CS769 Advanced NLP

Word Embeddings

Junjie Hu

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Slides adapted from Noah, Yulia

https://junjiehu.github.io/cs769-fall23/

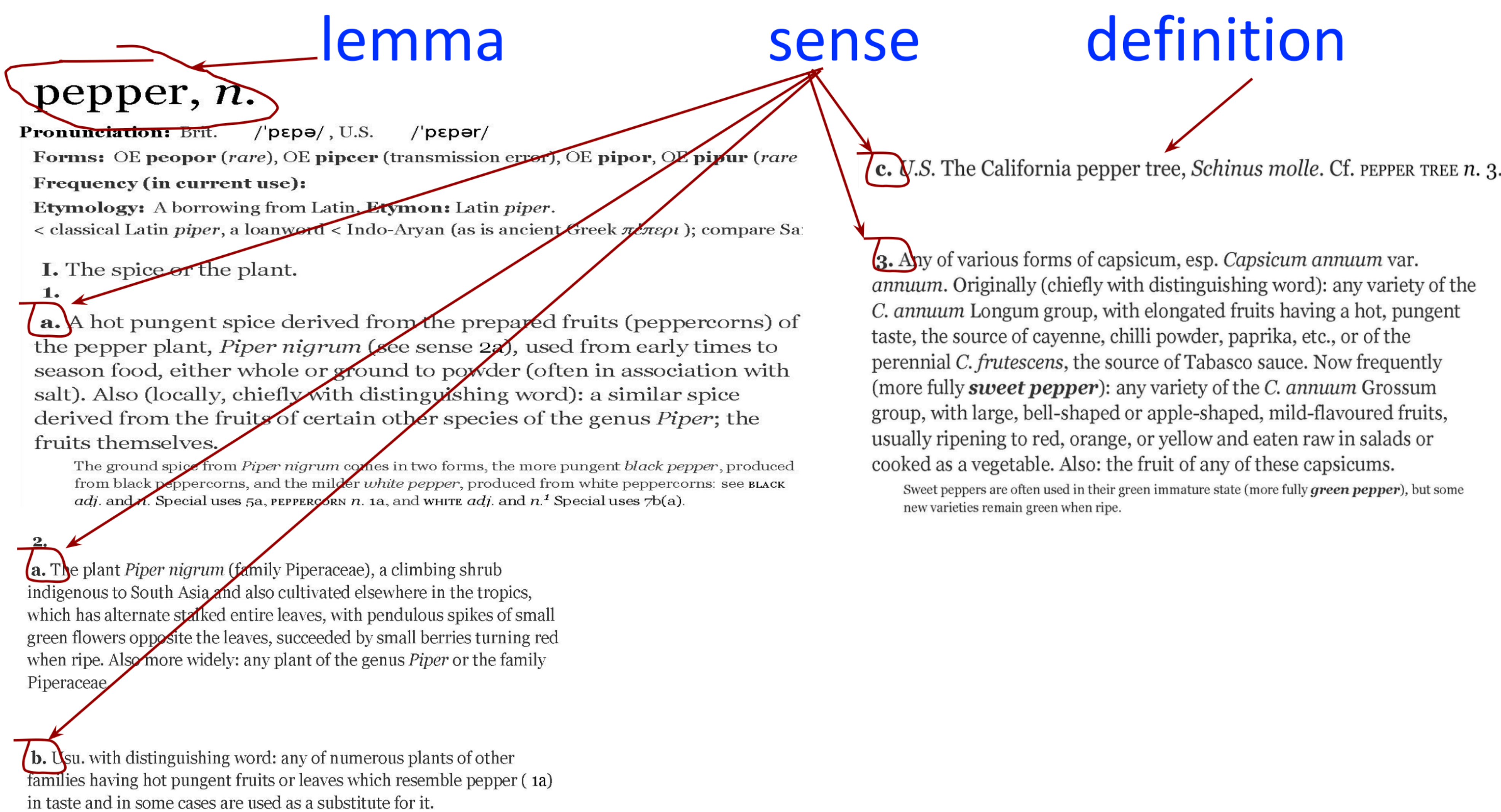
Goals for Today

- Lexical Semantics and Distributional Semantics
- **Count Based** Word Methods (e.g, TF-IDF, PMI)
- **Matrix Factorization** (e.g., topic modeling)
- **Word Embeddings** (e.g., Skip-gram, CBOW)
- Evaluation (intrinsic and extrinsic)

How should we represent the meaning of the word?

Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definition



pepper, *n.*

Pronunciation: Brit. /ˈpɛpə/ , U.S. /ˈpɛpər/

Forms: OE **peopor** (*rare*), OE **pipcer** (transmission error), OE **pipor**, OE **pipur** (*rare*

Frequency (in current use):

Etymology: A borrowing from Latin. **Etymon:** Latin *piper*.

< classical Latin *piper*, a loanword < Indo-Aryan (as is ancient Greek *πέπερι*); compare Sa

I. The spice or the plant.

1.

a. A hot pungent spice derived from the prepared fruits (peppercorns) of the pepper plant, *Piper nigrum* (see sense 2a), used from early times to season food, either whole or ground to powder (often in association with salt). Also (locally, chiefly with distinguishing word): a similar spice derived from the fruits of certain other species of the genus *Piper*; the fruits themselves.

The ground spice from *Piper nigrum* comes in two forms, the more pungent *black pepper*, produced from black peppercorns, and the milder *white pepper*, produced from white peppercorns: see BLACK *adj.* and *n.* Special uses 5a, PEPPERCORN *n.* 1a, and WHITE *adj.* and *n.*[1] Special uses 7b(a).

2.

a. The plant *Piper nigrum* (family Piperaceae), a climbing shrub indigenous to South Asia and also cultivated elsewhere in the tropics, which has alternate stalked entire leaves, with pendulous spikes of small green flowers opposite the leaves, succeeded by small berries turning red when ripe. Also more widely: any plant of the genus *Piper* or the family Piperaceae.

b. Usu. with distinguishing word: any of numerous plants of other families having hot pungent fruits or leaves which resemble pepper (1a) in taste and in some cases are used as a substitute for it.

c. *U.S.* The California pepper tree, *Schinus molle*. Cf. PEPPER TREE *n.* 3.

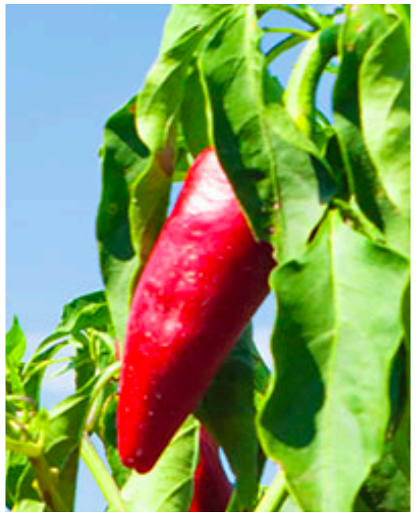
3. Any of various forms of capsicum, esp. *Capsicum annuum* var. *annuum*. Originally (chiefly with distinguishing word): any variety of the *C. annuum* Longum group, with elongated fruits having a hot, pungent taste, the source of cayenne, chilli powder, paprika, etc., or of the perennial *C. frutescens*, the source of Tabasco sauce. Now frequently (more fully ***sweet pepper***): any variety of the *C. annuum* Grossum group, with large, bell-shaped or apple-shaped, mild-flavoured fruits, usually ripening to red, orange, or yellow and eaten raw in salads or cooked as a vegetable. Also: the fruit of any of these capsicums.

Sweet peppers are often used in their green immature state (more fully ***green pepper***), but some new varieties remain green when ripe.

