

# Lecture 4:

# Multinomial Logistic Regression and Word Embeddings

*Instructor: Swabha Swayamdipta*

*USC CSCI 444 NLP*

*Sep 10, 2025*



# Announcements

- Project Pitch Voting
  - Grades will be released once I get all votes
  - Will also release the votes
- Please form teams by 9/15!
- Quiz grades (available at the end of class)
- Other deadlines:
  - HW1 due on 9/17

# Quiz 1 Solutions (Redacted)

# IV. Optimization: Stochastic Gradient Descent

# Our goal: minimize the loss

- Loss function is parameterized by weights:  $\theta = [\mathbf{w}; b]$
- We will represent  $\hat{y}$  as  $f(x; \theta)$  to make the dependence on  $\theta$  more obvious

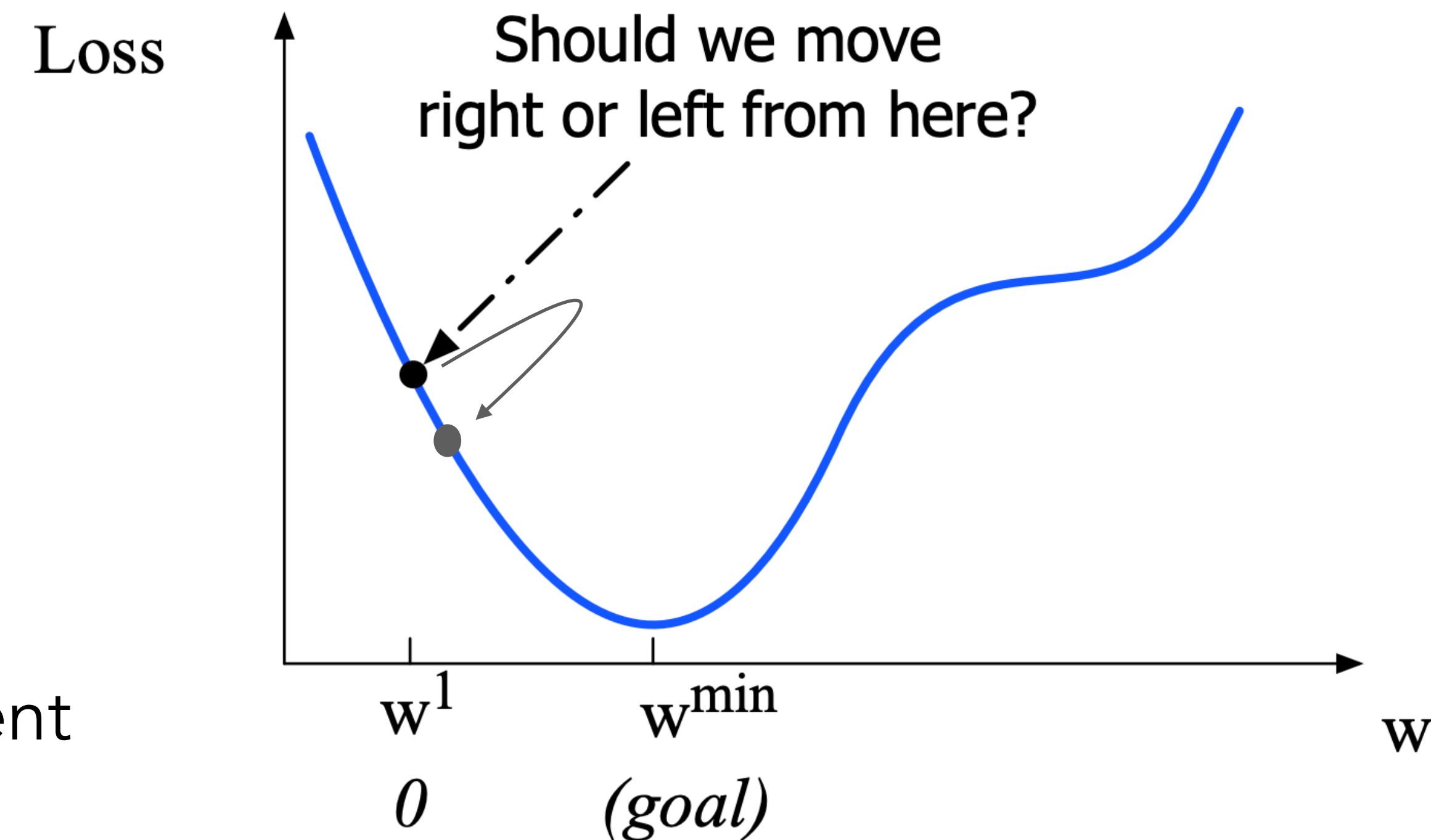
We want the weights that minimize the loss, averaged over all examples:

$$L_{CE}(f(\mathbf{x}^{(i)}; \theta), y^{(i)})$$

# Gradients

The **gradient** of a function of many variables is a vector pointing in the direction of the greatest increase in a function.

Find the gradient of the loss function at the current point and move in the **opposite** direction.



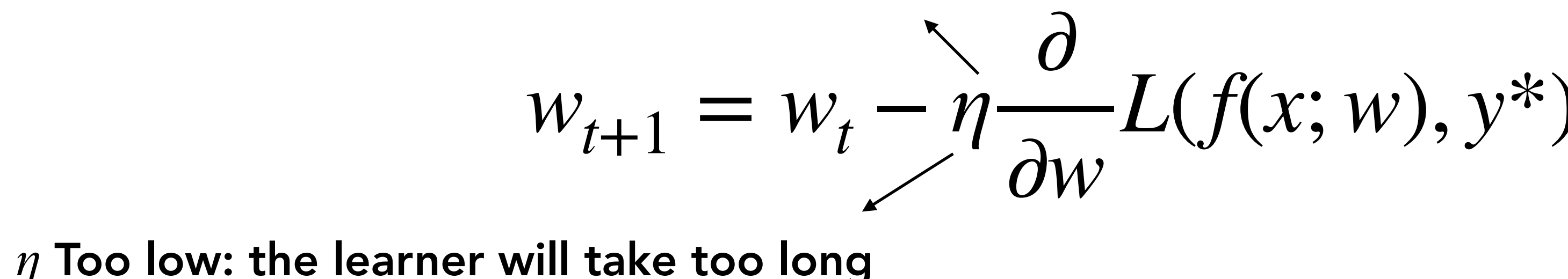
Gradient Descent

But by how much?

# Gradient Updates

- Move  $w$  by the value of the gradient  $\frac{\partial}{\partial w} L(f(x; w), y^*)$ , weighted by a learning rate  $\eta$
- Higher learning rate means move  $w$  faster

$\eta$  Too high: the learner will take big steps and overshoot

$$w_{t+1} = w_t - \eta \frac{\partial}{\partial w} L(f(x; w), y^*)$$


$\eta$  Too low: the learner will take too long

If parameter  $\theta$  is a vector of  $d$  dimensions:

The gradient is just such a vector; it expresses the directional components of the sharpest slope along each of the  $d$  dimensions.



