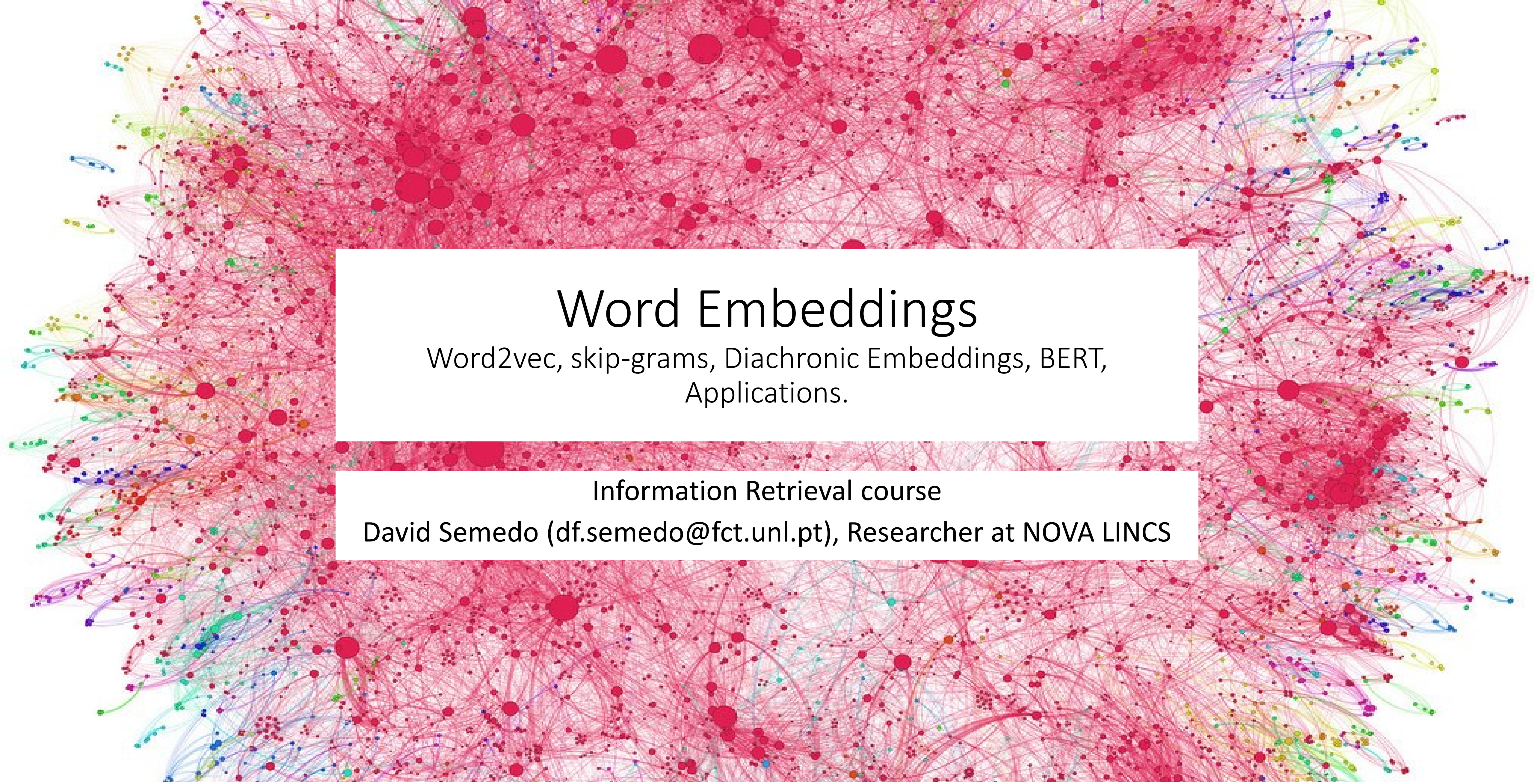


Word Embeddings

Word2vec, skip-grams, Diachronic Embeddings, BERT, Applications.

Information Retrieval course

David Semedo (df.semedo@fct.unl.pt), Researcher at NOVA LINCS

About me and my research

- Researcher at the Web and Media Search group, from NOVA LINCS since 2015.
- Ph.D. in Deep Learning for Multimedia Understanding.
- Topic:
  - *"Bridging Vision and Language over Time with Neural Cross-modal Embeddings"*.
- Main interests: Conversational Agents, multimodal machine learning, intersecting CV and NLP approaches, deep learning and data mining.

Lecture outline

- Representing words and documents;
- Distributional Semantics;
- Word Embeddings: word2vec model;
- Word embeddings Applications;
- State-of-the-art approaches for word and sentence representation.

How to represent a word?

$$\begin{array}{llccccccc}
 & & \text{dog} & \text{cat} & \text{person} & \text{plane} & \text{ate} & \text{The} & \text{floor} \\
\text{dog} & 1 & [1 & 0 & 0 & 0 & 0 & 0 & 0] \\
\text{cat} & 2 & [0 & 1 & 0 & 0 & 0 & 0 & 0] \\
\text{person} & 3 & [0 & 0 & 1 & 0 & 0 & 0 & 0]
\end{array}$$

How to represent a document?

D1: “The dog ate the cat”

D2: “The person on the plane”

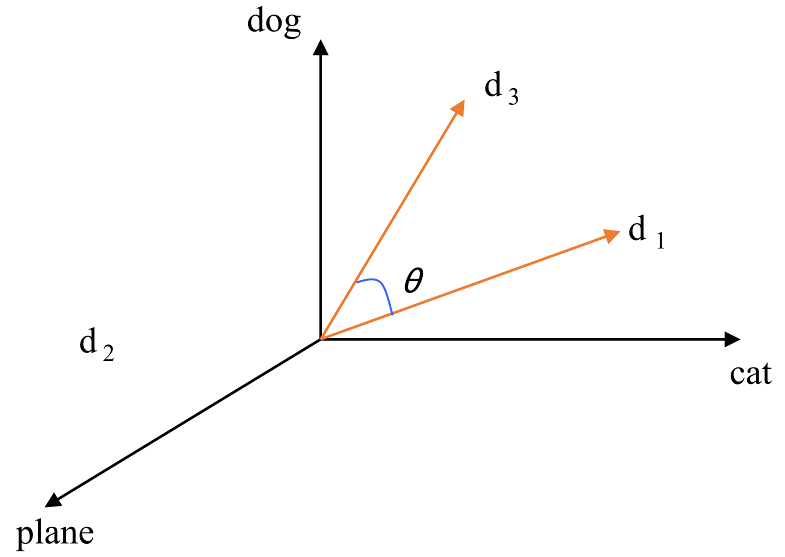
	dog	cat	person	plane	ate	The	on
D1:	[1	1	0	0	1	2	0]
D2:	[0	0	1	1	0	2	1]

Bag-of-words Model

Vector space model

- In the vector space model,
  each dimension corresponds to a term.

