Natural Language Processing
Lexical semantics

Sofia Serrano
sofias6@cs.washington.edu

Credit to Yulia Tsvetkov and Noah Smith for slides

Announcements

- Quiz 3 will be released on Canvas today at 2:20pm
  - Available through Thursday 2:20pm
  - 5 questions, 10 minutes
  - Will cover material from Wednesday, Friday, and Monday (so, language modeling and the first part of lexical semantics)
- Midterm course eval form (online) is out– please let us know how we're doing!

Two common solutions for word weighting

tf-idf: tf-idf value for word t in document d:

$$w_{t,d} = \mathrm{tf}_{t,d} \times \mathrm{idf}_t$$

Words like "the" or "it" have very low idf

PMI: Pointwise mutual information

$$\mathrm{PMI}(w_1, w_2) = log\frac{p(w_1,w_2)}{p(w_1)p(w_2)}$$

See if words like "good" appear more often with "great" than we would expect by chance

TF-IDF

- What to do with words that are evenly distributed across many documents?

$$\mathrm{tf}_{t,d} = \log_{10}(\mathrm{count}(t,d)+1)$$

Total # of docs in collection

$$\mathrm{idf}_i = \log\left(\frac{N}{\mathrm{df}_i}\right)$$

# of docs that have word i

Words like "the" or "good" have very low idf

$$w_{t,d} = \mathrm{tf}_{t,d} \times \mathrm{idf}_t$$

Positive Pointwise Mutual Information (PPMI)

- In word--context matrix
- Do words w and c co-occur more than if they were independent?

$$\text{PMI}(w,c) = \log_2 \frac{P(w,c)}{P(w)P(c)}$$

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

- PMI is biased toward infrequent events
  - Very rare words have very high PMI values
  - Give rare words slightly higher probabilities α=0.75

$$\text{PPMI}_\alpha(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P_\alpha(c)}, 0) \qquad P_\alpha(c) = \frac{count(c)^\alpha}{\sum_c count(c)^\alpha}$$

# name	formula	reference
1. **Joint probability**	$p(xy)$	(Giuliano, 1964)
2. **Conditional probability**	$p(y \mid x)$	(Gregory et al., 1999)
3. **Reverse cond. probability**	$p(x \mid y)$	(Gregory et al., 1999)
4. **Pointwise mutual inf. (*MI*)**	$\log \frac{p(xy)}{p(x\ast)p(\ast y)}$	(Church and Hanks, 1990)
5. **Mutual dependency (*MD*)**	$\log \frac{p(xy)^2}{p(x\ast)p(\ast y)}$	(Thanopoulos et al., 2002)
6. **Log frequency biased *MD***	$\log \frac{p(xy)^2}{p(x\ast)p(\ast y)} + \log p(xy)$	(Thanopoulos et al., 2002)
7. **Normalized expectation**	$\frac{2f(xy)}{f(x\ast)+f(\ast y)}$	(Smadja and McKeown, 1990)
8. **Mutual expectation**	$\frac{2f(xy)}{f(x\ast)+f(\ast y)} \cdot p(xy)$	(Dias et al., 2000)
9. **Salience**	$\log \frac{p(xy)^2}{p(x\ast)p(\ast y)} \cdot \log f(xy)$	(Kilgarriff and Tugwell, 2001)
10. **Pearson's χ^2 test**	$\sum_{i,j} \frac{(f_{ij}-\hat{f}_{ij})^2}{\hat{f}_{ij}}$	(Manning and Schütze, 1999)
11. **Fisher's exact test**	$\frac{f(x\ast)!f(\bar{x}\ast)!f(\ast y)!f(\ast\bar{y})!}{N!f(xy)!f(x\bar{y})!f(\bar{x}y)!f(\bar{x}\bar{y})!}$	(Pedersen, 1996)
12. **t test**	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{f(xy)(1-(f(xy)/N))}}$	(Church and Hanks, 1990)
13. **z score**	$\frac{f(xy)-\hat{f}(xy)}{\sqrt{\hat{f}(xy)(1-(\hat{f}(xy)/N))}}$	(Berry-Rogghe, 1973)
14. **Poisson significance**	$\frac{\hat{f}(xy)-f(xy)\log \hat{f}(xy)+\log f(xy)!}{\log N}$	(Quasthoff and Wolff, 2002)
15. **Log likelihood ratio**	$-2\sum_{i,j} f_{ij} \log \frac{f_{ij}}{\hat{f}_{ij}}$	(Dunning, 1993)
16. **Squared log likelihood ratio**	$-2\sum_{i,j} \frac{\log f_{ij}^2}{\hat{f}_{ij}}$	(Inkpen and Hirst, 2002)
17. **Russel-Rao**	$\frac{a}{a+b+c+d}$	(Russel and Rao, 1940)
18. **Sokal-Michiner**	$\frac{a+d}{a+b+c+d}$	(Sokal and Michener, 1958)
19. **Rogers-Tanimoto**	$\frac{a+d}{a+2b+2c+d}$	(Rogers and Tanimoto, 1960)
20. **Hamann**	$\frac{(a+d)-(b+c)}{a+b+c+d}$	(Hamann, 1961)
21. **Third Sokal-Sneath**	$\frac{b+c}{a+d}$	(Sokal and Sneath, 1963)
22. **Jaccard**	$\frac{a}{a+b+c}$	(Jaccard, 1912)
23. **First Kulczynsky**	$\frac{a}{b+c}$	(Kulczynski, 1927)
24. **Second Sokal-Sneath**	$\frac{a}{a+2(b+c)}$	(Sokal and Sneath, 1963)
25. **Second Kulczynski**	$\frac{1}{2}(\frac{a}{a+b}+\frac{a}{a+c})$	(Kulczynski, 1927)
26. **Fourth Sokal-Sneath**	$\frac{1}{4}(\frac{a}{a+b}+\frac{a}{a+c}+\frac{d}{d+b}+\frac{d}{d+c})$	(Kulczynski, 1927)
27. **Odds ratio**	$\frac{ad}{bc}$	(Tan et al., 2002)
28. **Yulle's ω**	$\frac{\sqrt{ad}-\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}$	(Tan et al., 2002)
29. **Yulle's Q**	$\frac{ad-bc}{ad+bc}$	(Tan et al., 2002)
30. **Driver-Kroeber**	$\frac{a}{\sqrt{(a+b)(a+c)}}$	(Driver and Kroeber, 1932)

