CS388: Natural Language Processing

Lecture 5: Word Embeddings

Greg Durrett

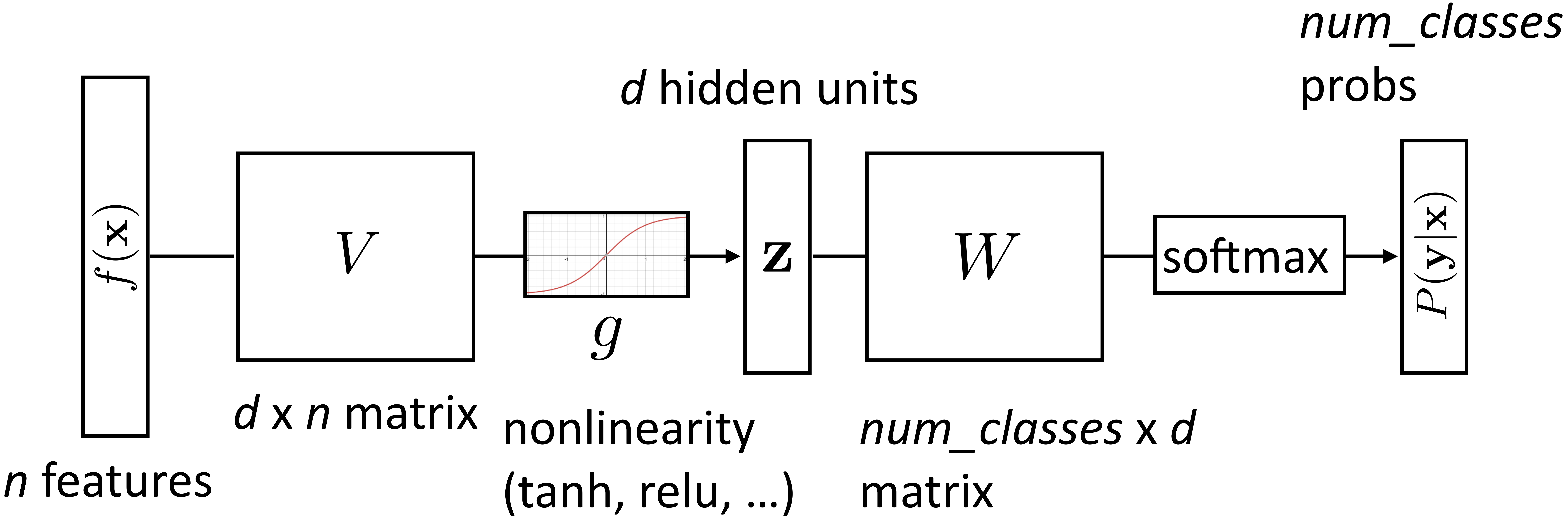
The University of Texas at Austin

Administrivia

- Project 1 due today
- Project 2 released today; material for it covered Thursday and finished next Tuesday

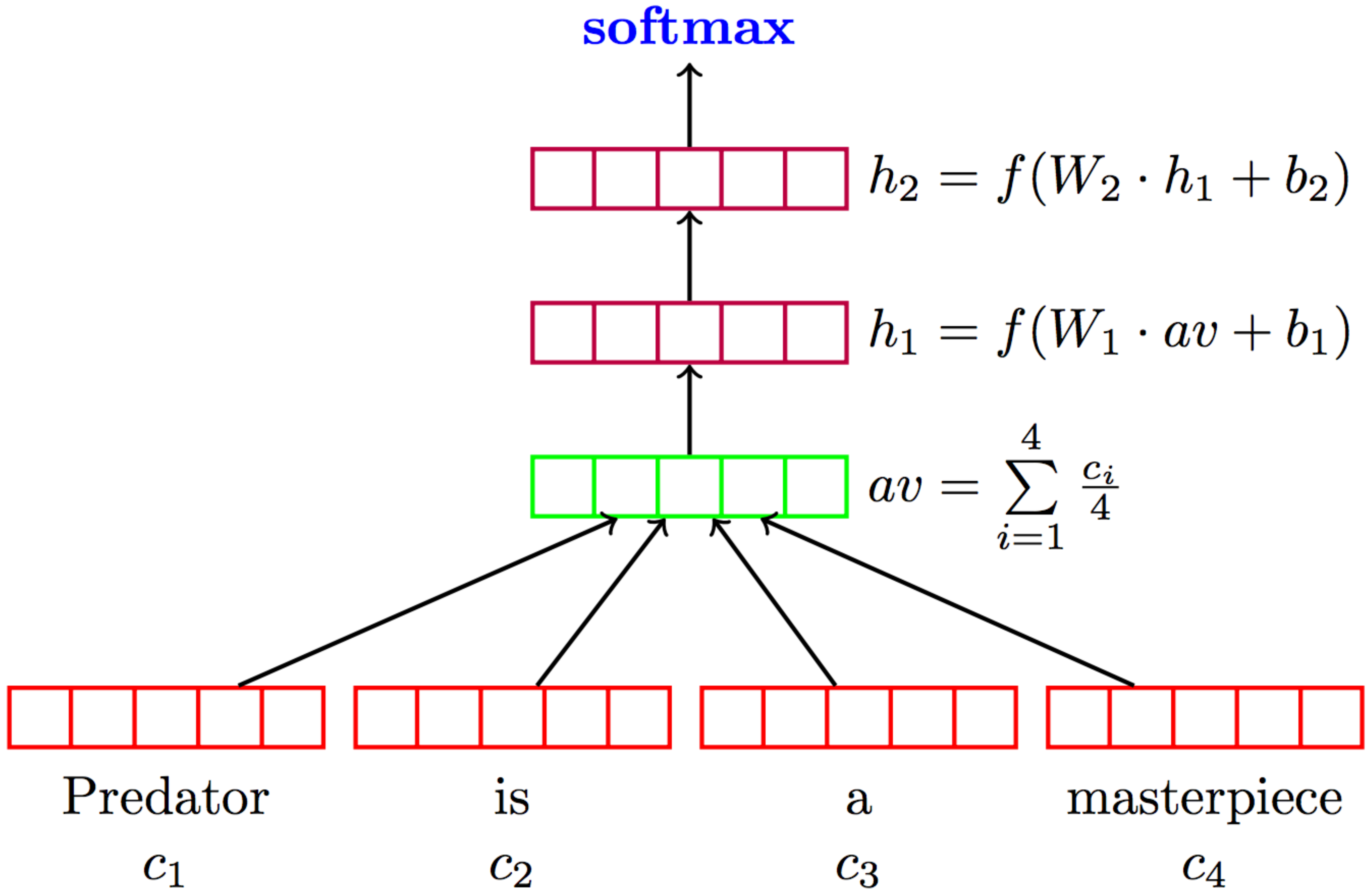
Recall: Feedforward NNs

$$P(\mathbf{y}|\mathbf{x}) = \mathrm{softmax}(Wg(Vf(\mathbf{x})))$$



Recall: Deep Averaging Networks

- Deep Averaging Networks: feedforward neural network on average of word embeddings from input



Iyyer et al. (2015)

