

CS 6120/CS 4120: Natural Language Processing

Instructor: Prof. Lu Wang

College of Computer and Information Science

Northeastern University

Webpage: www.ccs.neu.edu/home/luwang

Neural language models

- Skip-grams
- Continuous Bag of Words (CBOW)
- More details can be found at https://cs224d.stanford.edu/lecture_notes/notes1.pdf

Prediction-based models:

An alternative way to get dense vectors

- **Skip-gram** (Mikolov et al. 2013a), **CBOW** (Mikolov et al. 2013b)
- Learn embeddings as part of the process of word prediction
- Train a neural network to predict neighboring words
- Advantages:
 - Fast, easy to train (much faster than SVD)
 - Available online in the `word2vec` package
 - Including sets of pretrained embeddings!

Word2vec

- Popular embedding method
- Very fast to train
- Code available on the web

- Idea: **predict** rather than **count**

Word2vec

- Given a sentence:

... lemon, a **tablespoon of apricot jam** a pinch ...

- Instead of **counting** how often each word w occurs near "*apricot*"
- Train a classifier on a binary **prediction** task:
 - Is w likely to show up near "*apricot*"?
- We don't actually care about this task
 - But we'll take the learned weights (will be discussed later) as the word embeddings

Brilliant insight: Use running text as implicitly supervised training data!

- A word near *apricot*
 - Acts as gold ‘correct answer’ to the question
 - “Is word w likely to show up near *apricot*?”
- No need for hand-labeled supervision
- The idea comes from **neural language modeling**
 - Bengio et al. (2003)
 - Collobert et al. (2011)

Word2Vec: Skip-Gram Task

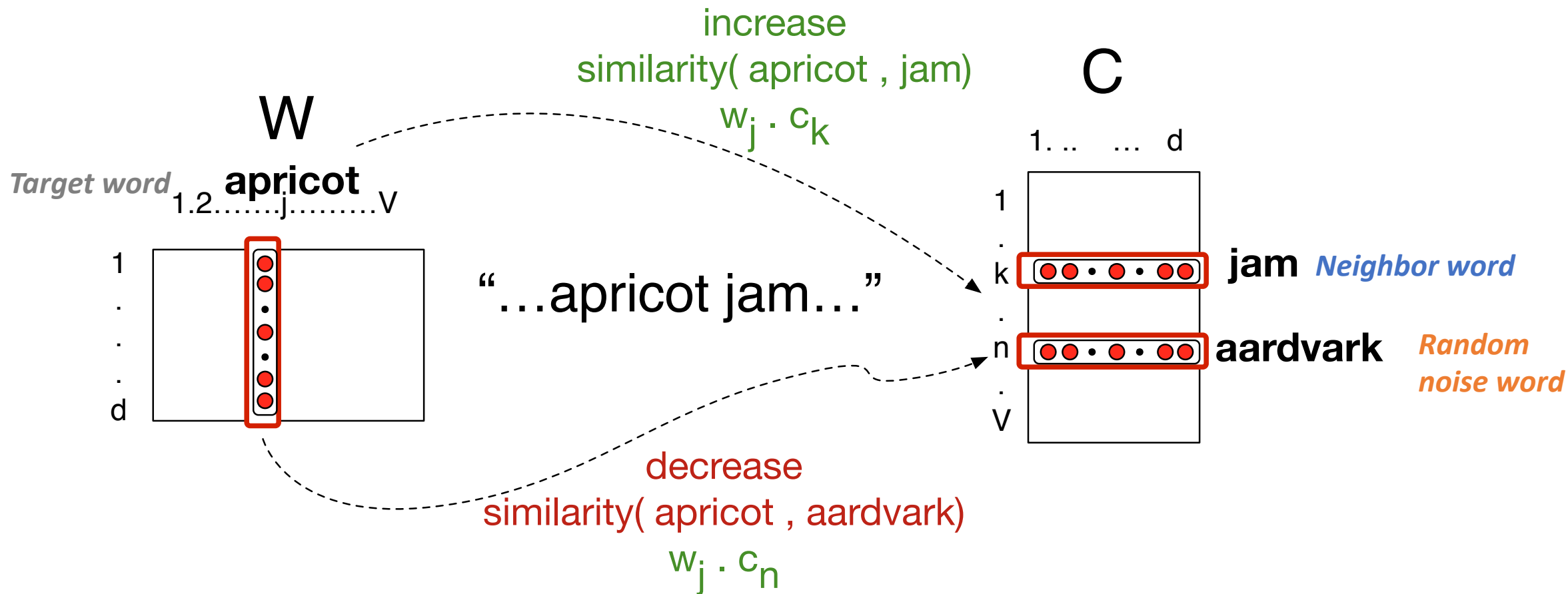
- Now we have positive samples.
- Where do the “negative samples” come from?

Word2Vec: Skip-Gram Task

- Word2vec provides a variety of options. Let's do
 - "skip-gram with negative sampling" (SGNS)

Skip-gram algorithm

1. Treat the target word and a neighboring context word as positive examples.
2. Randomly sample other words in the lexicon to get negative samples
3. Use logistic regression (will discuss formulation later) to train a classifier to distinguish those two cases
4. Use the weights as the embeddings



Skip-gram Training Data

- Training sentence:

... lemon, a **tablespoon of apricot jam** a pinch ...

c1 c2 **target** c3 c4

Assume context words are those in +/- 2 word window

Skip-gram Goal

- Given a tuple (t,c) = target, context
 - $(\textit{apricot}, \textit{jam})$
 - $(\textit{apricot}, \textit{aardvark})$
- Return probability that c is a real context word (or not):
 - $P(+ | t,c) \rightarrow$ positive
 - $P(- | t,c) = 1 - P(+ | t,c) \rightarrow$ negative

