AIE1007: Natural Language Processing

L4: Word Embeddings

Autumn 2024

Lecture plan

- Word embeddings = Vector representations of word meaning

Recommended reading:
JM3 6.2-6.4, 6.6

CHAPTER

6

Vector Semantics and Embeddings

荃者所以在鱼，得鱼而忘荃 Nets are for fish;
Once you get the fish, you can forget the net.
言者所以在意，得意而忘言 Words are for meaning;
Once you get the meaning, you can forget the words
庄子(Zhuangzi), Chapter 26

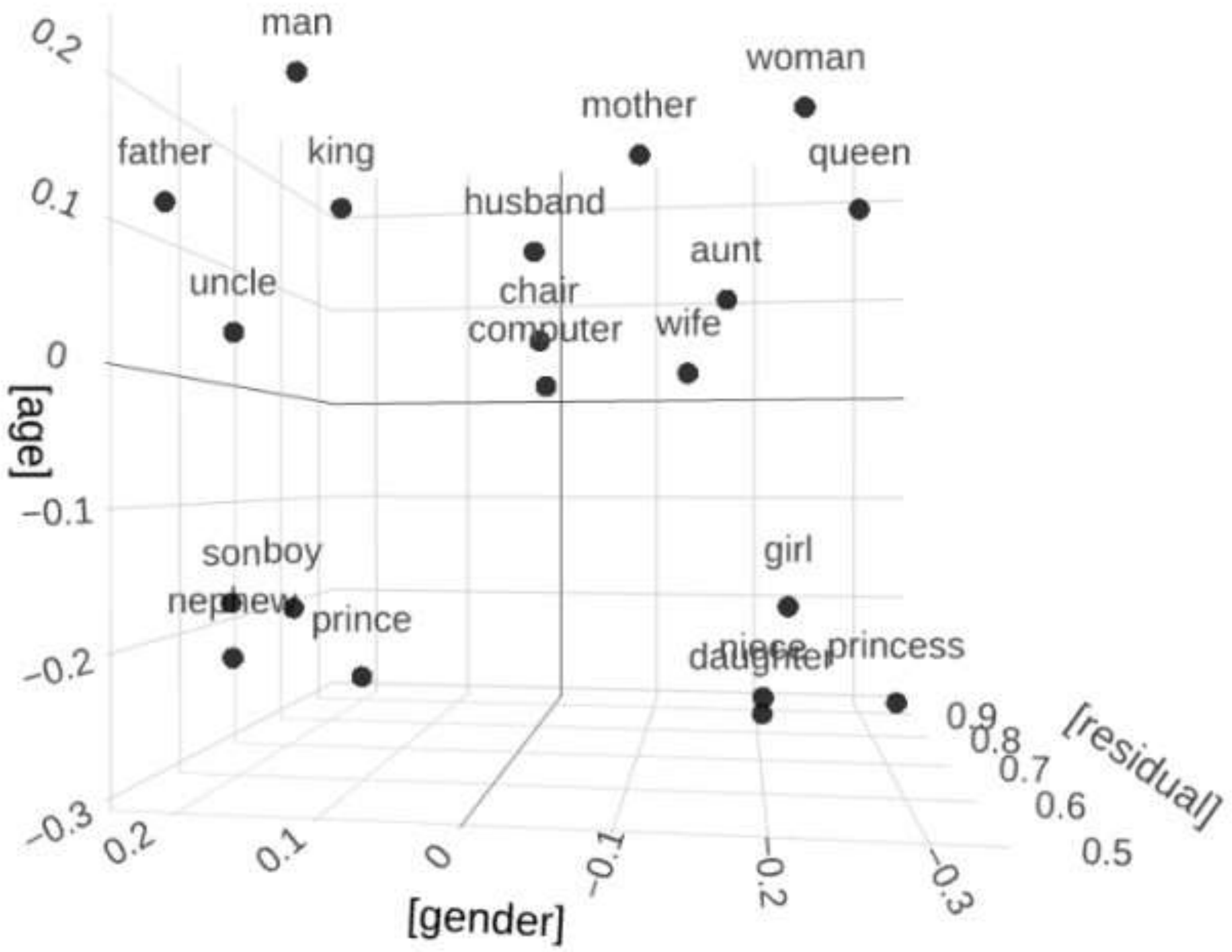
The big idea: model of meaning focusing on similarity

Each word = a vector

$$v_{\text{cat}} = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix} \quad v_{\text{dog}} = \begin{pmatrix} -0.124 \\ 0.430 \\ -0.200 \\ 0.329 \end{pmatrix}$$

$$v_{\text{the}} = \begin{pmatrix} 0.234 \\ 0.266 \\ 0.239 \\ 0.199 \end{pmatrix} \quad v_{\text{language}} = \begin{pmatrix} 0.290 \\ 0.441 \\ 0.762 \\ 0.982 \end{pmatrix}$$

Similar words are "**nearby in the vector space**"



(Bandyopadhyay et al. 2022)

How do we represent words in NLP models?

- n-gram models

$$P(w_1, w_2, ...w_n) = \prod_{i=1}^{n} P(w_i|w_{i-1})$$

$$P(w_i|w_{i-1}) = \frac{\boxed{C(w_{i-1}, w_i)} + \alpha}{C(w_{i-1}) + \alpha|V|}$$

- Naive Bayes

$$\hat{P}(w_i \mid c_j) = \frac{\boxed{\text{Count}(w_i, c_j)} + \alpha}{\sum_{w \in V} \text{Count}(w, c_j) + \alpha|V|}$$

Each word is just a string or indices w_i in the vocabulary list

cat = the 5th word in V

dog = the 10th word in V

cats = the 118th word in V

How do we represent words in NLP models?

- Logistic regression

string match

Var	Definition	Value in Fig. 5.2
x_1	count(positive lexicon) $\in$ doc)	3
x_2	count(negative lexicon) $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(64) = 4.15$

What do words mean?