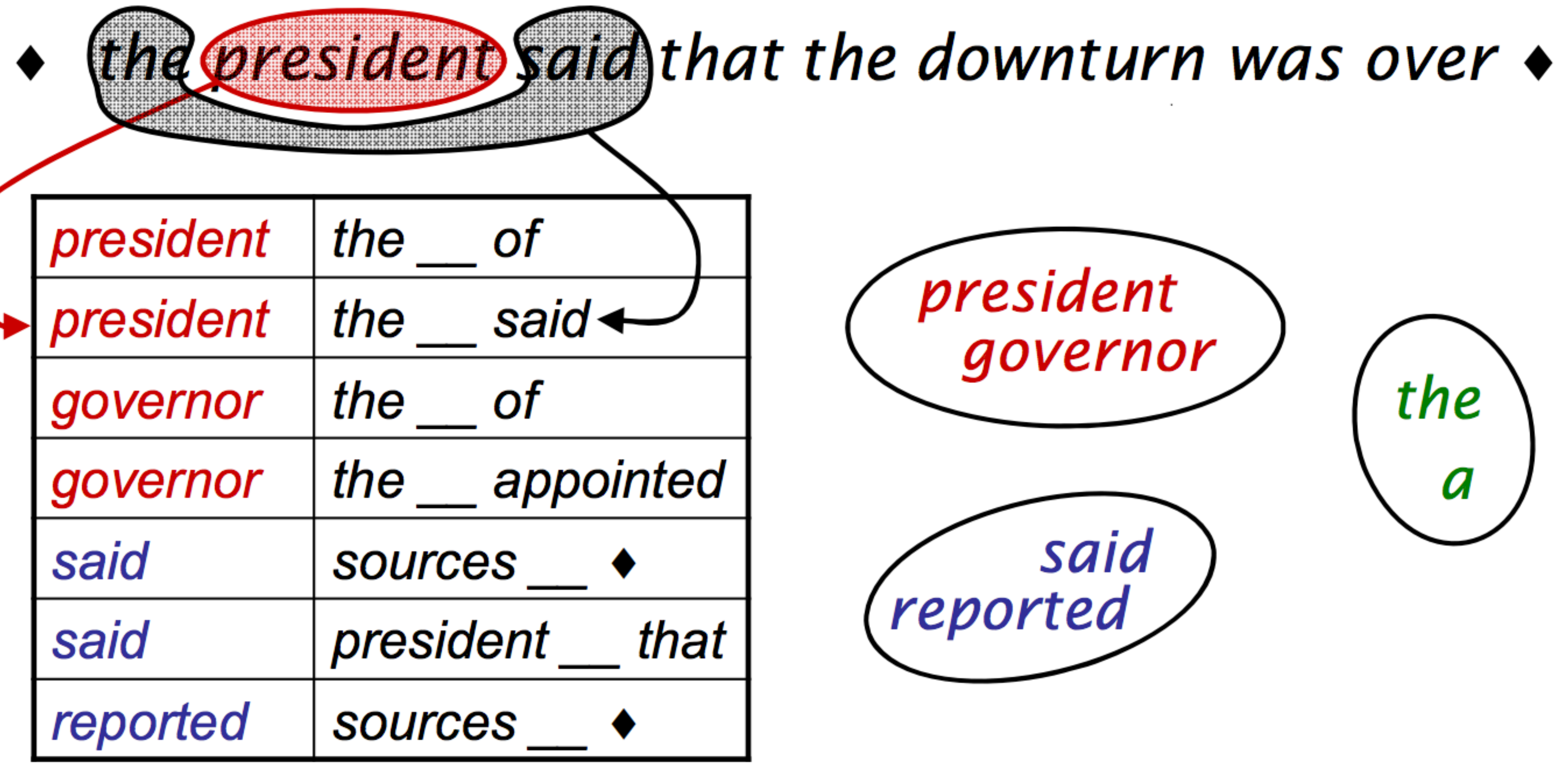
This Lecture

- Word representations
- Skip-gram
- GloVe
- Other word embedding methods
- Evaluating word embeddings

Word Representations

Word Representations

- Neural networks work very well at continuous data, but words are discrete
- Continuous model <-> expects continuous semantics from input
- “You shall know a word by the company it keeps” Firth (1957)



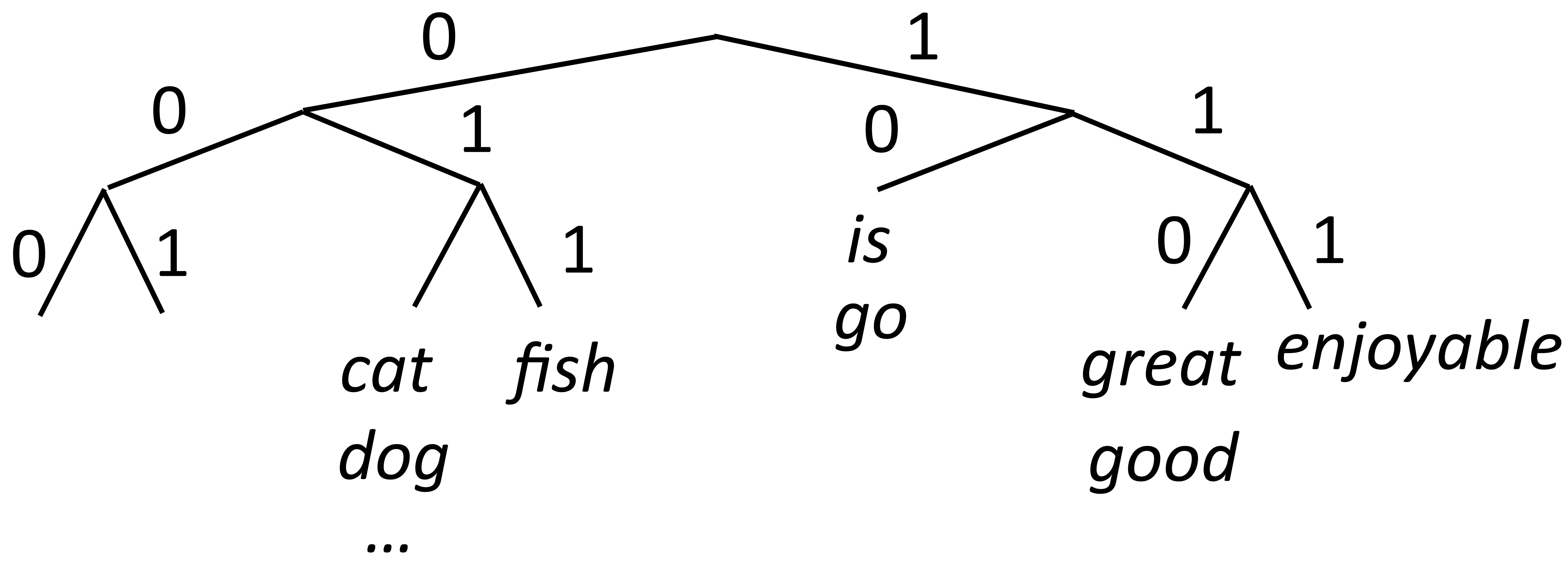
president	the __ of
president	the __ said
governor	the __ of
governor	the __ appointed
said	sources __ ♦
said	president __ that
reported	sources __ ♦

[Finch and Chater 92, Shuetze 93, many others]

slide credit: Dan Klein

Discrete Word Representations

- Brown clusters: hierarchical agglomerative *hard* clustering (each word has one cluster, not some posterior distribution like in mixture models)



- Maximize $P(w_i|w_{i-1}) = P(c_i|c_{i-1})P(w_i|c_i)$
- Useful features for tasks like NER, not suitable for NNs

Brown et al. (1992)

Word Embeddings

- Want a vector space where similar words have similar embeddings

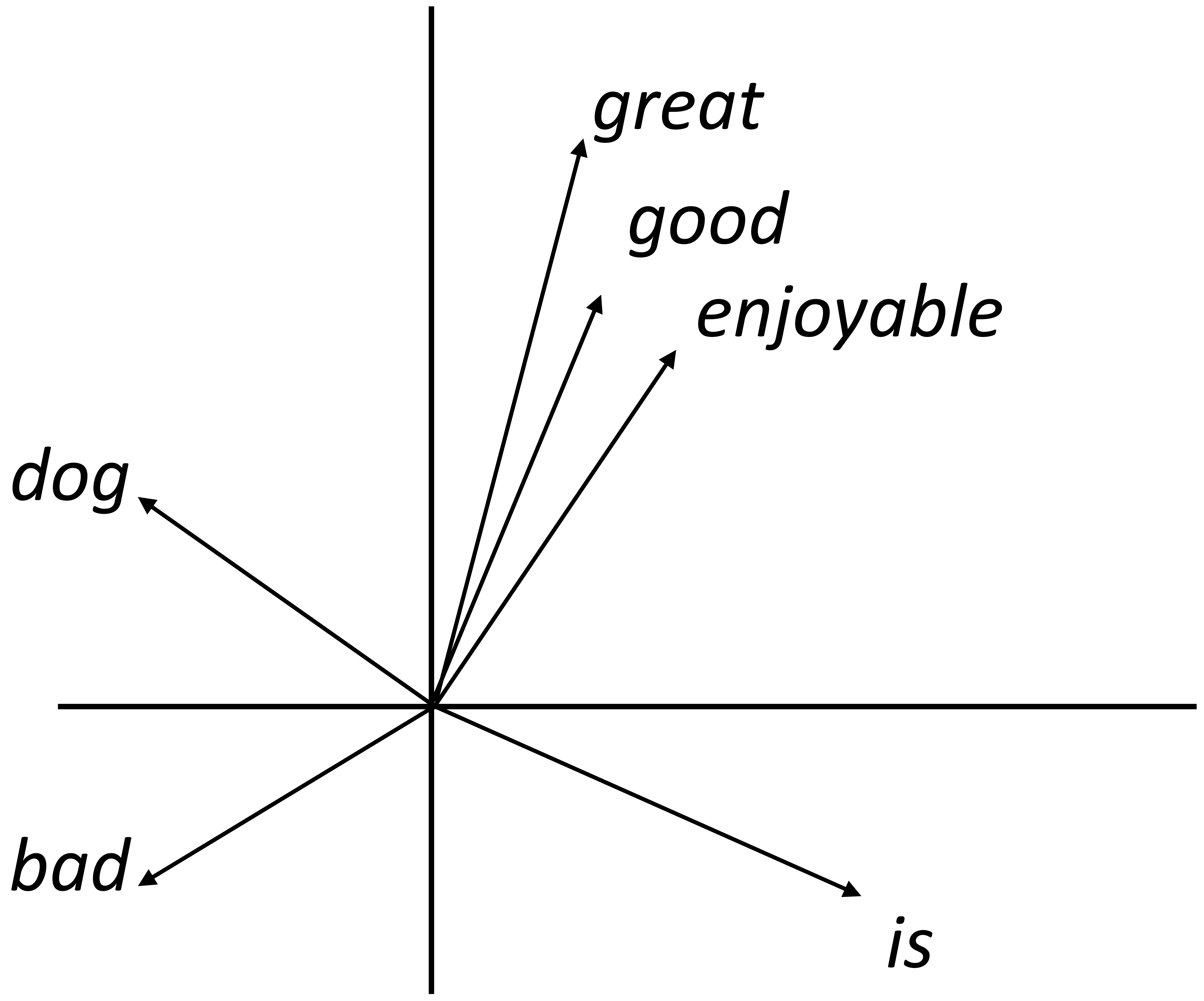
the movie was great

≈

the movie was good

- Goal: come up with a way to produce these embeddings

- For each word, want "medium" dimensional vector (50-300 dims) representing it



Skip-gram

Skip-Gram

- Input: a corpus of raw text. (Same as the input to "real" language modeling)

- Output: a set of *embeddings*: a real-valued vector for each word in the vocabulary