- The dimensionality V of the space
  corresponds to the size of
  the *vocabulary*.



- Each word is represented by a V dimensional vector, where only the dimension corresponding to that word is non-zero.

- Hence, each document is represented by the frequency of its terms.

Some Bag-of-words Problems

D1: “The dog ate the cat”

D4: “The cat ate the dog”

D5: “The dog ate the biscuit”

D6: “The person ate the cat”

$$
\begin{array}{l|ccccccc}
 & \text{dog} & \text{cat} & \text{person} & \text{plane} & \text{ate} & \text{The} & \text{biscuit} \\
\hline
\text{D1:} & [1 & 1 & 0 & 0 & 1 & 2 & 0] \\
\text{D4:} & [1 & 1 & 0 & 0 & 1 & 2 & 0] \\
\text{D5:} & [1 & 0 & 0 & 0 & 1 & 2 & 1] \\
\text{D6:} & [0 & 1 & 1 & 0 & 1 & 2 & 0]
\end{array}
$$

?

$d(D1, D4) = 0$

$d(D1, D5) = 1$

$d(D1, D6) = 1$

- Term-count rational fails to capture word/sentence semantics;
- Distance between words using one-hot encodings always the same.

Distributional Semantics

	food	walks	window	runs	mouse	invented	legs	sleeps	mirror	tail	…
dog	[5	5	0	5	0	0	5	5	0	2	…]
cat	[5	4	1	4	2	0	3	4	0	3	…]
person	[5	5	1	5	0	2	5	5	0	0	…]

$\longrightarrow$

This vocabulary can be extremely large

Distributional Semantics

- How similar is **pizza** to **pasta**?
- How related is **pizza** to **Italy**?

- **Representing words as vectors** allows easy computation of similarity and relatedness.
- We know how to compare two vectors. We just need the two vectors to **capture the semantics of each word**.

Approaches for Representing Words

Distributional Semantics (*Count*)

- Used since the 90's
- Sparse word-context PMI/PPMI matrix
- Decomposed with SVD

Word Embeddings (*Predict*)

- Inspired by deep learning
- `word2vec` *(Mikolov et al., 2013)*
- GloVe *(Pennington et al., 2014)*

Underlying Theory: **The Distributional Hypothesis** *(Harris, '54; Firth, '57)*

"Similar words occur in similar contexts"

Approaches for Representing Words

Both approaches:

- Rely on the **same linguistic theory**
- Use the **same data**
- Are **mathematically related**
  - “Neural Word Embedding as Implicit Matrix Factorization” (NIPS 2014)

- How come word embeddings are so much better?
  - “Don’t Count, Predict!” (Baroni et al., ACL 2014)

Word Embeddings with *Word2Vec*

Algorithms

(objective + training method)

- Skip Grams + Negative Sampling
- CBOW + Hierarchical Softmax
- Noise Contrastive Estimation
- GloVe
- …

Hyperparameters

(preprocessing, smoothing, etc.)

- Subsampling
- Dynamic Context Windows
- Context Distribution Smoothing
- Adding Context Vectors
- …

What is `word2vec`?

- `word2vec` is **not** a single algorithm
- It is a **software package** for representing words as vectors, containing:
  - Two distinct models
    - CBoW
    - Skip-Gram
  - Various training methods
    - Negative Sampling
    - Hierarchical Softmax
  - A rich preprocessing pipeline
    - Dynamic Context Windows
    - Subsampling
    - Deleting Rare Words

What is `word2vec`?

- `word2vec` is **not** a single algorithm
- It is a **software package** for representing words as vectors, containing:
  - Two distinct models
    - CBoW
    - **Skip-Gram** **(SG)**
  - Various training methods
    - **Negative Sampling** **(NS)**
    - Hierarchical Softmax
  - A rich preprocessing pipeline
    - **Dynamic Context Windows** **(DCW)**
    - Subsampling
    - Deleting Rare Words

Skip-Grams with Negative Sampling (SGNS)

"Marco saw a furry little cat hiding in the tree."

"word2vec Explained…"
Goldberg & Levy, arXiv 2014

Skip-Grams with Negative Sampling (SGNS)

"Marco saw a furry little cat hiding in the tree."

"word2vec Explained…"
Goldberg & Levy, arXiv 2014

Skip-Grams with Negative Sampling (SGNS)

"Marco saw a furry little cat hiding in the tree."

words	**contexts**
cat	furry
cat	little
cat	hiding
cat	in
…	…

D (data)

"word2vec Explained…"
Goldberg & Levy, arXiv 2014

Skip-Grams with Negative Sampling (SGNS)

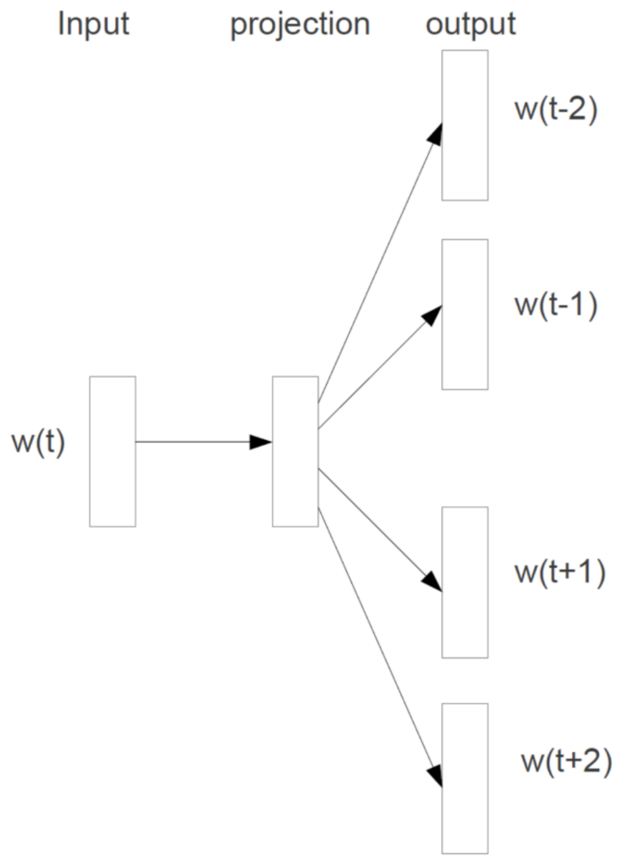
"Marco saw a furry little cat hiding in the tree."

- Word2vec models the distribution of words and context words.

