \theta)$$

We want to **maximize** the likelihood or equivalently minimize **negative log likelihood**:

$$J(\theta) = -\frac{1}{T} \sum_{i=1}^{L} \sum_{-m \leq j \leq m, j \neq 0} logP(w_{i+j} \mid w_i, \theta)$$

Since: $log(a \cdot b) = log(a) + log(b)$

How do we **minimize** this quantity?

Short answer: Gradient descent

Long answer: Lecture 4 on Neural Networks

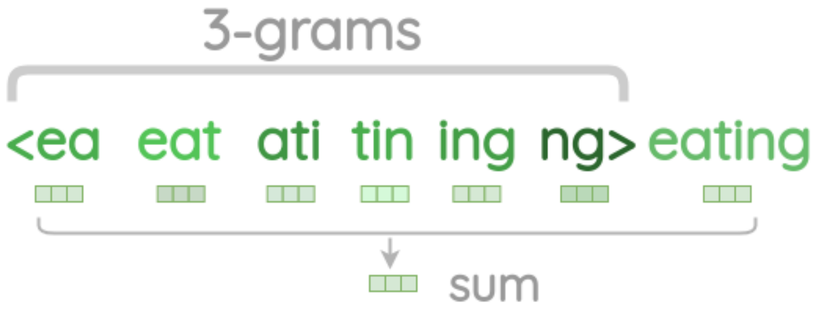
Overview

- We can learn **vector representations** of word
- These can be learned in multiple ways
  - Often based on **co-occurance** patterns
- We examined how to **compare** these **embeddings**
- They have properties representing:
  - **Semantics and syntax**: verb-tense, genderness, synonymy, etc.

Perspectives

What defines a word?

- Don’t —>
  [“do”, “not”]
  [“don’t”]
  [“don”, “t”]
  …
- Morphemes
- Word pieces
  - E.g. fastText