- We are going to learn these by setting up a fake prediction problem: predict a word's context from that word

the dog bit the man

(word = *bit,* context = *dog)*

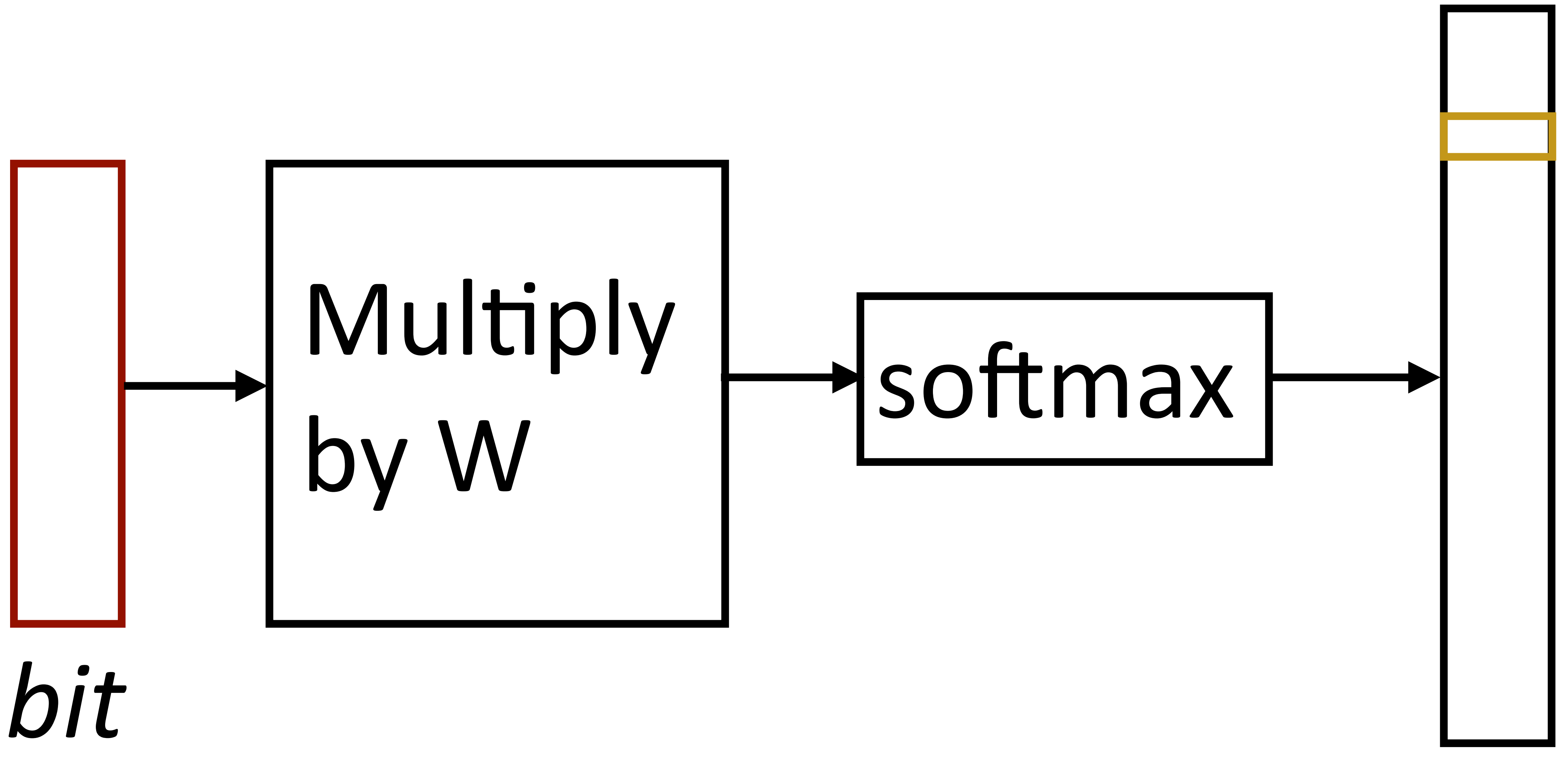
(word = *bit,* context = *the)*

Mikolov et al. (2013)

Skip-Gram

- Predict one word of context from word

the dog bit the man



gold = *dog*

$$P(w'|w) = \mathrm{softmax}(We(w))$$

- Another training example: *bit -> the*

- Parameters: *d* x |V| vectors, |V| x *d* output parameters (W) (also usable as vectors!). *d* is a hyperparameter

Mikolov et al. (2013)

Using Skip-Gram

- Context window size: how many words around the "center" word do we look?

the dog bit the man

k=1: two words of context

k=2: four words of context

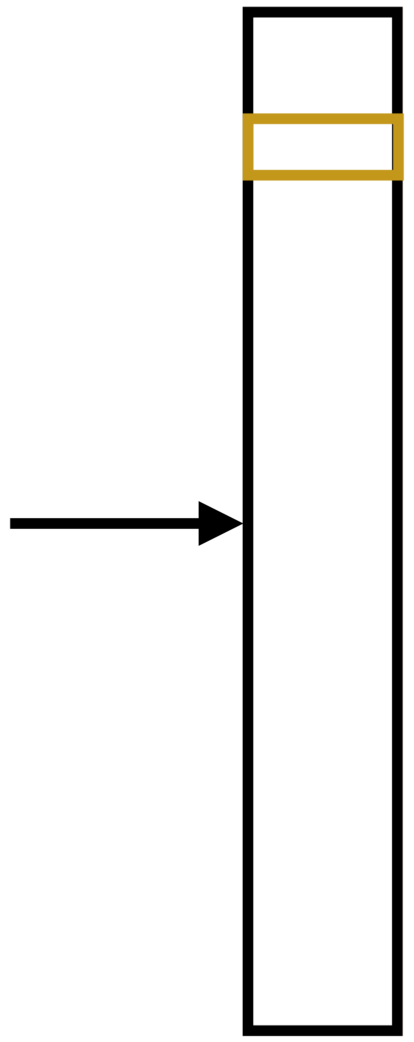
  - Advantages/disadvantages of different sizes of *k*?
- Training: maximize log likelihood of the examples derived given *k*, summed over a corpus (but we'll never use the model as is, only its embeddings)
- Initialization: need to randomly initialize in a reasonable way
- Vector size: controls capacity of model

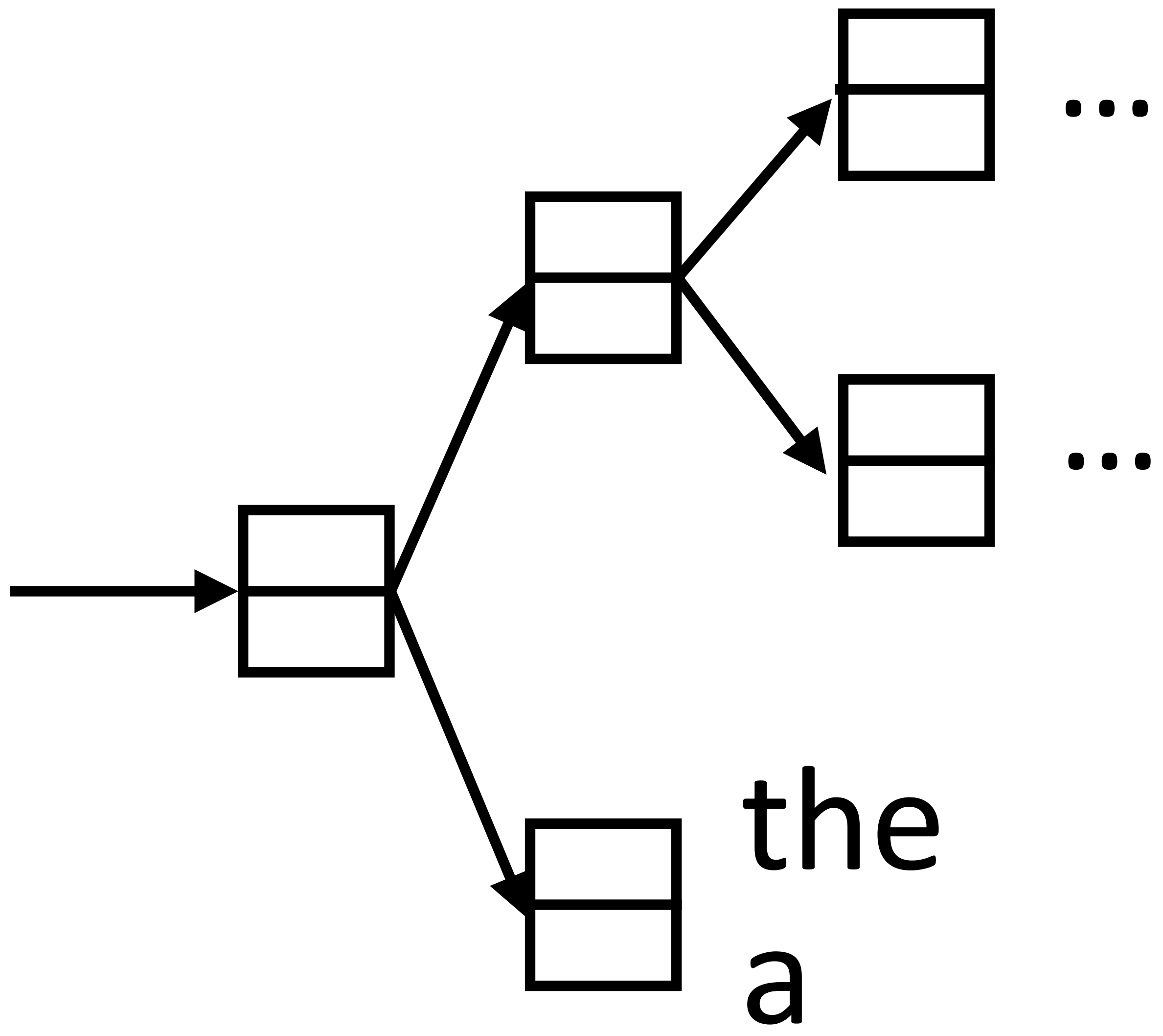
Mikolov et al. (2013)