- The model will maximize the log-likelihood:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

Context-window size



Skip-Grams with Negative Sampling (SGNS)

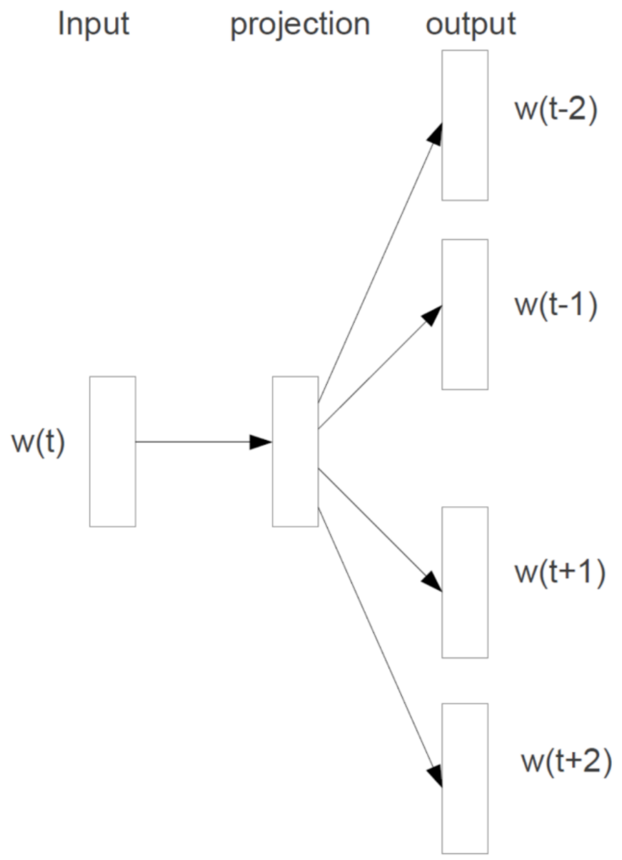
"Marco saw a furry little cat hiding in the tree."

We can think of it as learning a classifier that given a target w_{target} word and any other word w_i:

$$P(w_i|w_{target})$$

"A word is likely to occur near the target, if its embedding is similar to the target embedding":

Cosine similarity: $sim(w_{target}, w_i) = v_{w_{target}}^T \cdot v_{w_i}$



Skip-Grams with Negative Sampling (SGNS)

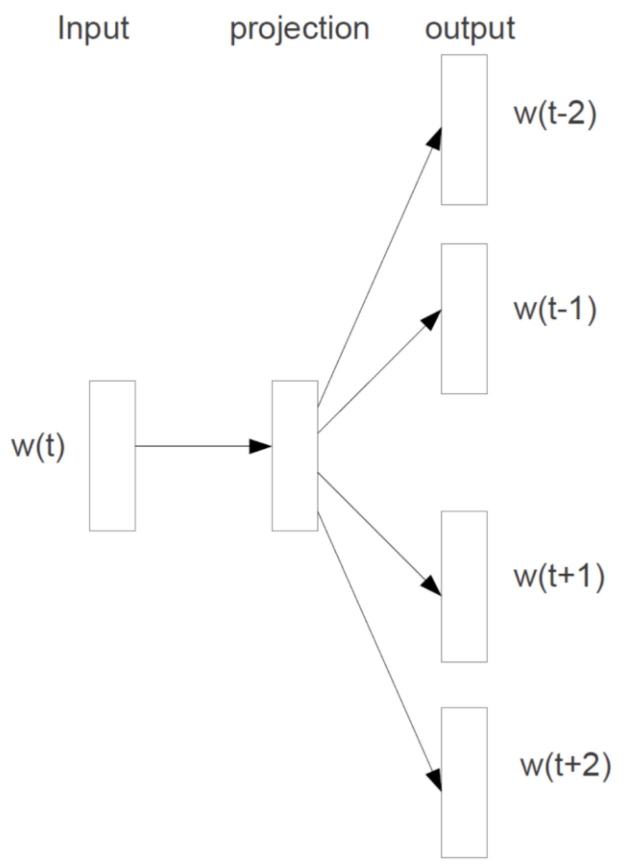
"Marco saw a furry little cat hiding in the tree."

"A word is likely to occur near the target, if its embedding is similar to the target embedding":

$$P(w_i|w_{target}) = sim(w_{target}, w_i) = v_{w_{target}}^T \cdot v_{w_i}$$

Recall the binary classifier:

$$sim(w_{target}, w_i) = \begin{cases} 1 & , if\ w_i\ is\ in\ the\ context\ of\ w_{target} \\ -1 & , otherwise \end{cases}$$



Softmax: words vs context words

- The $P(w_i|w_{target})$ is formalized as the softmax:

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^\top v_{w_I}\right)}{\sum_{w=1}^{W} \exp\left({v'_w}^\top v_{w_I}\right)}$$

where each word is represented by a vector $v_{w_*} = [v_{w*} \quad \cdots \quad v_{w*}]$, thus rendering the argument of the *softmax* function:

Marco saw a furry little cat hiding in the tree.

$$v_{w_O}^T \cdot v_{w_I} = \boxed{[v_{w_{O,1}} \quad \cdots \quad v_{w_{O,n}}]} \cdot \boxed{\begin{bmatrix} v_{w_{I,1}} \\ \cdots \\ v_{w_{I,n}} \end{bmatrix}}$$

$$p(furry|wampimuk) = \frac{\exp\left(\boxed{v_{furry_O}^T} \cdot \boxed{v_{cat_I}}\right)}{\sum_{w_I} \exp\left(\boxed{v_{furry_O}^T} \cdot v_{*_I}\right)}$$

Skip-Gram Architecture

$P(w_i|w_{target})$

Target word w_{target}: "ants"