# name	formula	reference
31. **Fifth Sokal-Sneath**	$\frac{ad}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$	(Sokal and Sneath, 1963)
32. **Pearson**	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$	(Pearson,1950)
33. **Baroni-Urbani**	$\frac{a+\sqrt{ad}}{a+b+c+\sqrt{ad}}$	(Baroni-Urbani and Buser, 1976)
34. **Braun-Blanquet**	$\frac{a}{\max(a+b,a+c)}$	(Braun-Blanquet, 1932)
35. **Simpson**	$\frac{a}{\min(a+b,a+c)}$	(Simpson, 1943)
36. **Michael**	$\frac{4(ad-bc)}{(a+d)^2+(b+c)^2}$	(Michael, 1920)
37. **Mountford**	$\frac{2a}{2bc+ab+ac}$	(Kaufman and Rousseeuw, 1990)
38. **Fager**	$\frac{a}{\sqrt{(a+b)(a+c)}} - \frac{1}{2}\max(b,c)$	(Kaufman and Rousseeuw, 1990)
39. **Unigram subtuples**	$\log \frac{ad}{bc} - 3.29\sqrt{\frac{1}{a}+\frac{1}{b}+\frac{1}{c}+\frac{1}{d}}$	(Blaheta and Johnson, 2001)
40. ***U* cost**	$\log(1+\frac{\min(b,c)+a}{\max(b,c)+a})$	(Tulloss, 1997)
41. ***S* cost**	$\log(1+\frac{\min(b,c)}{a+1})^{-\frac{1}{2}}$	(Tulloss, 1997)
42. ***R* cost**	$\log(1+\frac{a}{a+b}) \cdot \log(1+\frac{a}{a+c})$	(Tulloss, 1997)
43. ***T* combined cost**	$\sqrt{U \times S \times R}$	(Tulloss, 1997)
44. **Phi**	$\frac{p(xy)-p(x\ast)p(\ast y)}{\sqrt{p(x\ast)p(\ast y)(1-p(x\ast))(1-p(\ast y))}}$	(Tan et al., 2002)
45. **Kappa**	$\frac{p(xy)+p(\bar{x}\bar{y})-p(x\ast)p(\ast y)-p(\bar{x}\ast)p(\ast\bar{y})}{1-p(x\ast)p(\ast y)-p(\bar{x}\ast)p(\ast\bar{y})}$	(Tan et al., 2002)
46. ***J* measure**	$\max[p(xy)\log\frac{p(y \mid x)}{p(\ast y)}+p(x\bar{y})\log\frac{p(\bar{y} \mid x)}{p(\ast\bar{y})},\ p(xy)\log\frac{p(x \mid y)}{p(x\ast)}+p(\bar{x}y)\log\frac{p(\bar{x} \mid y)}{p(\bar{x}\ast)}]$	(Tan et al., 2002)
47. **Gini index**	$\max[p(x\ast)(p(y \mid x)^2+p(\bar{y} \mid x)^2)-p(\ast y)^2 + p(\bar{x}\ast)(p(y \mid \bar{x})^2+p(\bar{y} \mid \bar{x})^2)-p(\ast\bar{y})^2,\ p(\ast y)(p(x \mid y)^2+p(\bar{x} \mid y)^2)-p(x\ast)^2 + p(\ast\bar{y})(p(x \mid \bar{y})^2+p(\bar{x} \mid \bar{y})^2)-p(\bar{x}\ast)^2]$	(Tan et al., 2002)
48. **Confidence**	$\max[p(y \mid x), p(x \mid y)]$	(Tan et al., 2002)
49. **Laplace**	$\max[\frac{Np(xy)+1}{Np(x\ast)+2}, \frac{Np(xy)+1}{Np(\ast y)+2}]$	(Tan et al., 2002)
50. **Conviction**	$\max[\frac{p(x\ast)p(\ast y)}{p(x\bar{y})}, \frac{p(\bar{x}\ast)p(\ast y)}{p(\bar{x}y)}]$	(Tan et al., 2002)
51. **Piatersky-Shapiro**	$p(xy)-p(x\ast)p(\ast y)$	(Tan et al., 2002)
52. **Certainity factor**	$\max[\frac{p(y \mid x)-p(\ast y)}{1-p(\ast y)}, \frac{p(x \mid y)-p(x\ast)}{1-p(x\ast)}]$	(Tan et al., 2002)
53. **Added value (*AV*)**	$\max[p(y \mid x)-p(\ast y), p(x \mid y)-p(x\ast)]$	(Tan et al., 2002)
54. **Collective strength**	$\frac{p(xy)+p(\bar{x}\bar{y})}{p(x\ast)p(y)+p(\bar{x}\ast)p(\ast y)} \cdot \frac{1-p(x\ast)p(\ast y)-p(\bar{x}\ast)p(\ast\bar{y})}{1-p(xy)-p(\bar{x}\bar{y})}$	(Tan et al., 2002)
55. **Klosgen**	$\sqrt{p(xy)} \cdot AV$	(Tan et al., 2002)