# Real-life gradients

We will represent  $\hat{y}$  as  $f(x; \theta)$  to make the dependence on  $\theta$  more obvious

$$\nabla_{\theta} L(f(x; \theta), y) = \begin{bmatrix} \frac{\partial}{\partial w_1} L(f(x; \theta), y) \\ \frac{\partial}{\partial w_2} L(f(x; \theta), y) \\ \vdots \\ \frac{\partial}{\partial w_n} L(f(x; \theta), y) \end{bmatrix}$$

The final equation for updating  $\theta$  at time step  $t + 1$  based on the gradient is thus:

$$\theta_{t+1} = \theta_t - \eta \frac{\partial}{\partial \theta} L(f(x; \theta), y)$$



# Gradients for Logistic Regression

*Case: Sentiment Analysis*

Recall: the cross-entropy loss for logistic regression

$$L_{CE}(y, \hat{y}) = - [y \log \sigma(\mathbf{w} \cdot \mathbf{x} + b) + (1 - y) \log(\sigma(-\mathbf{w} \cdot \mathbf{x} + b))]$$

Derivatives have a closed form solution:

$$\frac{\partial L_{CE}(y, \hat{y})}{\partial w_j} = [\sigma(\mathbf{w} \cdot \mathbf{x} + b) - y] x_j$$

# Pseudocode

function STOCHASTIC GRADIENT DESCENT ( $L()$ ,  $f()$ ,  $x$ ,  $y$ ) returns  $\theta$

# where:  $L$  is the loss function

#  $f$  is a function parameterized by  $\theta$

#  $\mathbf{x}$  is the set of training inputs  $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}$

#  $y$  is the set of training outputs (labels)  $y^{(1)}, y^{(2)}, \dots, y^{(N)}$

$\theta \leftarrow 0$  (or randomly initialized)

**repeat** till done

for each training tuple( $x^{(i)}, y^{(i)}$ ): (in random order)

1. Compute  $\hat{y}^{(i)} = f(\mathbf{x}^{(i)}; \theta)$

# What is our estimated output  $\hat{y}^{(i)}$ ?

2. Compute the loss  $L(\hat{y}^{(i)}, y^{(i)})$

# How far off is  $\hat{y}^{(i)}$  from the true output  $y^{(i)}$ ?

3.  $g \leftarrow \nabla L(f(\mathbf{x}^{(i)}; \theta), y^{(i)})$

# How should we move  $\theta$  to maximize loss?

4.  $\theta \leftarrow \theta - \eta g$

# Go the other way instead

return  $\theta$



Stochastic Gradient Descent

# Mini-Batching

function STOCHASTIC GRADIENT DESCENT ( $L()$ ,  $f()$ ,  $x$ ,  $y$ ,  $m$ ) returns  $\theta$

# where:  $L$  is the loss function

#  $f$  is a function parameterized by  $\theta$

#  $\mathbf{x}$  is the set of training inputs  $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}$

#  $y$  is the set of training outputs (labels)  $y^{(1)}, y^{(2)}, \dots, y^{(N)}$  and  $m$  is the mini-batch size

$\theta \leftarrow 0$  (or randomly initialized)

**repeat** till done

for each randomly sampled minibatch of size  $m$ :

1. for each training tuple  $(\mathbf{x}^{(i)}, y^{(i)})$  in the minibatch: (in random order)

i. Compute  $\hat{y}^{(i)} = f(\mathbf{x}^{(i)}; \theta)$

# What is our estimated output  $\hat{y}^{(i)}$ ?

ii. Compute the loss  $L_{mini} \leftarrow L_{mini} + L(\hat{y}^{(i)}, y^{(i)})$

# How far off is  $\hat{y}^{(i)}$  from the true output  $y^{(i)}$ ?

2.  $g \leftarrow \frac{1}{m} \nabla L_{mini}(f(\mathbf{x}^{(i)}; \theta), y^{(i)})$

# How should we move  $\theta$  to maximize loss?

3.  $\theta \leftarrow \theta - \eta g$

# Go the other way instead

return  $\theta$

Why is this better than stochastic gradient descent?

# Lecture Outline

- Last Class: Logistic Regression
  - Data (Features)
  - Model (Logistic Regression)
  - Loss (Cross Entropy)
- Today: Logistic Regression Contd.
  - Optimization
  - Multinomial Logistic Regression
  - Regularization
- Word Embeddings



# Multinomial Logistic Regression



# Multinomial Logistic Regression

- Often we need more than 2 classes
  - Positive / negative / neutral sentiment of a document
  - Parts of speech of a word (noun, verb, adjective, adverb, preposition, etc.)
  - Actionable classes for emergency SMSs
- If  $>2$  classes we use multinomial logistic regression
  - = Softmax regression
  - = Multinomial logit
  - = (defunct names : Maximum entropy modeling or MaxEnt)
- So "logistic regression" will just mean binary (2 output classes)

# Multinomial Logistic Regression

The probability of everything must still sum to 1

$$P(+ | x) + P(- | x) + P(\sim | x) = 1$$

- Need a generalization of the sigmoid!
- Introducing the softmax function, which
  - Takes a vector  $\mathbf{z} = [z_1, z_2, \dots, z_K]$  of  $K$  arbitrary values
    - Each  $z_i$  corresponds to weighted sum of features for the  $K$ th class
  - Outputs a probability distribution
    - each value in the range  $[0,1]$
    - all the values summing to 1

Softmax



# The Softmax Function

Turns a vector  $\mathbf{z} = [z_1, z_2, \dots, z_K]$  of  $K$  arbitrary values into probabilities

$$\mathbf{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^K \exp(z_j)} \quad 1 \leq i \leq K$$

The denominator  $\sum_{i=1}^K \exp(z_i)$  is used to normalize all the values into probabilities.

$$\mathbf{softmax}(\mathbf{z}) = \left[ \frac{\exp(z_1)}{\sum_{i=1}^K \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^K \exp(z_i)}, \dots, \frac{\exp(z_K)}{\sum_{i=1}^K \exp(z_i)} \right]$$

# Softmax: Example

Turns a vector  $\mathbf{z} = [z_1, z_2, \dots, z_K]$  of  $K$  arbitrary values into probabilities

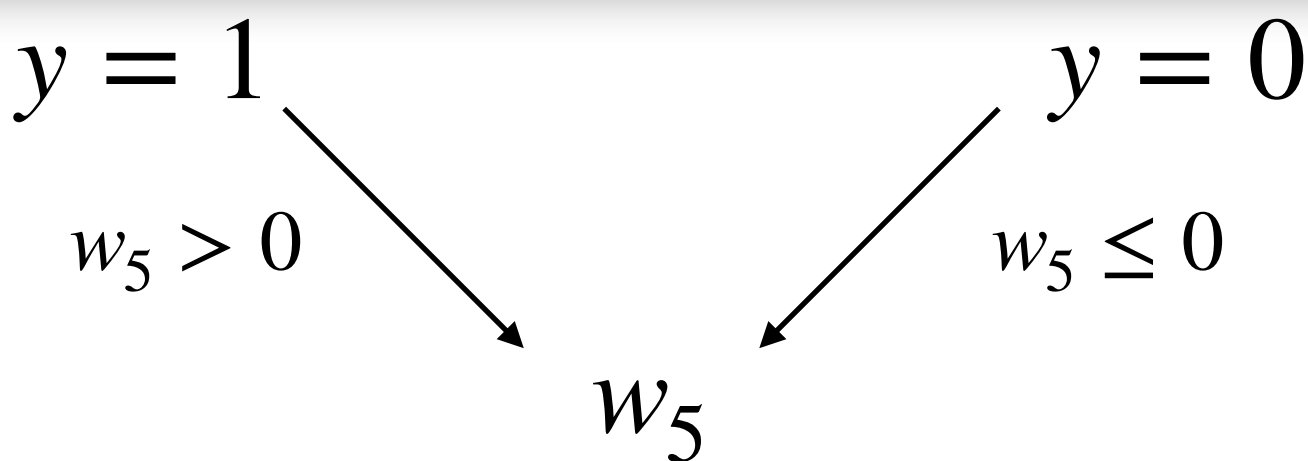
$$\mathbf{z} = [0.6, 1.1, 1.5, 1.2, 3.2, 1.1]$$

$$\text{softmax}(\mathbf{z}) = \left[ \frac{\exp(z_1)}{\sum_{i=1}^K \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^K \exp(z_i)}, \dots, \frac{\exp(z_K)}{\sum_{i=1}^K \exp(z_i)} \right]$$