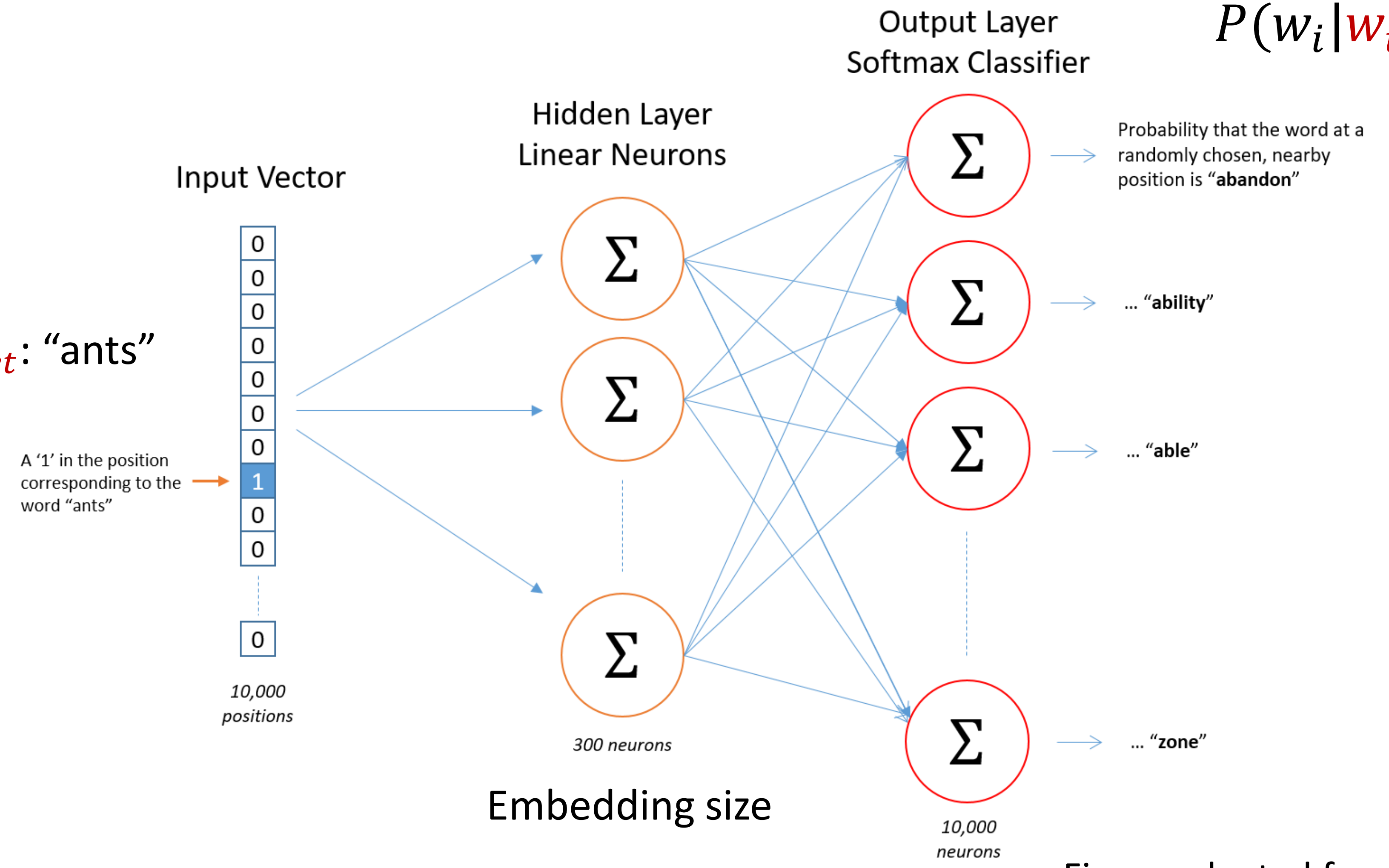


Figure adapted from here.

Stochastic Gradient Descent

- But Corpus may have a vocabulary with 40B tokens and windows
- You would wait a very long time before making a single update!
- **Very** bad idea for pretty much all neural nets!
- Instead: We will update parameters after each window t
  → Stochastic gradient descent (SGD)

$$v_{w_O}^{new} = v_{w_O}^{old} - \alpha \nabla_{v_{w_O}} p(w_o | w_I, D)$$

$$v_{w_I}^{new} = v_{w_I}^{old} - \alpha \nabla_{v_{w_I}} p(w_o | w_I, D)$$

Positive Samples + Negative Sampling

- **Maximize:** $\sigma(\vec{w} \cdot \vec{c})$
  - c was **observed** with w

words	**contexts**
cat	furry
cat	little
cat	hiding
cat	in

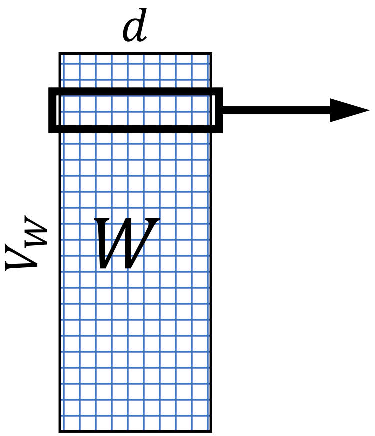
- **Minimize:** $\sigma(\vec{w} \cdot \vec{c}')$
  - c' was **hallucinated** with w

words	**contexts**
cat	Australia
cat	cyber
cat	the
cat	1985

"word2vec Explained…"
Goldberg & Levy, arXiv 2014

Word vectors

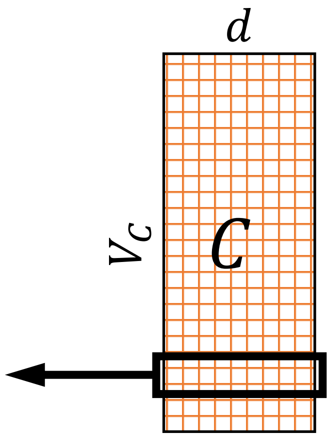
- SGNS finds a vector $\vec{w}$ for each word w in our vocabulary V_W
- Each such vector has d latent dimensions (e.g. $d = 300$)
- Effectively, it learns a matrix W whose rows represent V_W
- **Key point:** it also learns a similar auxiliary matrix C of context vectors
- In fact, each word has two embeddings



w:cat $= (-3.1, 4.15, 9.2, -6.5, \ldots)$

$\neq$

c:cat $= (-5.6, 2.95, 1.4, -1.3, \ldots)$



Why?
Think about P(cat|cat)

“word2vec Explained…”
Goldberg & Levy, arXiv 2014

Hyperparameters

- **Preprocessing** **(word2vec)**
  - Dynamic Context Windows
  - Subsampling
  - Deleting Rare Words

- **Association Metric** **(SGNS)**
  - Shifted PMI
  - Context Distribution Smoothing

Dynamic Context Windows

"Marco saw a furry little cat hiding in the tree."

Dynamic Context Windows

”saw a furry little cat hiding in the tree”

Dynamic Context Windows

saw a furry little **cat** hiding in the tree

word2vec:	$\frac{1}{4}$	$\frac{2}{4}$	$\frac{3}{4}$	$\frac{4}{4}$	$\frac{4}{4}$	$\frac{3}{4}$	$\frac{2}{4}$	$\frac{1}{4}$
GloVe:	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{1}$	$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$
Aggressive:	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{1}$	$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{8}$

The Word-Space Model *(Sahlgren, 2006)*

Context Distribution Smoothing

- SGNS samples $c' \sim P$ to form **negative** (w, c') examples
- Our analysis assumes P is the unigram distribution $P\,(c) = \frac{\#c}{\sum_{c' \in V_C} \#c'}$
- In practice, it's a **smoothed** unigram distribution

$$P^{0.75}(c) = \frac{(\#c)^{0.75}}{\sum_{c' \in V_C} (\#c')^{0.75}}$$

- Frequent words are sampled less often. This little change makes a big difference.

Slides from: Christopher Manning, Distributed word representations for IR, Stanford University, USA

Linear Relationships in word2vec

These representations are *very good* at encoding similarity and dimensions of similarity!