Oxford English Dictionary: https://www.oed.com/

Lemma pepper

- Sense 1: spice from pepper plant
- Sense 2: the pepper plant itself
- Sense 3: another similar plant (Jamaican pepper)
- Sense 4: plant with peppercorns (California pepper)
- Sense 5: capsicum (i.e., chili, paprika, bell pepper, etc)



Lexical Semantics

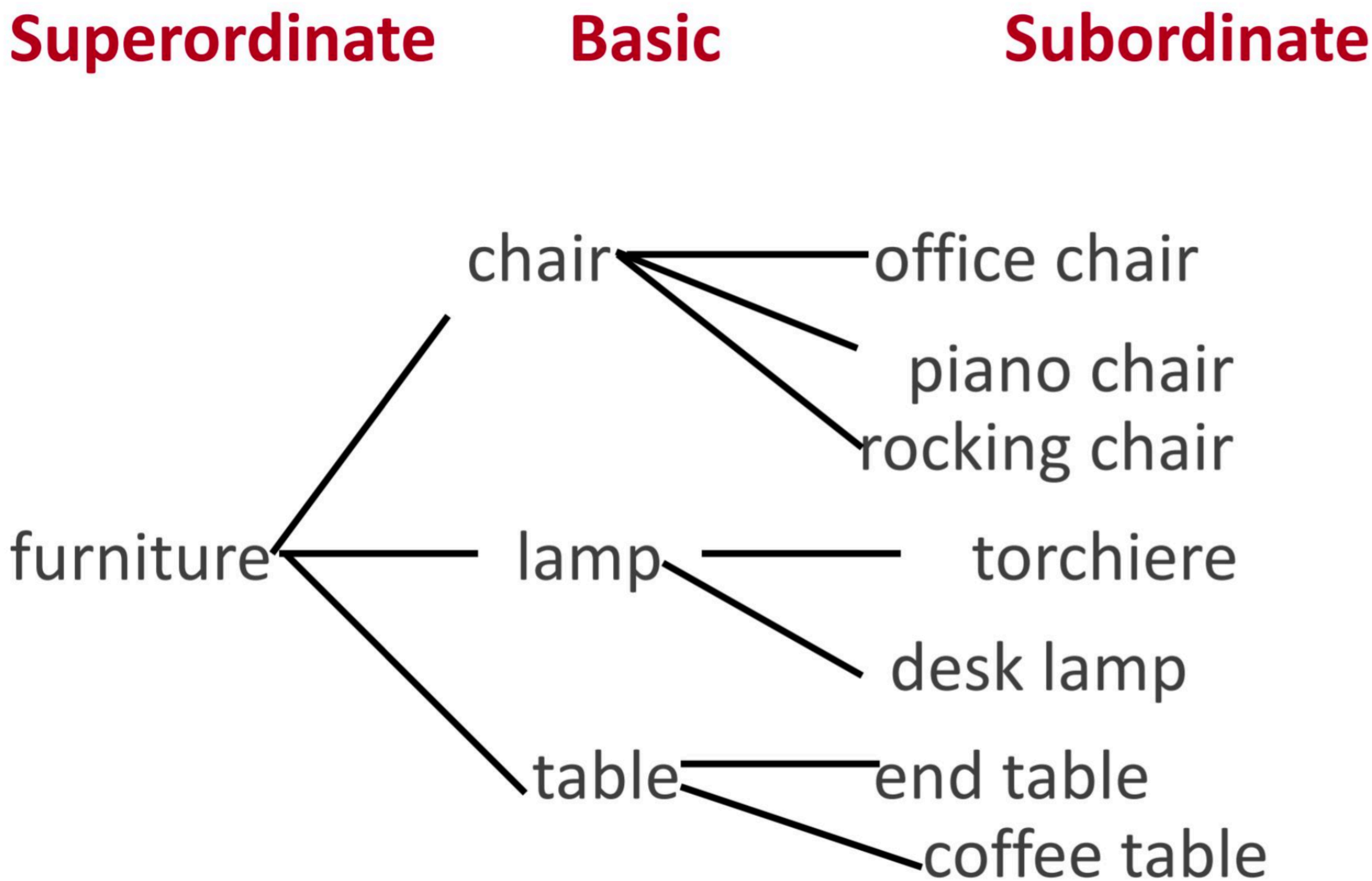
- How should we represent the meaning of the word?
  - Words, lemmas, senses, definition
  - Relationships between words or senses
    1. Synonymity: same meaning, e.g., couch/sofa
    2. Antonymy: opposite senses, e.g., hot/cold
    3. Similarity: similar meanings, e.g., car/bicycle
    4. Relatedness: association, e.g., car/gasoline
    5. Superordinate/Subordinate: e.g., car/vehicle, mango/fruit

Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definition
  - Relationships between words or senses
  - Taxonomy: abstract -> concrete

Taxonomy

- abstract -> concrete



Lexical Semantics

- How should we represent the meaning of the word?
  - Words, lemmas, senses, definition
  - Relationships between words or senses
  - Taxonomy: abstract -> concrete
  - Semantic frames and roles

Semantic Frame

- A set of words that denote perspectives or participants in an event
  - Tom brought a book from Bill.

buyer event from the perspective of the buyer

  - Bill sold a book to Tom.

seller event from the perspective of the seller

Mismatch

- Theories of language tend to view the data (words, sentences, documents) and abstractions over it as *symbolic* or categorical.
    - Uses symbols to represent linguistic information

- Machine learning algorithms built on optimization rely more on *continuous* data.
    - Uses floating-point numbers (vectors)

Documents and Words as Vectors

- A common thread: we derive the vectors from a corpus (collection of documents), with no annotation
  - a.k.a. “unsupervised” or “self-supervised” learning
  - Similar to language modeling
  - Human-written raw sentences have already provide supervision on how words co-exist in a sentence.

Problems with Discrete Representations

- Too coarse: *expert* ↔ *skillful*
- Sparse
- Subjective
- Expensive
- Hard to compute word relationships

expert [0 0 0 **1** 0 0 0 0 0 0 0 0 0 0 0]

skillful [0 0 0 0 0 0 0 0 0 0 **1** 0 0 0 0]

Distributional Hypothesis

"The meaning of a word is its use in the language"

[Wittgenstein 1943]

"You shall know a word by the company it keeps"

[Firth 1957]

"If A and B have almost identical environments we say that they are synonyms."

[Harris 1954]

Example

- What does “Ong Choy” mean?
  - Suppose you see these sentences:
    - Ong Choy is delicious **sautéed with garlic**
    - Ong Choy is superb **over rice**
    - Ong Choy **leaves** with salty sauces
  - And you’ve also seen these:
    - … water spinach **sautéed with garlic over rice**
    - Chard stems and **leaves** are delicious
    - Collard greens and other **salty leafy** greens

Ong Choy ≈ “Water Spinach”?

- Ong Choy is a leafy green like spinach, chard, or collard greens

Ong Choy: pronunciation of “蕹菜” in Cantonese

Model of Meaning Focusing on Similarity

- Each word = a vector
  - Similar words are “nearby in space”
  - the standard way to represent meaning in NLP

Approaches for encoding words as vectors

- Counting-based methods (e.g., TF-IDF)
- Matrix factorization (e.g., topic modeling)
- Brown clusters
- Word2vec (e.g., Skip-gram, CBOW)

Count-based Model

— A naive way to represent words in a corpus is to count their statistics.

Count-based Method

- Words are not independent, identically distributed (IID)!
  - **Predictable given history**: n-gram/Markov models
  - **Predictable given other words in the document**: topic models
- Let $\mathcal{Z} = \{1, \ldots, K\}$ be a set of "topic"/"themes" that will capture the interdependence of words in a document
  - Usually these are not named or characterized in advance; they're just K different values with no a prior meaning.

Notation

- $\mathbf{x}$ is the corpus
- $\mathbf{x}_c$ is the c-th document in the corpus
- ℓ_c is the length of $\mathbf{x}_c$ (in tokens)
- N is the total count of tokens in the corpus,
  $$N = \sum_{c=1}^{C} \ell_c$$
- V, C are the vocabulary size and document size respectively.

Word-Document Matrix

- Let $\mathbf{A} \in \mathbb{R}^{V \times C}$ contain the statistics of association between words in the vocabulary and documents.
  - Example: three documents

$\boldsymbol{x}_1$: `yes , we have no bananas`

$\boldsymbol{x}_2$: `say yes for bananas`

$\boldsymbol{x}_3$: `no bananas , we say`

	1	2	3
`,`	1	0	1
`bananas`	1	1	1
`for`	0	1	0
`have`	1	0	0
`no`	1	0	1
`say`	0	1	1
`we`	1	0	1
`yes`	1	1	0

For example, $\mathbf{A}$ could be defined as a count matrix: count of word v in the c-th document

$$[\mathbf{A}]_{v,c} = \text{count}_{\boldsymbol{x}_c}(v)$$

Note: $\mathbf{A}$ could be other statistics like TF-IDF, PMI, more.