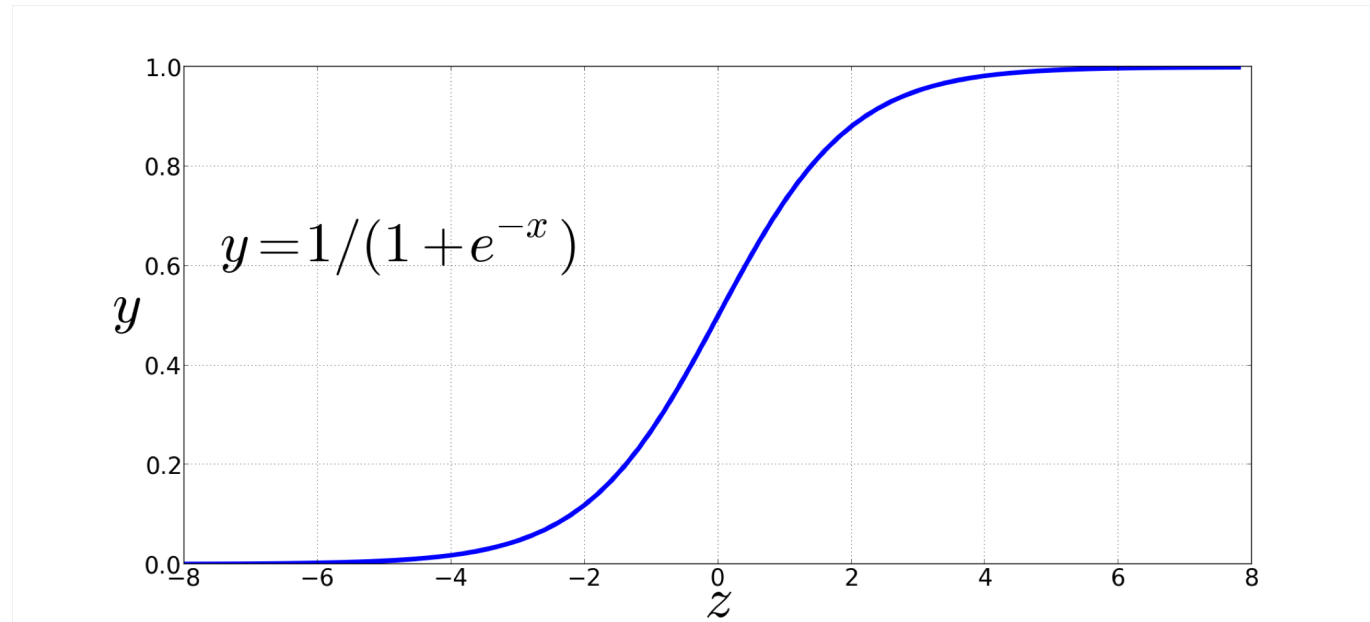
How to compute $p(+ | t, c)$?

- Intuition:
 - Words are likely to appear near similar words
 - Model similarity with dot-product!
 - $\text{Similarity}(t, c) \propto t \cdot c$
- *Problem:*
 - *Dot product is not a probability!*
 - *(Neither is cosine)*

Turning dot product into a probability

- The sigmoid lies between 0 and 1:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Turning dot product into a probability

$$P(+|t, c) = \frac{1}{1 + e^{-t \cdot c}}$$

$$\begin{aligned} P(-|t, c) &= 1 - P(+|t, c) \\ &= \frac{e^{-t \cdot c}}{1 + e^{-t \cdot c}} \end{aligned}$$

For all the context words:

- Assume all context words are independent

$$P(+|t, c_{1:k}) = \prod_{i=1}^k \frac{1}{1 + e^{-t \cdot c_i}}$$

$$\log P(+|t, c_{1:k}) = \sum_{i=1}^k \log \frac{1}{1 + e^{-t \cdot c_i}}$$

Skip-gram Training Data

- Training sentence:

... lemon, a **tablespoon of apricot jam** a pinch ...

c1

c2

t

c3

c4

- Training data: input/output pairs centering on *apricot*
- Assume a +/- 2 word window

Skip-gram Training Data

- Training sentence:

... lemon, a **tablespoon** of **apricot** jam a pinch ...

c1 c2 t c3 c4

- Training data: input/output pairs centering on *apricot*
- Assume a +/- 2 word window

positive examples +

t c

apricot tablespoon

apricot of

apricot preserves

apricot or

Skip-gram Training Data

- Training sentence:

... lemon, a tablespoon of apricot jam a pinch ...

c1 c2 t c3 c4

positive examples +

t

c

apricot tablespoon

apricot of

apricot preserves

apricot or

- For each positive example, we'll create k negative examples.
- Using *noise* words
- Any random word that isn't t

Skip-gram Training Data

- Training sentence:

... lemon, a tablespoon of apricot jam a pinch ...

c1 c2 t c3 c4

positive examples +

t	c
apricot	tablespoon
apricot	of
apricot	preserves
apricot	or

negative examples - $k=2$

t	c	t	c
apricot	aardvark	apricot	twelve
apricot	puddle	apricot	hello
apricot	where	apricot	dear
apricot	coaxial	apricot	forever