(Pecina '09)

Dense vectors (part 1)

Term-document matrix from Monday

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	17
soldier	2	80	62	89
fool	36	58	1	4
clown	20	15	2	3

These word vectors are still the length of our number of documents! Hmmm...

Dimensionality Reduction

- Wikipedia: ~29 million English documents. Vocab: ~1M words.
  - High dimensionality of word--document matrix
    - Sparsity
    - The order of rows and columns doesn’t matter
- Goal:
  - good similarity measure for words or documents
  - dense representation
- Sparse vs Dense vectors
  - Short vectors may be easier to use as features in machine learning (less weights to tune)
  - Dense vectors may generalize better than storing explicit counts
    - They may do better at capturing synonymy
    - In practice, they work better

A	0
a	0
aa	0
aal	0
aalii	0
aam	0
Aani	0
aardvark	**1**
aardwolf	0
...	0
zymotoxic	0
zymurgy	0
Zyrenian	0
Zyrian	0
Zyryan	0
zythem	0
Zythia	0
zythum	0
Zyzomys	0
Zyzzogeton	0

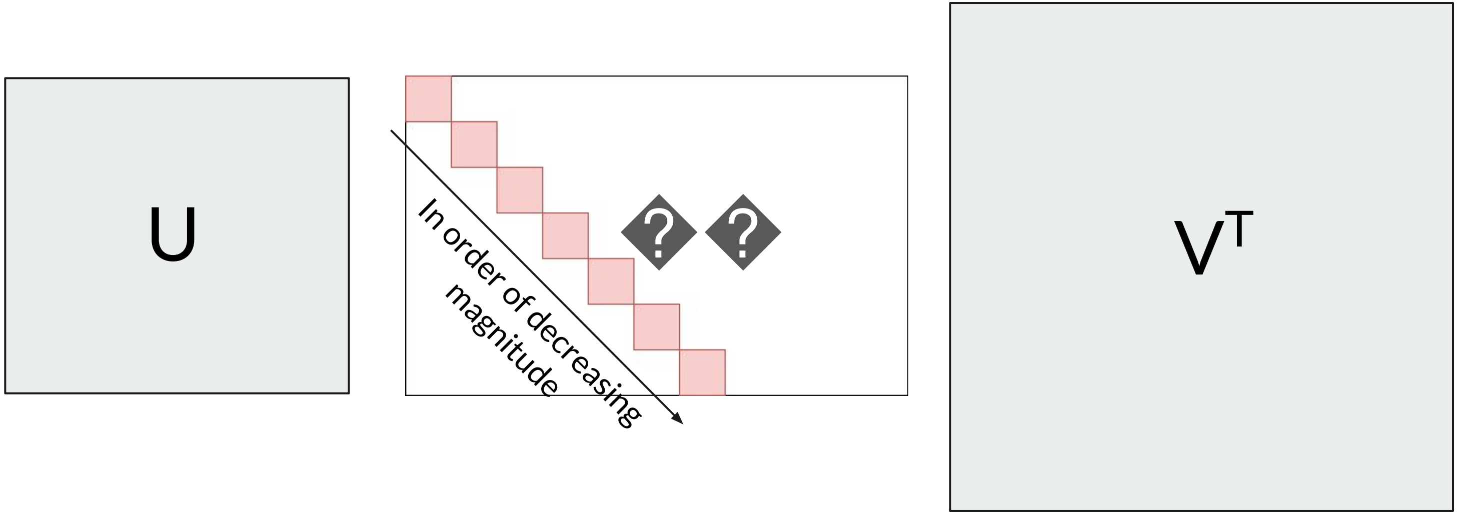
Solution idea

- Find a projection into a low-dimensional space (~300 dim)...
- … that, up to a certain vector-length budget, preserves the most important information

We turn to Singular Value Decomposition (SVD)