Encoding context with TF-IDF

- Problem for word-doc matrix: useless signal from *the, they, and*
- ***Solution:*** **TF-IDF** incorporates two terms that capture these conflicting constraints:
  - **Term frequency (tf):** frequency of the word in the document

$$\mathrm{tf}_{v,c} = \log(\mathrm{count}(v,c) + 1)$$

  - **Document frequency (df):** number of documents that a term occurs in. Inverse document frequency (idf) just takes the inverse:

$$\mathrm{idf}_v = \log\left(\frac{|N|}{|\{c|v \in c,\ \forall c \in C\}|}\right)$$

Higher for words that occur in fewer documents

where N is the no. of documents.

$$[A]_{v,c} = \mathrm{tf}_{v,c} \cdot \mathrm{idf}_v$$

TF-IDF Example

word	df	idf
Romeo	1	1.57
salad	2	1.27
Falstaff	4	0.967
forest	12	0.489
battle	21	0.246
wit	34	0.037
fool	36	0.012
good	37	0
sweet	37	0

Example: 4 documents in red

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	0.074	0	0.22	0.28
good	0	0	0	0
fool	0.019	0.021	0.0036	0.0083
wit	0.049	0.044	0.018	0.022

Association Score

- Let $\frac{\text{count}_{\boldsymbol{x}}(v)}{N}$ be the percentage of word v in all docs, and $\text{count}_{\boldsymbol{x}_c}(v)$ be the word count in a doc $\boldsymbol{c}$.

- By **chance** (under a unigram model), we expect that $\frac{\text{count}_{\boldsymbol{x}}(v)}{N}$ (percentage) of words in document $\boldsymbol{c}$ of length $\ell_{\boldsymbol{c}}$ are the word v

- As document $\boldsymbol{c}$ may consist of *different topics*, is the occurrence of word v in $\boldsymbol{c}$ surprisingly high (or low), comparing to **chance**?

- Intuition: consider **the ratio of observed frequency** ($\text{count}_{\boldsymbol{x}_c}(v)$) **to "chance"** $\left(\frac{\text{count}_{\boldsymbol{x}}(v)}{N} \cdot \ell_{\boldsymbol{c}}\right)$

Pointwise Mutual Information

- A common measurement is to define $\mathbf{A}$ as positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{\text{count}_{\boldsymbol{x}_c}(v)}{\frac{\text{count}_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot \text{count}_{\boldsymbol{x}_c}(v)}{\text{count}_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

where $[x]_+ = \max(0, x)$.

$$[\mathbf{A}]_{\text{bananas},1} = \log \frac{15 \cdot 1}{3 \cdot 6} \approx -0.18 \rightarrow 0$$

$$[\mathbf{A}]_{\text{for},2} = \log \frac{15 \cdot 1}{1 \cdot 4} \approx 1.32$$

	1	2	3
,	1	0	1
bananas	1	1	1
for	0	1	0
have	1	0	0
no	1	0	1
say	0	1	1
we	1	0	1
yes	1	1	0

$\mathbf{A}$: count matrix

	1	2	3
,			
bananas	0	0	0
for		1.32	
have			
no			
say			
we			
yes			

$\mathbf{A}$: PMI

A Nod to Information Theory

- *Single* event: pointwise mutual information for two random variables (r.v.) A and B taking values a and b:

$$\begin{aligned}\text{PMI}(a, b) &= \log \frac{p(A = a, B = b)}{p(A = a) \cdot p(B = b)} \\ &= \log \frac{p(A = a \mid B = b)}{p(A = a)} \\ &= \log \frac{p(B = b \mid A = a)}{p(B = b)}\end{aligned}$$

- *All possible* events: **average mutual information**

$$\text{MI}(A, B) = \sum_{a,b} p(A = a, B = b) \cdot \mathrm{PMI}(a, b)$$

- PMI, MI: amount of information each r.v. offers about the other.
- Recall entropy: amount of information or uncertainty in a single r.v.

Pointwise Mutual Information

- If a word v appears with nearly the same frequency in every doc, its row $[\mathbf{A}]_{v,*}$ is nearly 0.
- If a word v appears only in doc c, their PMI $([\mathbf{A}]_{v,c})$ is large and positive
- PMI is very sensitive to rare occurrences: smooth the frequencies and filter rare words.
- PMI: tells us where a unigram model is most wrong.

	1	2	3
,	1	0	1
bananas	1	1	1
for	0	1	0
have	1	0	0
no	1	0	1
say	0	1	1
we	1	0	1
yes	1	1	0

$\mathbf{A}$: count matrix

	1	2	3
,			
bananas	0	0	0
for		1.32	
have			
no			
say			
we			
yes			

$\mathbf{A}$: PMI

Reflection

- Can we use the **rows** of this association matrix $\mathbf{A}$ as **word vectors** in a neural net model?
  - Word embedding's dimension is linear to no. of document, since $\mathbf{A} \in \mathbb{R}^{V \times C}$. Too large & not generalizable to other documents.
  - Too many zeros for each **word vector** (sparse)

- Can we use the **columns** of this association matrix $\mathbf{A}$ as **document vectors** in a neural net model?
  - Yes. If we use a count function for $\mathbf{A}$, then this is essentially the bag-of-word representation for each document.
  - Too many zeros for each **document vector** (sparse)

Matrix Factorization Based Method

Topic Models: Latent Semantic Indexing/Analysis

Deerwester et al., 1990, LSA

- LSA or LSI seeks to solve:

$$\underset{V \times C}{\mathbf{A}} \approx \hat{\mathbf{A}} = \underset{V \times d}{\mathbf{M}} \times \underset{d \times d}{\operatorname{diag}}(\mathbf{s}) \times \underset{d \times C}{\mathbf{C}}^{\top}$$

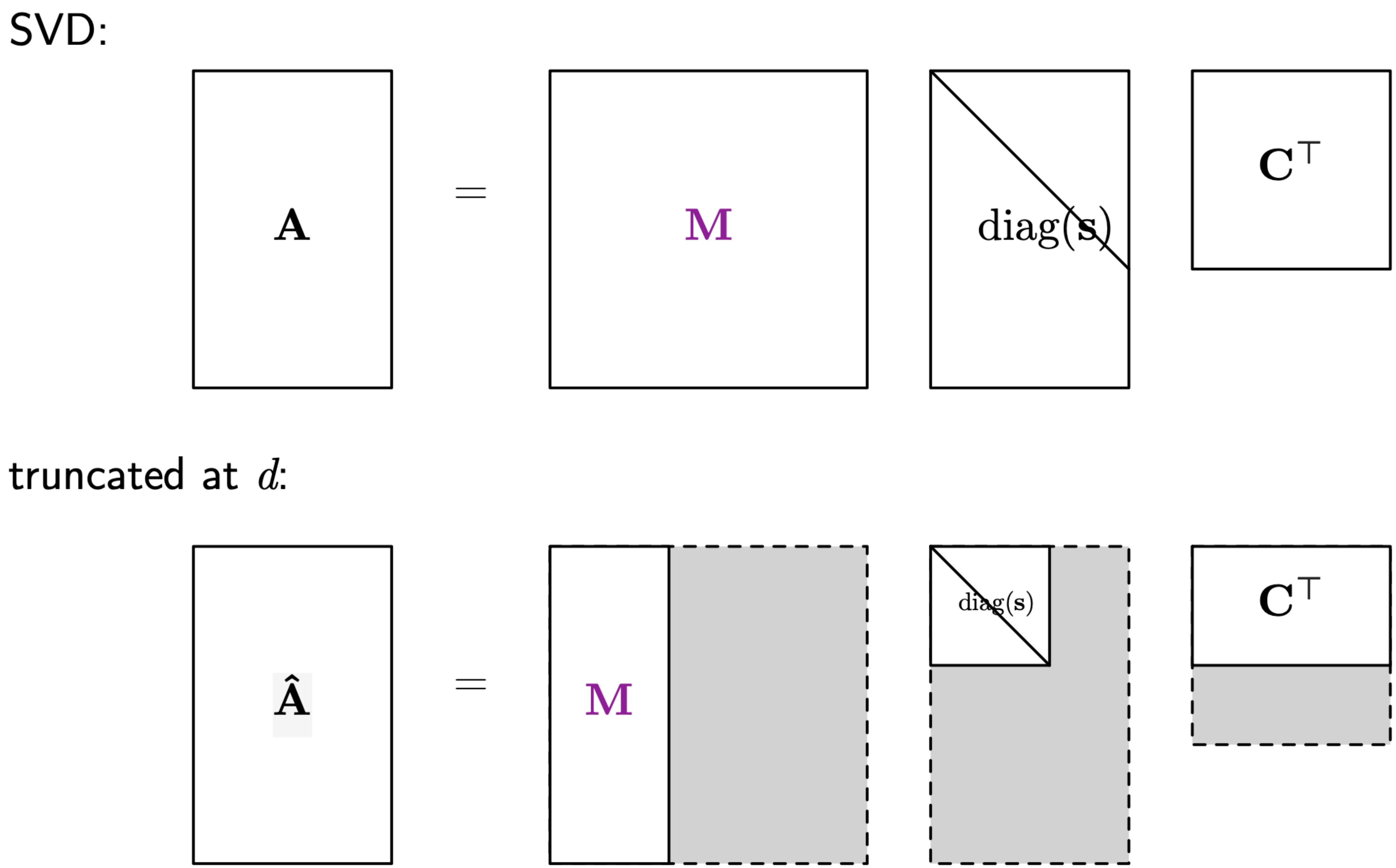
where $\mathbf{M}$ is the word embedding matrix, $\mathbf{C}$ is the document embedding matrix.