Choosing noise words

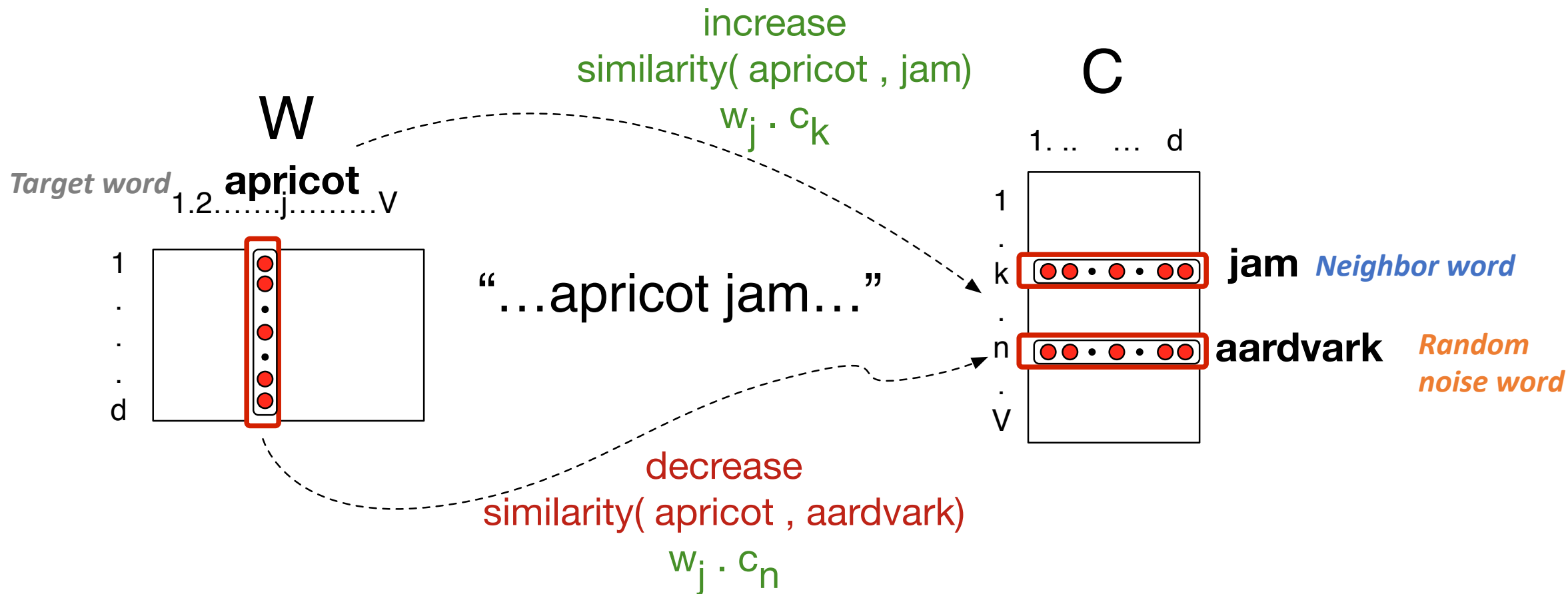
- Could pick w according to their unigram frequency $P(w)$
- More common to chosen then according to $p_\alpha(w)$

$$P_\alpha(w) = \frac{\text{count}(w)^\alpha}{\sum_w \text{count}(w)^\alpha}$$

- $\alpha = \frac{3}{4}$ works well because it gives rare noise words slightly higher probability
- To show this, imagine two events $p(a) = .99$ and $p(b) = .01$:

$$P_\alpha(a) = \frac{.99^{.75}}{.99^{.75} + .01^{.75}} = .97$$

$$P_\alpha(b) = \frac{.01^{.75}}{.99^{.75} + .01^{.75}} = .03$$



Learning the classifier (W and C)

- Iterative process on training data
- Then adjust the word weights to
 - make the positive pairs more likely
 - and the negative pairs less likely

Setup

- Let's represent words as vectors of some length (say 300), randomly initialized.
- So we start with $300 * V$ random parameters
- Over the entire training set, we'd like to adjust those word vectors such that we
 - **Maximize** the similarity of the **target word, context word** pairs (t,c) drawn from the **positive data**
 - **Minimize** the similarity of the (t,c) pairs drawn from the **negative data**

Formally

- We want to maximize the following objective

$$\sum_{(t,c) \in +} \log P(+|t, c) + \sum_{(t,c) \in -} \log P(-|t, c)$$

- Maximize the **+** label for the pairs from the positive training data, and the **-** label for the pairs sample from the negative data.

Focusing on one target word t :

$$\begin{aligned}L(\theta) &= \log P(+|t, c) + \sum_{i=1}^k \log P(-|t, n_i) \\&= \log \sigma(c \cdot t) + \sum_{i=1}^k \log \sigma(-n_i \cdot t) \\&= \log \frac{1}{1 + e^{-c \cdot t}} + \sum_{i=1}^k \log \frac{1}{1 + e^{n_i \cdot t}}\end{aligned}$$

Focusing on one target word t :

$$L(\theta) = \log P(+|t, c) + \sum_{i=1}^k \log P(-|t, n_i)$$

$$= \log \sigma(c \cdot t) + \sum_{i=1}^k \log \sigma(-n_i \cdot t)$$

$$= \log \frac{1}{1 + e^{-c \cdot t}} + \sum_{i=1}^k \log \frac{1}{1 + e^{n_i \cdot t}}$$

Logistic regression

Train using gradient descent (not required)

- Idea: gradually changing W and C
- Finally learns two separate embedding matrices W and C
- Can use W and throw away C , or merge them

Summary: How to learn skip-gram embeddings

- Start with V random 300-dimensional vectors as initial embeddings
- Use logistic regression, the second most basic classifier used in machine learning after naïve bayes
 - Take a corpus and take pairs of words that co-occur as positive examples
 - Take pairs of words that don't co-occur as negative examples
 - Train the classifier to distinguish these by slowly adjusting all the embeddings to improve the classifier performance
 - Throw away the classifier code and keep the embeddings.

(Dense) Word embeddings you can download!