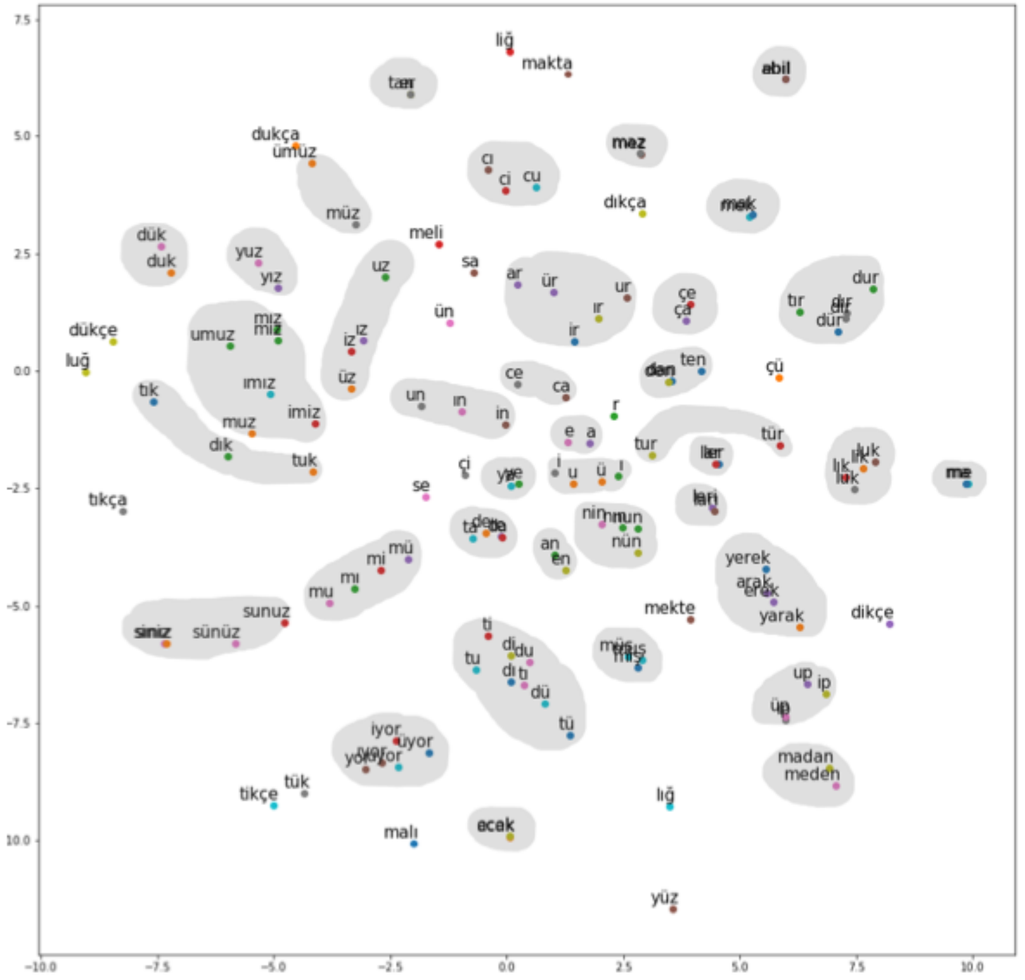


Figure 3: Turkish allomorph vectors learned by morph2vec. Some morphemes are blurry because of the overlapping of a few allomorphs.

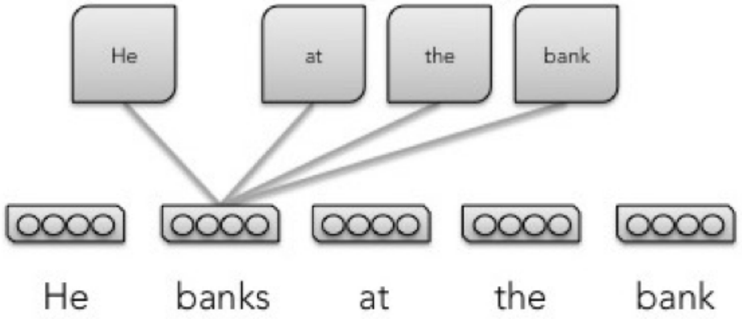
Sources & Notes

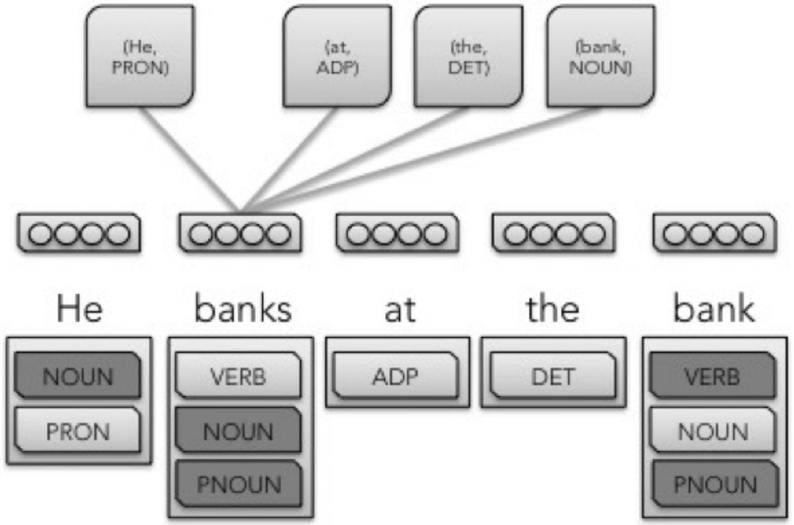
Ahmet Üstün, Murathan Kurfalı, and Burcu Can. 2018. Characters or Morphemes: How to Represent Words?. In Proceedings of the Third Workshop on Representation Learning for NLP, pages 144–153, Melbourne, Australia. Association for Computational Linguistics.