$$[\mathbf{A}]_{v,c} \approx \sum_{i=1}^{d} [\mathbf{v}_v]_i \cdot [\mathbf{s}]_i \cdot [\mathbf{c}_c]_i$$

- This can be solved by **singular value decomposition** to $\mathbf{A}$, then truncating to *d* dimensions.
  - $\mathbf{M}$ contains left singular vectors of $\mathbf{A}$
  - $\mathbf{C}$ contains right singular vectors of $\mathbf{A}$
  - $\mathbf{s}$ are singular values of $\mathbf{A}$: nonnegative and conventionally organized in decreasing order.

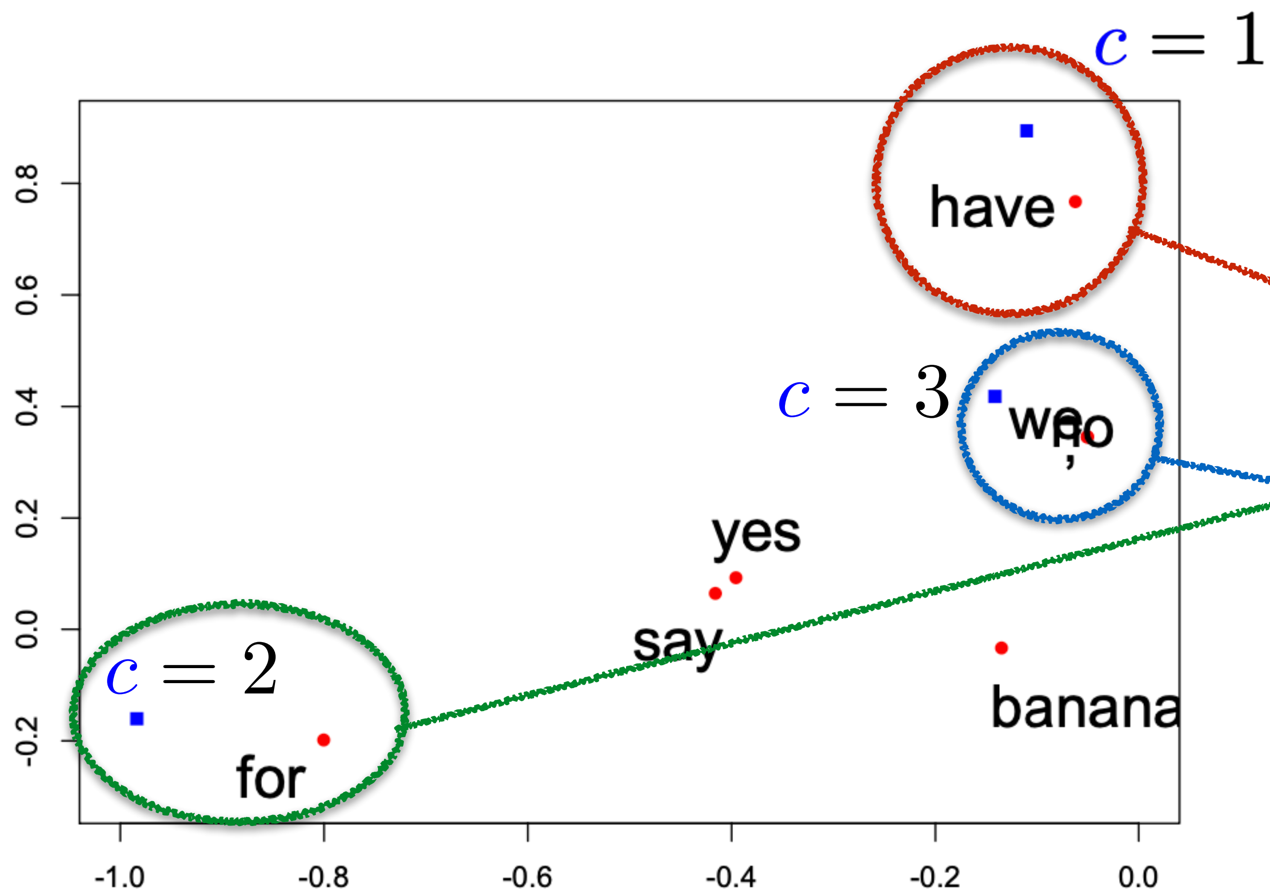
Truncated Singular Value Decomposition

- Some element of $\mathbf{s}$ are nearly 0: delete these values to obtain a "low-rank" approximation of $\mathbf{A}$



LSI/A Example

- *d*=2, project vectors of words and documents to two dimensional space.



	1	2	3
,	1	0	1
bananas	1	1	1
for	0	1	0
have	1	0	0
no	1	0	1
say	0	1	1
we	1	0	1
yes	1	1	0

$\mathbf{A}$: count matrix

Note: "no", "we" and "," are all in the exact same spot. Why?

- These words have the same statistics in this example, but this doesn't imply that they have the same semantic meaning.

Refection

- LSA creates a mapping of words and documents into the same low-dimensional space. Remove the reliance on no. of documents for word embeddings.
- $\mathbf{A}$ is sparse and noisy. LSA “squeezes” the zeros, finds the relationship between words and documents through topics (features), and finds the best rank-*d* approximation to $\mathbf{A}$.
- More variants of LSA
    - Probabilistic Latent Semantic Indexing (PLSI)
    - Latent Dirichlet Allocation (LDA)
    - Nonnegative Matrix Factorization (NMF)