- **Word2vec** (Mikolov et al.)
<https://code.google.com/archive/p/word2vec/>
- **Fasttext** <http://www.fasttext.cc/>
- **Glove** (Pennington, Socher, Manning)
<http://nlp.stanford.edu/projects/glove/>

Evaluating embeddings

- Compare to human scores on word similarity-type tasks:
 - WordSim-353 (Finkelstein et al., 2002)
 - Stanford Contextual Word Similarity (SCWS) dataset (Huang et al., 2012)
- TOEFL dataset:
 - *Levied is closest in meaning to:*
 - ***imposed, believed, requested, correlated***

Properties of embeddings

- Nearest words to some embeddings (Mikolov et al. 2013)

target:	Redmond	Havel	ninjutsu	graffiti	capitulate
	Redmond Wash.	Vaclav Havel	ninja	spray paint	capitulation
	Redmond Washington	president Vaclav Havel	martial arts	grafitti	capitulated
	Microsoft	Velvet Revolution	swordsmanship	taggers	capitulating

Properties of embeddings

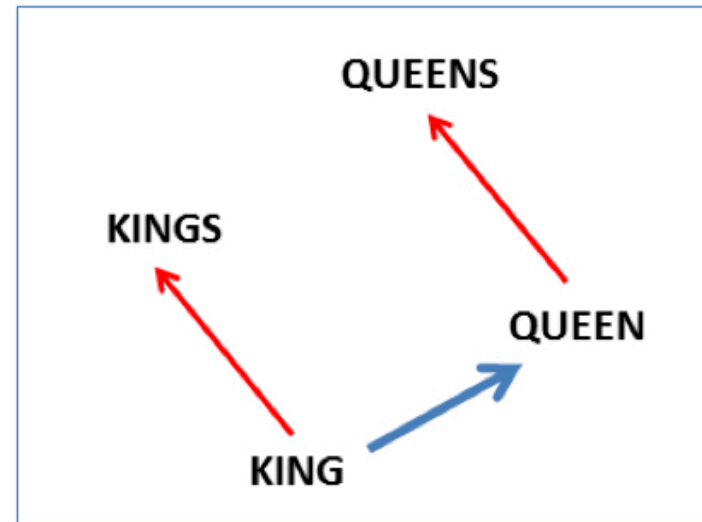
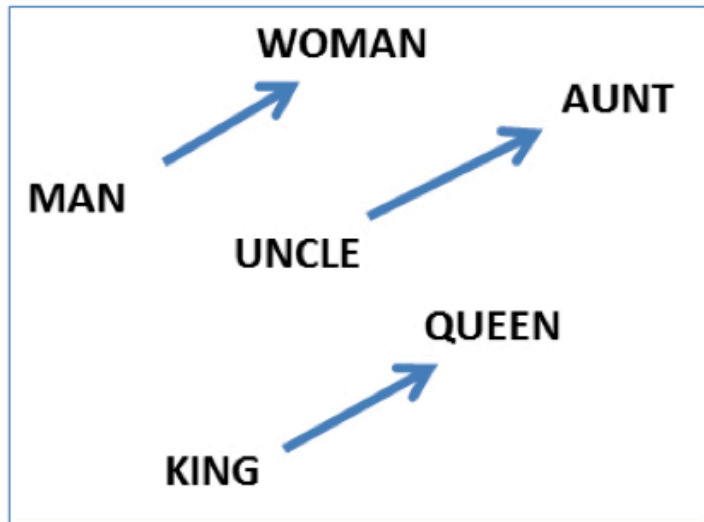
Similarity depends on window size C

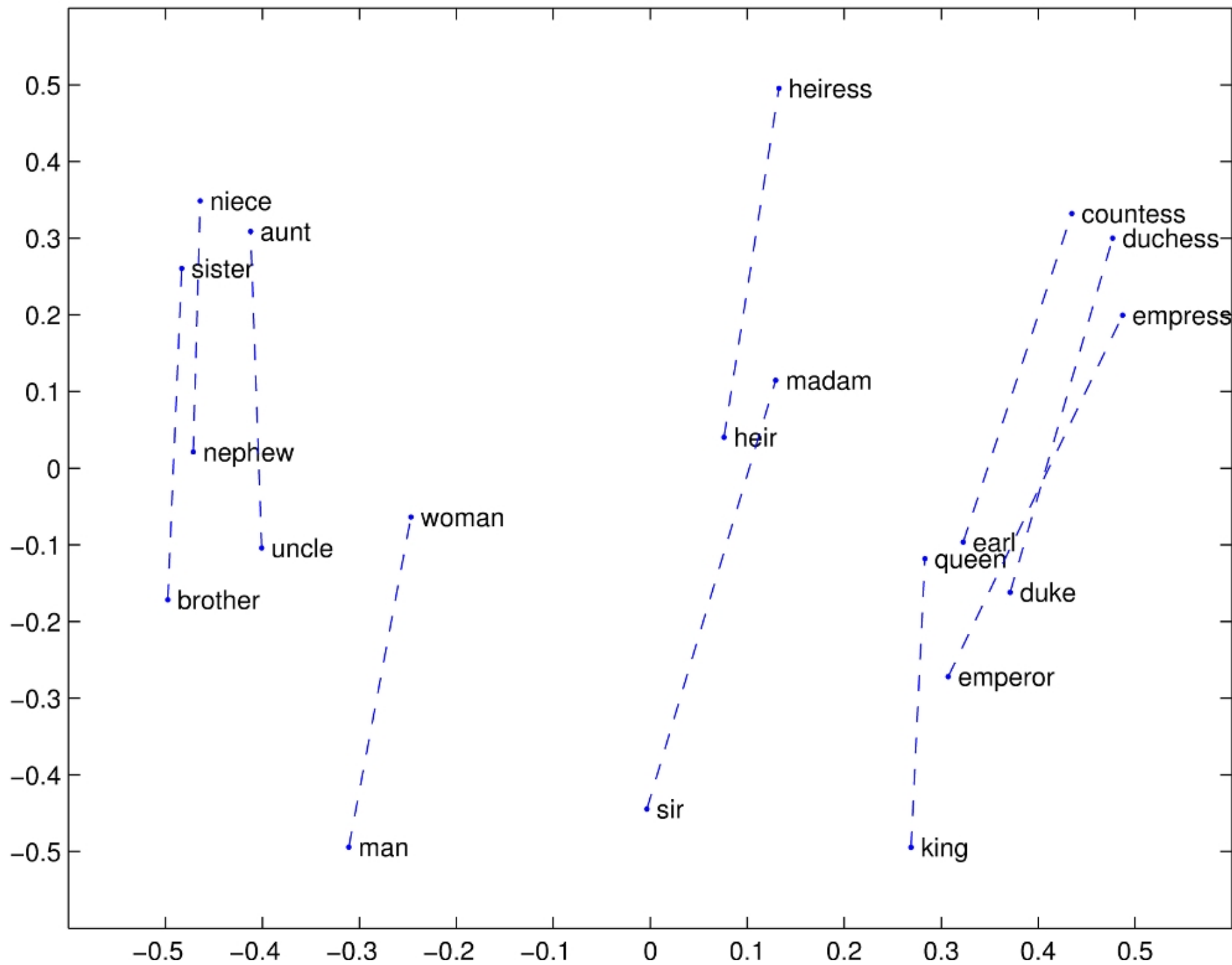
- $C = \pm 2$ The nearest words to *Hogwarts*:
 - *Sunnydale*
 - *Evernight*
- $C = \pm 5$ The nearest words to *Hogwarts*:
 - *Dumbledore*
 - *Malfoy*
 - *halfblood*