Hierarchical Softmax

$$P(w'|w) = \mathrm{softmax}(We(w))$$

- Matmul + softmax over |V| is very slow to compute for skip-gram





- Huffman encode vocabulary, use binary classifiers to decide which branch to take
- log(|V|) binary decisions

- Standard softmax: |V| dot products of size *d*

- Hierarchical softmax: log(|V|) dot products of size *d*, |V| x *d* parameters

Mikolov et al. (2013)

Skip-Gram with Negative Sampling

- Take (word, context) pairs and classify them as "real" or not. Create random negative examples by sampling from unigram distribution

(*bit, the*) => +1

(*bit, cat*) => -1

(*bit, a*) => -1

(*bit, fish*) => -1

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

words in similar contexts select for similar *c* vectors

- *d* x |V| vectors, *d* x |V| context vectors (same # of params as before)

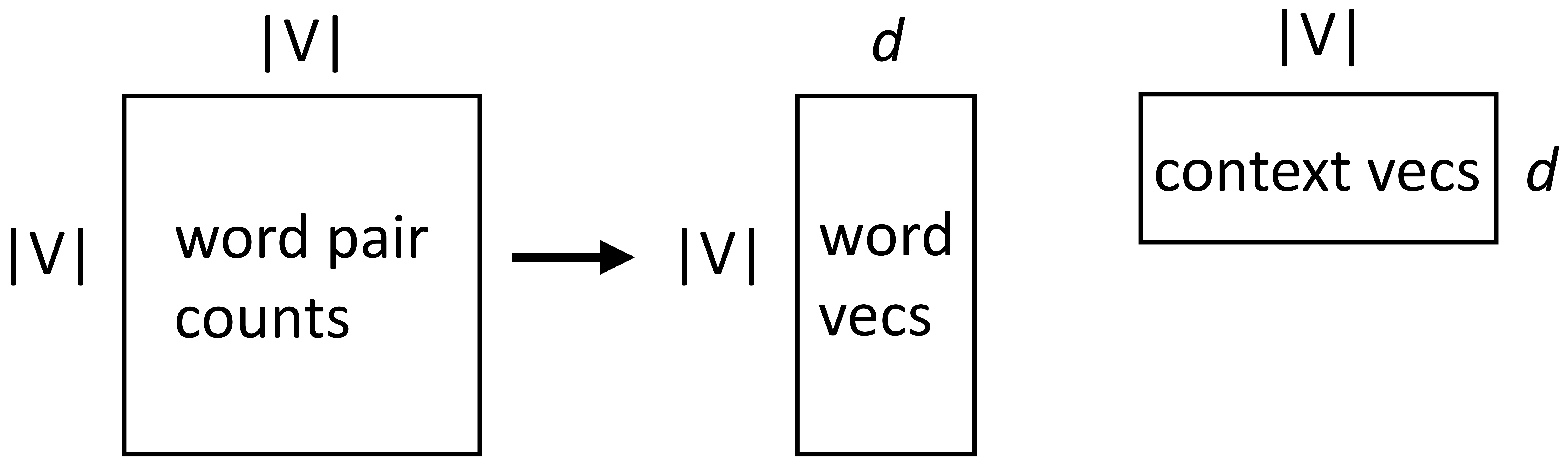
sampled

- Objective = $\log P(y=1|w,c) + \frac{1}{k}\sum_{i=1}^{n} \log P(y=0|w_i,c)$

Mikolov et al. (2013)

Connections with Matrix Factorization

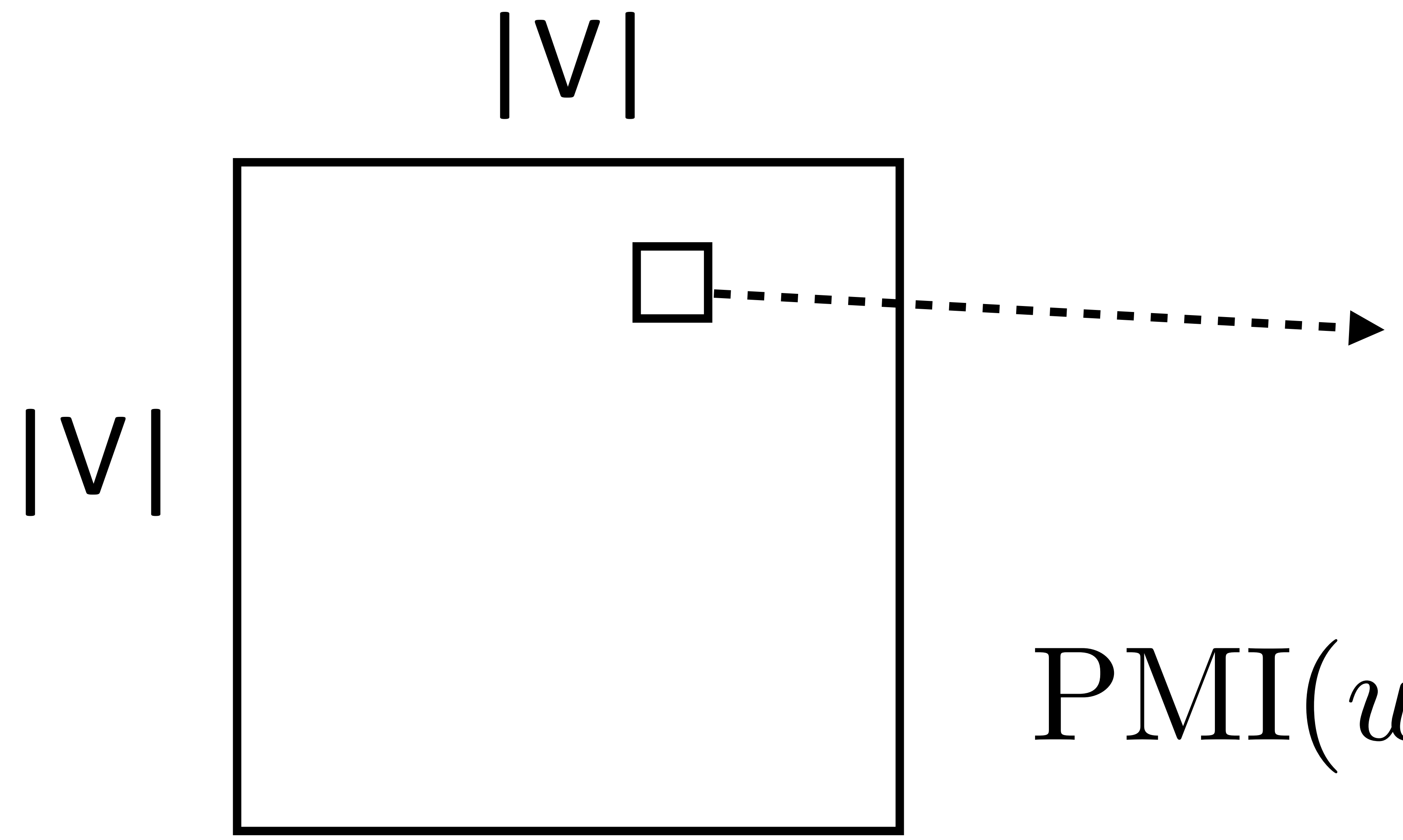
- Skip-gram model looks at word-word co-occurrences and produces two types of vectors



- Looks almost like a matrix factorization…

Levy et al. (2014)

Skip-Gram as Matrix Factorization



num negative samples

$$M_{ij} = \mathrm{PMI}(w_i, c_j) - \log k$$

$$\mathrm{PMI}(w_i, c_j) = \frac{P(w_i, c_j)}{P(w_i)P(c_j)} = \frac{\frac{\mathrm{count}(w_i, c_j)}{D}}{\frac{\mathrm{count}(w_i)}{D}\frac{count(c_j)}{D}}$$

Skip-gram objective *exactly* corresponds to factoring this matrix:

- *If* we sample negative examples from the unigram distribution over words
- …and it's a *weighted* factorization problem (weighted by word freq)

Levy et al. (2014)

GloVe

GloVe (**Glo**bal **Ve**ctors)

|V|

|V| word pair counts

- Also operates on counts matrix, weighted regression on the log co-occurrence matrix
- Objective = $\sum_{i,j} f(\mathrm{count}(w_i, c_j)) \left(w_i^\top c_j + a_i + b_j - \log \mathrm{count}(w_i, c_j)\right)^2$
- Constant in the dataset size (just need counts), quadratic in voc size
- By far the most common word vectors used today (30000+ citations)

Pennington et al. (2014)

GloVe (**Glo**bal **Ve**ctors): Example

- Objective = $\sum_{i,j} f(\text{count}(w_i, c_j)) \left(w_i^\top c_j + a_i + b_j - \log \text{count}(w_i, c_j)\right)^2$

	the	*dog*	*cat*	*ran*
the	0	200	200	0
dog	200	0	0	15
cat	200	0	0	15
ran	0	15	15	0

Linear regression with 6 pairs: each element is plugged into the above equation

+ constant = log count of pair

(made up values — matrix will generally be symmetric, though)