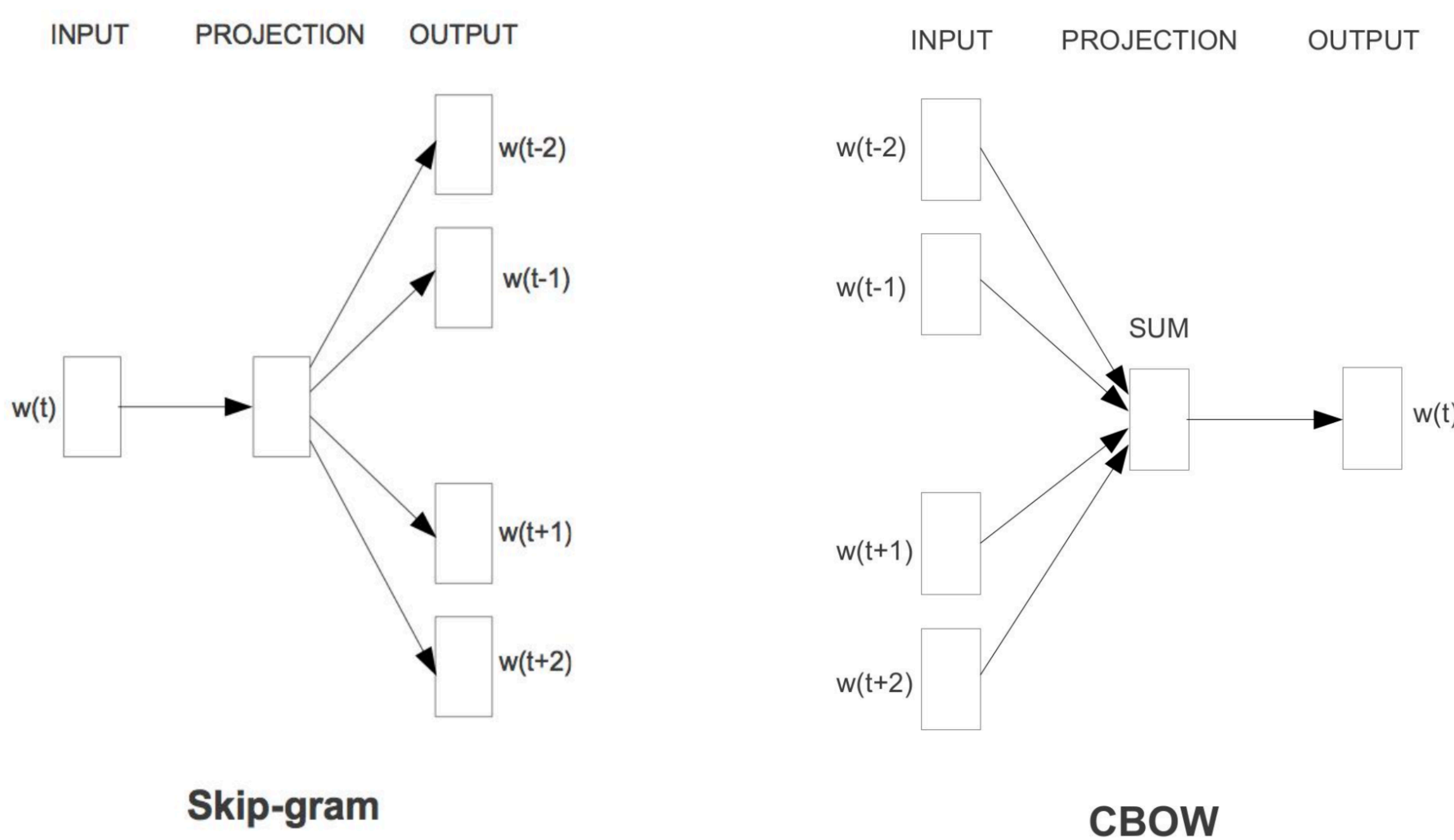
Distributed Word Embeddings

Word Vector Models

- These models are designed to "guess" a word at position i given a word at a position in $\{i-w,\ldots,i-1\} \cup \{i+1,\ldots,i+w\}$
- "Pre-train" word vectors are used in other larger models (e.g., neural LM)

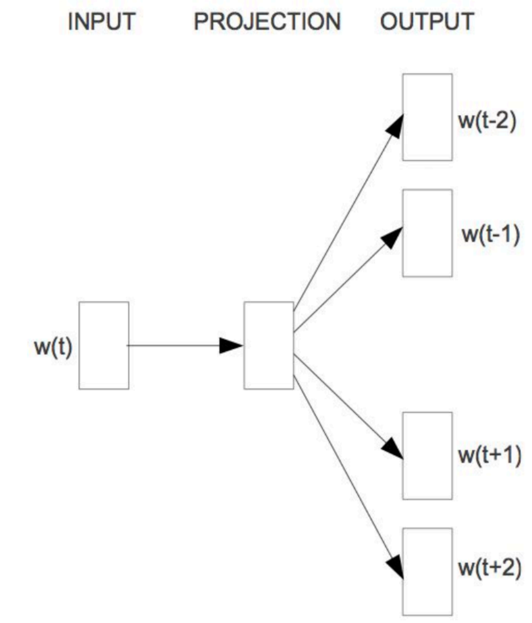
Word2vec

- Continuous bag of words (CBOW): $p(v \mid c)$
  - Similar to feedforward neural LM w/o the feedforward layers in Lecture 3.
- Skip-gram: $p(c \mid v)$