Singular Value Decomposition in a nutshell

Any matrix can be decomposed into



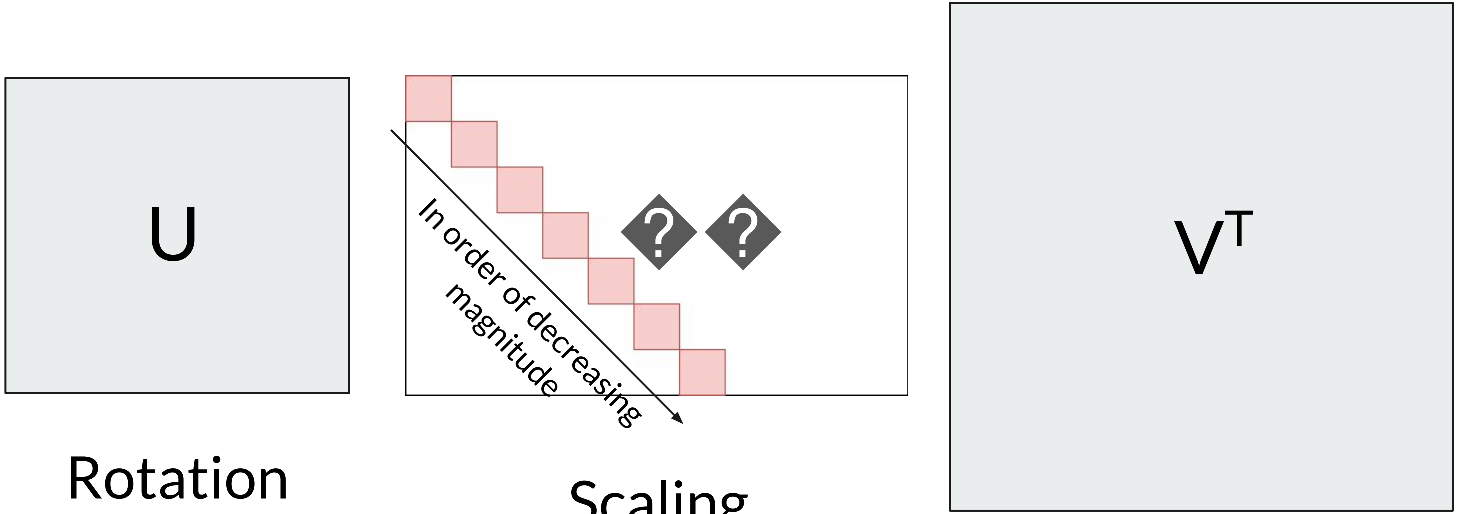
Orthonormal, unitary

(Rectangular) diagonal

Orthonormal, unitary

Singular Value Decomposition in a nutshell