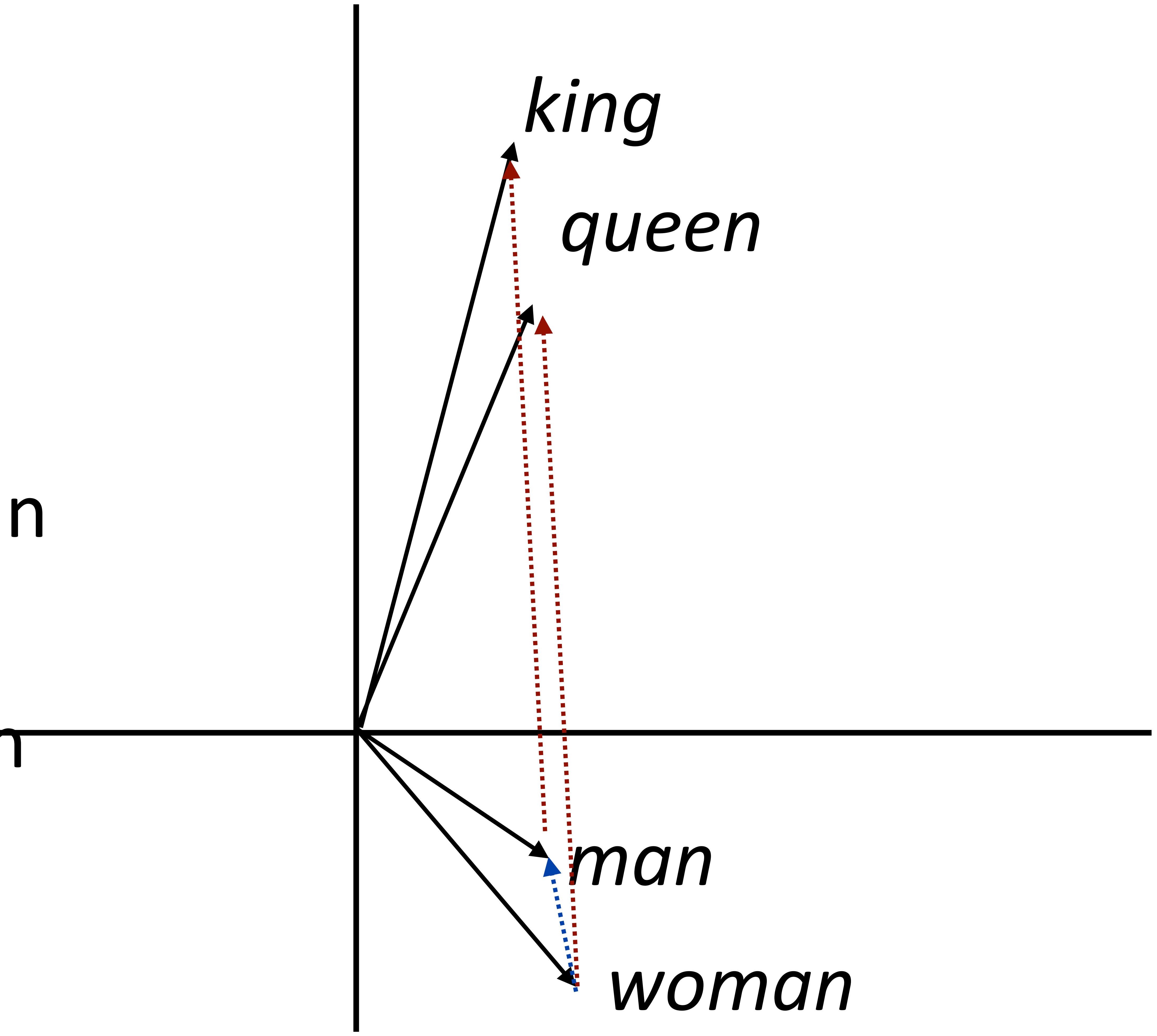
Pennington et al. (2014)

Analogies

(king - man) + woman = queen

king + (woman - man) = queen

- Why would this be?
- woman - man captures the difference in the contexts that these occur in
- Dominant change: more “he” with man and “she” with woman — similar to difference between king and queen
- Can evaluate on this as well



GloVe Motivation

Table 1: Co-occurrence probabilities for target words *ice* and *steam* with selected context words from a 6 billion token corpus. Only in the ratio does noise from non-discriminative words like *water* and *fashion* cancel out, so that large values (much greater than 1) correlate well with properties specific to ice, and small values (much less than 1) correlate well with properties specific of steam.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

- GloVe objective is derived to preserve regularities in cooccurrence of words with other words

$$F\left((w_i - w_j)^T \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}}$$

Pennington et al. (2014)

Other Methods

fastText: Sub-word Embeddings

- Same as SGNS, but break words down into n-grams with n = 3 to 6

  where:
  3-grams: <wh, whe, her, ere, re>
  4-grams: <whe, wher, here, ere>,
  5-grams: <wher, where, here>,
  6-grams: <where, where>

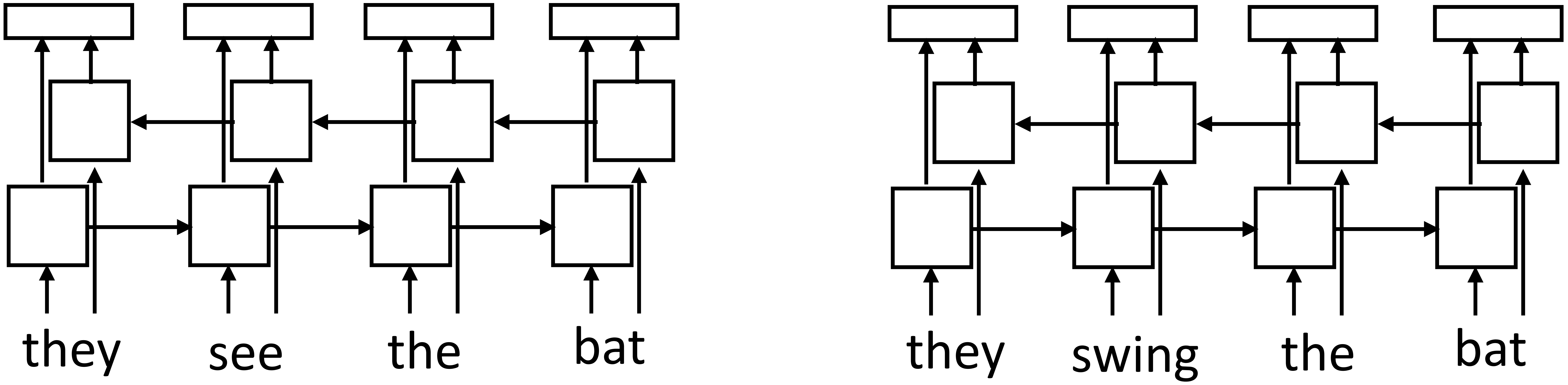
- Replace $w \cdot c$ in skip-gram computation with $\left(\sum_{g \in \text{ngrams}} w_g \cdot c \right)$

- Advantages?

Bojanowski et al. (2017)

Preview: Context-dependent Embeddings

- How to handle different word senses? One vector for *bat*



- Train a neural language model to predict the next word given previous words in the sentence, use its internal representations as word vectors
- *Context-sensitive* word embeddings: depend on rest of the sentence
- *Huge* improvements across nearly all NLP tasks over GloVe

Peters et al. (2018)

Compositional Semantics

- What if we want embedding representations for whole sentences?
- Skip-*thought* vectors (Kiros et al., 2015), similar to skip-gram generalized to a sentence level (more later)
- Is there a way we can compose vectors to make sentence representations? Summing?
- Will return to this in a few weeks as we move on to syntax and semantics

Using Word Embeddings

- Approach 1: learn embeddings as parameters from your data
  - Often works pretty well
- Approach 2: initialize using GloVe, keep fixed
  - Faster because no need to update these parameters
- Approach 3: initialize using GloVe, fine-tune
  - Works best for some tasks