$$\text{softmax}(\mathbf{z}) = [0.055, 0.090, 0.0067, 0.10, 0.74, 0.010]$$

# Binary versus Multinomial

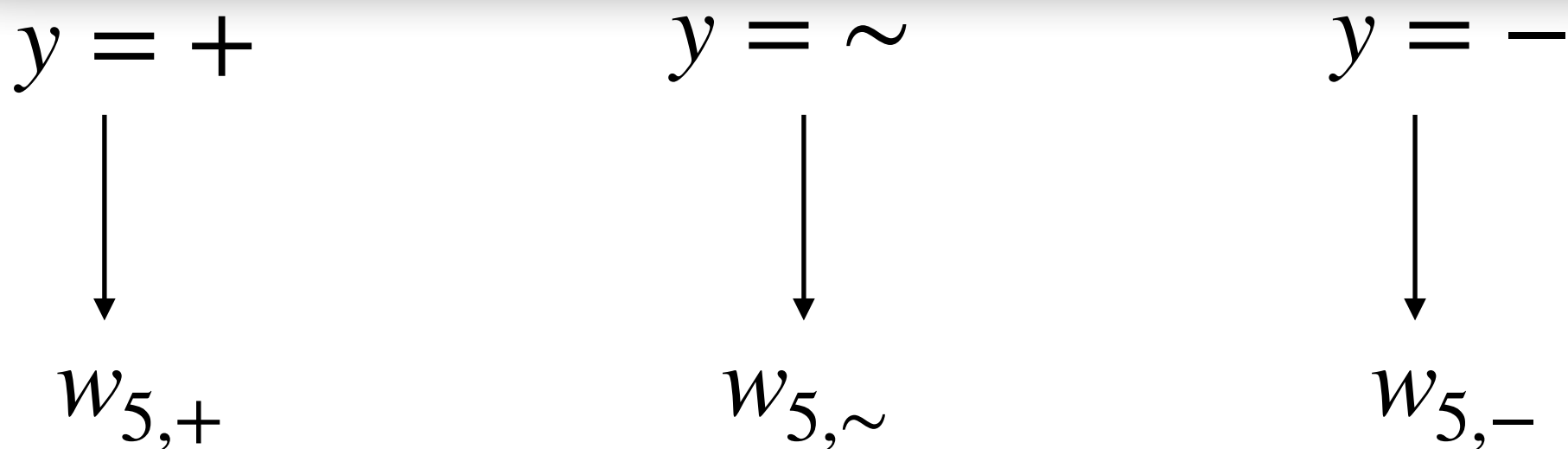
## Binary Logistic Regression



$$x_5 = \begin{cases} 1 & \text{if “!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases} \qquad w_5 = 3.0$$

Why do we NOT need a different weight for each class in binary logistic regression?

## Multinomial Logistic Regression



Separate weights for each class

Feature	Definition	$w_{5,+}$	$w_{5,-}$	$w_{5,0}$
$f_5(x)$	$\begin{cases} 1 & \text{if “!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	3.5	3.1	-5.3

# Softmax in multinomial logistic regression

Parameters are now a matrix  $\mathbf{W} \in \mathbb{R}^{d \times K}$  and  $b \in \mathbb{R}^1$

$$P(y = c \mid \mathbf{x}; \theta) = \frac{\exp(\mathbf{w}_c \cdot \mathbf{x} + b)}{\sum_{j=1}^K \exp(\mathbf{w}_j \cdot \mathbf{x} + b)}$$

- Input is still the dot product between weight vector  $\mathbf{w}_c$  and input vector  $\mathbf{x}$ , offset by  $b$
- But **separate weight vectors for each of the  $K$  classes, each of dimension  $d$**

Multinomial LR Loss:

$$L_{CE} = -\log P(y = c \mid \mathbf{x}; \theta) = -(\mathbf{w}_c \cdot \mathbf{x} + b) + \log \left[ \sum_{j=1}^K \exp(\mathbf{w}_j \cdot \mathbf{x} + b) \right]$$



# Case Study: Word Neighborhood Predictor

- Yet do I fear thy nature. It is too full o'the milk of human kindness To catch the nearest way.
- If it were done when 'tis done, then 'twere well It were done quickly.
- Is this a dagger which I see before me, The handle toward my hand?
- My hands are of ;your color, but I shame To wear a heart so white.
- Knock, knock, knock! Who's there,

Where is it more likely to find the word "doth"?





# Case Study: Word Neighborhood Predictor

What words are likely to co-occur with the word "garnish"?



## Ingredients

12 Large Eggs, Hard-boiled And Peeled  
(See Note)

1/2 c. Mayonnaise

2 tbsp. Prepared Yellow Mustard

1/4 tsp. Kosher Salt

[See Nutritional Information](#) ▼

## Directions

Slice eggs in half lengthwise. Pop out yolks and place them in a medium-sized mixing bowl. Mash yolks with a fork and then add mayonnaise, mustard, and salt. Mix until smooth. To achieve maximum smoothness, blitz the yolk mixture a few times with an immersion blender. If you are planning to use a pastry bag and piping tip to add the yolk mixture to the egg whites, it helps to get the yolk mixture nice and smooth.

Divide the yolk mixture evenly between the egg white halves. Garnish as desired. Serve immediately or refrigerate until ready to serve.

Next Class: We will use multinomial logistic regression to learn word embeddings!

# Lecture Outline

- Last Class: Logistic Regression
  - Data (Features)
  - Model (Logistic Regression)
  - Loss (Cross Entropy)
- Today: Logistic Regression Contd.
  - Optimization
  - Multinomial Logistic Regression
  - Regularization
- Word Embeddings



# Regularization

# Overfitting

- A model that perfectly match the training data has a problem
- It will also overfit to the data, modeling noise
  - A random word that perfectly predicts  $y$  (it happens to only occur in one class) will get a very high weight
  - Failing to generalize to a test set without this word

Why?

A good model should be able to generalize

What happens when a feature only occurs with one class?

e.g. word "wow" for positive reviews

# Overfitting in Sentiment Classification

This movie drew me in, and it'll do the same to you.

Useful or harmless features

$x_1 = \text{"this"}$

$x_2 = \text{"movie"}$

$x_3 = \text{"hated"}$

$x_4 = \text{"drew me in"}$

I can't tell you how much I hated this movie. It sucked.

4-gram features that just "memorize" training set and might cause problems

$x_5 = \text{"the same to you"}$

$x_6 = \text{"tell you how much"}$

# Overfitting in $n$ -gram models

- 4-gram model on tiny data will just memorize the data
  - 100% accuracy on the training set
- But it will be surprised by the novel 4-grams in the test data
  - Low accuracy on test set
- Models that are too powerful can overfit the data
  - Fitting the details of the training data so exactly that the model doesn't generalize well to the test set

How to avoid overfitting?

Regularization in logistic regression and other neural nets

# Regularization

- A solution for overfitting: Add a regularization term  $R(\theta)$  to the loss function
  - (for now written as maximizing logprob rather than minimizing loss)
- Idea: choose an  $R(\theta)$  that penalizes large weights
  - fitting the data well with lots of big weights not as good as
  - fitting the data a little less well, with small weights

$$\hat{\theta} = \arg \max_{\theta} \sum_{i=1}^n \log P(y^{(i)} | \mathbf{x}^{(i)}) - \alpha R(\theta)$$

# $L_2$ / Ridge Regularization

- The sum of the squares of the weights
- The name is because this is the (square of the)  $L_2$  norm  $\|\theta\|_2^2$ , = Euclidean distance of  $\theta$  to the origin.

$$R(\theta) = \|\theta\|_2^2 = \sum_{j=1}^d \theta_j^2$$

$L_2$  regularized objective function:

$$\hat{\theta} = \arg \max_{\theta} \sum_{i=1}^n \log P(y^{(i)} | x^{(i)}) - \alpha \sum_{j=1}^d \theta_j^2$$