- Analogies testing dimensions of similarity can be solved quite well just by doing vector subtraction in the embedding space

  Syntactically

  - $x_{apple} - x_{apples} \approx x_{car} - x_{cars} \approx x_{family} - x_{families}$
  - Similarly for verb and adjective morphological forms

  Semantically (Semeval 2012 task 2)

  - $x_{shirt} - x_{clothing} \approx x_{chair} - x_{furniture}$
  - $x_{king} - x_{man} \approx x_{queen} - x_{woman}$

Word Analogies

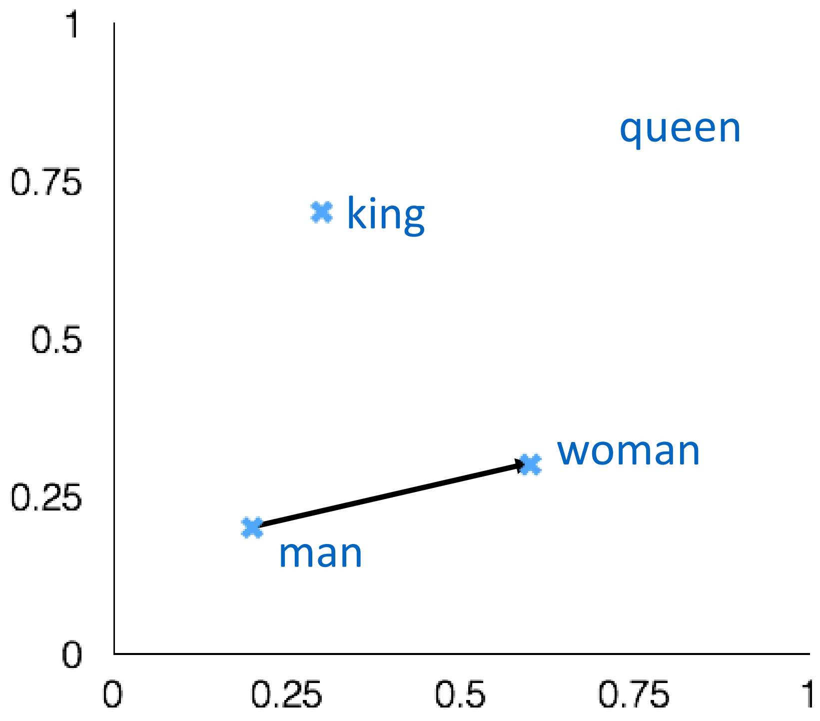
Test for linear relationships, examined by Mikolov et al.

a:b :: c:?

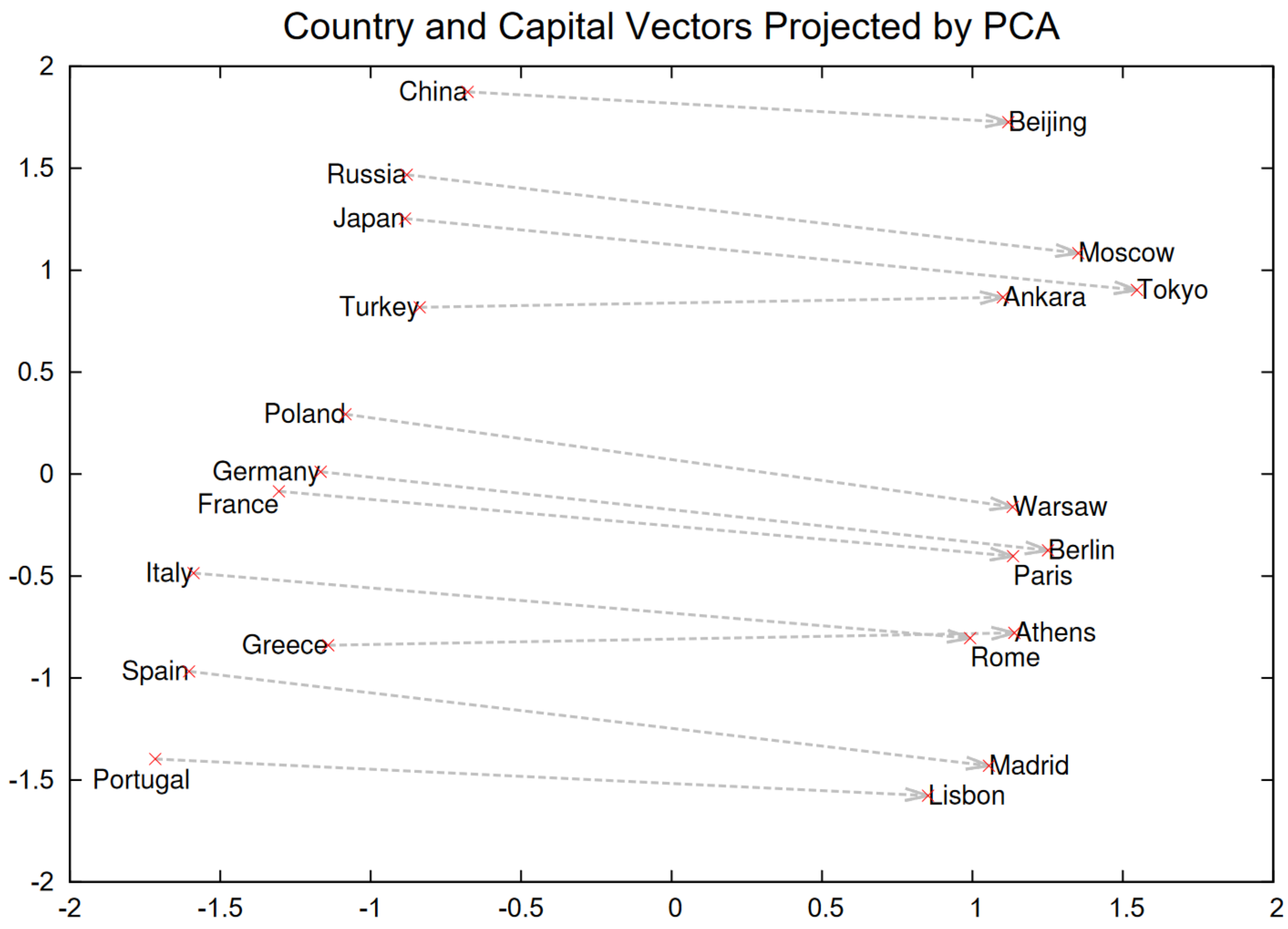
$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