- **Synonyms**: couch/sofa, car/automobile, filbert/hazelnut
- **Antonyms**: dark/light, rise/fall, up/down
- Some words are not synonyms but they share some element of meaning
  - cat/dog, car/bicycle, cow/horse
- Some words are not similar but they are **related**
  - coffee/cup, house/door, chef/menu
- **Affective meanings** or **connotations**:

vanish	disappear	9.8
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

SimLex-999

valence: the pleasantness of the stimulus

arousal: the intensity of emotion provoked by the stimulus

dominance: the degree of control exerted by the stimulus

	Valence	Arousal	Dominance
courageous	8.05	5.5	7.38
music	7.67	5.57	6.5
heartbreak	2.45	5.65	3.58
cub	6.71	3.95	4.24

(Osgood et al., 1957)

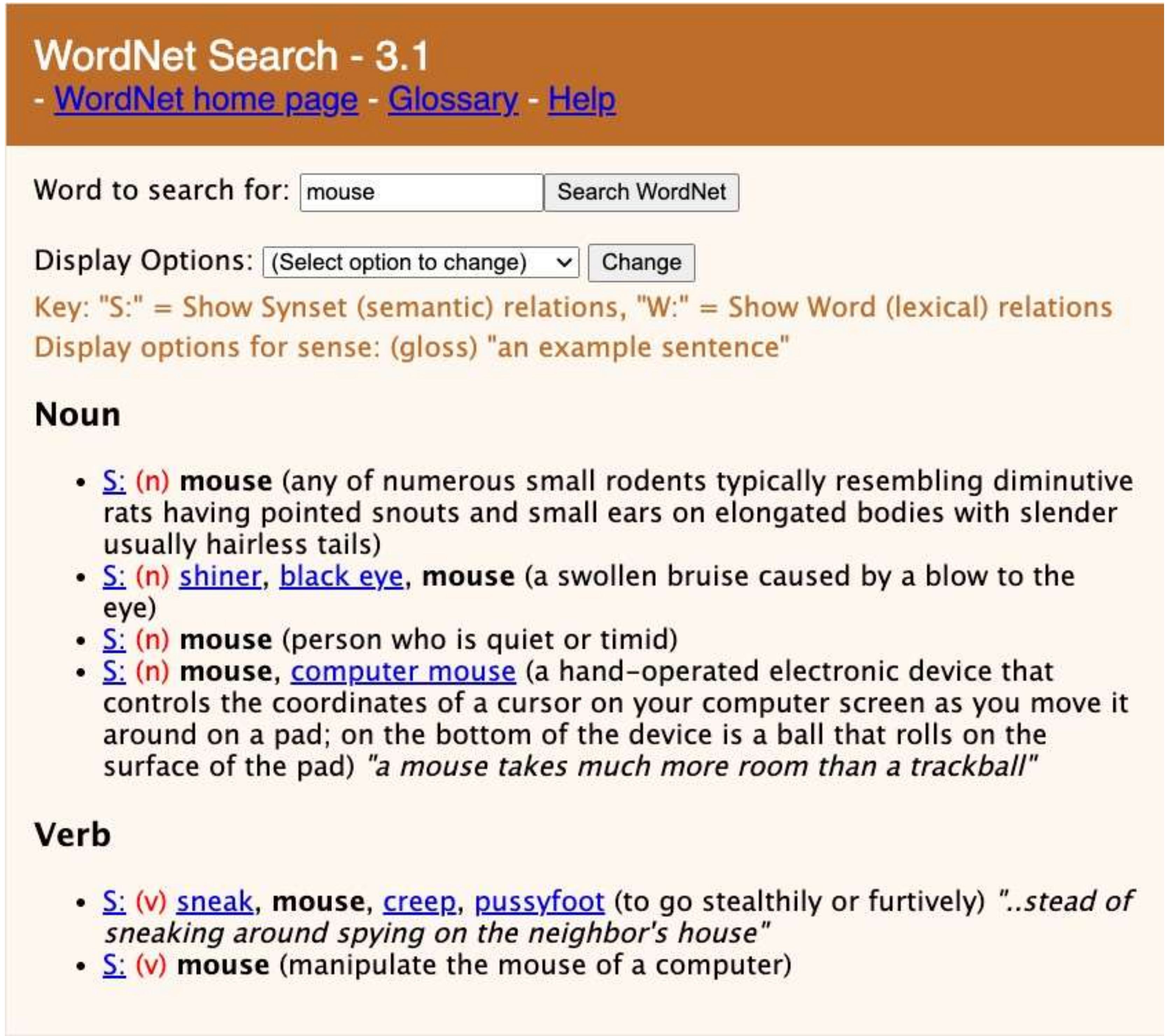
Need for word meaning in NLP models

- With words, a feature is a word identity (= string)
  - Feature 5: `The previous word was "terrible"'
  - Requires **exact same word** to be in the training and testing set

"terrible" ≠ "horrible"

- If we can represent word meaning in vectors:
  - The previous word was vector [35, 22, 17, …]
  - Now in the test set we might see a similar vector [34, 21, 14, …]
  - We can generalize to **similar but unseen** words!!!