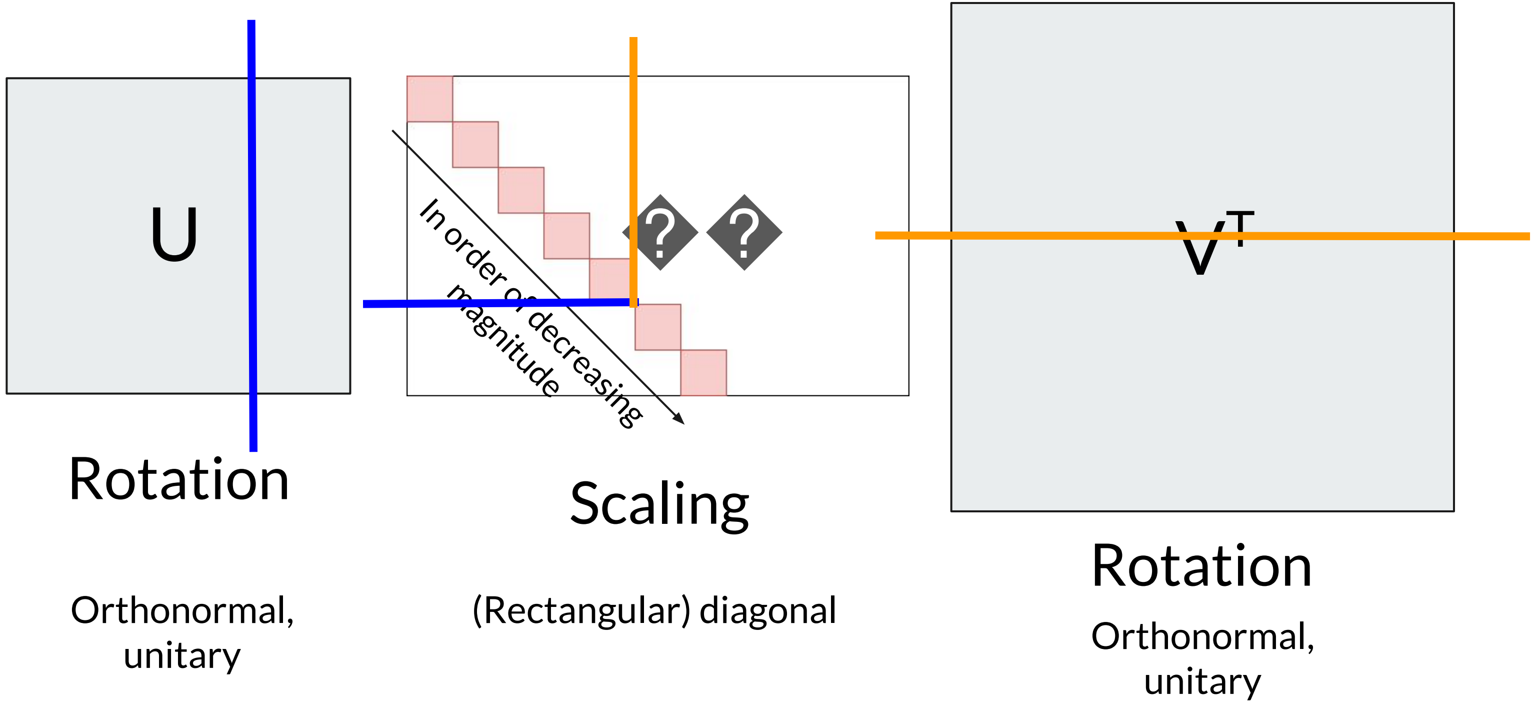
Any matrix can be decomposed into



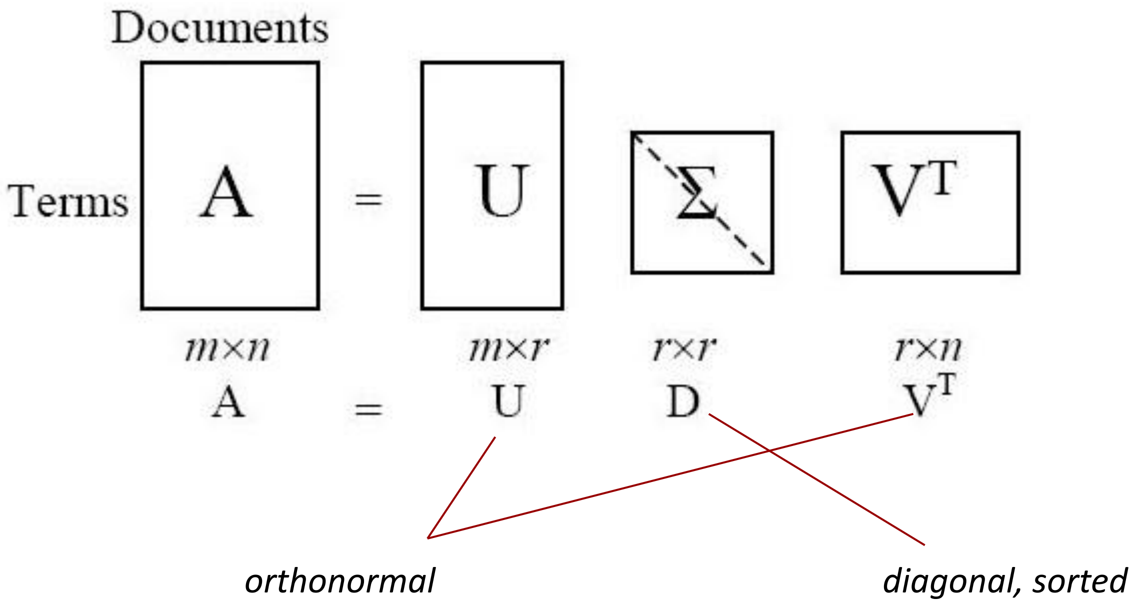
Rotation	Scaling	Rotation
Orthonormal, unitary	(Rectangular) diagonal	Orthonormal, unitary