CENTER FOR HUMANITIES COMPUTING

Contextualizing Word embeddings

- **What are the benefits?**
- **What are the problems?**

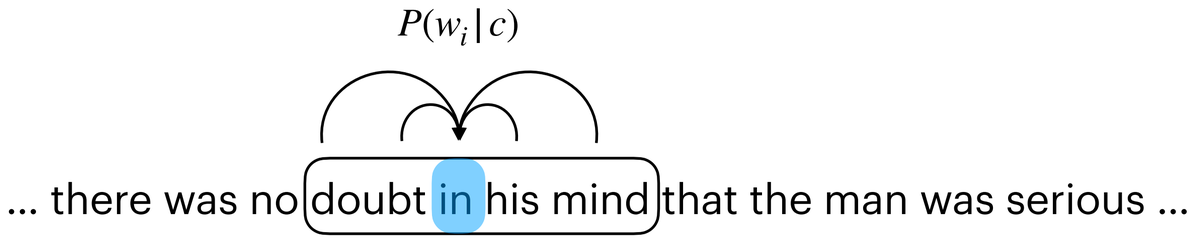


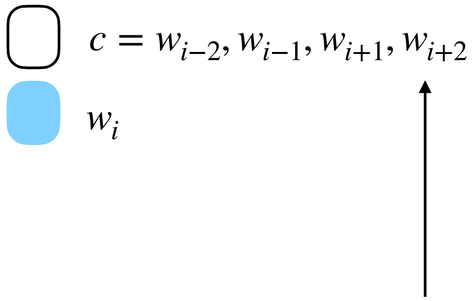


Sources & Notes

https://arxiv.org/abs/1511.06388

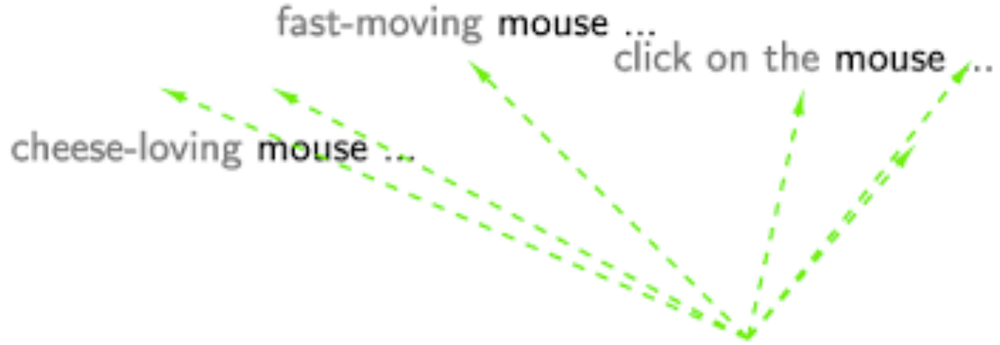
Contextualizing Word embeddings





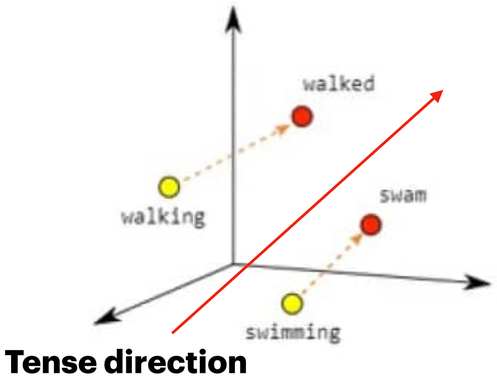
What about adding some **positional information** here?