+	king	[0.30 0.70]
−	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



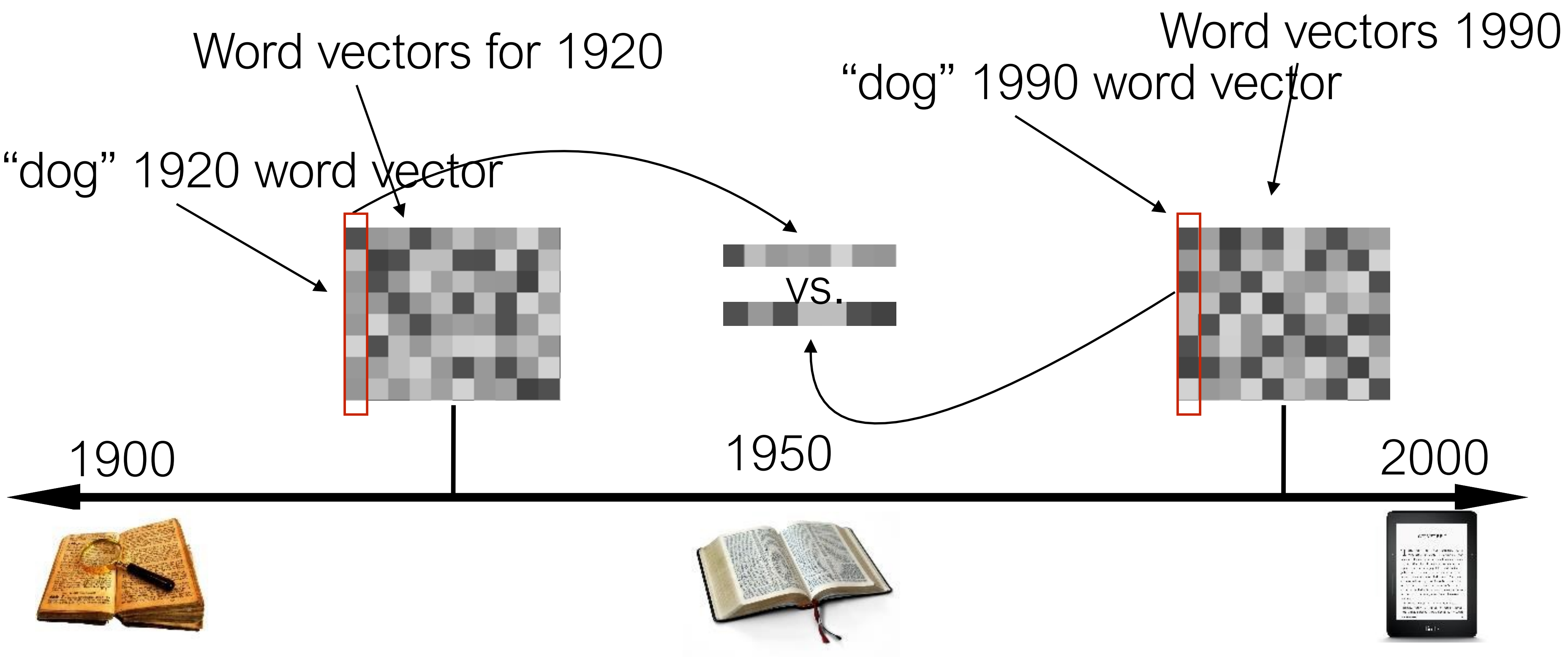
Slides from: Christopher Manning, Distributed word representations for IR, Stanford University, USA



During training no information is provided regarding what a capital city means!

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* (*NIPS'13*)

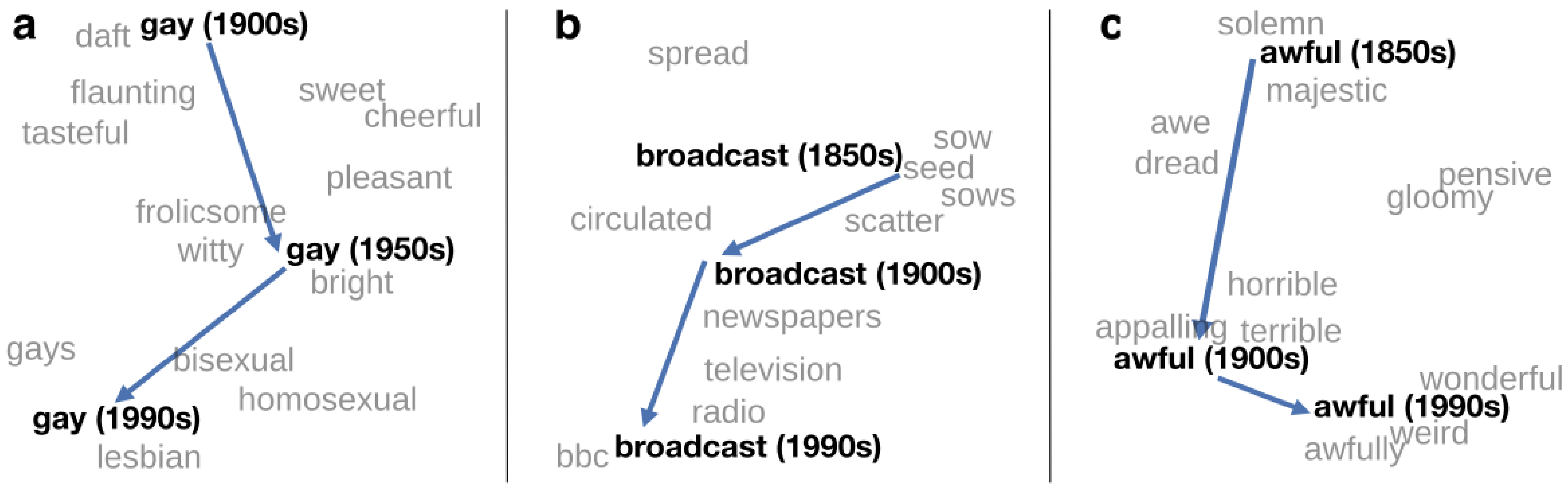
Diachronic word embeddings for studying language change!



Hamilton, William L., Jure Leskovec, and Dan Jurafsky. "Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change." In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1489-1501. 2016.