Singular Value Decomposition in a nutshell

Let’s trim away the zero scaling factors

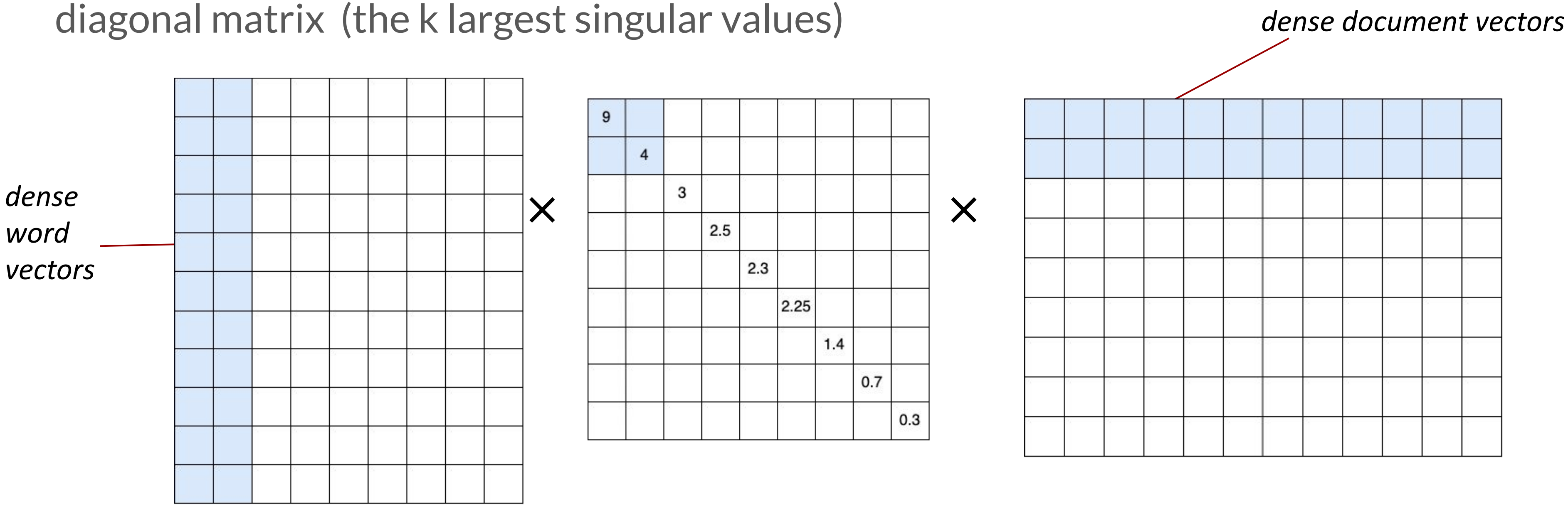


Singular Value Decomposition in a nutshell



Truncated SVD

We can approximate the full matrix by only considering the leftmost k terms in the diagonal matrix (the k largest singular values)



$$A_{m\times n} \approx U_{m\times k}\Sigma_{k\times k}V_{k\times n}^{\top} \qquad k \ll m, n$$

Latent Semantic Analysis

#0	#1	#2	#3	#4	#5
we	music	company	how	program	10
said	film	mr	what	project	30
have	theater	its	about	russian	11
they	mr	inc	their	space	12
not	this	stock	or	russia	15
but	who	companies	this	center	13
be	movie	sales	are	programs	14
do	which	shares	history	clark	20
he	show	said	be	aircraft	sept
this	about	business	social	ballet	16
there	dance	share	these	its	25
you	its	chief	other	projects	17
are	disney	executive	research	orchestra	18
what	play	president	writes	development	19
if	production	group	language	work	21

(Deerwester et al., 1990)

How do we tell whether a set of word embeddings is any good?

Evaluation

- Intrinsic
- Extrinsic
- Qualitative

WORD	d1	d2	d3	d4	d5	...	d50
summer	0.12	0.21	0.07	0.25	0.33	...	0.51
spring	0.19	0.57	0.99	0.30	0.02	...	0.73
fall	0.53	0.77	0.43	0.20	0.29	...	0.85
light	0.00	0.68	0.84	0.45	0.11	...	0.03
clear	0.27	0.50	0.21	0.56	0.25	...	0.32
blizzard	0.15	0.05	0.64	0.17	0.99	...	0.23

Extrinsic Evaluation

- Chunking
- POS tagging
- Parsing
- MT
- SRL
- Topic categorization
- Sentiment analysis
- Metaphor detection
- etc.
-

Intrinsic Evaluation

word1	word2	similarity (humans)	similarity (embeddings)
vanish	disappear	9.8	1.1
behave	obey	7.3	0.5
belief	impression	5.95	0.3
muscle	bone	3.65	1.7
modest	flexible	0.98	0.98
hole	agreement	0.3	0.3

- WS-353 (Finkelstein et al. ‘02)
- MEN-3k (Bruni et al. ‘12)
- SimLex-999 dataset (Hill et al., 2015)

Spearman's rho (human ranks, model ranks)

Computing word similarity

The dot product between two vectors is a scalar:

$$\text{dot product}(\mathbf{v}, \mathbf{w}) = \mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^{N} v_i w_i = v_1 w_1 + v_2 w_2 + \ldots + v_N w_N$$

- The dot product tends to be high when the two vectors have large values in the same dimensions
- Dot product can thus be a useful similarity metric between vectors

Problem with raw dot-product

- Dot product favors long vectors
  - Dot product is higher if a vector is longer (has higher values in many dimension) Vector length:

$$|\mathbf{v}| = \sqrt{\sum_{i=1}^{N} v_i^2}$$

- Frequent words (of, the, you) have long vectors (since they occur many times with other words).
  - So dot product overly favors frequent words

Alternative: cosine for computing word similarity

$$\text{cosine}(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

Based on the definition of the dot product between two vectors a and b

$$\mathbf{a}\cdot\mathbf{b} = |\mathbf{a}||\mathbf{b}|\cos\theta$$

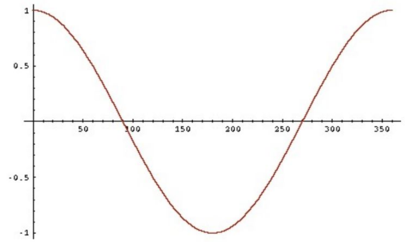
$$\frac{\mathbf{a}\cdot\mathbf{b}}{|\mathbf{a}||\mathbf{b}|} = \cos\theta$$

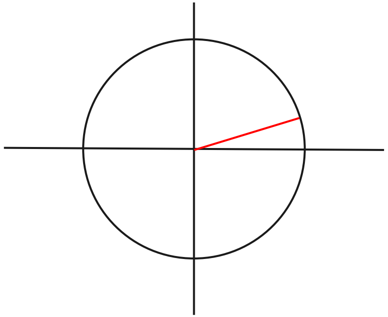
Cosine as a similarity metric

-1: vectors point in opposite directions

+1: vectors point in same directions

0: vectors are orthogonal





- But since raw frequency values are non-negative, the cosine for term-term matrix vectors ranges from 0–1