# $L_1$ / Lasso Regularization

- The sum of the (absolute value of the) weights
- Named after the  $L_1$  norm  $\|\theta\|_1$  = sum of the absolute values of the weights = Manhattan distance

$$R(\theta) = \|\theta\|_1 = \sum_{j=1}^d |\theta_j|$$

$L_1$  regularized objective function:

$$\hat{\theta} = \arg \max_{\theta} \sum_{i=1}^n \log P(y^{(i)} | x^{(i)}) - \alpha \sum_{j=1}^d |\theta_j|$$



# Lecture Outline

- Last Class: Logistic Regression
  - Data (Features)
  - Model (Logistic Regression)
  - Loss (Cross Entropy)
- Today: Logistic Regression Contd.
  - Optimization
  - Multinomial Logistic Regression
  - Regularization
- Word Embeddings

~~Words, words, words~~

Types, types, types

# What do words mean?

A **sense** or “concept” is the meaning component of a word

## Lemmas

- Canonical form
- For example, break, breaks, broke, broken and breaking all share the lemma “break”
- Different from “stem”

Can be polysemous (have multiple senses)

## Dictionary

Definitions from [Oxford Languages](#) · [Learn more](#)



ob·jec·tive

/əbˈjektiv/

Lemma

### adjective

1. (of a person or their judgment) not influenced by personal feelings or opinions in considering and representing facts.

"historians try to be objective and impartial"

Similar:

impartial

unbiased

unprejudiced

nonpartisan

disinterested



2. **GRAMMAR**

relating to or denoting a case of nouns and pronouns used as the object of a transitive verb or a preposition.

### noun

1. a thing aimed at or sought; a goal.

"the system has achieved its objective"

Similar:

aim

intention

purpose

target

goal

intent

object

end



2. **GRAMMAR**

the objective case.

Sense

“You shall know a word by the company  
it keeps.”

- Firth (1957)

# Word Meaning via Language Use

- The meaning of a word can be given by its distribution in language usage:
  - One way to define "usage": words are defined by their environments
    - Neighboring words or grammatical environments
- Intuitions: Zellig Harris (1954):
  - "oculist and eye-doctor ... occur in almost the same environments"
  - "If A and B have almost identical environments we say that they are synonyms."

A bottle of tesgüino is on the table  
Everybody likes tesgüino  
Tesgüino makes you drunk  
We make tesgüino out of corn.



Two words are similar if they have similar word contexts



# Word Meanings via Language Properties

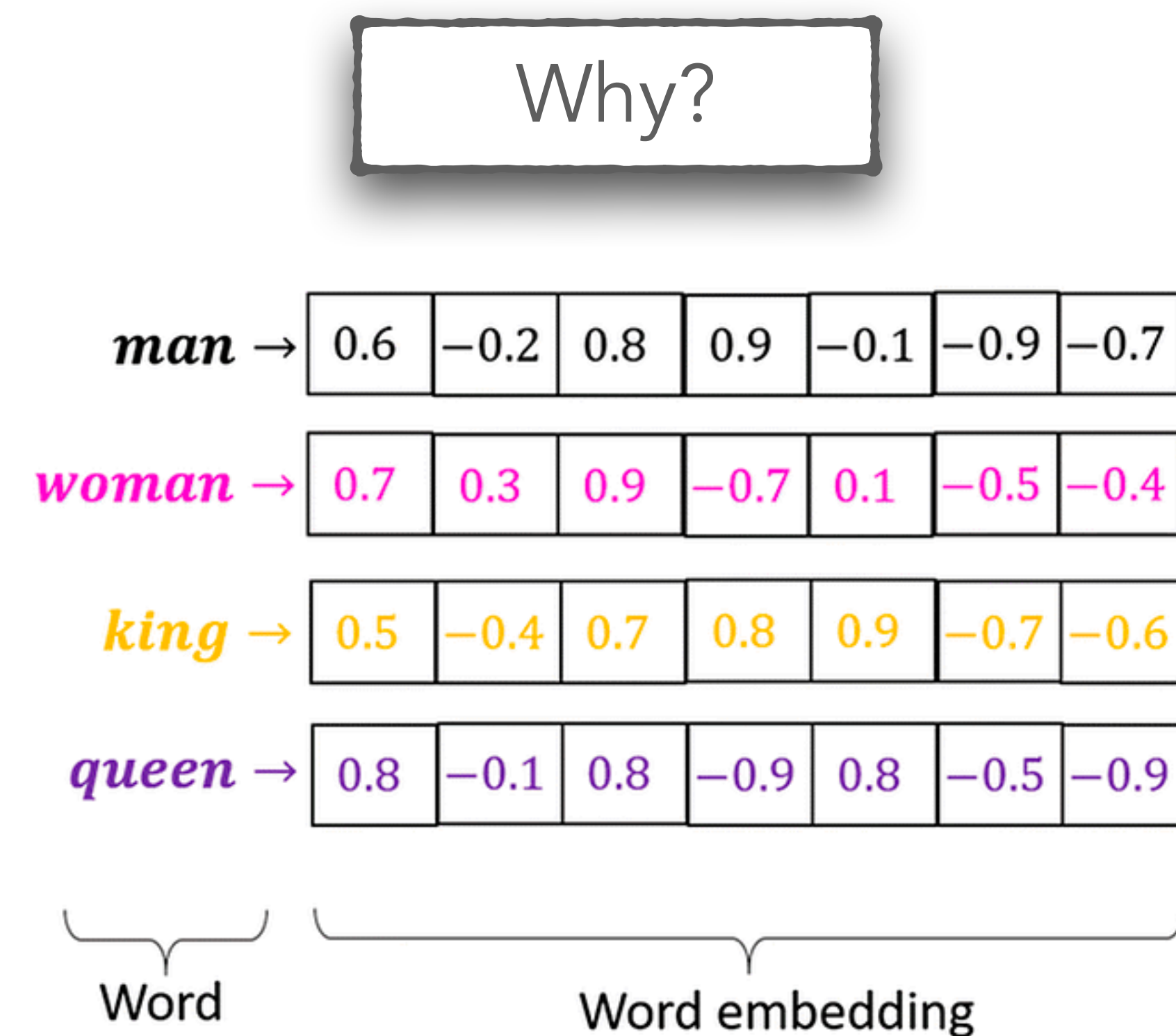
- Meaning of a word can be determined by some properties of the word
- Point in space (Osgood et al., 1957)
- Example Properties: Affective Dimensions

	Word	Score		Word	Score
<b>Valence</b>	love	1.000		toxic	0.008
	happy	1.000		nightmare	0.005
<b>Arousal</b>	elated	0.960		mellow	0.069
	frenzy	0.965		napping	0.046
<b>Dominance</b>	powerful	0.991		weak	0.045
	leadership	0.983		empty	0.081

# Words as Vectors

In NLP, we commonly represent word types with vectors!