Evaluating Word Embeddings

Evaluating Word Embeddings

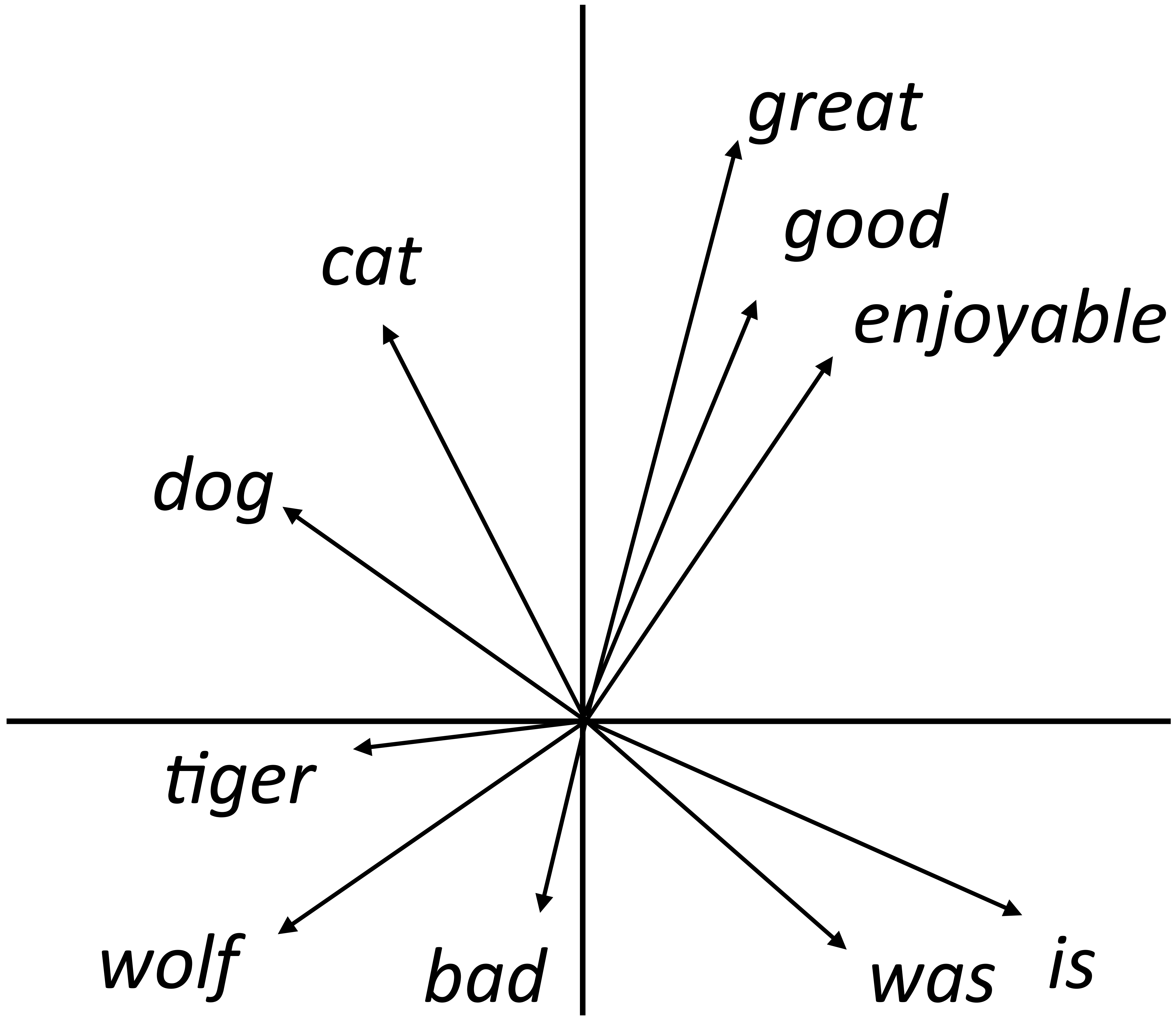
- What properties of language should word embeddings capture?

- Similarity: similar words are close to each other

- Analogy:

  good is to best as smart is to ???

  Paris is to France as Tokyo is to ???



Similarity

Method	WordSim Similarity	WordSim Relatedness	Bruni et al. MEN	Radinsky et al. M. Turk	Luong et al. Rare Words	Hill et al. SimLex
PPMI	.755	**.697**	.745	.686	.462	.393
SVD	**.793**	.691	**.778**	.666	**.514**	.432
SGNS	**.793**	.685	.774	**.693**	.470	**.438**
GloVe	.725	.604	.729	.632	.403	.398

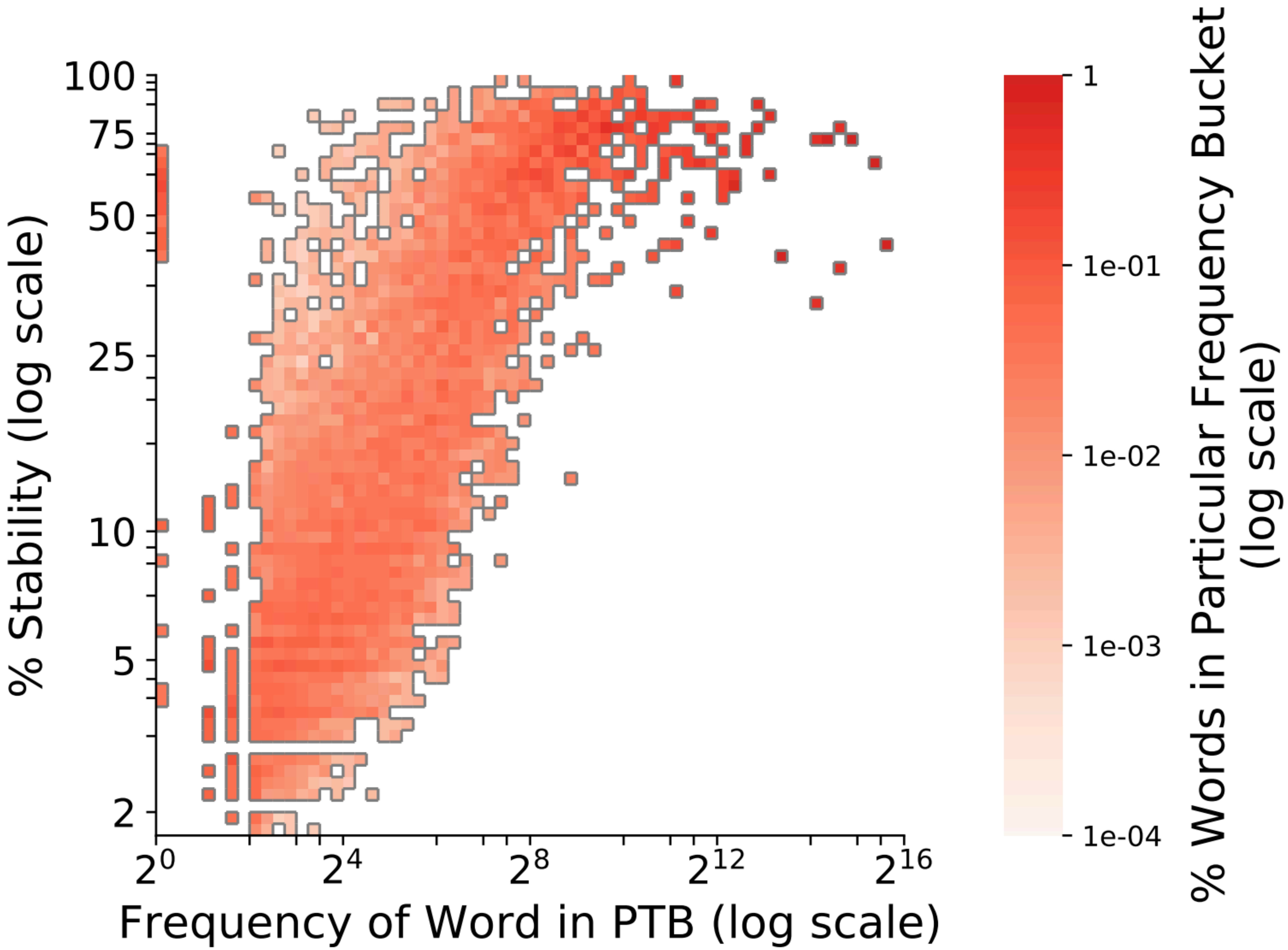
- SVD = singular value decomposition on PMI matrix
- GloVe does not appear to be the best when experiments are carefully controlled, but it depends on hyperparameters + these distinctions don't matter in practice

Levy et al. (2015)

Stability

- To what extent are the relationships captured by word embeddings consistent?

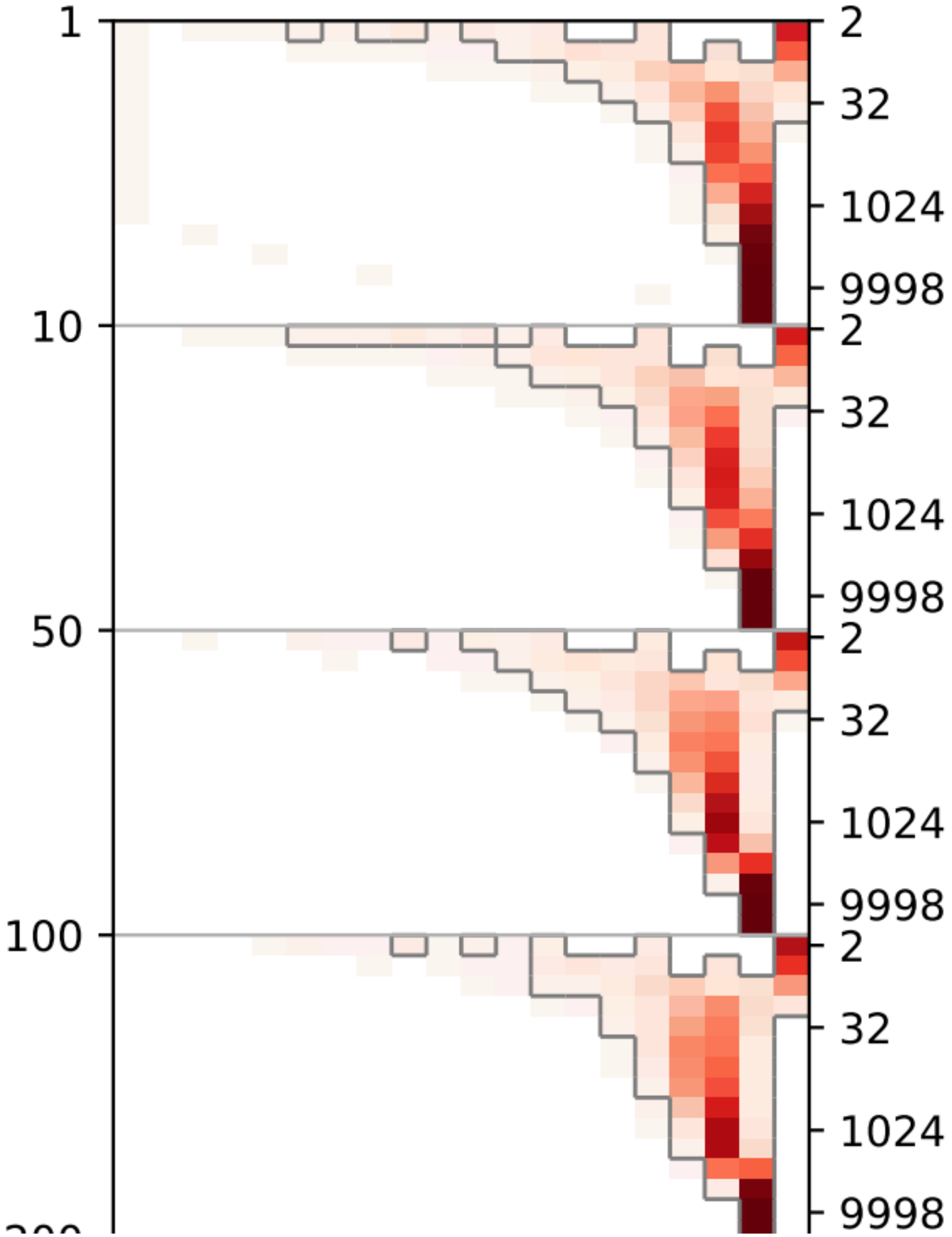
- Stability: percent overlap between nearest neighbors in embedding space if you retrain embeddings from different initialization