Cosine examples

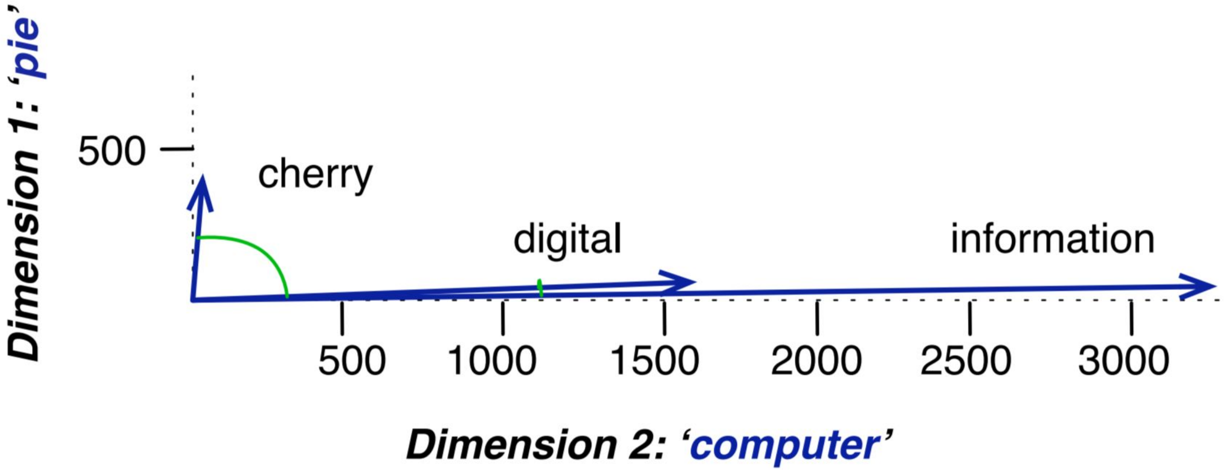
$$\cos(\vec{v},\vec{w}) = \frac{\vec{v}\bullet\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\vec{v}}{|\vec{v}|}\bullet\frac{\vec{w}}{|\vec{w}|} = \frac{\sum_{i=1}^{N} v_i w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

	pie	data	computer
cherry	442	8	2
digital	114	80	62
information	36	58	1

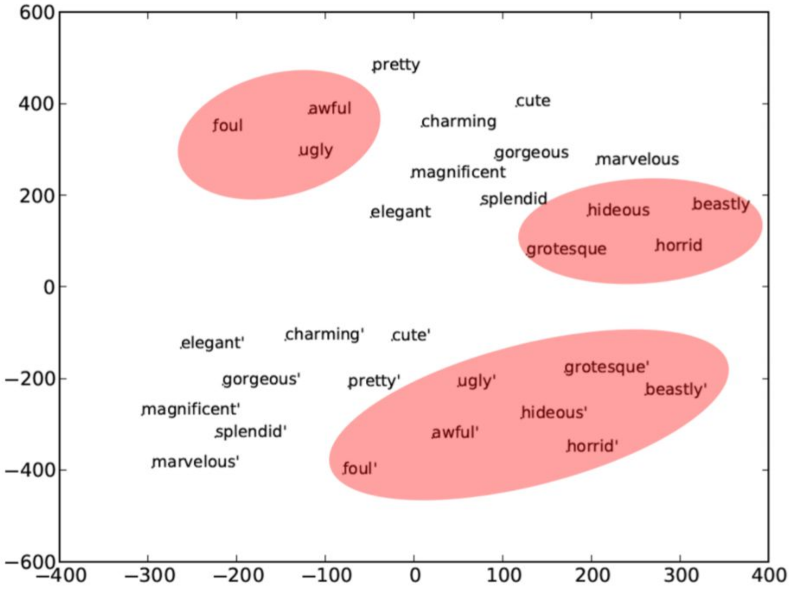
$$\cos(\text{cherry}, \text{information}) = \frac{442*5+8*3982+2*3325}{\sqrt{442^2+8^2+2^2}\sqrt{5^2+3982^2+3325^2}} = .017$$

$$\cos(\text{digital}, \text{information}) = \frac{5*5+1683*3982+1670*3325}{\sqrt{5^2+1683^2+1670^2}\sqrt{5^2+3982^2+3325^2}} = .996$$

Visualizing angles



Visualisation



(Faruqui et al., 2014)

Figure 6.5: Monolingual (top) and multilingual (bottom; marked with apostrophe) word projections of the antonyms (shown in red) and synonyms of "beautiful".