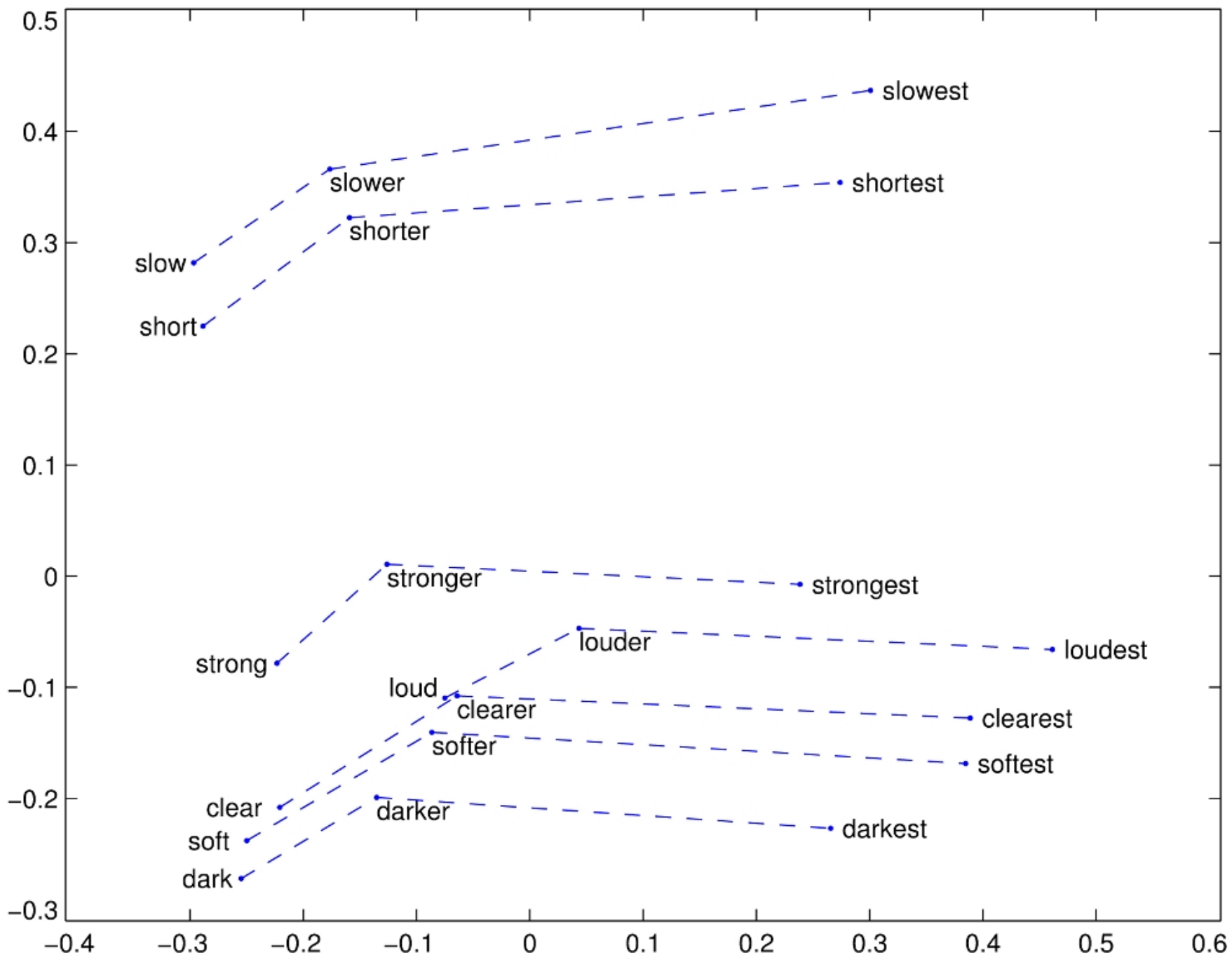
Analogy: Embeddings capture relational meaning!