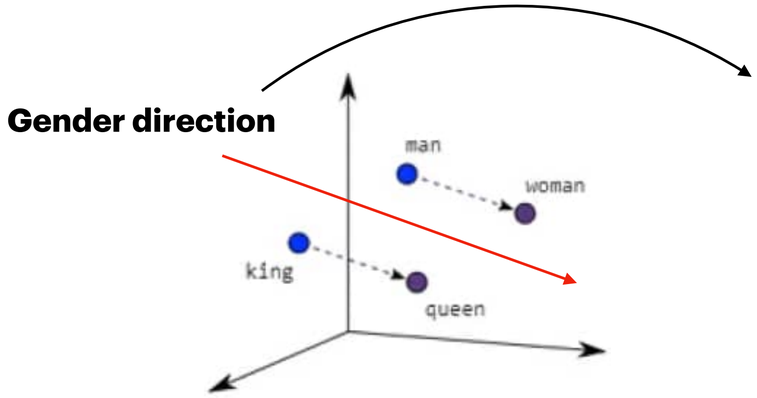
⇒ BERT



Making Dimensions Interpretable Again

- Rediscover interpretable direction
- Transform the matrix such that the dimensions in the given matrix represent the interpretable direction
- **Is there any problems?**





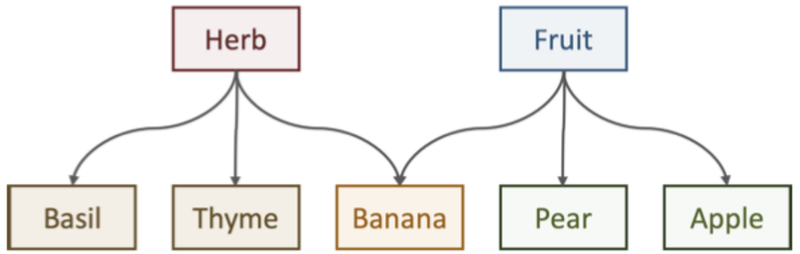
	Man (5391)	Woman (9853)	King (4914)	Queen (7157)
Gender	−1	1	-0.95	0.97
Royal	0.01	0.02	0.93	0.95
Age	0.03	0.02	0.70	0.69
Food	0.09	0.01	0.02	0.01

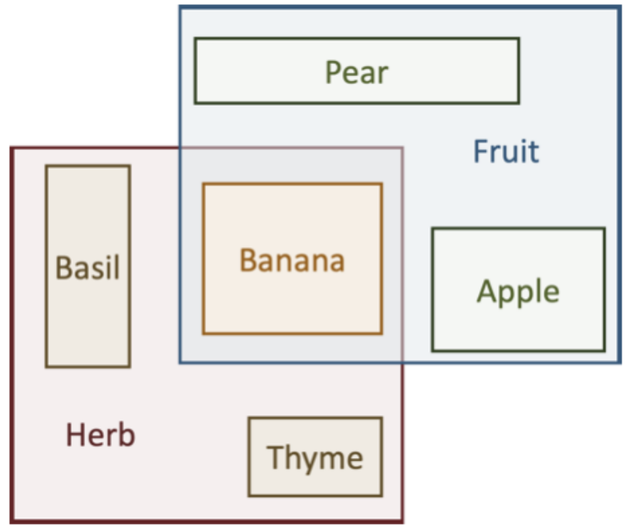
https://ar5iv.labs.arxiv.org/html/1807.07279v4

Alternative to vectors

- Graphs
- Box embeddings

  ...
- **What do we need for an embedding?**





Word Embeddings for Translations

Ingredients:

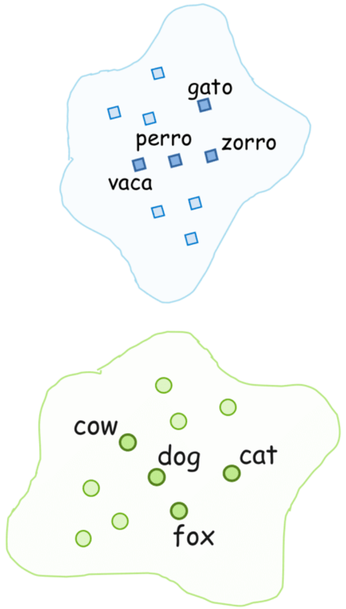
- corpus in one language (e.g., English)
- corpus in another language (e.g., Spanish)