- Very useful in capturing similarity between words, and other forms of lexical semantics (e.g. synonymy, hypernyms, antonymy)
- Computing the similarity between two words (or phrases, or documents) is extremely useful for many NLP tasks
  - Q: How **tall** is Mount Everest?
  - A: The official **height** of Mount Everest is 29029 ft

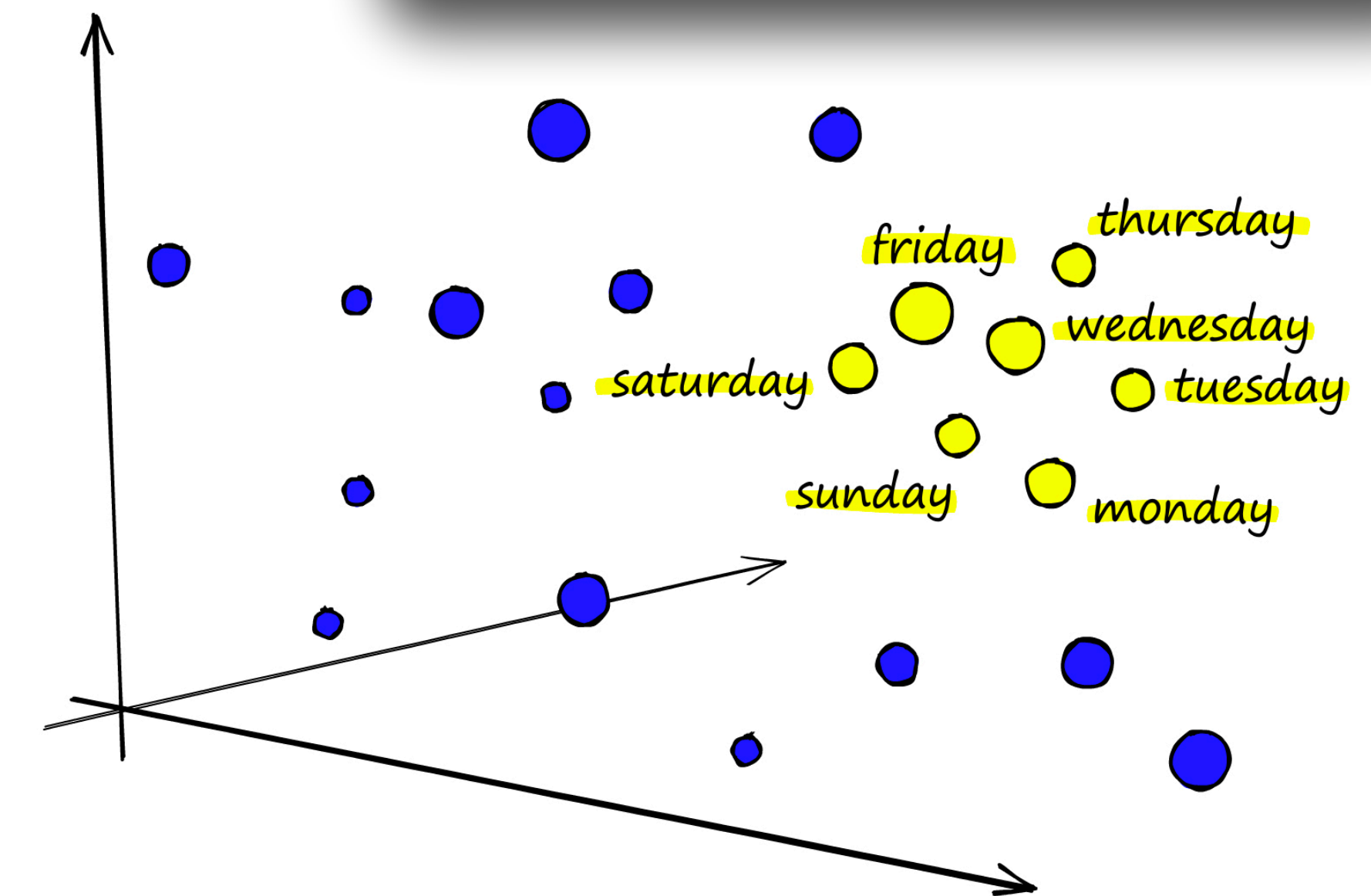




# Word Embeddings

## Vector Semantics

- Represent a word as a point in a multidimensional semantic space
  - Space itself constructed from distribution of word neighbors
- Called an "embedding" because it's embedded into a space
- Fine-grained model of meaning for **similarity**



Every modern NLP algorithm uses embeddings as the representation of word meaning

# Defining meaning as a point in space based on distribution

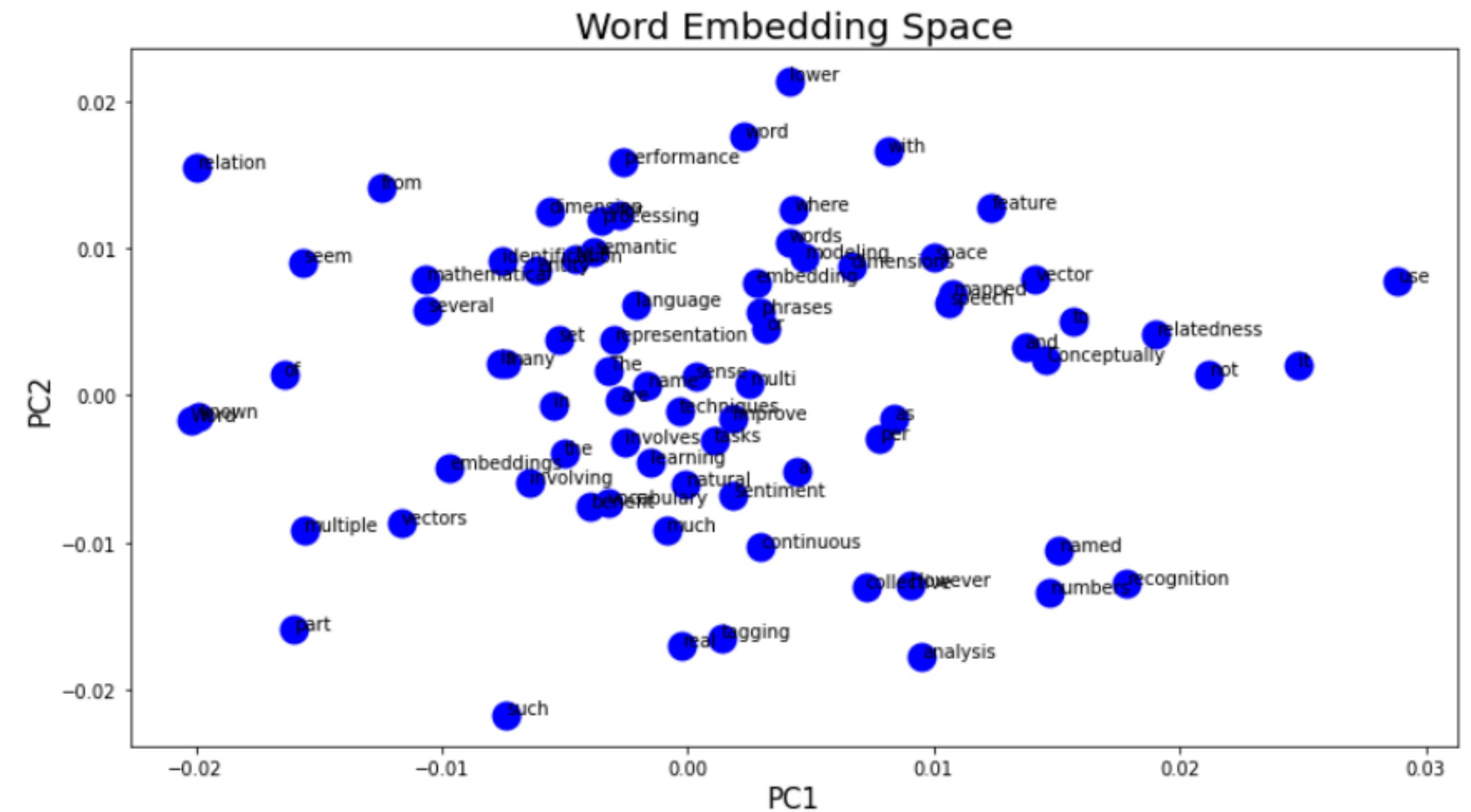
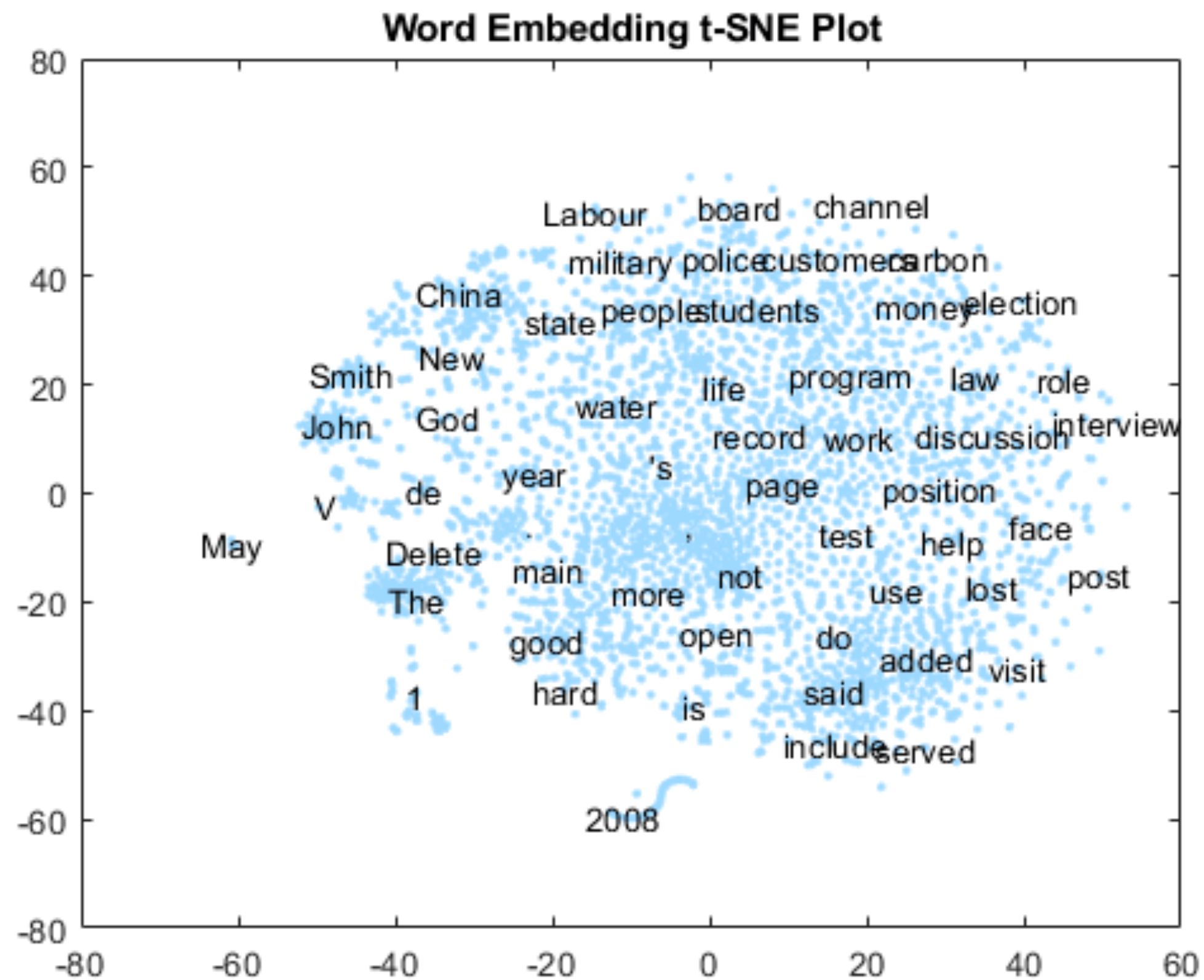
- Each word = a vector
  - not just “good” or “word#45”
- Similar words are “nearby in semantic space”
- Can build this space automatically by seeing which words are nearby in text
- 2-D representation