- Visualizing Data using t-SNE (van der Maaten & Hinton '08)

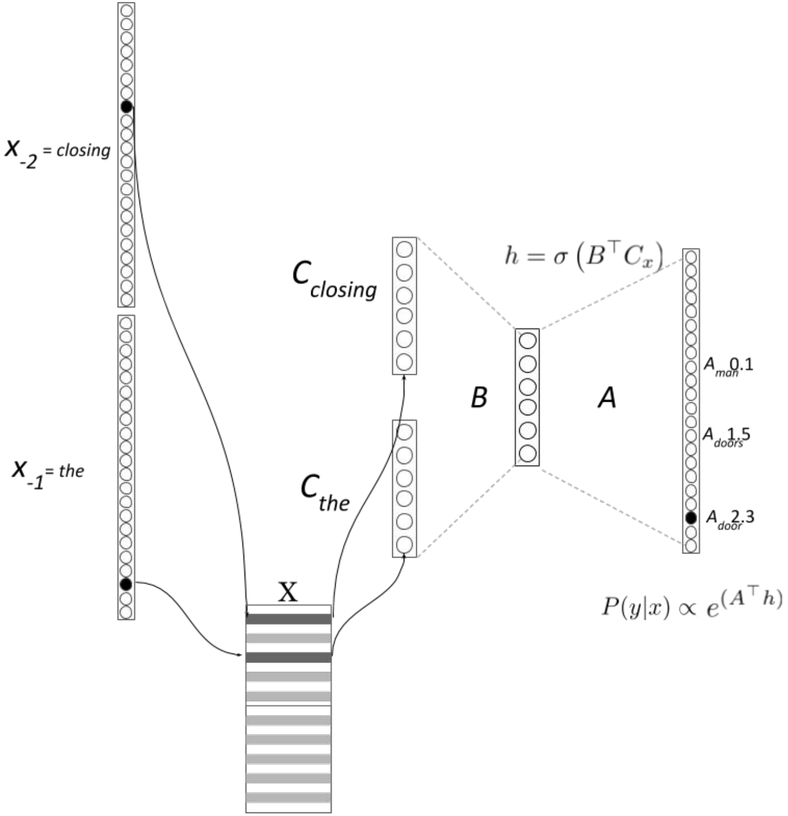
Dense vectors (part 2)

Distributed representations

Word Vectors

WORD	d1	d2	d3	d4	d5	...	d50
summer	0.12	0.21	0.07	0.25	0.33	...	0.51
spring	0.19	0.57	0.99	0.30	0.02	...	0.73
fall	0.53	0.77	0.43	0.20	0.29	...	0.85
light	0.00	0.68	0.84	0.45	0.11	...	0.03
clear	0.27	0.50	0.21	0.56	0.25	...	0.32
blizzard	0.15	0.05	0.64	0.17	0.99	...	0.23

“One hot” vectors and dense word vectors (embeddings)

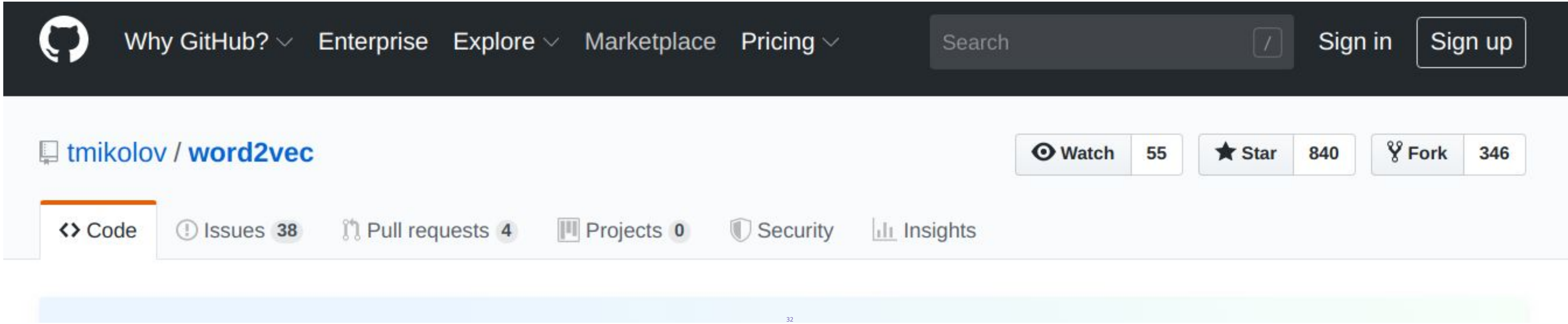


Low-dimensional word representations

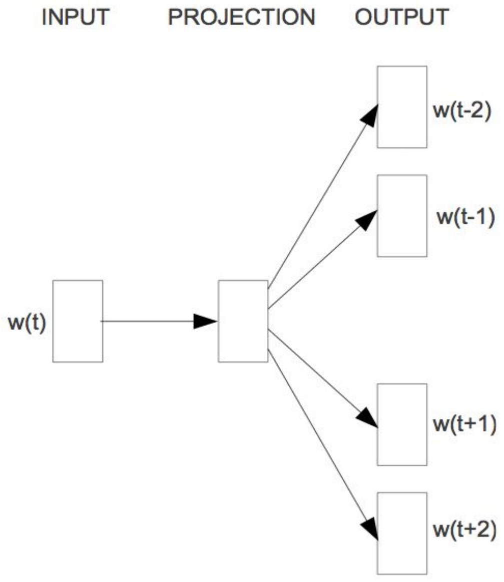
- Learning representations by back-propagating errors
  - Rumelhart, Hinton & Williams, 1986
- A neural probabilistic language model
  - Bengio et al., 2003
- Natural Language Processing (almost) from scratch
  - Collobert & Weston, 2008
- Word representations: A simple and general method for semi-supervised learning
  - