#### CS769 Advanced NLP

# Word Embeddings

Junjie Hu



Slides adapted from Noah, Yulia <a href="https://junjiehu.github.io/cs769-spring22/">https://junjiehu.github.io/cs769-spring22/</a>

# How should we represent the meaning of the word?

## Lexical Semantics

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

lemma sense definition

#### pepper, n.

Pronunciation: Brit. /'pspə/, U.S. /'pspər/

Forms: OE peopor (rare), OE pipcer (transmission error), OE pipor, OF 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.

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. Special uses 7b(a).

**a.** The plant *Piper nigrum* (family Piperaceae), a climbing shrub indigenous to South Asia and also cultivated elsewhere in the tropics, which has alternate starked 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.** V.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: <a href="https://www.oed.com/">https://www.oed.com/</a>

## 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
    - Superordinate/Subordinate: e.g., car/vehicle, mango/ fruit

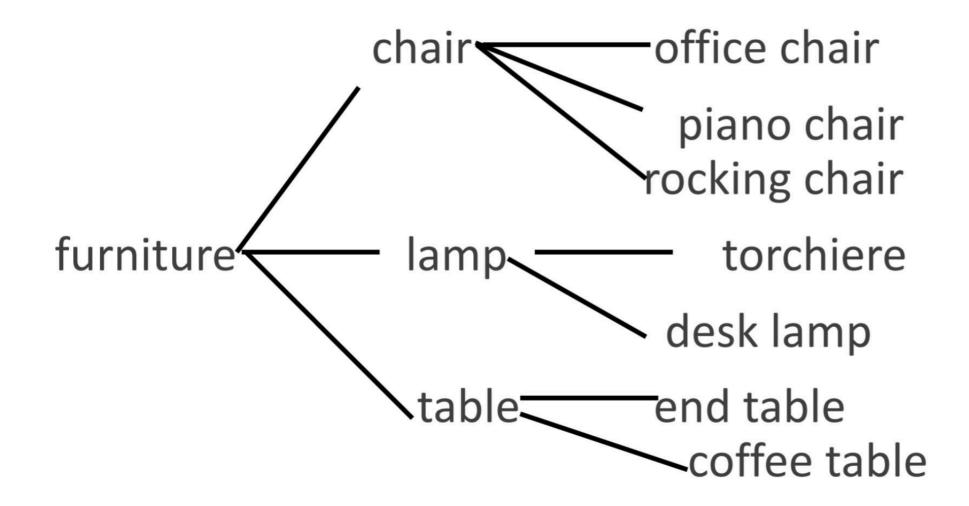
## Lexical Semantics

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

## Taxonomy

abstract -> concrete

Superordinate Basic Subordinate



### 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]
skillful [0 0 0 0 0 0 0 0 0 1 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

```
not good
                                                               bad
       by
                                                    dislike
to
                                                                    worst
                                                    incredibly bad
that
        now
                       are
                                                                      worse
                 vou
 than
          with
                                           incredibly good
                              very good
                      amazing
                                          fantastic
                                                    wonderful
                  terrific
                                       nice
                                      good
```

#### Approaches for encoding words as vectors

- Counting-based methods (e.g., Tf-idf)
- Matrix factorization (e.g., topic modeling)
- Brown clusters
- Word2vec

## Count-based Model

#### Count-based Method: Topic Models

- Words are not independent, identically distributed (IID)!
  - Predictable given history: n-gram/Markov models
  - Predictable given other words in the document: topic models
- Let  $\mathcal{Z} = \{1, \dots, K\}$  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

- x is the corpus
- $\mathbf{x}_c$  is the c-th document in the corpus
- $\ell_c$  is the length of  $\mathbf{x}_c$  (in tokens)
- N is the total count of tokens in the corpus,

$$N = \sum_{c=1}^{C} \ell_c$$

### Word-Document Matrix

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

```
oldsymbol{x}_1: yes , we have no bananas
```