# Visualizing Embeddings

# Project high-dimensional embeddings down into 2 dimensions

- Most common projection method: t-SNE
- Also: Principal Component Analysis (PCA)





- Similarity for plagiarism detection
- Word similarity can lead to sentence and document similarity

enough scale for companies to make profit from it. In order to be competitive with new technologies, the challenge of today's large companies is to create new business within their business (Garvin & Levesque, 2006). Furthermore, the two researchers emphasize a switch from downsizing and cost cutting to the creation, development and assistance of innovative new businesses. For existing companies the implementation of corporate entrepreneurship, in order to develop innovative businesses, is risky. Are the three types of entrepreneurship linked together over time? How long does it take to change behavior of the firm as a whole? If the five attributes are created, do all grow together equally, or do some grow faster and earlier than others? How do the importance and intensities of the attributes differ both absolutely and relatively in each type? These are the questions that a longitudinal study such as this can attempt to answer to shed light on the nature of organizations' adjustments to hostile environments. According to Garvin and Levesque (2006) implementing new ventures face several barriers, and can only be successful if a blend of old and new organizational traits is done. To achieve a blend of old and new, an organization needs to rely on employee innovative behavior in order to succeed in dynamic business environments (Yuan & Woodman, 2010).



Figure 1 - The prototypical model - innovation strategy, 2006

The downside is potentially very damaging to a startup's lifespan: if a startup lands a pilot or POC with the corporation running the accelerator, they have very little bargaining power or time to find other partners to test their solution with. The transition from manufacturing economy to service economy has led to a shift in business agenda from

1 Original source  
[onlinelibrary.wiley.com/stor...](https://onlinelibrary.wiley.com/stor...)

...all grow together equally, or do some grow faster and earlier than others? These are the questions that a longitudinal study such as this can attempt to answer to shed light on the nature of organizations' adjustments to hostile environments. Of the many ways to adjust, two stand out at

# Cosine Similarity for Word Similarity

Cosine similarity of two vectors

$$\begin{aligned}\cos(\vec{v}, \vec{w}) &= \frac{\vec{v} \cdot \vec{w}}{|\vec{v}| |\vec{w}|} \\ &= \frac{\sum_{i=1}^N v_i w_i}{\sqrt{\sum_{i=1}^N v_i^2} \sqrt{\sum_{i=1}^N w_i^2}}\end{aligned}$$

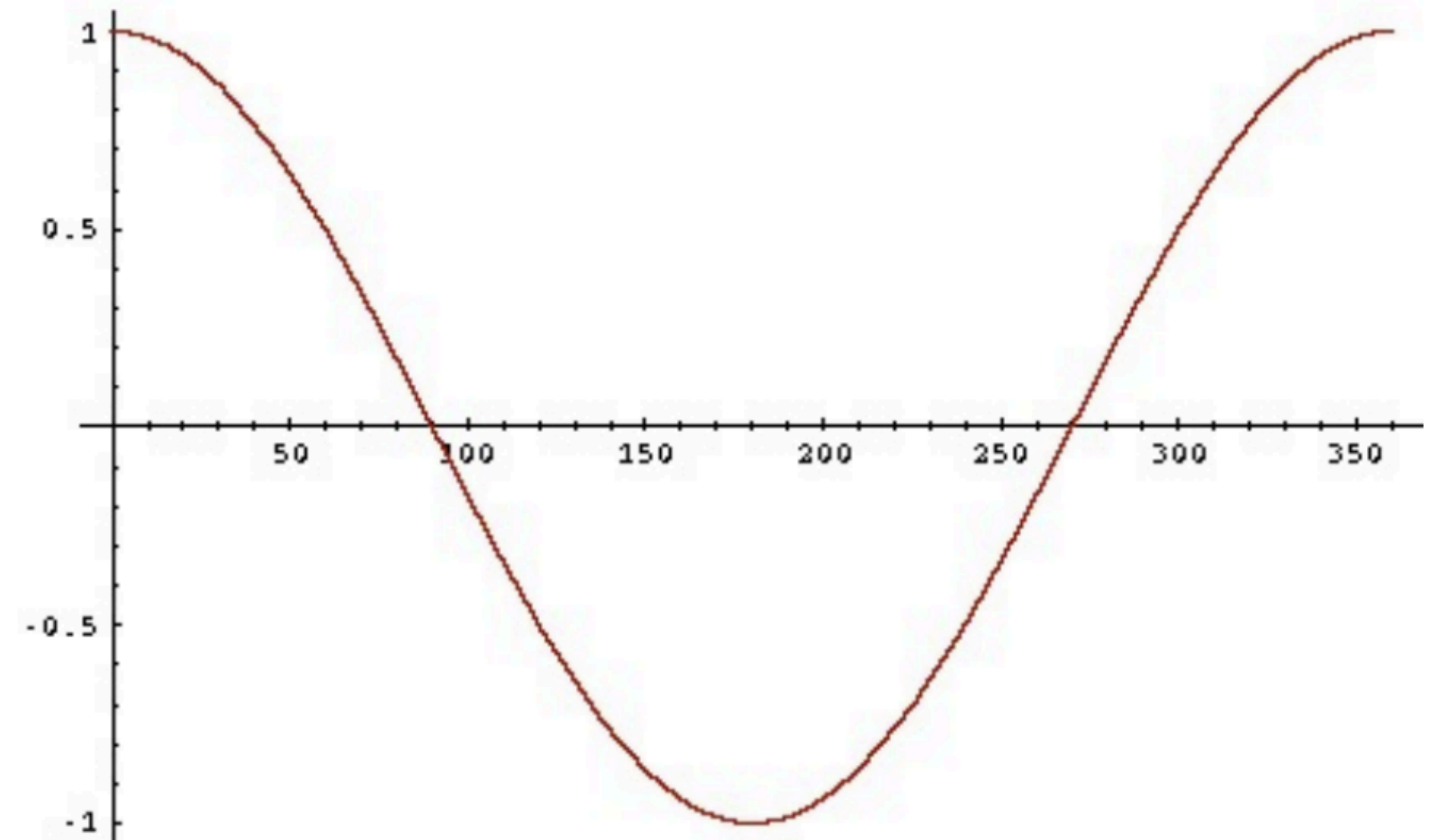
Based on the definition of the dot product between two vectors  $\vec{a}$  and  $\vec{b}$

$$\vec{v} \cdot \vec{w} = |\vec{v}| |\vec{w}| \cos \theta$$

$$\cos \theta = \frac{\vec{v} \cdot \vec{w}}{|\vec{v}| |\vec{w}|}$$

# Cosine as a similarity metric

- 1: vectors point in opposite directions
- +1: vectors point in same directions
- 0: vectors are orthogonal



Greater the cosine, more similar the words



Word embeddings teach us  
about our own language use!

荃者所以在鱼，得鱼而忘荃

言者所以在意，得意而忘言

*"Nets are for fish; Once you get the fish, you can forget the net."*

*Words are for meaning; Once you get the meaning, you can forget the words"*

*– Zhuangzi*

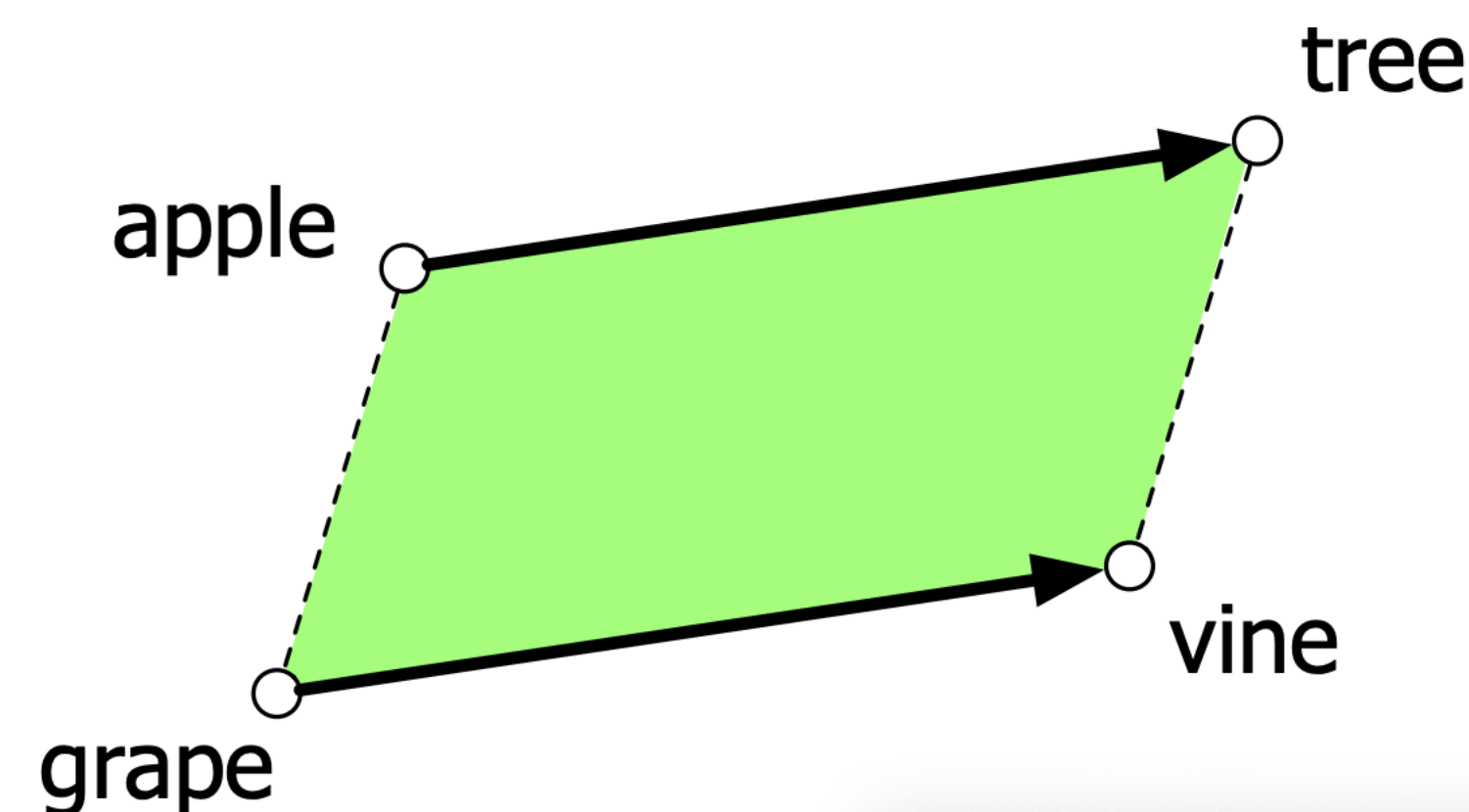
庄子

# Analogy Relations

Maximize similarity = minimize distance

- The classic parallelogram model of analogical reasoning
- Word analogy problem:
  - "Apple is to tree as grape is to ..."

Add  $(\mathbf{w}_{apple} - \mathbf{w}_{tree})$  to  $\mathbf{w}_{grape}$  ...  
Should result in  $\mathbf{w}_{vine}$



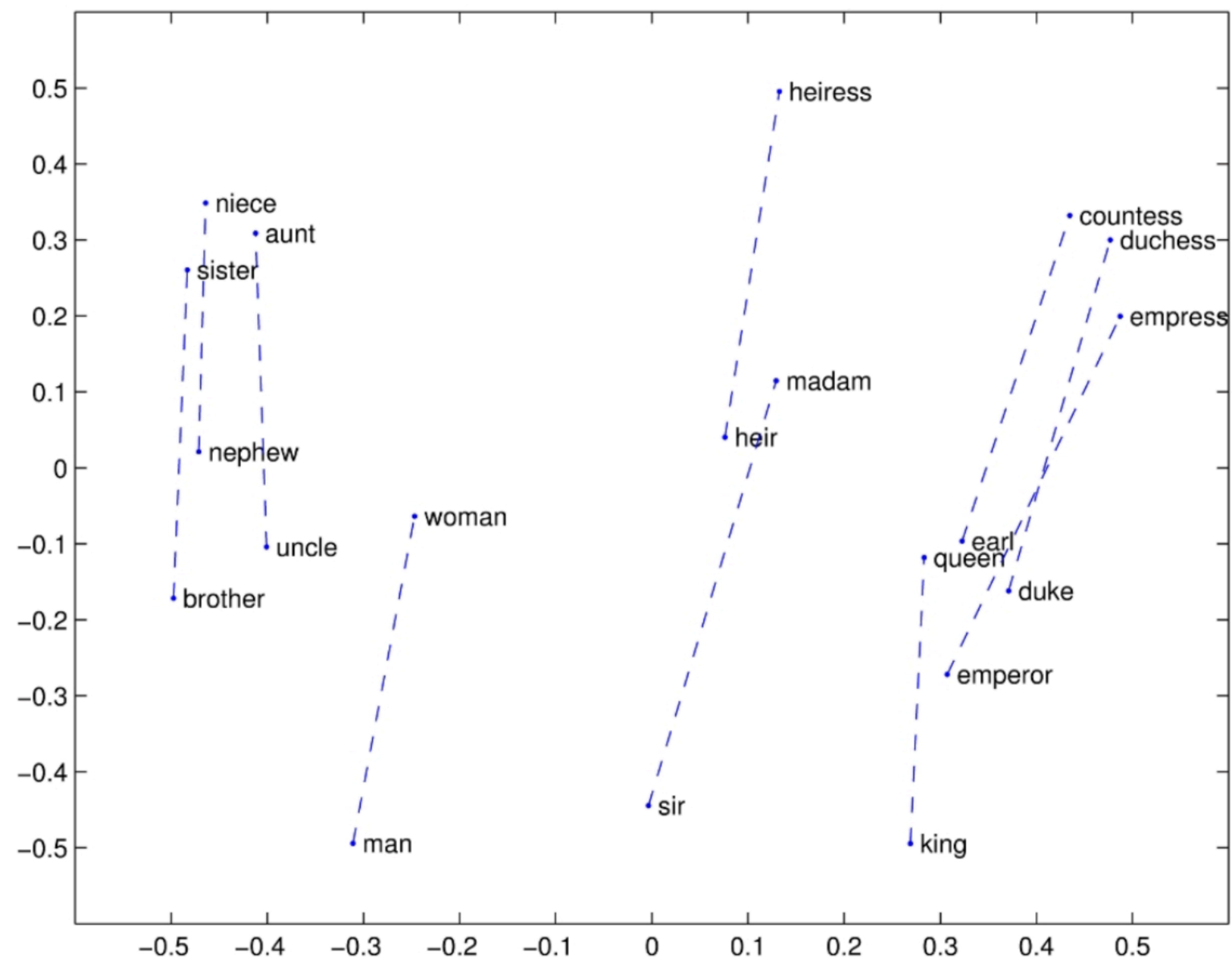
Rumelhart and Abrahamson, 1973

For a problem  $a : a^* :: b : b^*$ , the parallelogram method is:

$$\hat{b}^* = \arg \max_{\mathbf{w}} \text{sim}(\mathbf{w}, \mathbf{b} - \mathbf{a} + \mathbf{a}^*)$$

# Analogy Relations: GloVe

- Relational properties of the GloVe vector space, shown by projecting vectors onto two dimensions
- $\mathbf{w}_{king} - \mathbf{w}_{man} + \mathbf{w}_{woman}$  is similar to  $\mathbf{w}_{queen}$
- Caveats: Only works for frequent words, small distances and certain relations
  - Understanding analogy is an open area of research



# $n$ -grams and Semantics

- Were feature representations! And so were Bag-of-Words
  - $\mathbf{x}$ 's in machine learning, which are associated with parameters
- Just strings - atomic symbols!
  - As  $n$  increases, we get strings that co-occur
- $n$ -grams do not represent meaning well
  - Do not tell us that the word "rancor" is close in meaning to the word "hatred"
  - Or that "Rise" and "Fall" have opposite meanings
  - Let alone more complex is-a or part-of relations
- **Discrete** representations of meaning!
- Later: feature representations which are continuous



# $n$ -grams as One-hot Vectors

## vocabulary

i

hate

love

the

movie

film

movie =  $\langle 0,0,0,0,1,0 \rangle$ film =  $\langle 0,0,0,0,0,1 \rangle$ 

Unigram Vectors: Represent each word as a vector of zeros with a single 1 identifying the index of the word

One hot vector

How can we compute a vector representation such that the dot product correlates with word similarity?

Dot product is zero! These vectors are orthogonal

# Embeddings reflect cultural bias!



Offensive Content Warning

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

Allocational Harms

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

# Embeddings as a tool to study cultural bias!

- Compute a gender or ethnic bias for each adjective:  
e.g., how much closer the adjective is to "woman" synonyms than "man" synonyms, or names of particular ethnicities
  - Embeddings for competence adjective (smart, wise, brilliant, resourceful, thoughtful, logical) are biased toward men, a bias slowly decreasing 1960-1990
  - Embeddings for dehumanizing adjectives (barbaric, monstrous, bizarre) were biased toward Asians in the 1930s, bias decreasing over the 20th century
- These match the results of old surveys done in the 1930s

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

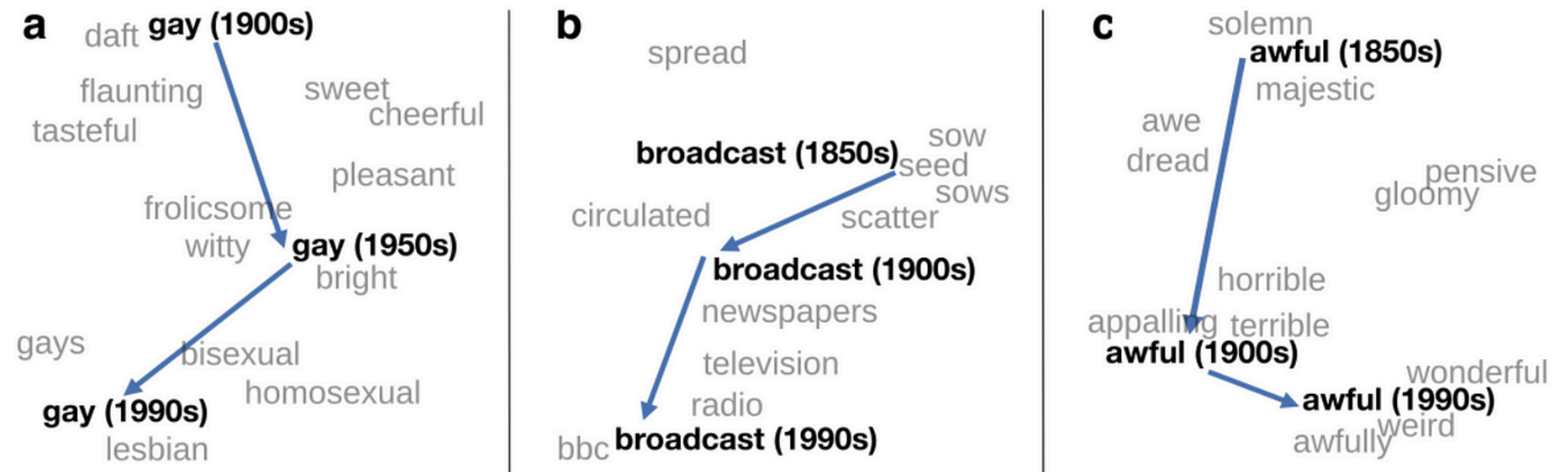
Garg, N., Schiebinger, L., Jurafsky, D., and Zou, J. (2018). Word embeddings quantify 100 years of gender and ethnic stereotypes. Proceedings of the National Academy of Sciences 115(16), E3635–E3644.

Representational Harms



# Embeddings uncover semantic histories

- Visualizing semantic change over time
- New words: dank, cheugy, rizz, shook, situationship



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

# Concluding Thoughts

Next Class: Word embeddings, inspired by neural language models

- Word2vec (skip-gram, CBOW)
- Based on logistic regression