Lexical resources


WordNet Search - 3.1
- WordNet home page - Glossary - Help

Word to search for: mouse [Search WordNet]

Display Options: (Select option to change) [Change]

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

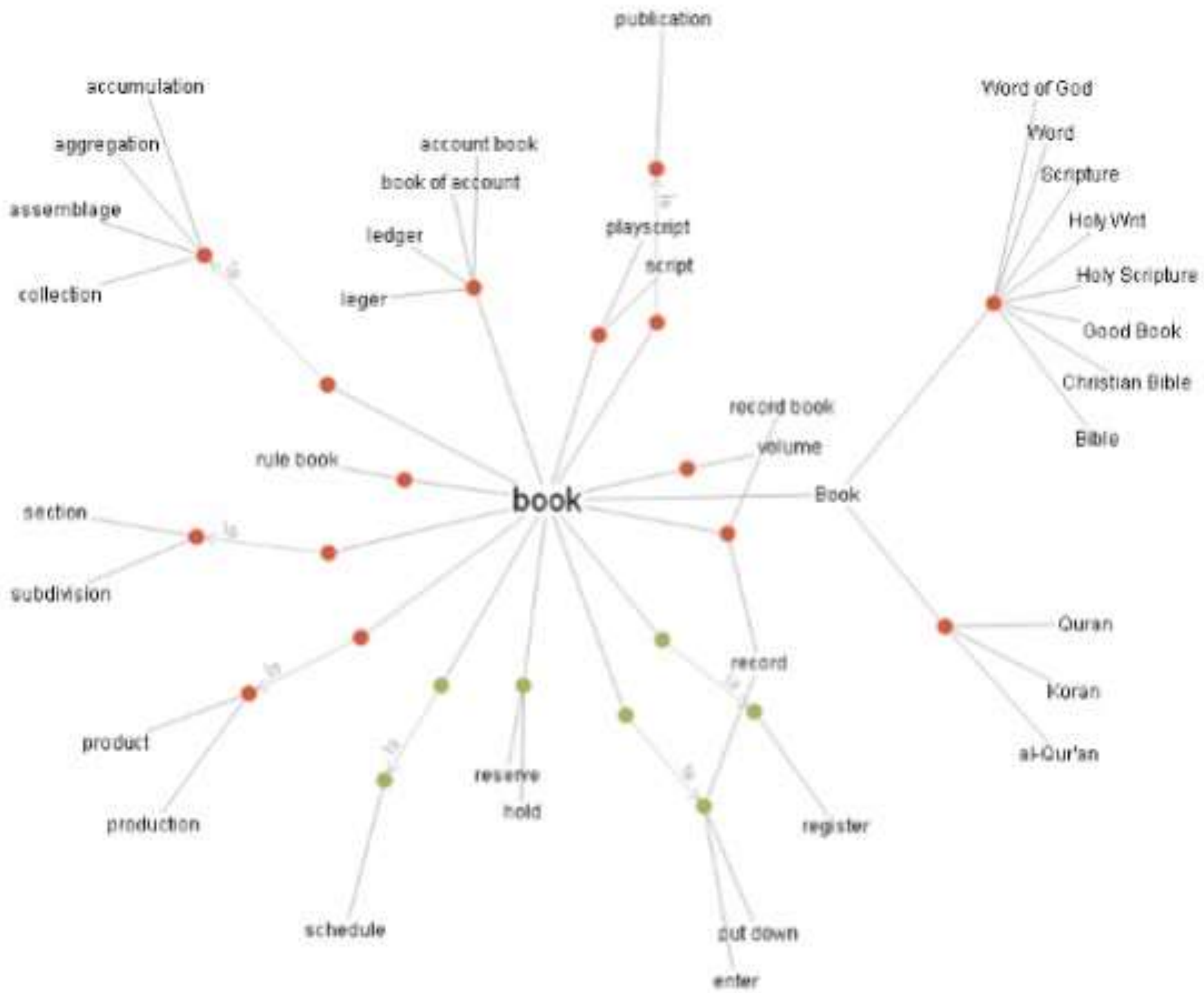
Display options for sense: (gloss) "an example sentence"

Noun

- S: (n) **mouse** (any of numerous small rodents typically resembling diminutive rats having pointed snouts and small ears on elongated bodies with slender usually hairless tails)
- S: (n) shiner, black eye, **mouse** (a swollen bruise caused by a blow to the eye)
- S: (n) **mouse** (person who is quiet or timid)
- S: (n) **mouse**, computer mouse (a hand-operated electronic device that controls the coordinates of a cursor on your computer screen as you move it around on a pad; on the bottom of the device is a ball that rolls on the surface of the pad) *"a mouse takes much more room than a trackball"*

Verb

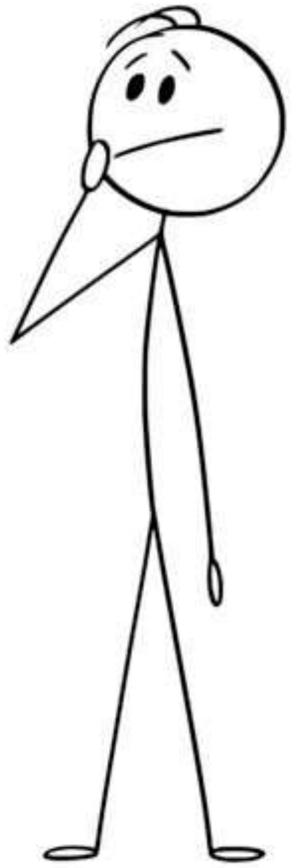
- S: (v) sneak, **mouse**, creep, pussyfoot (to go stealthily or furtively) *"..stead of sneaking around spying on the neighbor's house"*
- S: (v) **mouse** (manipulate the mouse of a computer)



http://wordnetweb.princeton.edu/

(-) Huge amounts of human labor to create and maintain

Distributional hypothesis



- "The meaning of a word is its use in the language" [Wittgenstein PI 43]
- "If A and B have almost identical environments we say that they are synonyms." [Harris 1954]
- "You shall know a word by the company it keeps" [Firth 1957]

Distributional hypothesis

Distributional hypothesis: words that occur in similar **contexts** tend to have similar meanings

J.R.Firth 1957

- "You shall know a word by the company it keeps"
- One of the most successful ideas of modern statistical NLP!

When a word *w* appears in a text, its context is the set of words that appear nearby (within a fixed-size window).