- very small dictionary

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

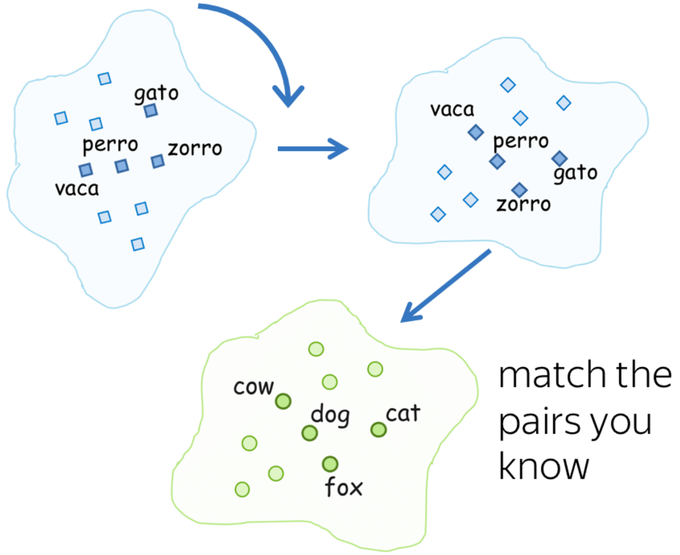
Step 1:

- train embeddings for each language



Step 2:

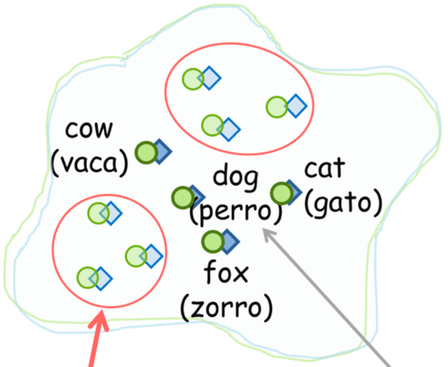
- linearly map one embeddings to the other to match words from the dictionary



match the pairs you know

Step 3:

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



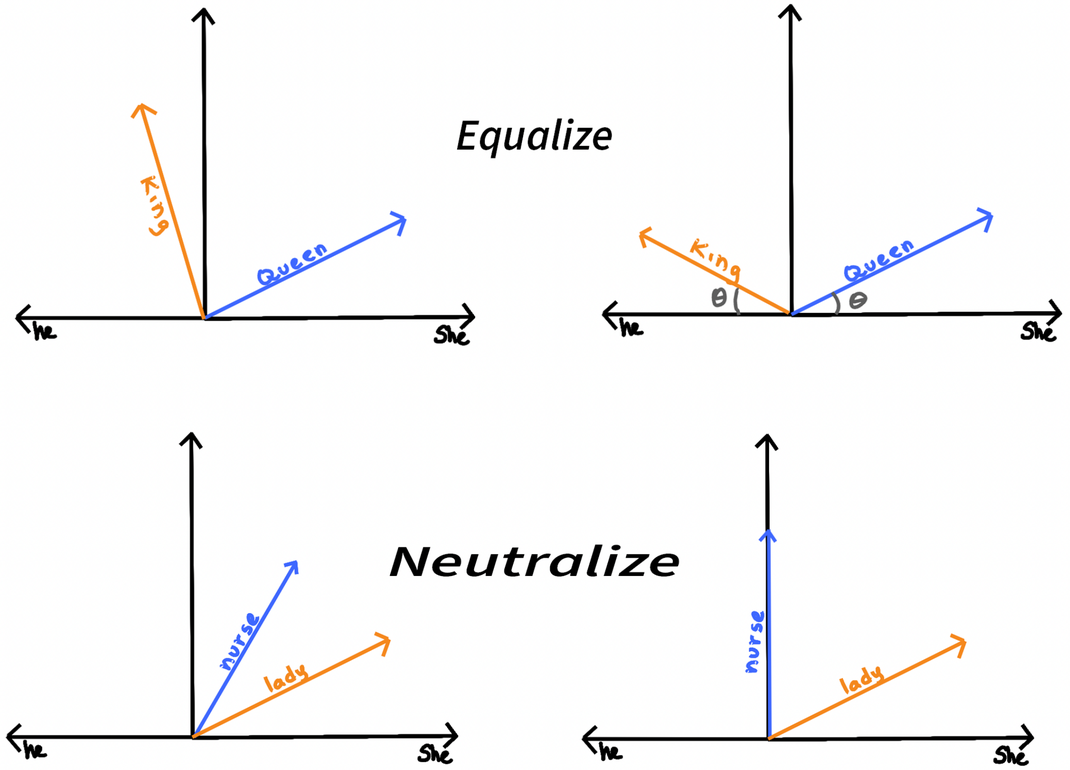
old translations – the ones you knew

new translations – you learned them!

Sources & Notes

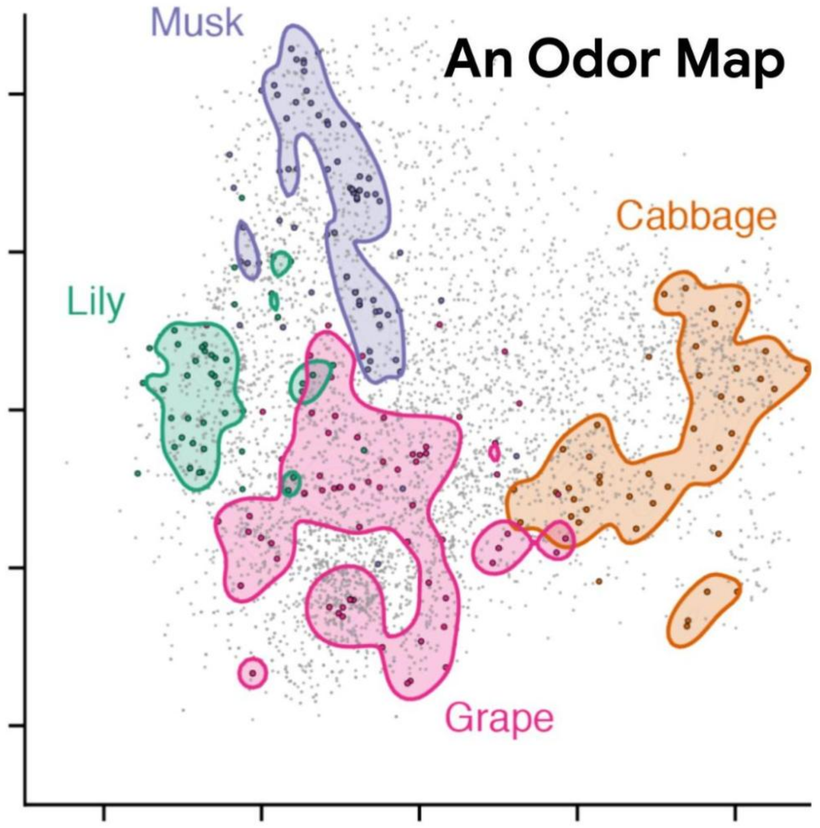
CENTER FOR HUMANITIES COMPUTING

Debiasing Embeddings



Other modalities

- Embeddings work for many areas:
  - Frequently used in:
    - Images
    - Audio
    - Protein folding
  - Also seen in:
    - odors
    - genetics
- In the future we could likely see more of them in:
  - Video (already some)



Sources & Notes

https://wandb.ai/telidavies/ml-news/reports/Google-s-Smell-AI-Can-Predict-Scents-Repel-Mosquitoes--VmlldzoyNTg1OTE3