$\text{vector}('king') - \text{vector}('man') + \text{vector}('woman') \approx \text{vector}('queen')$

$\text{vector}('Paris') - \text{vector}('France') + \text{vector}('Italy') \approx \text{vector}('Rome')$



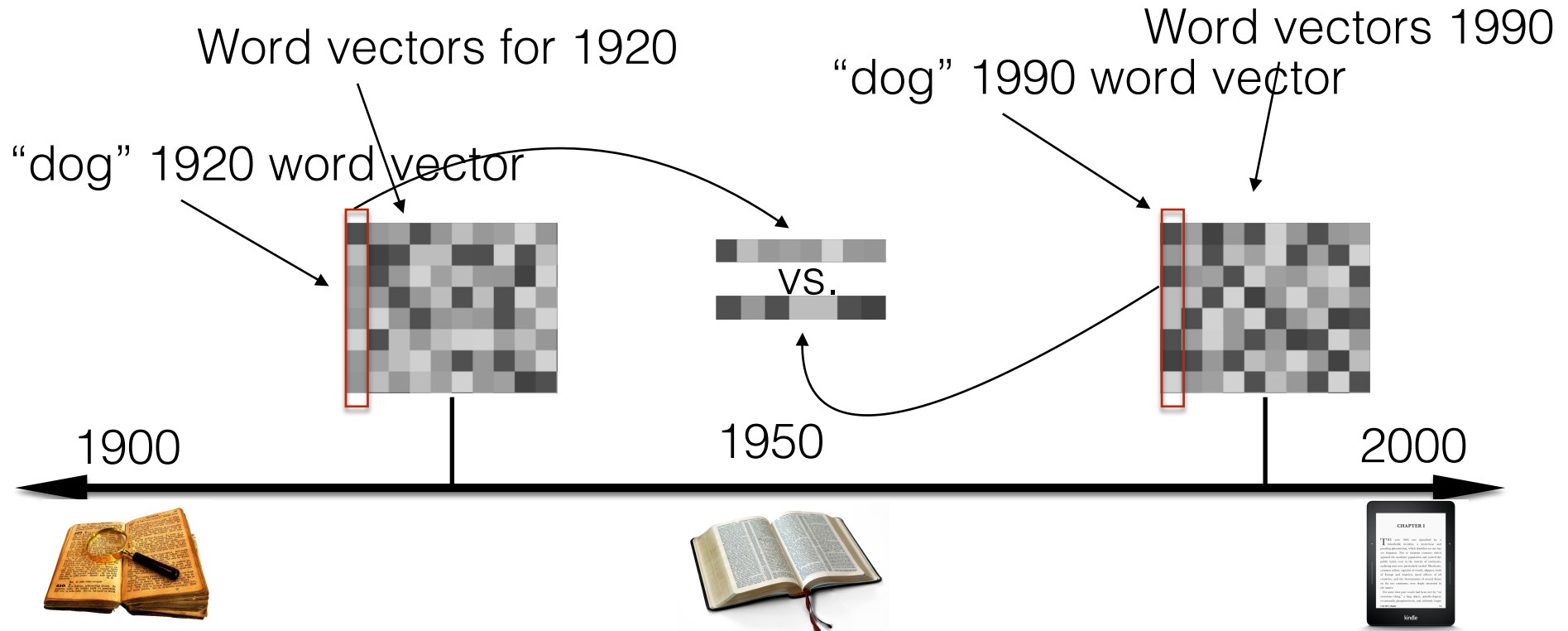




Embeddings can help study word history!

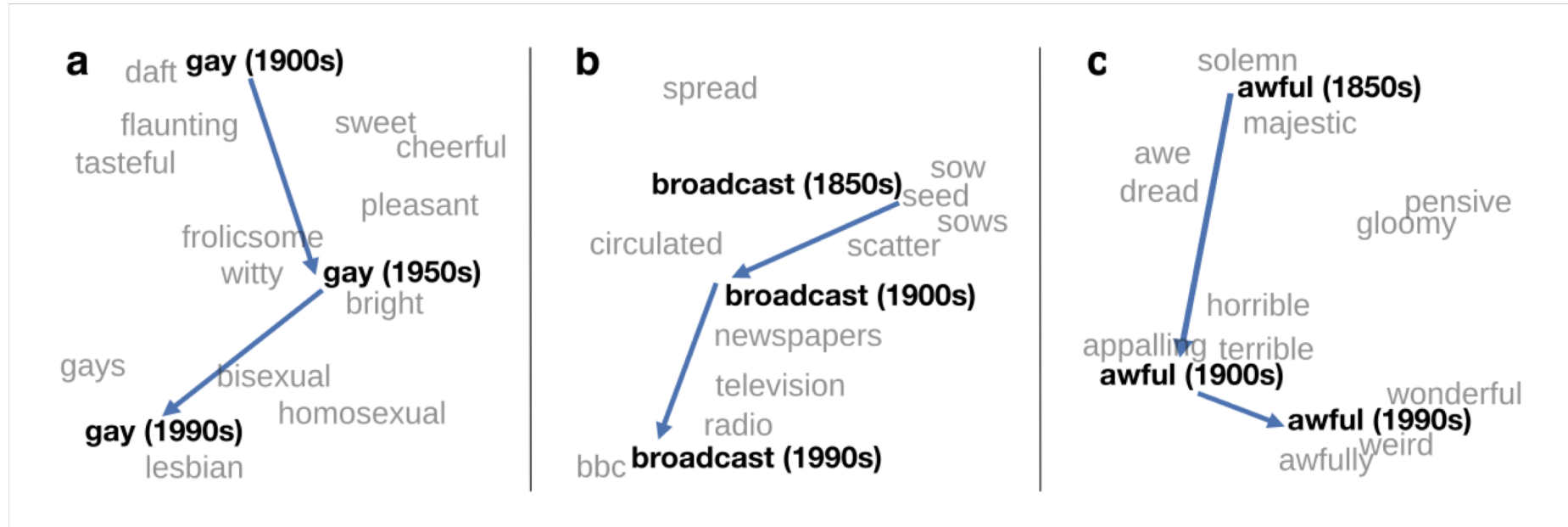
- Train embeddings on old books to study changes in word meaning!!

Diachronic word embeddings for studying language change!



Visualizing changes

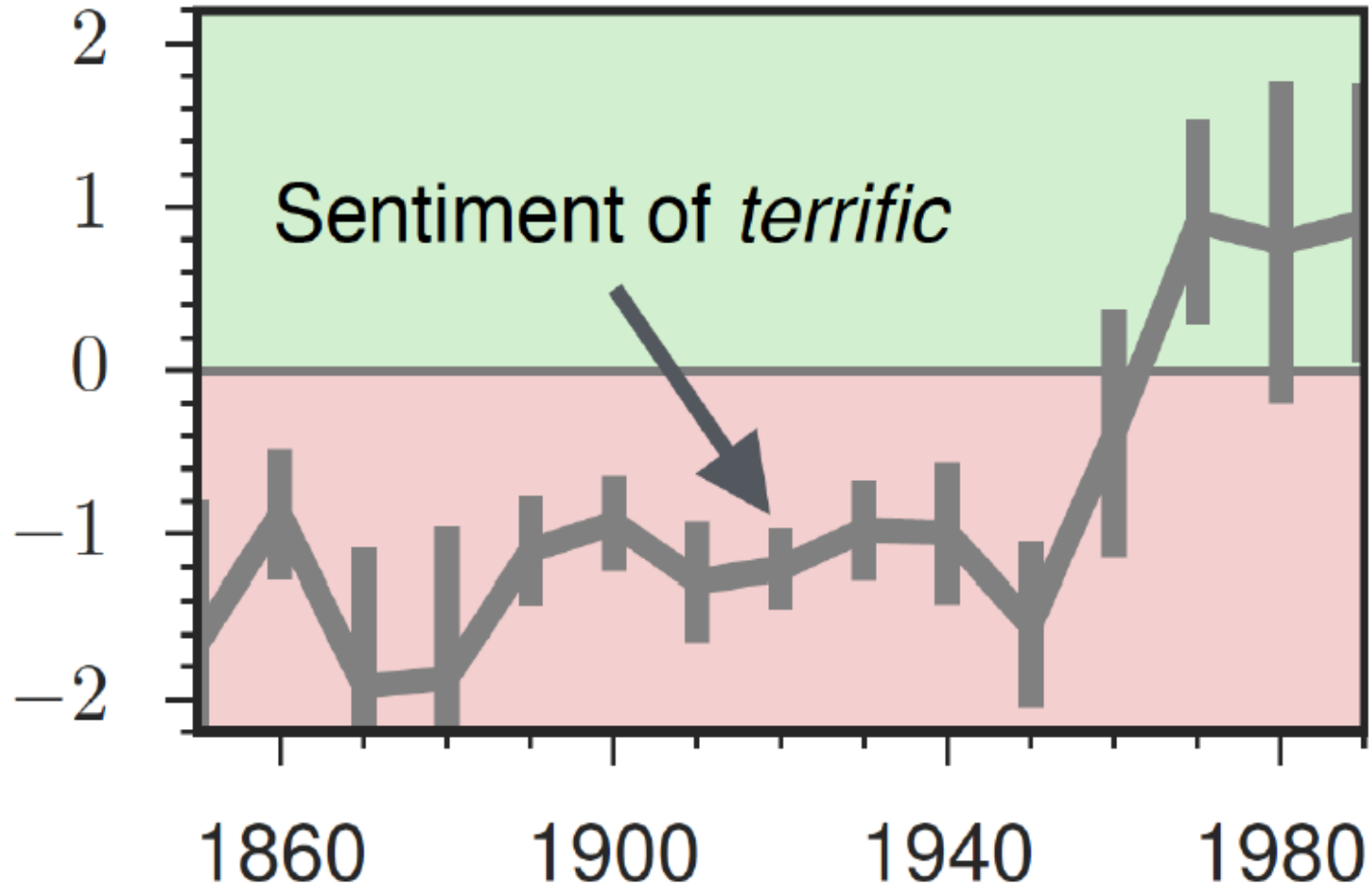
Project 300 dimensions down into 2



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

The evolution of sentiment words

Negative words change faster than positive words



Embeddings and bias

Embeddings reflect cultural bias

Bolukbasi, Tolga, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai.
"Man is to computer programmer as woman is to homemaker? debiasing word embeddings." In *Advances in Neural Information Processing Systems*, pp. 4349-4357. 2016.

- Ask “Paris : France :: Tokyo : x”
 - x = Japan
- Ask “father : doctor :: mother : x”
 - x = nurse
- Ask “man : computer programmer :: woman : x”
 - x = homemaker

Embeddings reflect cultural bias

Caliskan, Aylin, Joanna J. Brusson and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science* 356:6334, 183-186.

- Implicit Association test (Greenwald et al 1998):
 - How associated are **concepts** (*flowers, insects*) & **attributes** (*pleasantness, unpleasantness*)?
 - Studied by measuring timing latencies for categorization.
- Psychological findings on US participants:
 - African-American names are associated with unpleasant words (more than European-American names)
 - Male names associated more with math, female names with arts
 - Old people's names with unpleasant words, young people with pleasant words.