…government debt problems turning into	***banking***	*crises as happened in 2009…*
…saying that Europe needs unified	***banking***	*regulation to replace the hodgepodge…*
…India has just given its	***banking***	*system a shot in the arm…*

These context words will help represent "*banking".*

Distributional hypothesis

"Ongchoi"

Ongchoi is delicious sautéed with garlic

Ongchoi is superb over rice

Ongchoi leaves with salty sauces

Q: What do you think 'Ongchoi' means?

A) a savory snack
B) a green vegetable
C) an alcoholic beverage
D) a cooking sauce

Distributional hypothesis

"Ongchoi"

Ongchoi is delicious sautéed with garlic

Ongchoi is superb over rice

Ongchoi leaves with salty sauces

You may have seen these sentences before:

spinach **sautéed with garlic over rice**

chard stems and **leaves** are **delicious**

collard greens and other **salty** leafty greens

Distributional hypothesis

“Ongchoi”

Ongchoi is a leafty green like spinach, chard or collard greens

空心菜
kangkong
rau muống
...

How we can do the same thing computationally?

- Count the words in the context of ongchoi
- See what other words occur in those contexts

We can represent a word’s context using vectors!

Words and vectors

Q: What is the dimension of each such vector?

A: |V|

First solution: Let's use **word-word co-occurrence counts** to represent the meaning of words!

Each word is represented by the corresponding **row vector**

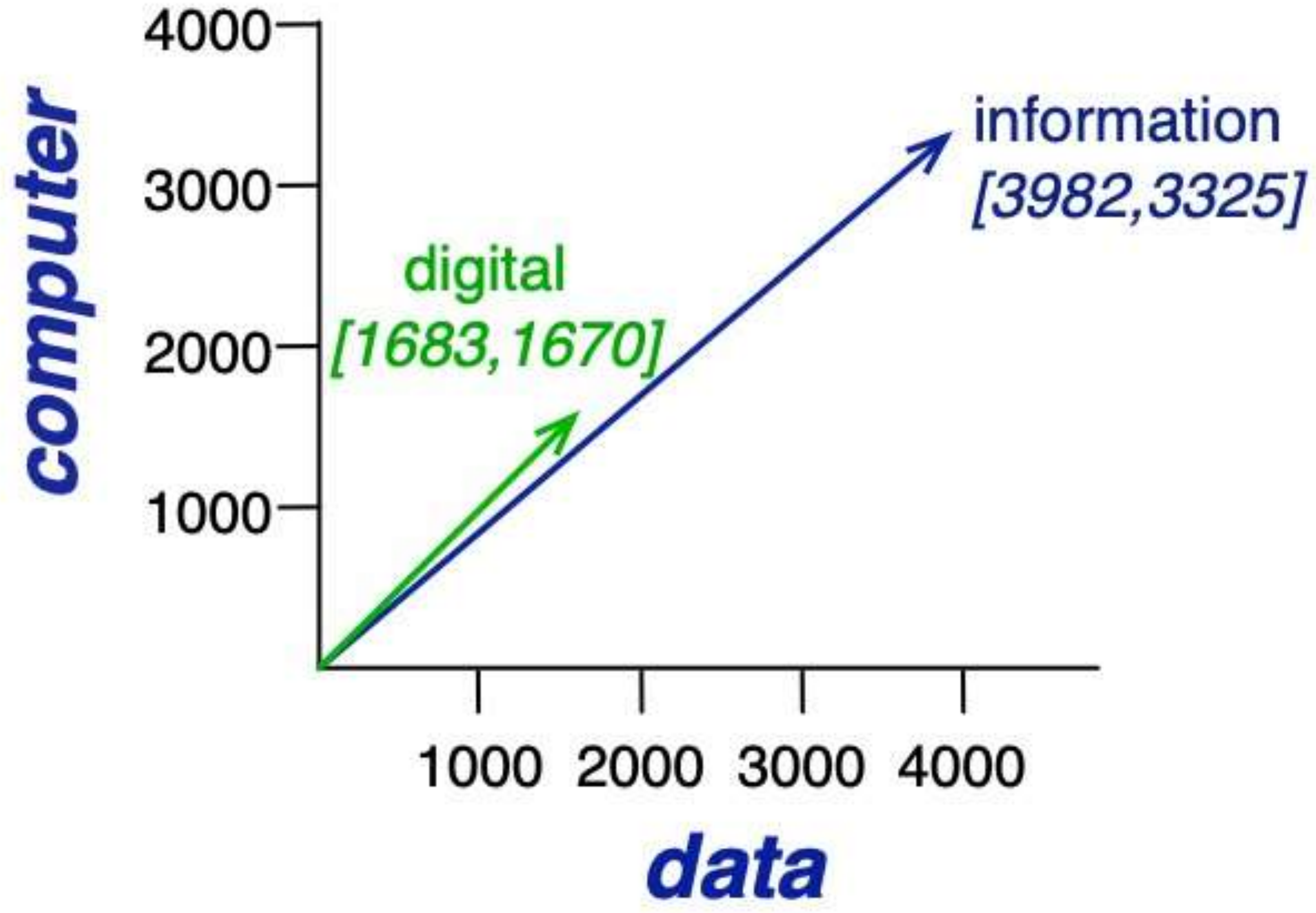
context words:
4 words to the left +
4 words to the right

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

	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	...

Most entries are 0s $\Longrightarrow$ sparse vectors

Measuring similarity



A common similarity metric: **cosine** of the angle between the two vectors (the larger, the more similar the two vectors are)

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$$

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i=1}^{|V|} u_i v_i}{\sqrt{\sum_{i=1}^{|V|} u_i^2}\sqrt{\sum_{i=1}^{|V|} v_i^2}}$$

Q: Why cosine similarity instead of dot product ?

What is the range of $\cos(u, v)$ if u, v are **count vectors**?

(a) [-1, 1]

(b) [0, 1]

(c) [0, $+\infty$)

(d) (, $+\infty$)

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\sum_{i=1}^{|V|} u_i v_i}{\sqrt{\sum_{i=1}^{|V|} u_i^2}\sqrt{\sum_{i=1}^{|V|} v_i^2}}$$

The answer is (b). Cosine similarity ranges between -1 and 1 in general. In this model, all the values of u_i, v_i are non-negative.

Any issues with this model?

Raw frequency count is a bad representation!

- Frequency is clearly useful; if "pie" appears a lot near "cherry", that's useful information.
- But overly frequent words like "the", "it", or "they" also appear a lot near "cherry". They are not very informative about the context.

Solution: use a **weighted function** instead of raw counts!

Pointwise Mutual Information (PMI):

Do events *x* and *y* co-occur more or less than if they were independent?

$$\mathrm{PMI}(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)} \qquad \mathrm{PMI}(w = \text{cherry}, c = \text{pie}) = \log_2 \frac{P(w = \text{cherry}, c = \text{pie})}{P(w = \text{cherry})P(c = \text{pie})}$$

Positive Pointwise Mutual Information (PPMI)

- PMI ranges from to $+\infty$
- PMI$(w, c) > 0 \Longrightarrow \quad P(w, c) > P(w)P(c)$
- PMI$(w, c) < 0 \Longrightarrow \quad P(w, c) < P(w)P(c)$
- Negative values of PMI are frequently not reliable unless the corpus is enormous
    - Unclear whether it is possible to evaluate scores of “unrelatedness” with human judgements
- **A simple fix:** replace all the negative PMI values by 0s

$$\mathrm{PPMI}(w, c) = \max\left(\log_2 \frac{P(w, c)}{P(w)P(c)}, 0\right)$$

PPMI - A running example

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

p(w=information,c=data) = 3982/ 11716 = .3399

p(w=information) = 7703/11716 = .6575

p(c=data) = 5673/11716 = .4842

$$p(w_i) = \frac{\sum_{j=1}^{C} f_{ij}}{N} \qquad p(c_j) = \frac{\sum_{i=1}^{W} f_{ij}}{N}$$

	p(w,context)					p(w)
	computer	data	result	pie	sugar	p(w)
cherry	0.0002	0.0007	0.0008	0.0377	0.0021	0.0415
strawberry	0.0000	0.0000	0.0001	0.0051	0.0016	0.0068
digital	0.1425	0.1436	0.0073	0.0004	0.0003	0.2942
information	0.2838	0.3399	0.0323	0.0004	0.0011	0.6575
p(context)	0.4265	0.4842	0.0404	0.0437	0.0052	

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

	computer	data	result	pie	sugar	count(w)
cherry	2	8	9	442	25	486
strawberry	0	0	1	60	19	80
digital	1670	1683	85	5	4	3447
information	3325	3982	378	5	13	7703
count(context)	4997	5673	473	512	61	11716

p(w=information,c=data) = 3982/111716 = .3399

p(w=information) = 7703/11716 = .6575

p(c=data) = 5673/11716 = .4842

$$p(w_i) = \frac{\sum_{j=1}^{C} f_{ij}}{N} \qquad p(c_j) = \frac{\sum_{i=1}^{W} f_{ij}}{N}$$

Assume that we have a text corpus of 1M tokens, we use 4 words before and 4 words after as context **c** for each word **w**, what is N (the denominator for computing these probabilities) approximately?

(a) 1M

(b) 4M

(c) 8M

(d) not enough information

The answer is (c). For every word w_i in the corpus, we need to collect 8 pairs (w_i, w_{i+j}), for j = -4, -3, -2, -1, 1, 2, 3, 4.

PPMI - A running example

	p(w,context)					p(w)
	computer	data	result	pie	sugar	p(w)
cherry	0.0002	0.0007	0.0008	0.0377	0.0021	0.0415
strawberry	0.0000	0.0000	0.0001	0.0051	0.0016	0.0068
digital	0.1425	0.1436	0.0073	0.0004	0.0003	0.2942
information	0.2838	0.3399	0.0323	0.0004	0.0011	0.6575
p(context)	0.4265	0.4842	0.0404	0.0437	0.0052	

$$\mathrm{PMI}(\mathrm{cherry}, \mathrm{pie}) = \log_2(0.0377/0.0415/0.0437) = 4.38$$

$$\mathrm{PMI}(\mathrm{cherry}, \mathrm{result}) = \log_2(0.0008/0.0415/0.0404) = 1.07$$

$$\mathrm{PMI}(\mathrm{digital}, \mathrm{result}) = \log_2(0.0073/0.2942/0.0404) = -0.70$$

Resulting PPMI matrix (negatives replaced by 0)

	computer	data	result	pie	sugar
cherry	0	0	0	4.38	3.30
strawberry	0	0	0	4.10	5.51
digital	0.18	0.01	0	0	0
information	0.02	0.09	0.28	0	0

Sparse vs dense vectors

- The vectors in the word-word occurrence matrix are
  - **Long**: vocabulary size
  - **Sparse**: most are 0’s

- Alternative: we want to represent words as **short** (50-300 dimensional) & **dense** (real-valued) vectors
  - The basis for modern NLP systems

$$v_{\text{cat}} = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix} \quad v_{\text{dog}} = \begin{pmatrix} -0.124 \\ 0.430 \\ -0.200 \\ 0.329 \end{pmatrix}$$

$$v_{\text{the}} = \begin{pmatrix} 0.234 \\ 0.266 \\ 0.239 \\ 0.199 \end{pmatrix} \quad v_{\text{language}} = \begin{pmatrix} 0.290 \\ 0.441 \\ 0.762 \\ 0.982 \end{pmatrix}$$

Why dense vectors?