Visualizing changes

Project 300 dimensions down into 2



~30 million books, 1850-1990, Google Books data

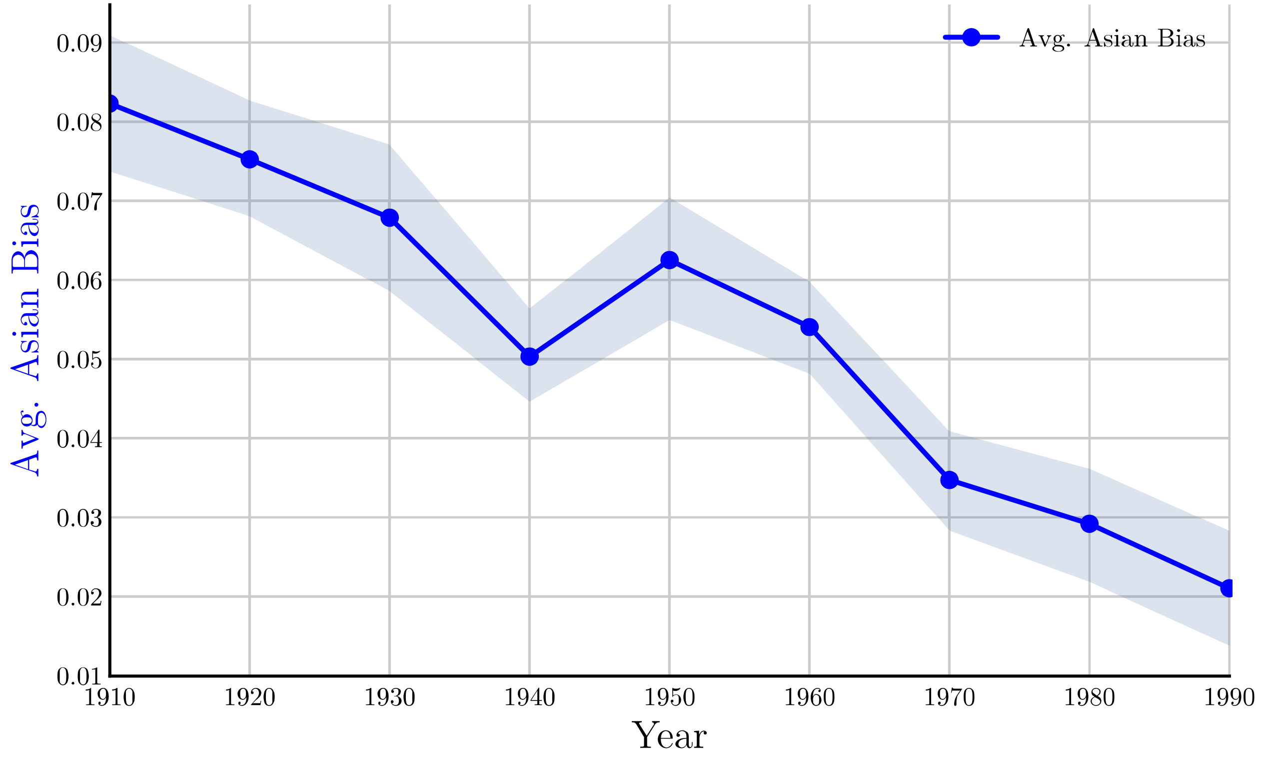
Embeddings as a window onto history

- The cosine similarity of embeddings for decade X for occupations (like teacher) to male vs female names
  - Is correlated with the actual percentage of women teachers in decade X
- Embeddings for competence adjectives are biased towards men:
  - Smart, wise, brilliant, intelligent, resourceful, thoughtful, logical, etc.
- This bias is slowly decreasing.

Garg, Nikhil, Schiebinger, Londa, Jurafsky, Dan, and Zou, James (2018). Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, *115*(16), E3635–E3644

Change in linguistic framing 1910-1990

Change in association of Chinese names with adjectives framed as "othering" (*barbaric*, *monstrous*, *bizarre*)

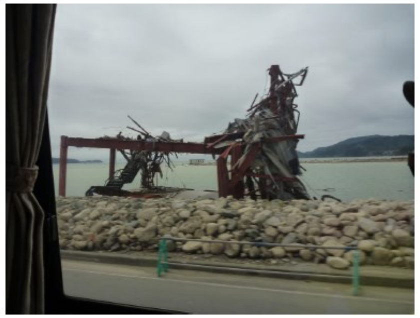


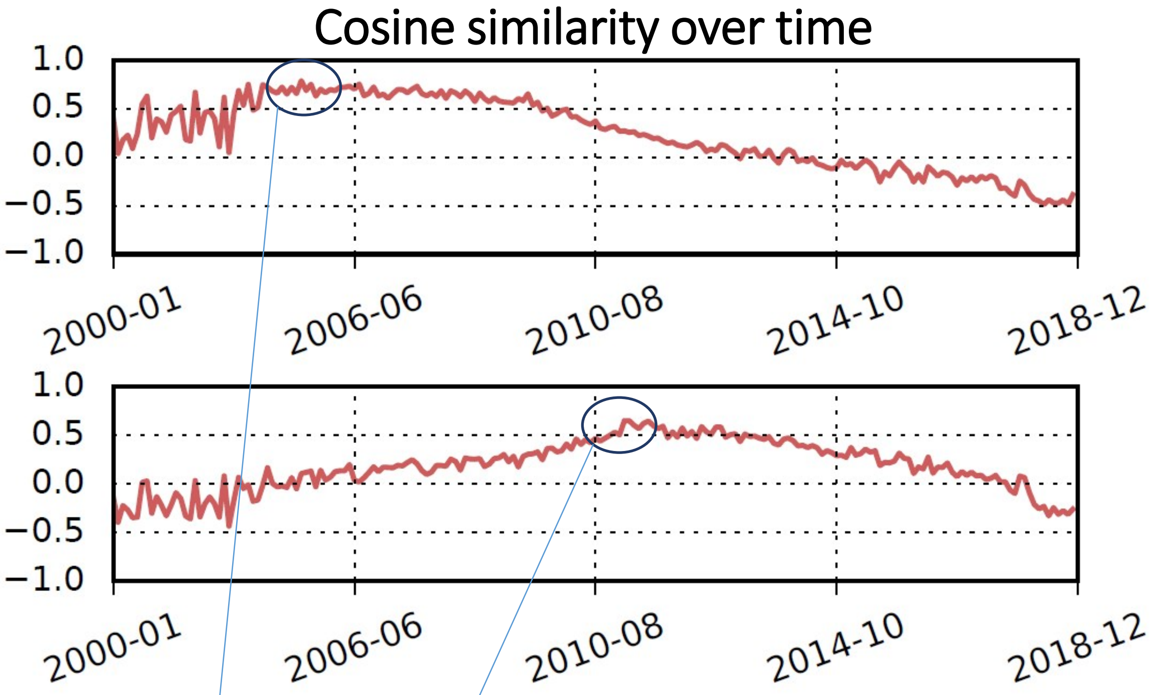
Garg, Nikhil, Schiebinger, Londa, Jurafsky, Dan, and Zou, James (2018). Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, *115*(16), E3635–E3644

Multimodal Diachronic embeddings for studying Vision and Language evolution!