Burdick et al. (2018)

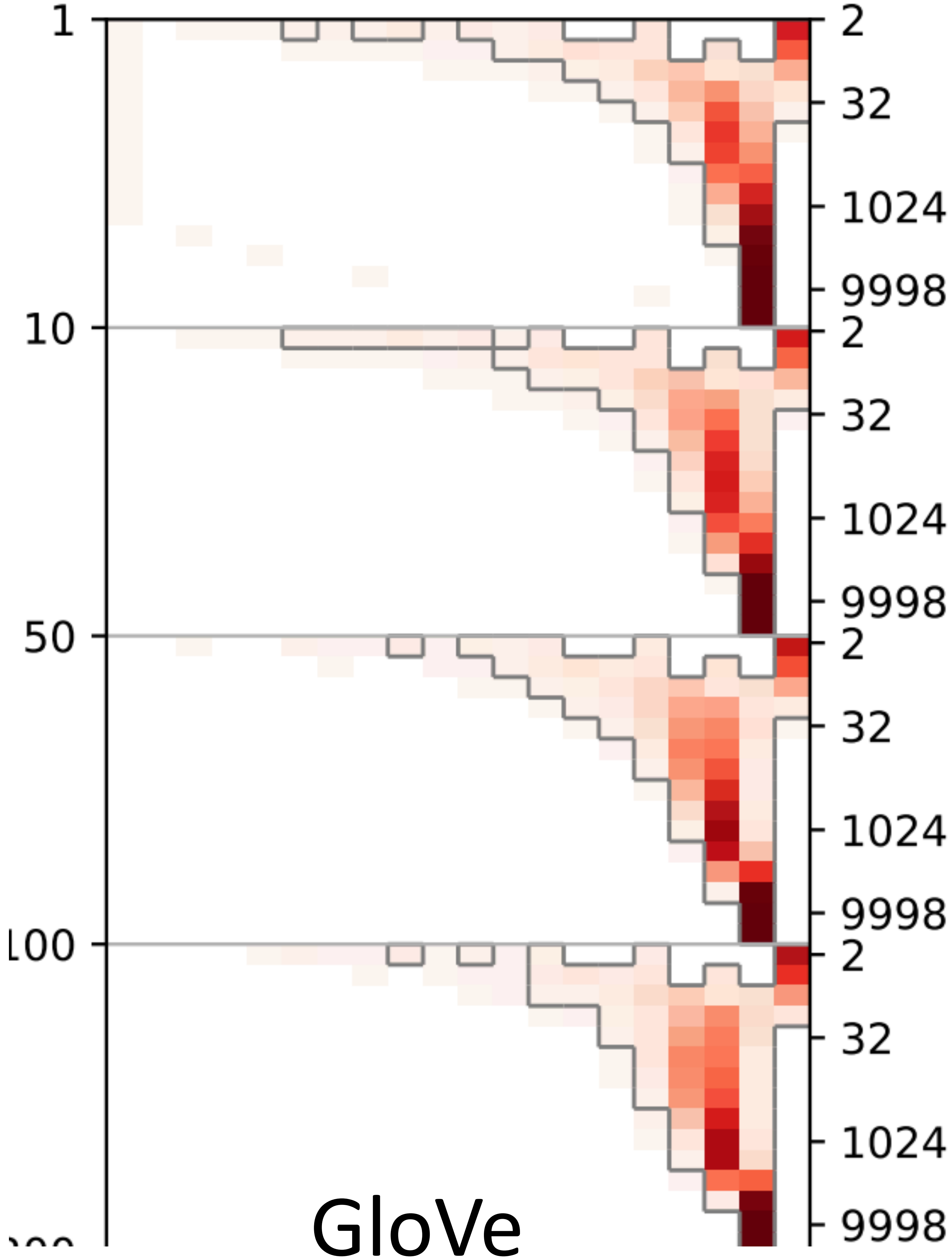
Stability: GloVe

- Left y-axis: bucketed corpus frequency
- Right y-axis: number of neighbors
- x-axis: percent of neighbors stable across samples
- Being all the way to the right is better (most neighbors are stable)

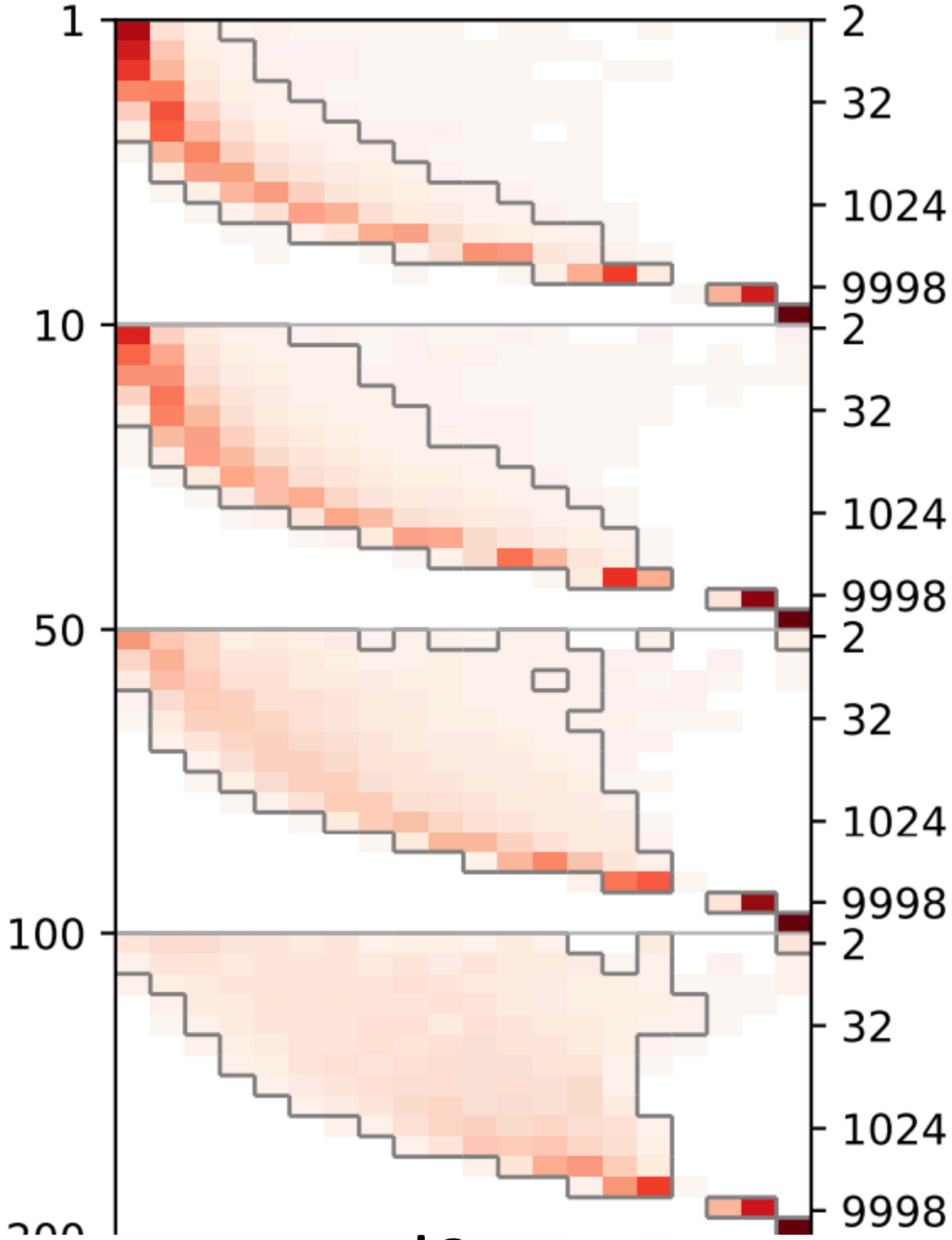


Burdick et al. (2018)

GloVe vs. word2vec: w2v much less stable!



GloVe



word2vec

Burdick et al. (2018)

What can go wrong with word embeddings?

- What's wrong with learning a word's "meaning" from its usage?

Ukraine's deputy defense minister resigns amid corruption allegations

From 2015 through 2020, there were at least 2,070 unintentional shootings by children under 18 in the US, according to a report from Everytown. Those shootings resulted in 765 deaths and 1,366 injuries.