- Short vectors are easier to use as **features** in ML systems
- Dense vectors generalize better than explicit counts (points in real space vs points in integer space)
- Sparse vectors can't capture higher-order co-occurrence
  - w_1 co-occurs with "car", w_2 co-occurs with "automobile"
  - They should be similar but they aren't because "car" and "automobile" are distinct dimensions
- In practice, they work better!

How to get short dense vectors?

- **Count-based methods**: Singular value decomposition (SVD) of count matrix

Singular value decomposition (SVD) of PPMI weighted co-occurrence matrix

$$\underset{|V|\times|V|}{\begin{bmatrix} \\ \\ X \\ \\ \\ \end{bmatrix}} = \underset{|V|\times|V|}{\begin{bmatrix} \\ \\ W \\ \\ \\ \end{bmatrix}} \underset{|V|\times|V|}{\begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_V \end{bmatrix}} \underset{|V|\times|V|}{\begin{bmatrix} \\ \\ C \\ \\ \\ \end{bmatrix}}$$

$$\underset{|V|\times|V|}{\begin{bmatrix} \\ \\ X \\ \\ \\ \end{bmatrix}} = \underset{|V|\times k}{\begin{bmatrix} \\ \\ W \\ \\ \\ \end{bmatrix}} \underset{k\times k}{\begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 \\ 0 & 0 & \sigma_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \sigma_k \end{bmatrix}} \begin{bmatrix} C \\ k\times|V| \end{bmatrix}$$

embedding for word i

$$\underset{|V|\times k}{\begin{bmatrix} \\ \boxed{\quad\quad} \\ W \\ \\ \end{bmatrix}}$$

We can approximate the full matrix by only keeping the top k (e.g., 100) singular values!

How to get short dense vectors?

- **Count-based methods**: Singular value decomposition (SVD) of count matrix

- **Prediction-based methods**:
  - Vectors are created by training a classifier to predict whether a word *c* (“pie”) is likely to appear in the context of a word w (“cherry”)
  - Examples: **word2vec** (Mikolov et al., 2013), **Glove** (Pennington et al., 2014), **FastText** (Bojanowski et al., 2017)

***Don’t count, predict!* A systematic comparison of context-counting vs. context-predicting semantic vectors**

Marco Baroni and **Georgiana Dinu** and **Germán Kruszewski**
Center for Mind/Brain Sciences (University of Trento, Italy)

(Baroni et al., 2014)

Also called word **embeddings!**

Word2vec and other variants

Word embeddings

- **Learned** representations from text for representing words
  - Input: a large text corpora, *V*, *d*
    - V: a pre-defined vocabulary
    - d: dimension of word vectors (e.g. 300)
    - Text corpora:
      - Wikipedia + Gigaword 5: 6B tokens
      - Twitter: 27B tokens
      - Common Crawl: 840B tokens
  - Output: $f : V \to \mathbb{R}^d$

$$v_{\text{cat}} = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix} \quad v_{\text{dog}} = \begin{pmatrix} -0.124 \\ 0.430 \\ -0.200 \\ 0.329 \end{pmatrix}$$

$$v_{\text{the}} = \begin{pmatrix} 0.234 \\ 0.266 \\ 0.239 \\ 0.199 \end{pmatrix} \quad v_{\text{language}} = \begin{pmatrix} 0.290 \\ 0.441 \\ 0.762 \\ 0.982 \end{pmatrix}$$

Each word is represented by a low-dimensional (e.g., d = 300), real-valued vector

Each coordinate/dimension of the vector doesn't have a particular interpretation

Trained word embeddings available

- word2vec: https://code.google.com/archive/p/word2vec/
- GloVe: https://nlp.stanford.edu/projects/glove/
- FastText: https://fasttext.cc/

Download pre-trained word vectors

- Pre-trained word vectors. This data is made available under the Public Domain Dedication and License v1.0 whose full text can be found at: http://www.opendatacommons.org/licenses/pddl/1.0/.
  - Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download): glove.6B.zip
  - Common Crawl (42B tokens, 1.9M vocab, uncased, 300d vectors, 1.75 GB download): glove.42B.300d.zip
  - Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors, 2.03 GB download): glove.840B.300d.zip
  - Twitter (2B tweets, 27B tokens, 1.2M vocab, uncased, 25d, 50d, 100d, & 200d vectors, 1.42 GB download): glove.twitter.27B.zip
- Ruby script for preprocessing Twitter data

Differ in algorithms, text corpora, dimensions, cased/uncased…

Applied to many other languages

Word embeddings

- Basic property: similar words have similar vectors

word w^*= “sweden”

$$\arg\max_{w \in V} \cos(e(w), e(w^*))$$

Word	Cosine distance
norway	0.760124
denmark	0.715460
finland	0.620022
switzerland	0.588132
belgium	0.585835
netherlands	0.574631
iceland	0.562368
estonia	0.547621
slovenia	0.531408

$\cos(u, v)$ ranges between -1 and 1

Word embeddings

- Basic property: similar words have similar vectors

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

leptodactylidae

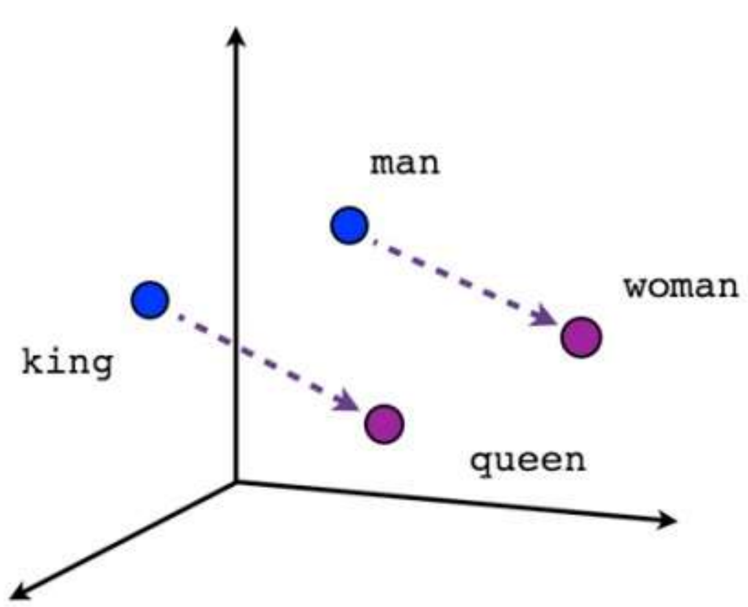
rana

eleutherodactylus

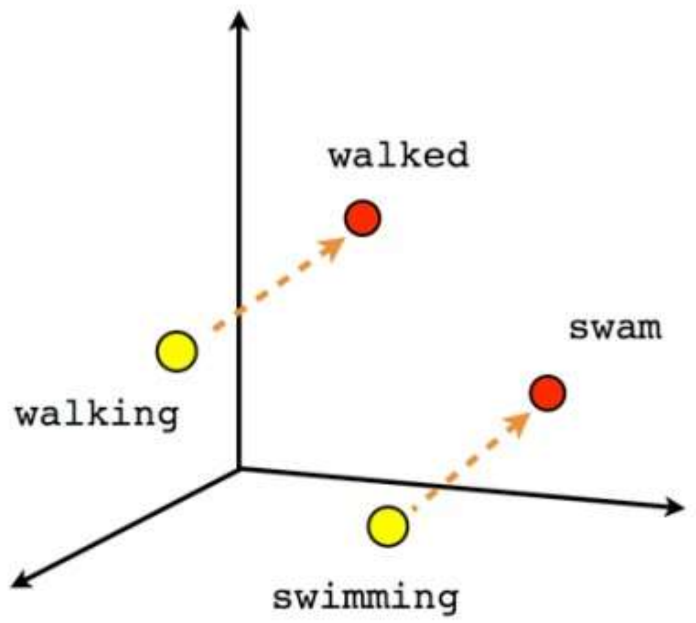
(Pennington et al, 2014): GloVe: Global Vectors for Word Representation

Word embeddings

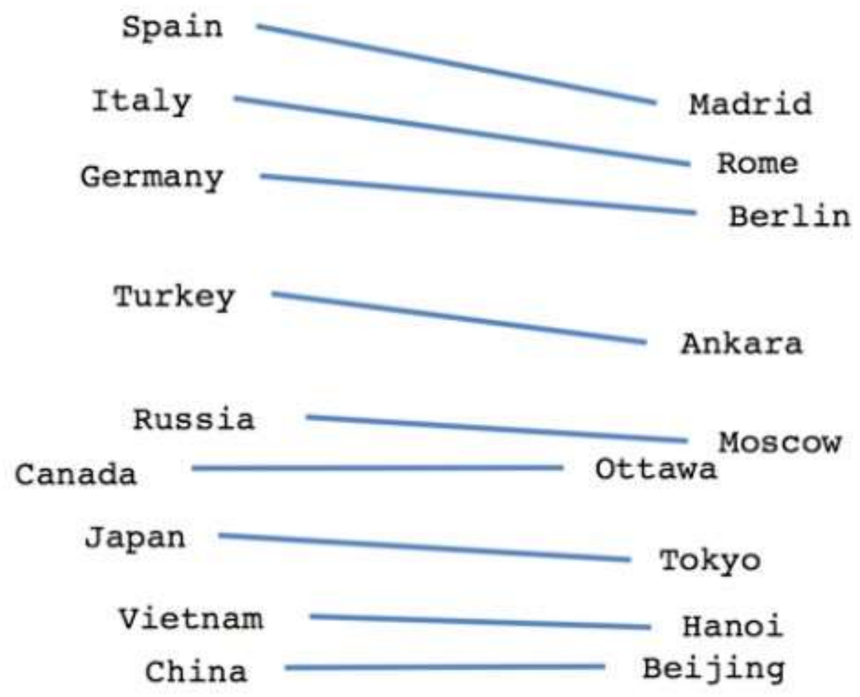
- They have some other nice properties too!



Male-Female



Verb tense



Country-Capital

$$v_{\text{man}} - v_{\text{woman}} \approx v_{\text{king}} - v_{\text{queen}}$$

$$v_{\text{Paris}} - v_{\text{France}} \approx v_{\text{Rome}} - v_{\text{Italy}}$$