Skip-gram Prediction

- Predict vs Count



Skip-gram

the cat sat on the mat

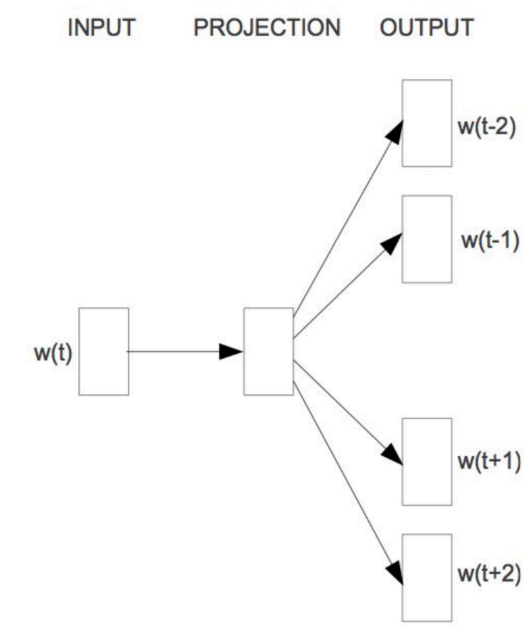
w_t = the → **CLASSIFIER** →

w_{t-2} = <start$_{-2}$>
w_{t-1} = <start$_{-1}$>
w_{t+1} = cat
w_{t+2} = sat

context size = 2

Skip-gram Prediction

- Predict vs Count



Skip-gram

the <u>cat</u> sat on the mat

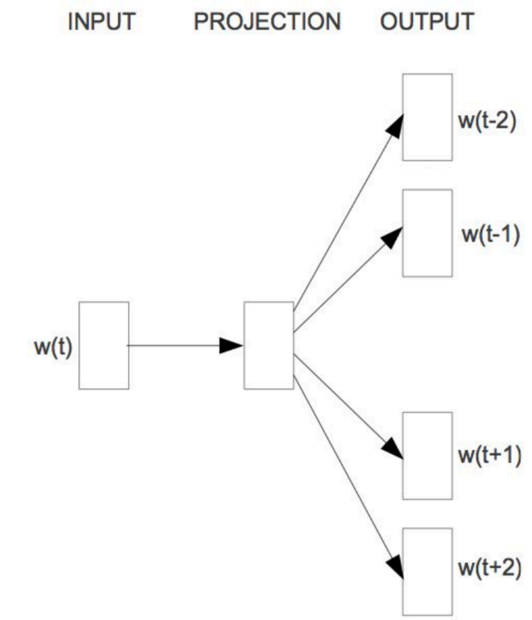
w_t = cat → **CLASSIFIER** →

w_{t-2} = <start$_{-1}$>
w_{t-1} = the
w_{t+1} = sat
w_{t+2} = on

context size = 2

Skip-gram Prediction

- Predict vs Count



Skip-gram

the cat <u>sat</u> on the mat

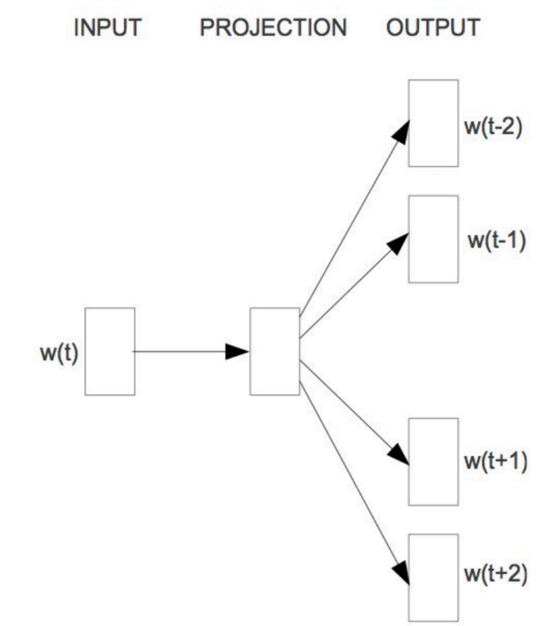
w_t = sat → **CLASSIFIER** →

w_{t-2} = the
w_{t-1} = cat
w_{t+1} = on
w_{t+2} = the

context size = 2

Skip-gram Prediction

- Predict vs Count



Skip-gram

the cat sat on the mat

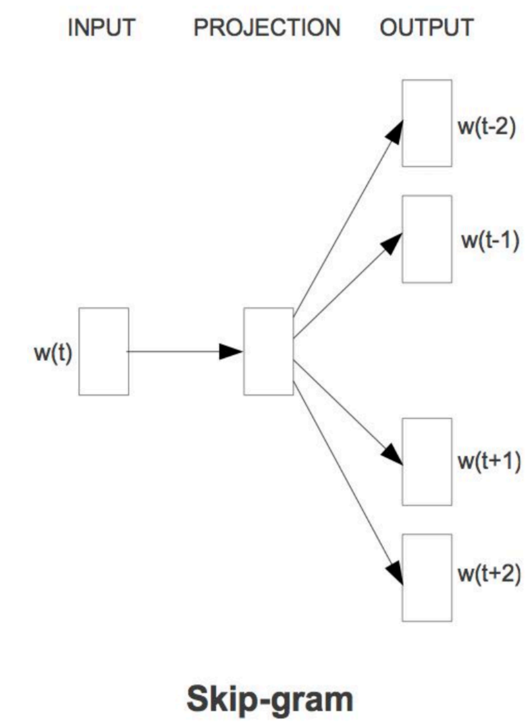
w_t = on → **CLASSIFIER** →

w_{t-2} = cat
w_{t-1} = sat
w_{t+1} = the
w_{t+2} = mat

context size = 2

Skip-gram Prediction

- Predict vs Count



the cat sat on <u>the</u> mat

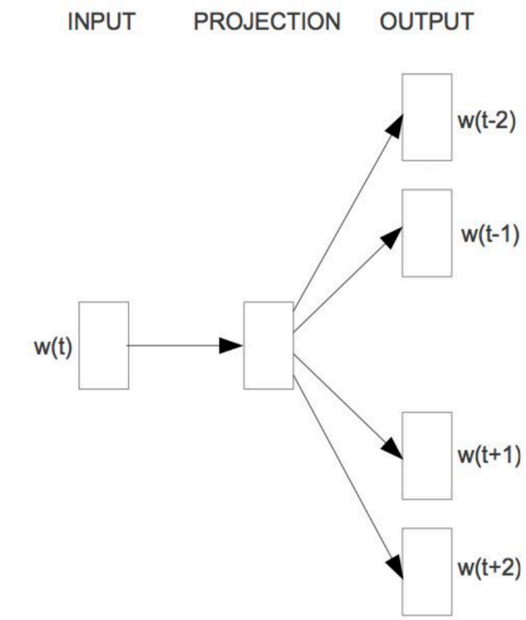
w_t = the → **CLASSIFIER** →

w_{t-2} = sat
w_{t-1} = on
w_{t+1} = mat
w_{t+2} = <end$_{+1}$>

context size = 2

Skip-gram Prediction

- Predict vs Count



Skip-gram

the cat sat on the mat

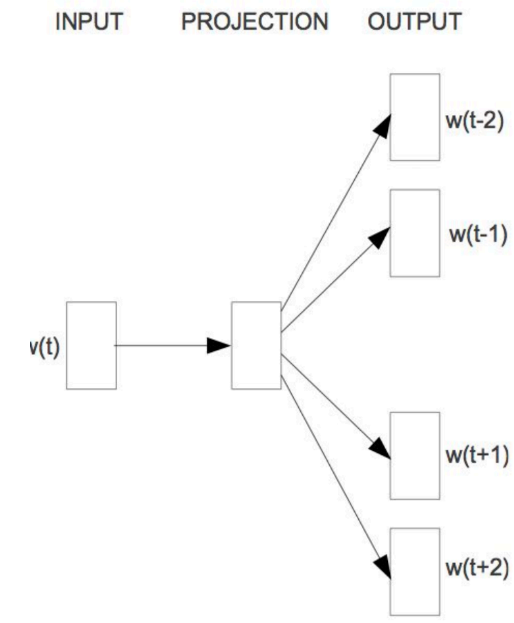
w_t = mat → CLASSIFIER →

w_{t-2} = on
w_{t-1} = the
w_{t+1} = <end$_{+1}$>
w_{t+2} = <end$_{+2}$>

context size = 2

Skip-gram Prediction

- The same word can appear in different context.



Skip-gram

w_t = the → **CLASSIFIER** →

w_{t-2} = sat
w_{t-1} = on
w_{t+1} = mat
$w_{t+2} = \langle \text{end}_{+1} \rangle$

w_t = the → **CLASSIFIER** →

$w_{t-2} = \langle \text{start}_{-2} \rangle$
$w_{t-1} = \langle \text{start}_{-1} \rangle$
w_{t+1} = cat
w_{t+2} = sat

context size = 2

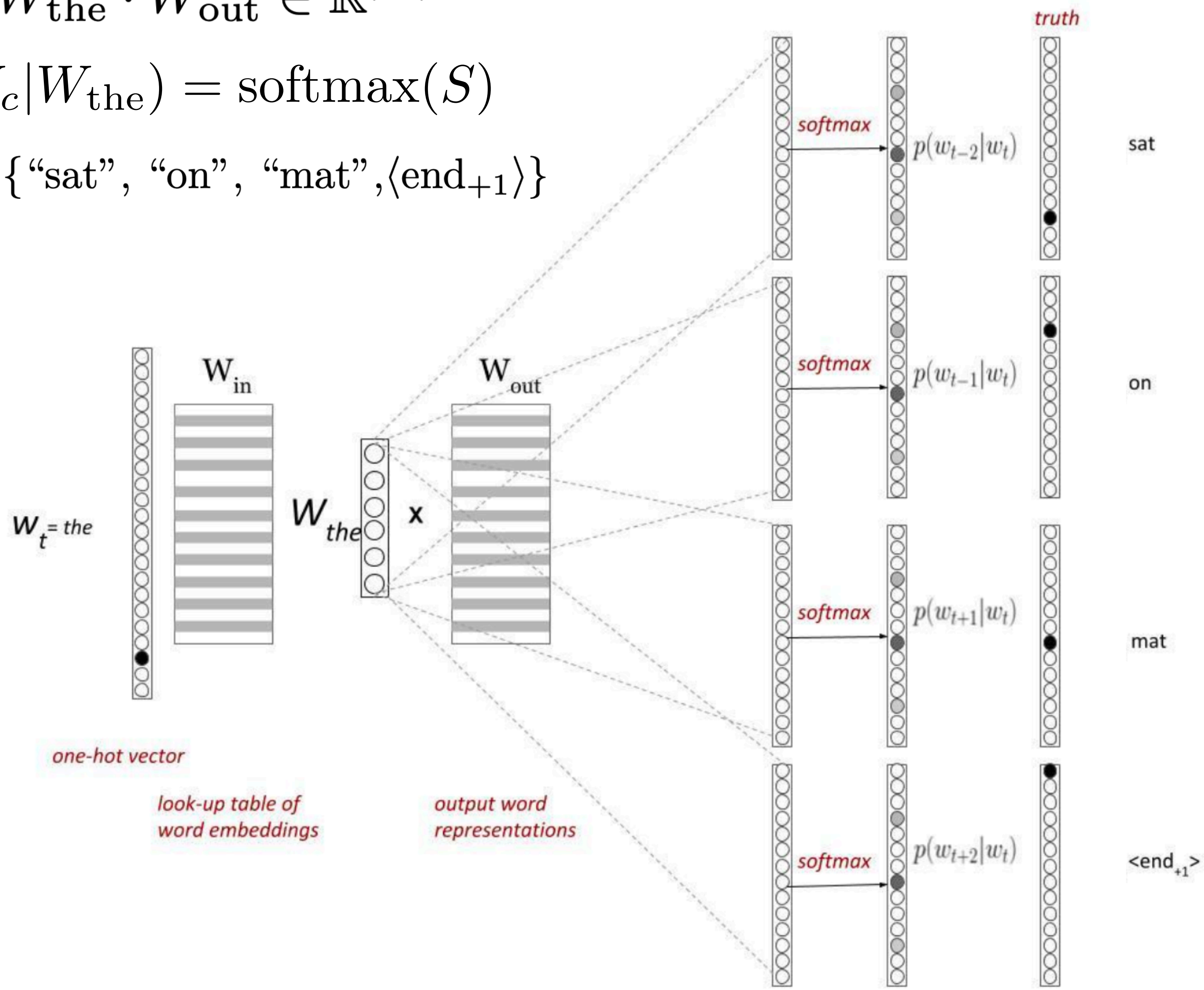
Skip-gram Prediction

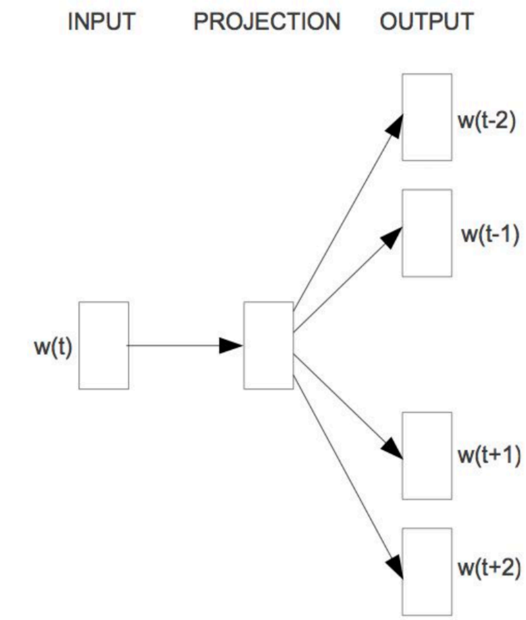
$$W_{\text{the}} = \text{LookUp}(W_{\text{in}}, \text{“the”})) \in \mathbb{R}^d, W_{\text{in}} \in \mathbb{R}^{|V| \times d}$$

$$S = W_{\text{the}} \cdot W_{\text{out}} \in \mathbb{R}^{|V|}$$

$$P(W_c | W_{\text{the}}) = \text{softmax}(S)$$

$$W_c = \{\text{“sat”},\ \text{“on”},\ \text{“mat”}, \langle \text{end}_{+1} \rangle\}$$





Skip-gram

Skip-gram Objective

- For each word in the corpus

$$J(\Theta) = \prod_{t=1}^{T} \prod_{-m \leq j \leq m, j \neq 0} p(w_{t+j} | w_t; \Theta)$$

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

Maximize the probability of any context window given the current center word

Skip-gram Objective

- For each word in the corpus

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

dot product (similarity) between outside and center word vectors

$$p(w_{t+j}|w_t) = p(o|c) = \frac{\exp\boxed{(u_o^\top v_c)}}{\sum_{i=1}^{V}\exp(u_i^\top v_c)}$$

Notation simplification:
o = index of outside (context) word
c = index of center word (w_t)
V = vocab size, V can be large 50K - 30M

Skip-gram w/ negative sampling

- V=50K-30M, too large!

$$p(w_{t+j}|w_t) = p(o|c) = \frac{\exp(u_o^\top v_c)}{\boxed{\sum_{i=1}^{V} \exp(u_i^\top v_c)}}$$

- Negative sampling:
  - Treat the center word and a neighboring context word as positive examples.
  - Randomly sample other words in the lexicon to get negative samples.

(banking, regulation) (banking, aardvark)

Skip-gram w/ negative sampling

- Convert the task to binary classification rather than multiclass:

$$P(o \mid c) = \frac{\exp(u_o^T v_c)}{\sum_{i=1}^{V} \exp(u_i^T v_c)} \longrightarrow P(o \mid c) = \frac{1}{1 + \exp(-u_o^T v_c)} = \sigma(u_o^T v_c)$$

- New objective (single context word, k negative samples):

$$\log P(o_+ \mid c) + \sum_{i=1}^{k} \log(1 - P(o_i \mid c))$$

Choosing negative samples

- Pick negative samples according to unigram frequency *P(w)*

- More common to choose according to:

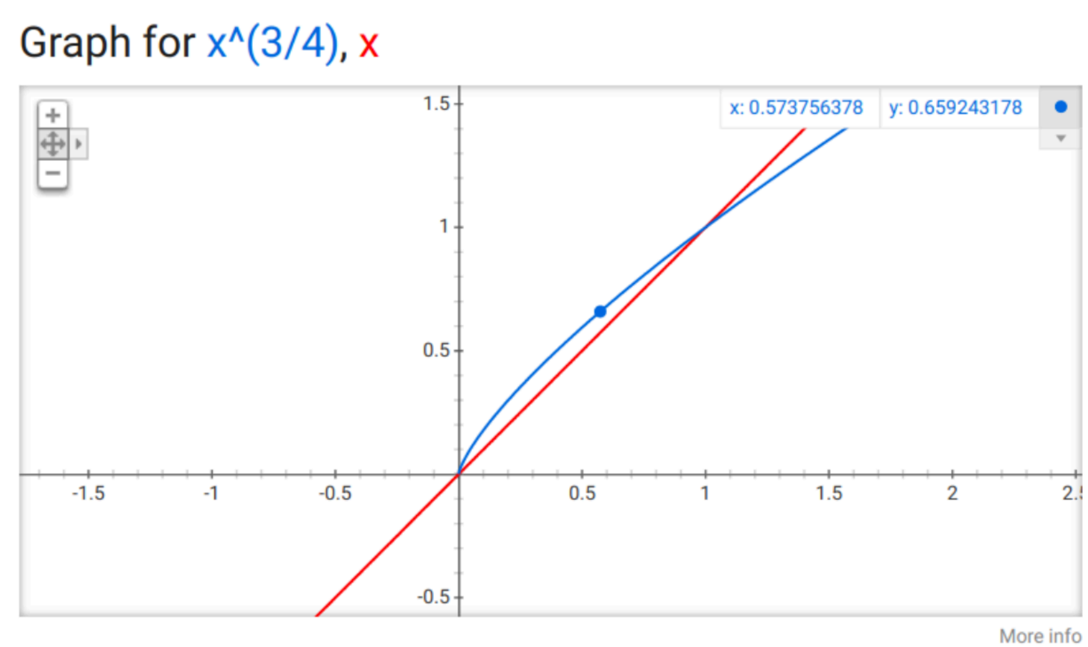
$$P_\alpha(w) = \frac{count(w)^\alpha}{\sum_w count(w)^\alpha}$$

- $\alpha = 0.75$ works well empirically

- Gives rare words slightly higher probability
  - e.g., P(a) = 0.99, P(b) = 0.01

$$P_\alpha(a) = \frac{0.99^{0.75}}{0.99^{0.75} + 0.01^{0.75}} = 0.97$$

$$P_\alpha(b) = \frac{0.01^{0.75}}{0.99^{0.75} + 0.01^{0.75}} = 0.03$$



Available dense embeddings

- Word2vec (Mikolov et a. 2013)
  - https://code.google.com/archive/p/word2vec/
- GloVe (Pennington et al. 2014)
  - http://nlp.stanford.edu/projects/glove/
- Fasttext (Bojanowsi et al. 2017)
  - http://www.fasttext.cc/

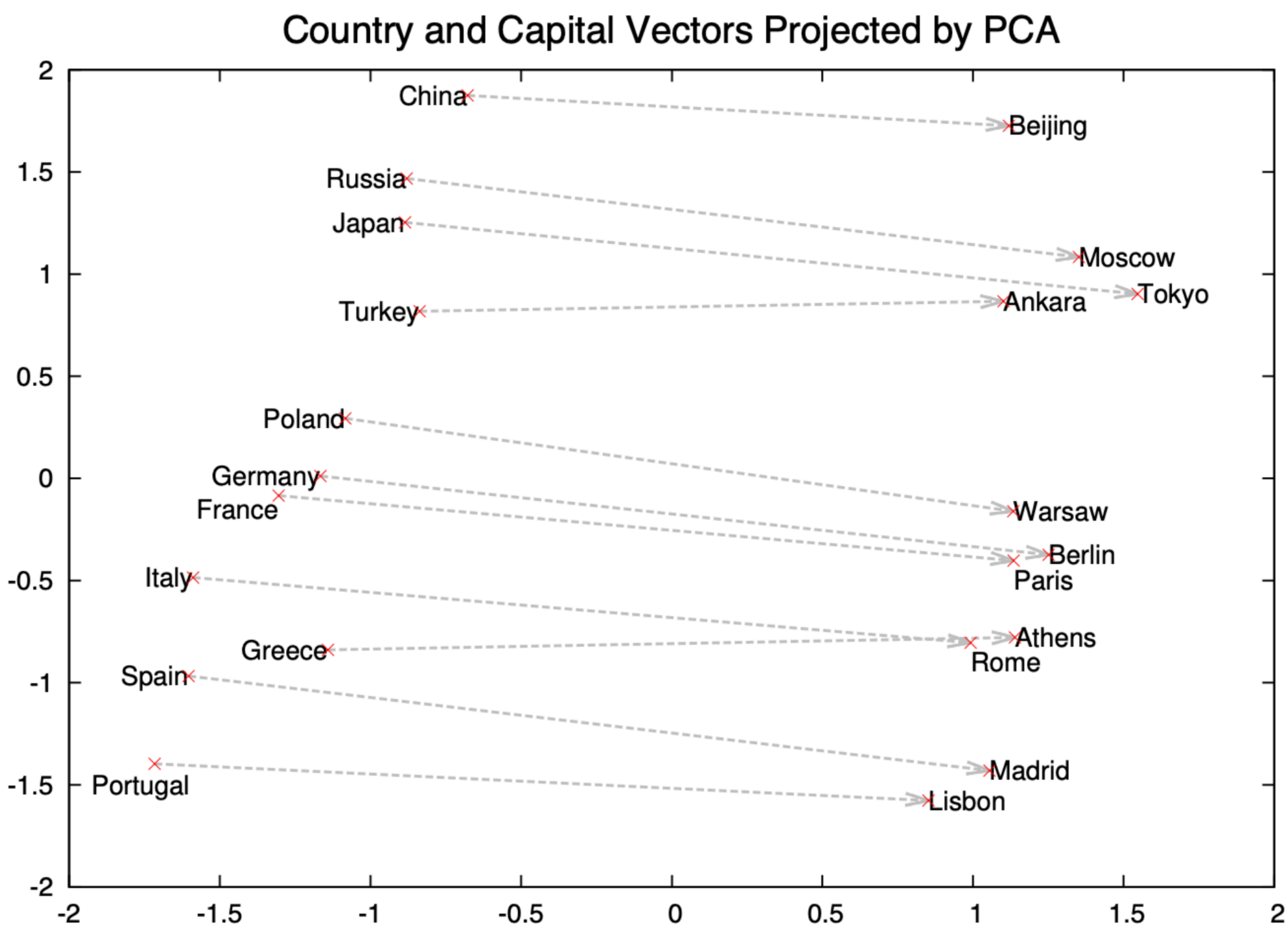
Evaluation

— how well do word vectors capture embedding similarity?

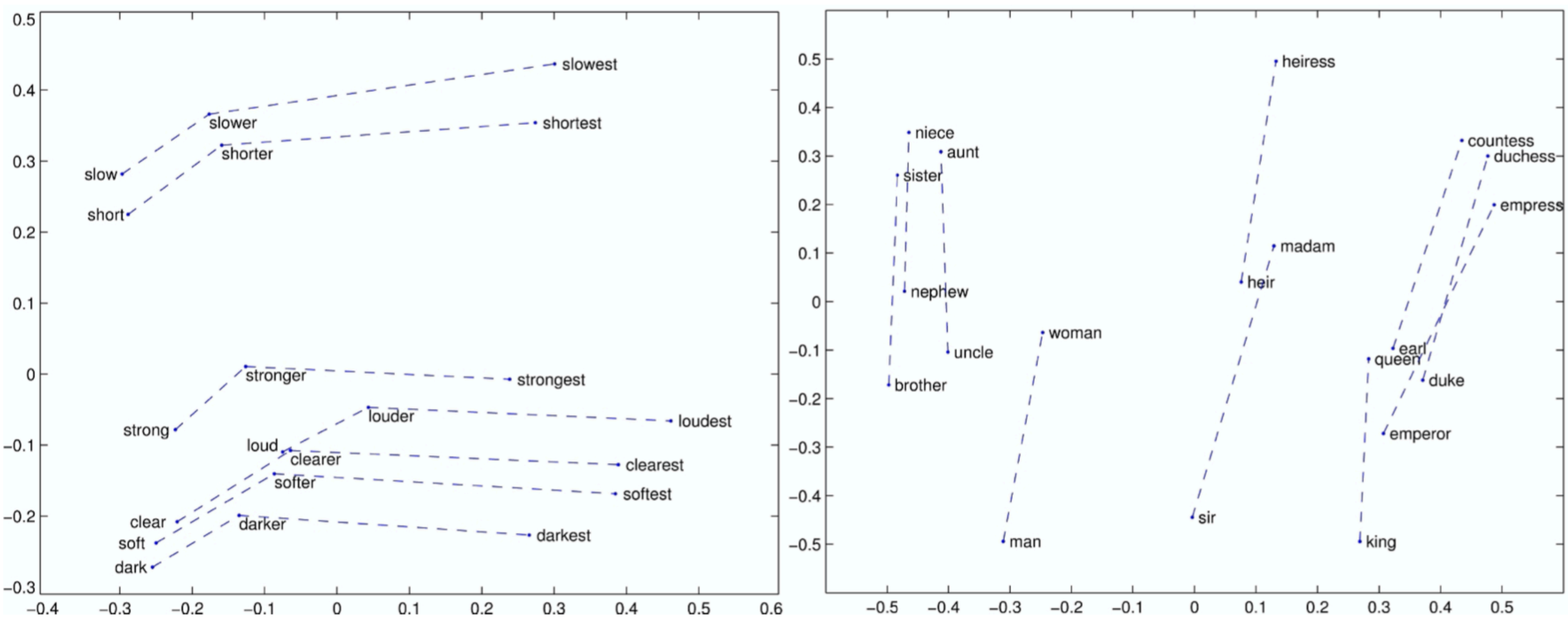
Evaluating word vectors

- **Intrinsic evaluation**: test whether the representations align with our intuitions about word meaning.
    - How well does cosine similarity of word embeddings correlate with human judgements?
    - Completing analogies: a:b <-> c: ?
- **Extrinsic evaluation**: test whether the representations are useful for downtream tasks, such as tagging, parsing, QA, …
    - Provide embeddings as input to the same classifier, how well does a model w/ pre-trained embeddings perform?

A:B <-> C:?

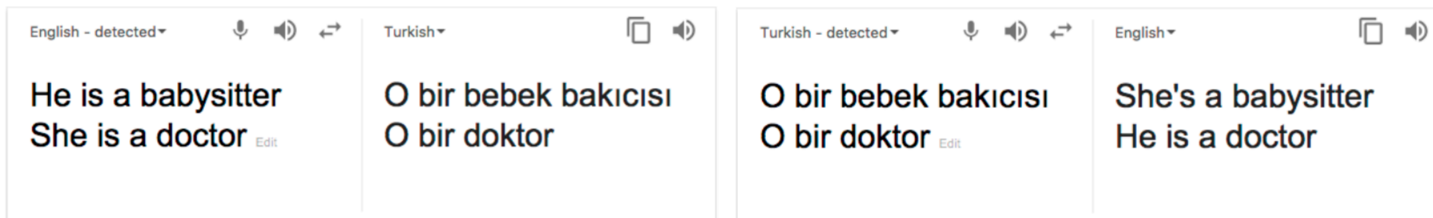

Country and Capital Vectors Projected by PCA
China
Beijing
Russia
Japan
Moscow
Tokyo
Turkey
Ankara
Poland
Germany
France
Warsaw
Berlin
Paris
Italy
Greece
Spain
Athens
Rome
Madrid
Portugal
Lisbon
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

A:B <-> C:?


slowest
slower
shortest
shorter
slow
short
stronger
strongest
strong
louder
loudest
loud
clearer
clearest
softer
softest
clear
soft
darker
darkest
dark
heiress
niece
aunt
sister
nephew
uncle
brother
woman
man
madam
heir
sir
countess
duchess
empress
earl
queen
duke
emperor
king

Other topics

- Bias in word embeddings (gender bias)



- Multilingual word embeddings
- Pre-trained contextualized word embeddings (e.g., Elmo, BERT, Roberta)

Any Questions?