Tsunami
Indonesia

Tsunami
Japan



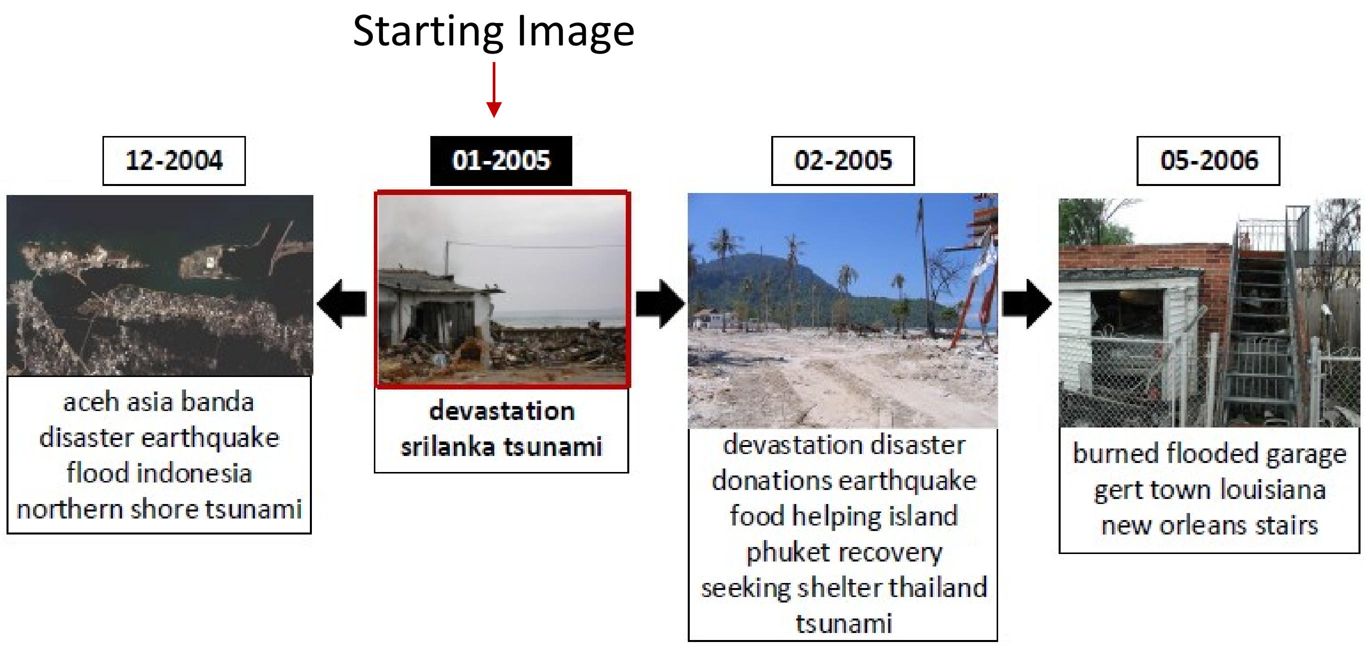


Corresponds to the timestamps of the images

Semedo D. and Magalhaes J. 2019. Diachronic Cross-modal Embeddings.
In *Proceedings of the 27th ACM International Conference on Multimedia* (*MM '19*).

Multimodal Evolution – Summarizing Trajectories

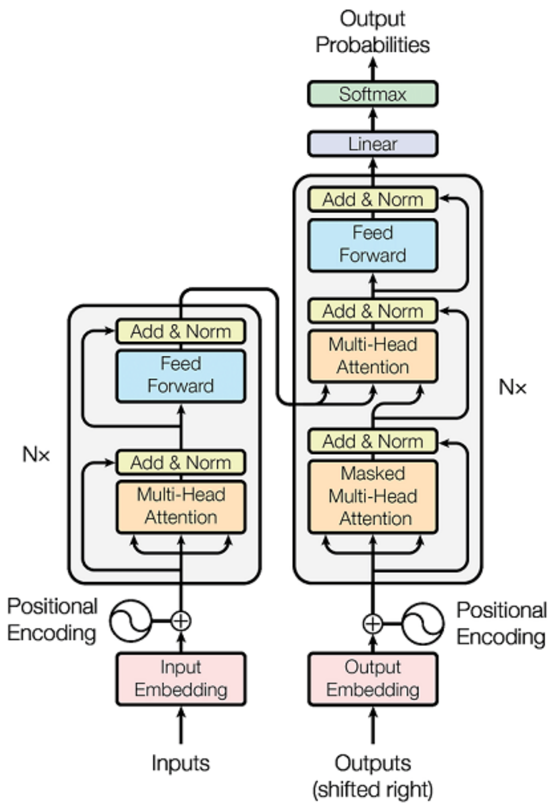
Automatically generate summaries given a single Image/Text, **based on temporal trajectories**



If you are interested in working in these kind of problems, feel free to contact us!

What about the State-of-the-art?

- BERT is a deep learning model that has achieved state-of-the-art results on a wide variety of NLP tasks.
- BERT large instance comprises 345 million parameters!
- Pre-trained on large corpora: BooksCorpus and Wikipedia;
- Self-supervised approach – no need for manual data annotation;
- More about this on today's lab …



Based on the Transformer architecture.

If you want to dive in this model, the Web Search course (next semester) is right for you ☺

Devlin, J., Ming-Wei Chang, Kenton Lee and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." NAACL-HLT (2019).

Out-of-the-box tools

For word2vec:

- Gensim: https://radimrehurek.com/gensim/models/word2vec.html

For Diachronic Word Embeddings:

- https://github.com/williamleif/histwords

For BERT:

- HuggingFace: https://huggingface.co/ (PyTorch and Tensorflow)

Summary: Embed all the things!

- Lots of applications wherever knowing word context or similarity helps prediction:
  - Synonym handling in search, Document aboutness, Ad serving, …

- Fundamental to all other NLP tasks:
  - Language models: from spelling correction to email response
  - Machine translation
  - Sentiment analysis
  - …

- Readings:
  - Dan Jurafsky and James H. Martin, *Speech and Language Processing* (*3rd ed*. *draft*), Chapter 6 https://web.stanford.edu/~jurafsky/slp3/6.pdf

Paper references

- **Word2Vec:** Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013.
    - Omer Levy, Yoav Goldberg, Ido Dagan, Improving Distributional Similarity with Lessons Learned from Word Embeddings, Transactions of ACL, 2015
- **FastTex:** Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. "Enriching word vectors with subword information." *Transactions of the Association for Computational Linguistics* 5 (2017): 135-146.
- **CNN:** Kim, Yoon. "Convolutional Neural Networks for Sentence Classification." Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014.
- **Diachronic Text Embeddings:** Hamilton, William L., Jure Leskovec, and Dan Jurafsky. "Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change." In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1489-1501. 2016.
- **Multimodal Diachronic Embeddings:** David Semedo and Joao Magalhaes. "Diachronic Cross-modal Embeddings". In Proceedings of the 27th ACM International Conference on Multimedia (MM '19). Association for Computing Machinery, New York, NY, USA, pp. 2061–2069.
- **Biases:** Garg, Nikhil, Londa Schiebinger, Dan Jurafsky, and James Zou. "Word embeddings quantify 100 years of gender and ethnic stereotypes." Proceedings of the National Academy of Sciences 115, no. 16 (2018).