 $oldsymbol{x}_2$ : say yes for bananas

 $oldsymbol{x}_3$ : no bananas , we say

1	2	3	
1	0	1	
1	1	1	
0	1	0	
1	0	0	
1	0	1	
0	1	1	
1	0	1	
1	1	0	
	1 1 0 1 1 0 1	1 0 1 1 0 1 1 0 1 0 0 1 1 0	1     0     1       1     1     1       0     1     0       1     0     0       1     0     1       0     1     1       1     0     1       1     0     1

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

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

Note: **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

$$tf_{v,c} = \log(count(v,c) + 1)$$

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

$$\mathrm{idf}_v = \log\left(\frac{|N|}{|\{c|v\in c,\ \forall c\in C\}|}\right) \quad \begin{array}{l} \text{Higher for words} \\ \text{that occur in} \\ \text{fewer documents} \end{array}$$

where N is the no. of documents.

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

# 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

	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	

# Matrix Factorization Based Method

#### Association Score

- Change (under a unigram model):  $\frac{\operatorname{count}_{\boldsymbol{x}}(v)}{N}$  words out of the  $\ell_c$  words in document c are the word v
- As document c may consist of different topics, is the occurrence of word v in c surprisingly high (or low), comparing to chance?
- Intuition: consider the ratio of observed frequency  $(\operatorname{count}_{\boldsymbol{x}_c}(v))$  to "chance"  $(\frac{\operatorname{count}_{\boldsymbol{x}}(v)}{N}\cdot\ell_c)$

#### Pointwise Mutual Information

 A common measurement is to define A as positive pointwise mutual information:

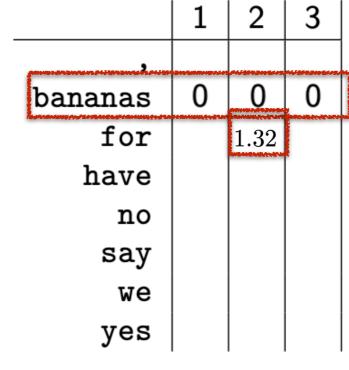
$$[\mathbf{A}]_{v,c} = \left[\log \frac{\operatorname{count}_{\boldsymbol{x}_c}(v)}{\frac{\operatorname{count}_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_{+} = \left[\log \frac{N \cdot \operatorname{count}_{\boldsymbol{x}_c}(v)}{\operatorname{count}_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_{+}$$

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

$$\begin{split} [\mathbf{A}]_{\mathsf{bananas},1} &= \log \frac{15 \cdot 1}{3 \cdot 6} \approx -0.18 \rightarrow 0 \\ [\mathbf{A}]_{\mathsf{for},2} &= \log \frac{15 \cdot 1}{1 \cdot 4} \approx 1.32 \end{split}$$

	1	2	3
amerananan eritaran an aristera	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

A: count matrix



**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:

$$\mathsf{PMI}(a,b) = \log \frac{p(A=a,B=b)}{p(A=a) \cdot p(B=b)}$$
$$= \log \frac{p(A=a \mid B=b)}{p(A=a)}$$
$$= \log \frac{p(B=b \mid A=a)}{p(B=b)}$$

All possible events: average mutual information

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

- 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  $[{\bf A}]_{v,*}$  is nearly 0.
- If a word v appears only in doc c, their PMI  $([{f A}]_{v,c})$  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	2	3	
em-es-es-es-es-es-es-es-es-es-es-es-es-es-	1	0	1		300 lahili 11 lahili 12 M	_~~2 <b>50</b> 200 <b>2</b> 02	M/8603409443	
bananas	1	1	1	bananas	0	0	0	
for	0	1	0	for		1.32		
have	1	0	0	have				
no	1	0	1	no				
say	0	1	1	say				
we	1	0	1	we				
yes	1	1	0	yes				

A: count matrix

**A**: PMI

### Reflection

- Can we use this association matrix A as word embedding matrix 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.

#### Topic Models: Latent Semantic Indexing/Analysis

Deerwester et al., 1990, LSA

LSA or LSI seeks to solve:

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

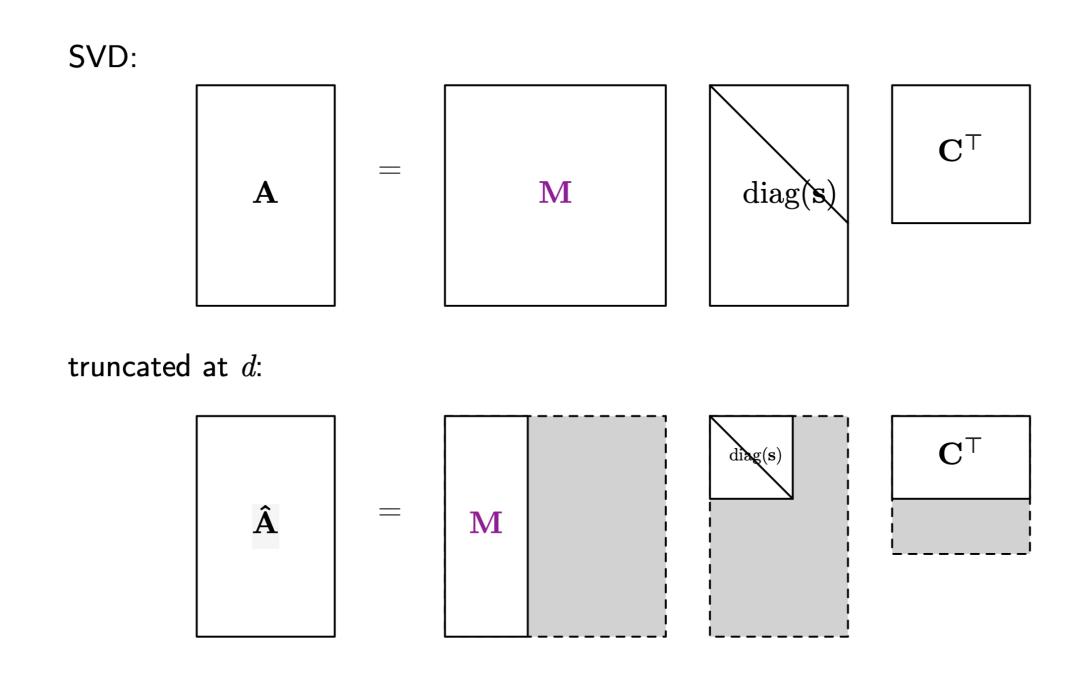
where  $\mathbf{M}$  is the word embedding matrix,  $\mathbf{C}$  is the document embedding matrix.

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

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

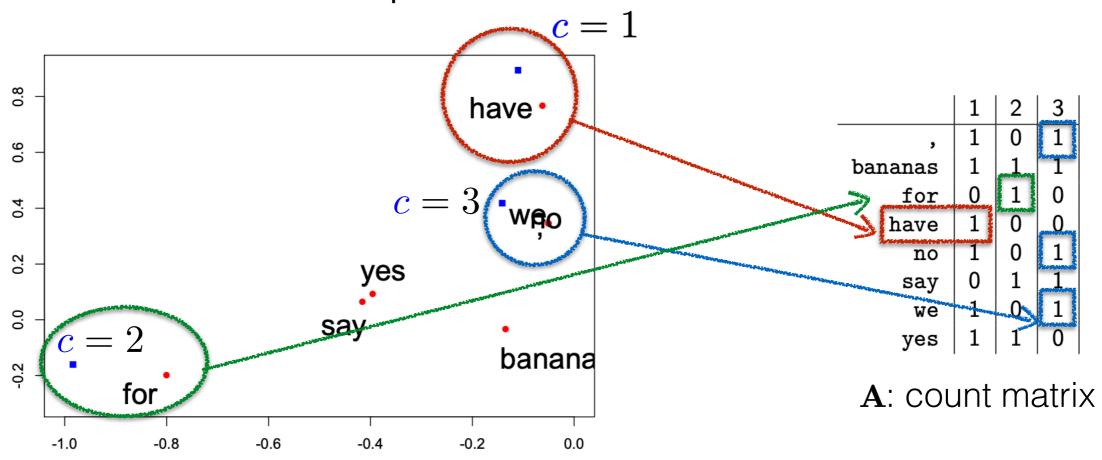
#### Truncated Singular Value Decomposition

• 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.



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.
- 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 A.
- More variants of LSA
  - Probabilistic Latent Semantic Indexing (PLSI)
  - Latent Dirichlet Allocation (LDA)
  - Nonnegative Matrix Factorization (NMF)

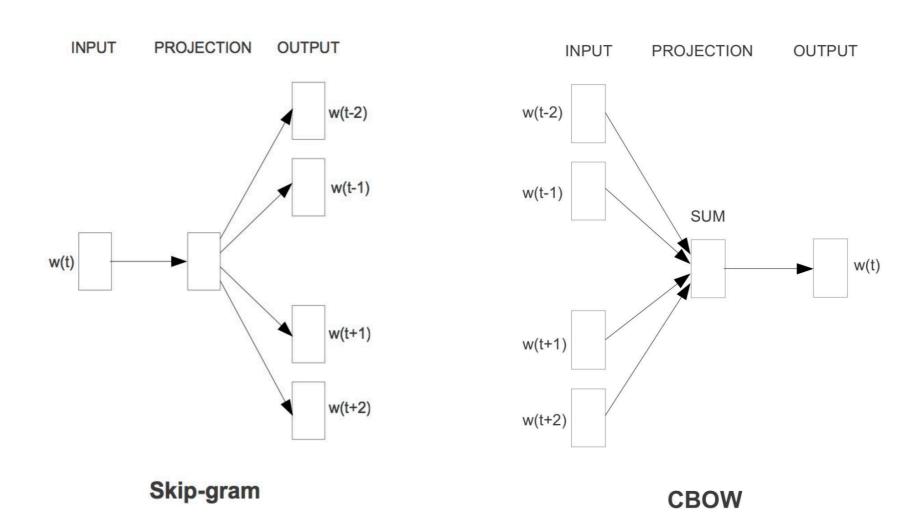
# Distributed Word Embeddings

#### 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

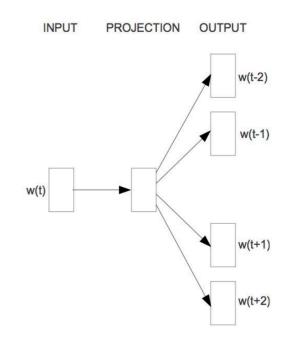
- Continuous bag of words (CBOW):  $p(v \mid c)$ 
  - Similar to feedforward neural LM w/o the feedforward layers in Lecture 3.
- Skip-gram:  $p(c \mid v)$



36

Predict vs Count

the cat sat on the mat

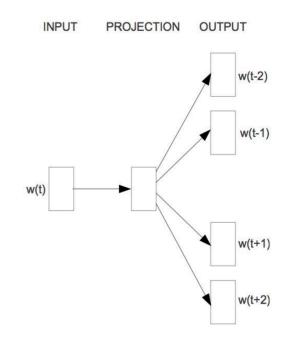


Skip-gram

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

Predict vs Count

the cat sat on the mat

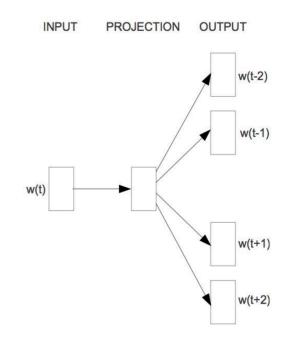


Skip-gram

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

Predict vs Count

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

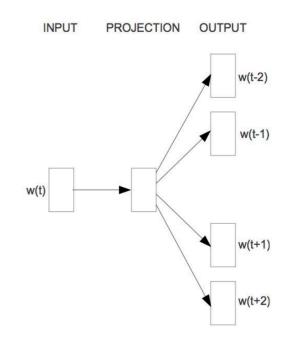


Skip-gram

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

Predict vs Count

the cat sat on the mat

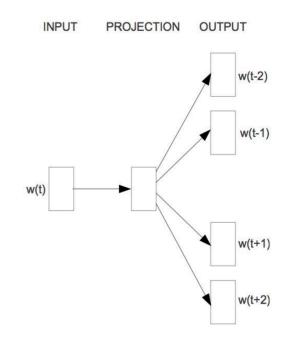


Skip-gram

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

Predict vs Count

the cat sat on the mat

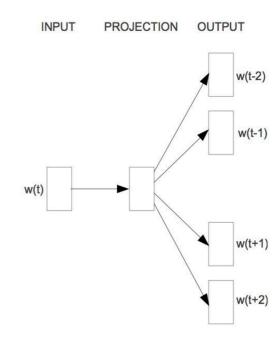


Skip-gram

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

Predict vs Count

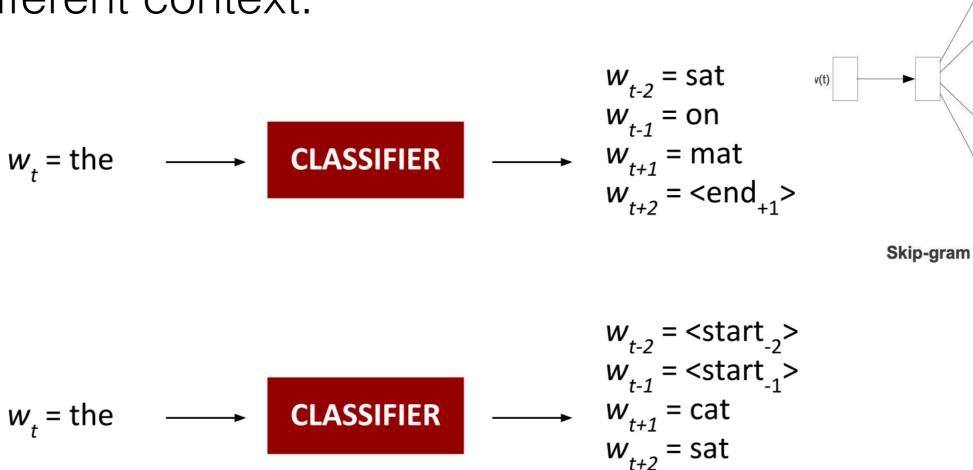




Skip-gram

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

 The same word can appear in different context.



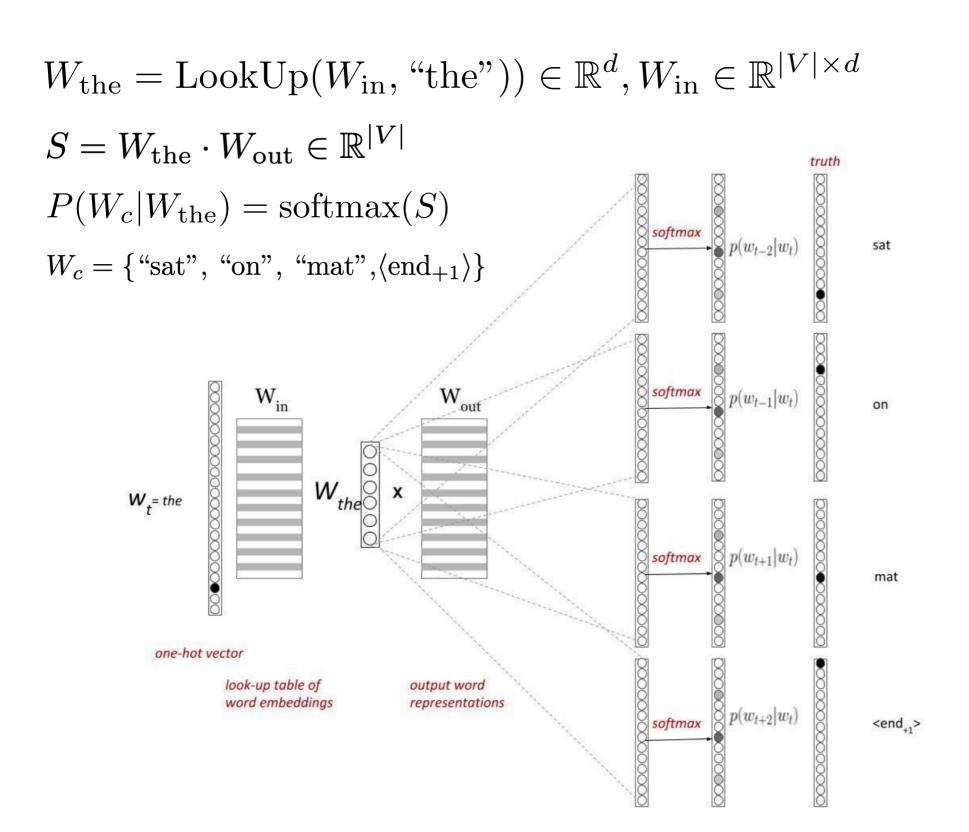
context size = 2

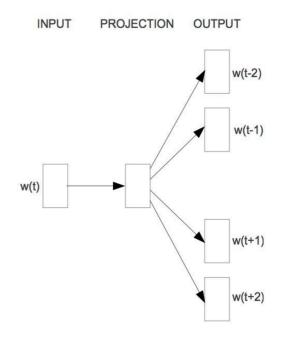
**PROJECTION** 

w(t-2)

w(t+1)

w(t+2)





Skip-gram

## Skip-gram Objective

For each word in the corpus

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

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

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

## Skip-gram Objective

For each word in the corpus

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

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

dot product
(similarity)
between outside
and center word
vectors

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!

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

- 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

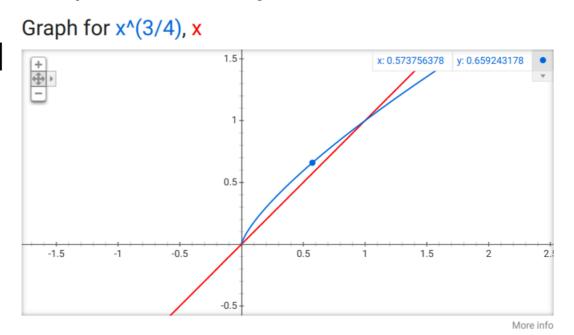
- Pick negative samples according to unigram frequency P(w)
- More common to choose according to:

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

- $\alpha = 0.75$  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/

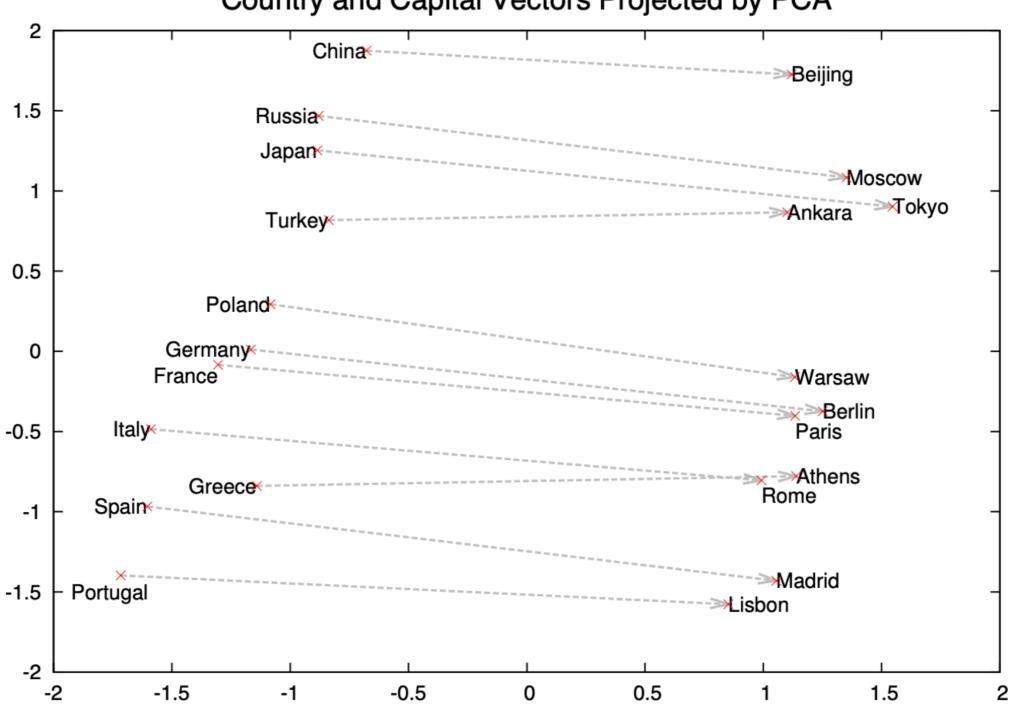
## Evaluating word vectors

- 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: ?

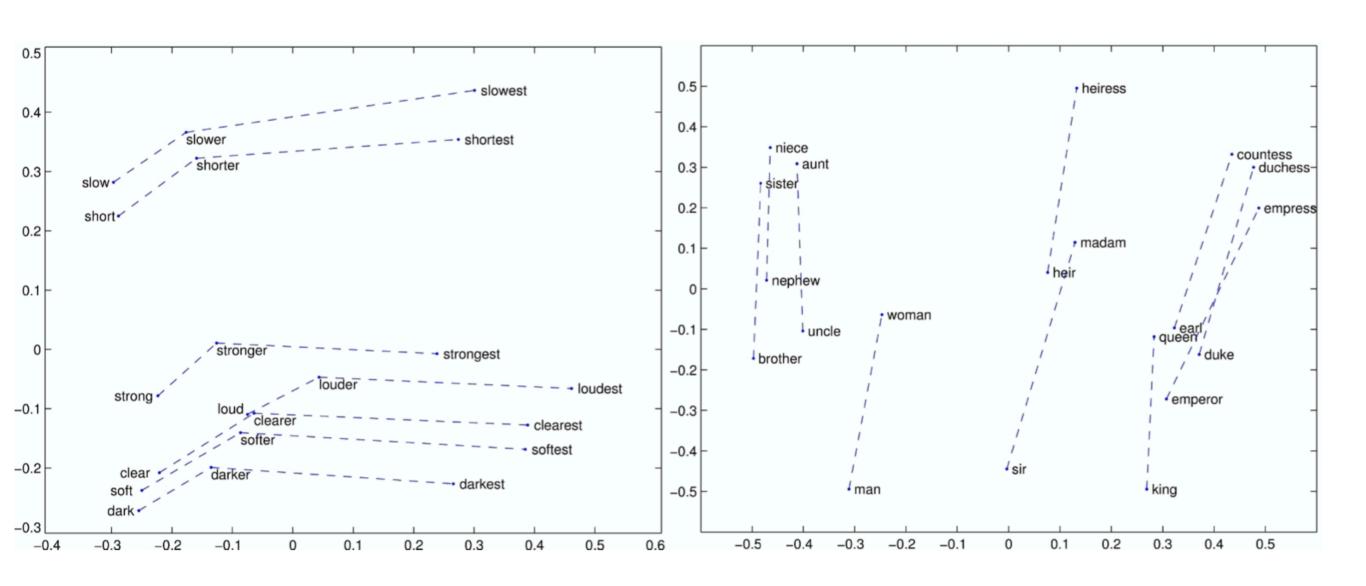
- Extrinsic evaluation: test whether the representations are useful for downtream tasks, such as tagging, parsing, QA, ...
  - Provide embeddings as input to the same classifier, how well does a model w/ pre-trained embeddings perform?

#### A:B <-> C:?

#### Country and Capital Vectors Projected by PCA

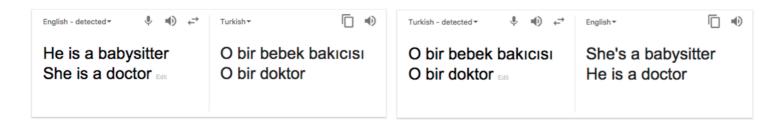


### A:B <-> C:?



## Other topics

Bias in word embeddings (gender bias)



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

# Any Questions?