Convicted child sex trafficker Ghislaine Maxwell has said a decades-old photograph of Prince Andrew with his sexual abuse accuser Virginia Giuffre is "fake," in a series of interviews from prison.

What do we mean by bias?

- Identify *she* - *he* axis in word vector space, project words onto this axis

- Nearest neighbor of (b - a + c)

Extreme *she* occupations

1. homemaker	2. nurse	3. receptionist
4. librarian	5. socialite	6. hairdresser
7. nanny	8. bookkeeper	9. stylist
10. housekeeper	11. interior designer	12. guidance counselor

Extreme *he* occupations

1. maestro	2. skipper	3. protege
4. philosopher	5. captain	6. architect
7. financier	8. warrior	9. broadcaster
10. magician	11. figher pilot	12. boss

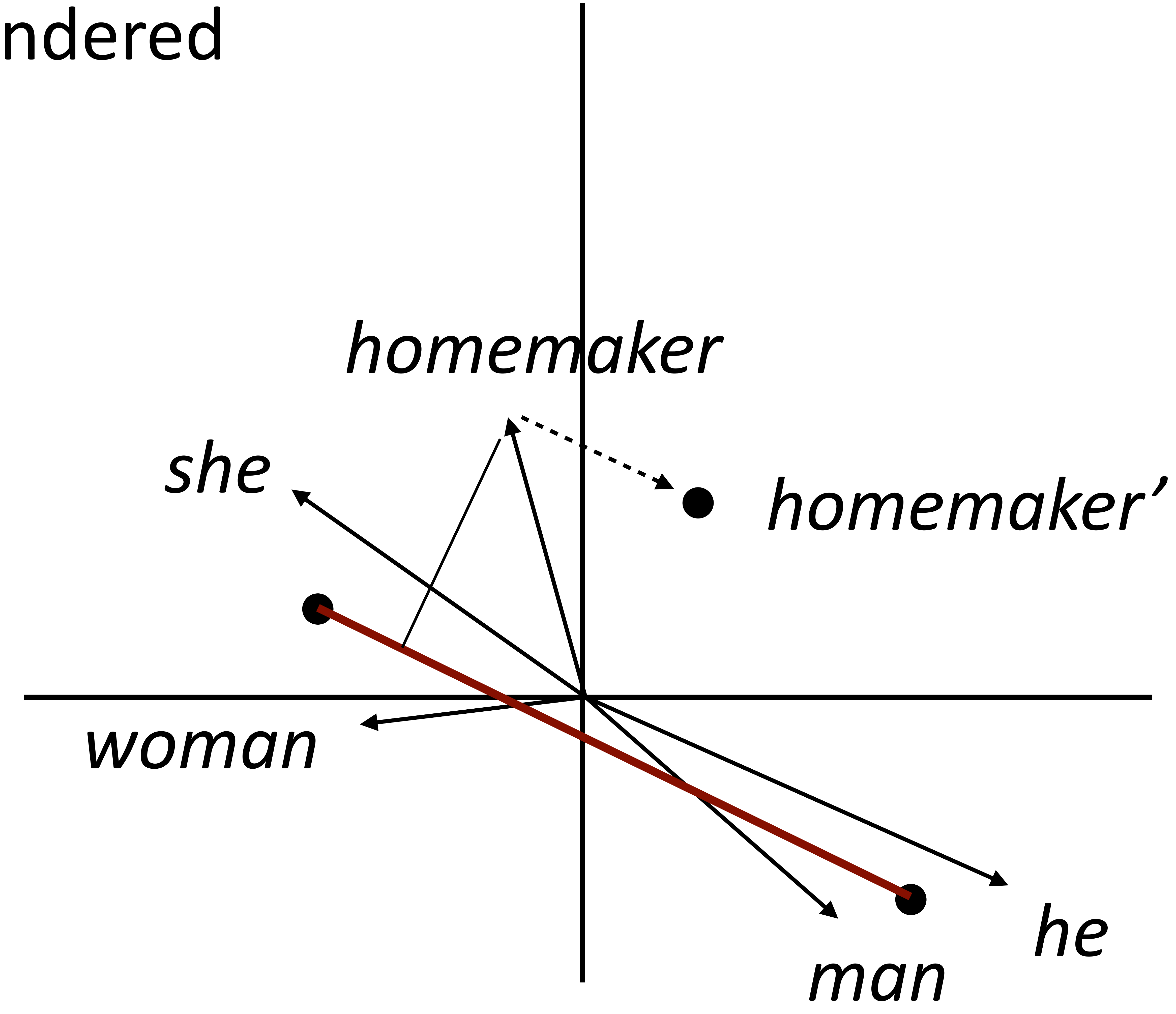
Bolukbasi et al. (2016)

Racial Analogies	
black → homeless	caucasian → servicemen
caucasian → hillbilly	asian → suburban
asian → laborer	black → landowner
Religious Analogies	
jew → greedy	muslim → powerless
christian → familial	muslim → warzone
muslim → uneducated	christian → intellectually

Manzini et al. (2019)

Debiasing

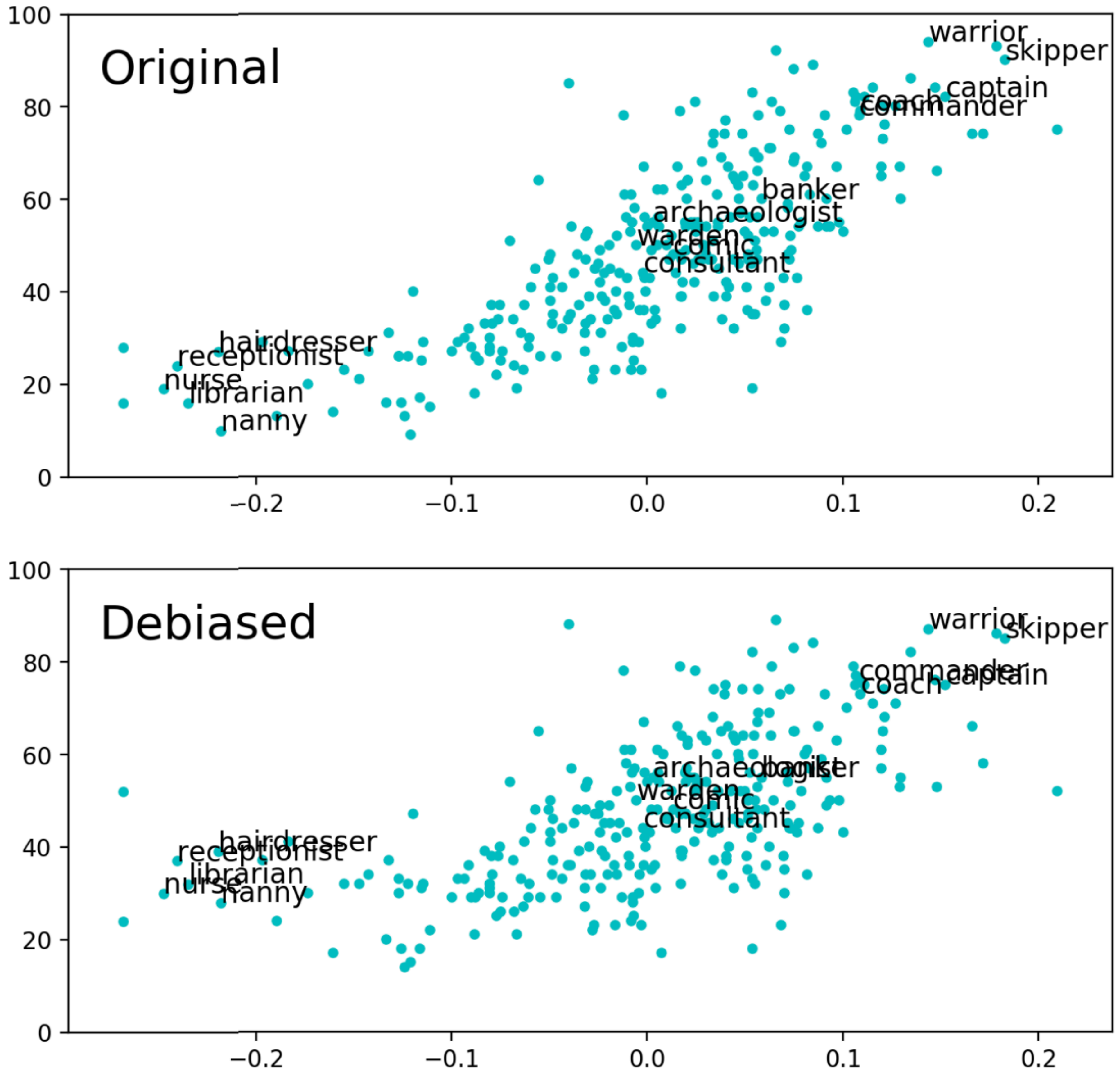
- Identify gender subspace with gendered words
- Project words onto this subspace
- Subtract those projections from the original word



Bolukbasi et al. (2016)

Hardness of Debiasing

- Not that effective…and the male and female words are still clustered together
- Bias pervades the word embedding space and isn't just a local property of a few words



(a) The plots for HARD-DEBIASED embedding, before (top) and after (bottom) debiasing.

Gonen and Goldberg (2019)

Takeaways

- Word vectors: learning word -> context mappings has given way to matrix factorization approaches (constant in dataset size)
- Lots of pretrained embeddings work well in practice, they capture some desirable properties
- Even better: context-sensitive word embeddings (ELMo)
- Next time: language modeling and Transformers