Embeddings reflect cultural bias

Caliskan, Aylin, Joanna J. Brusson and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science* 356:6334, 183-186.

- Implicit Association test (Greenwald et al 1998):
 - How associated are **concepts** (*flowers, insects*) & **attributes** (*pleasantness, unpleasantness*)?
 - Studied by measuring timing latencies for categorization.
- Psychological findings on US participants:
 - African-American names are associated with unpleasant words (more than European-American names)
 - Male names associated more with math, female names with arts
 - Old people's names with unpleasant words, young people with pleasant words.
- Caliskan et al. replication with embeddings:
 - African-American names (*Leroy, Shaniqua*) had a higher GloVe (word embeddings learning method) cosine with unpleasant words (*abuse, stink, ugly*)
 - European American names (*Brad, Greg, Courtney*) had a higher cosine with pleasant words (*love, peace, miracle*)
- Embeddings reflect and replicate all sorts of pernicious biases.

Embeddings as a window onto history

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

- The cosine similarity of embeddings for decade X for occupations or adjectives (e.g. teacher or smart) to male vs female names
 - Find its correlation with the actual percentage of women teachers in decade X

History of biased framings of women

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

- Embeddings for competence adjectives are biased toward men
 - *Smart, wise, brilliant, intelligent, resourceful, thoughtful, logical, etc.*
- This bias is slowly decreasing

Embeddings reflect ethnic stereotypes over time

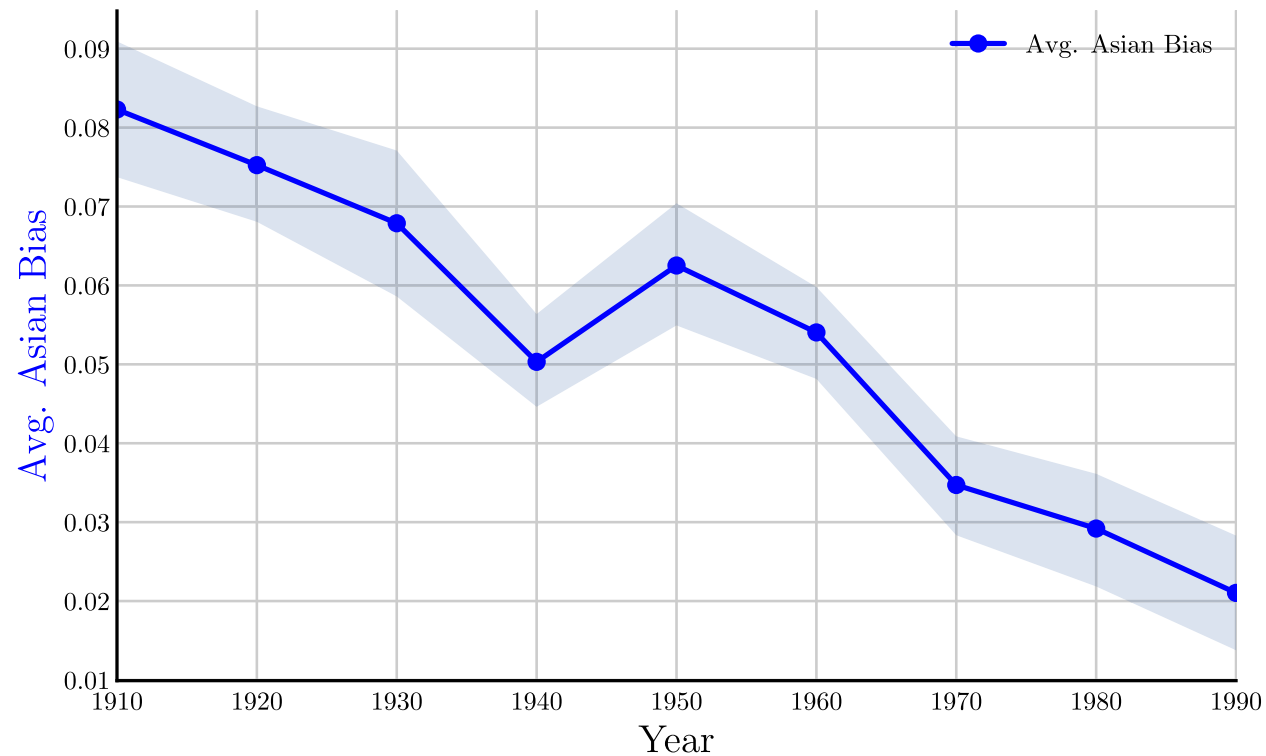
Garg, Nikhil, Schiebinger, Londa, Jurafsky, Dan, and Zou, James (2018). Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16), E3635–E3644

- Princeton trilogy experiments
- Attitudes toward ethnic groups (1933, 1951, 1969) scores for adjectives
 - *industrious, superstitious, nationalistic, etc*
- Cosine of Chinese name embeddings with those adjective embeddings correlates with human ratings.

Change in linguistic framing 1910-1990

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



Changes in framing: adjectives associated with Chinese

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

1910	1950	1990
Irresponsible	Disorganized	Inhibited
Envious	Outrageous	Passive
Barbaric	Pompous	Dissolute
Aggressive	Unstable	Haughty
Transparent	Effeminate	Complacent
Monstrous	Unprincipled	Forceful
Hateful	Venomous	Fixed
Cruel	Disobedient	Active
Greedy	Predatory	Sensitive
Bizarre	Boisterous	Hearty

Directions

- Debiasing algorithms for embeddings
 - Bolukbasi, Tolga, Chang, Kai-Wei, Zou, James Y., Saligrama, Venkatesh, and Kalai, Adam T. (2016). Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, pp. 4349–4357.
- Use embeddings as a historical tool to study bias