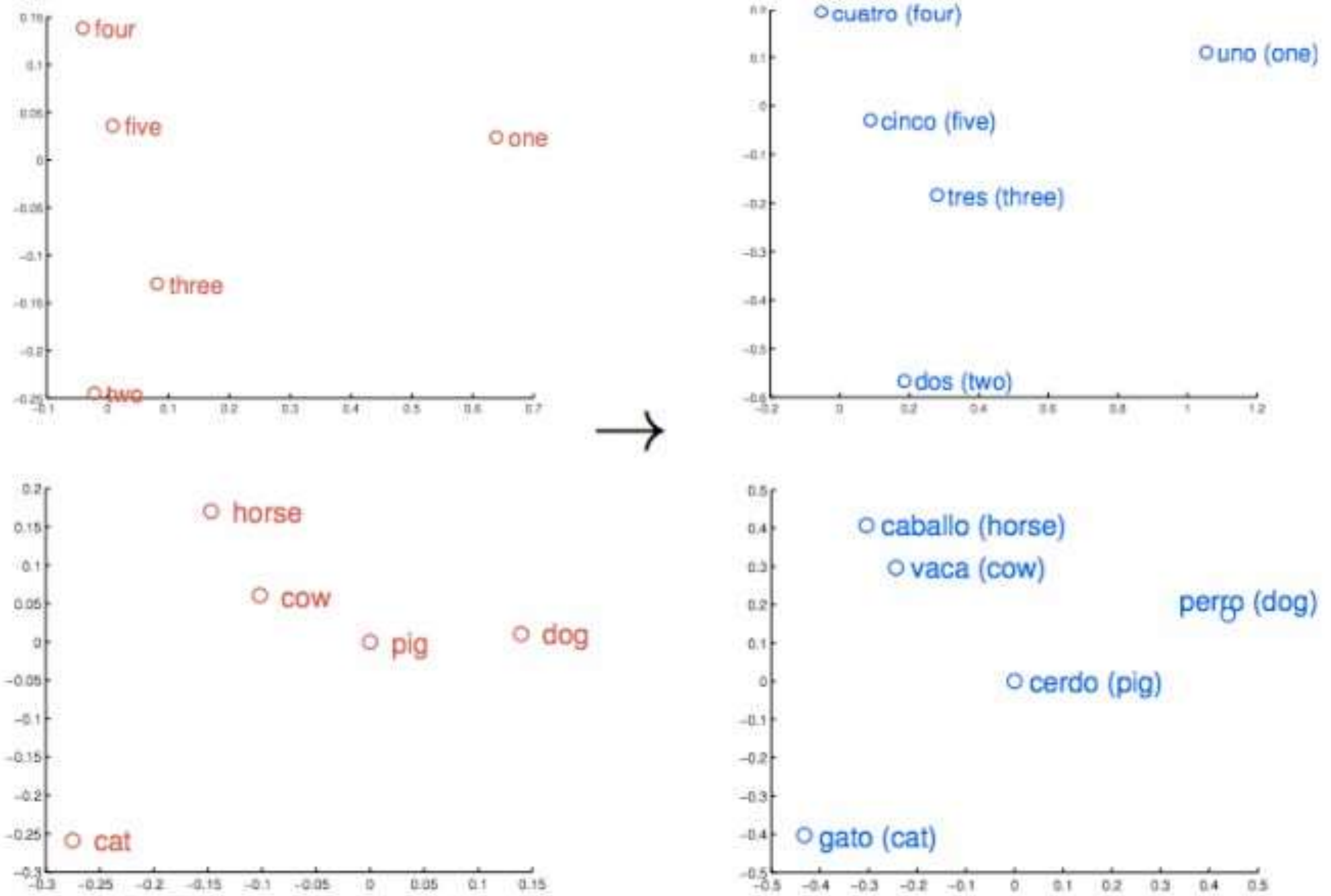
Word analogy test: $a : a^* :: b : b^*$

$$b^* = \arg\max_{w \in V} \cos(e(w), e(a^*) - e(a) + e(b))$$

Word embeddings

- They have some other nice properties too!

$$v(\text{cuatro}) \approx W v(\text{four})$$

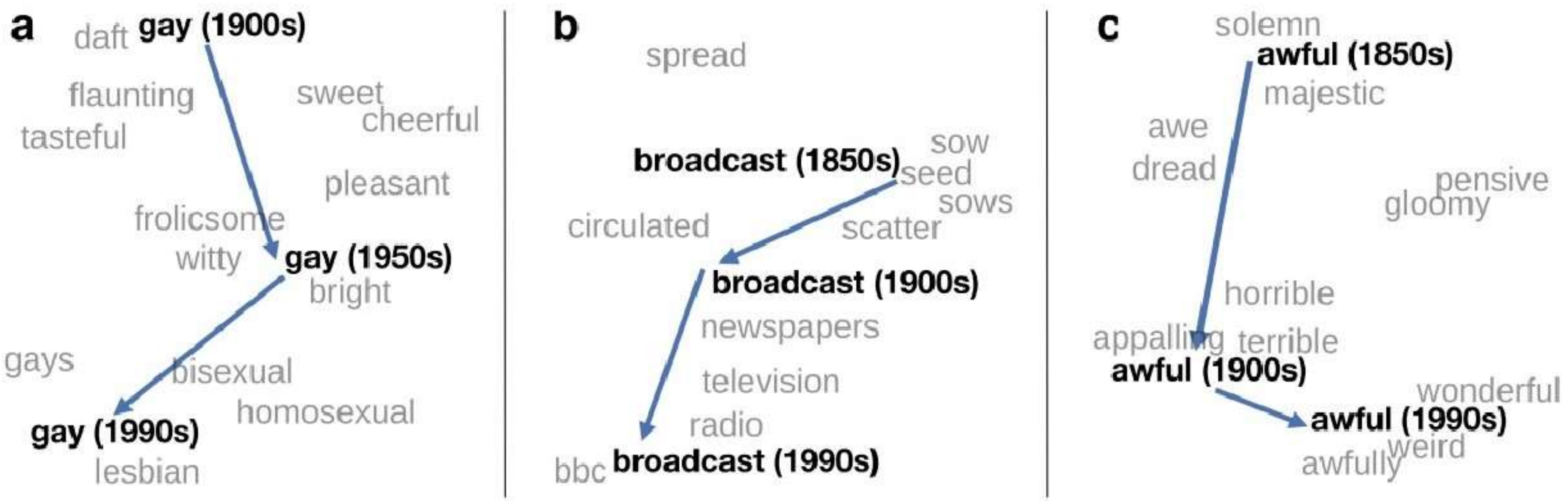


(Mikolov et al, 2013): Exploiting Similarities among Languages for Machine Translation

Embeddings as a window onto historical semantics

Train embeddings on different decades of historical text to see meanings shift

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



William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. Proceedings of ACL.

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 *NeurIPS*, 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

Algorithms that use embeddings as part of e.g., hiring searches for programmers, might lead to bias in hiring

Next lecture: word2vec

A classification problem!

- **Key idea:** Use each word to **predict** other words in its context
- Assume that we have a large corpus $w_1, w_2, \ldots, w_T \in V$
- Context: a fixed window of size $2\boldsymbol{m}$ (m = 2 in the example)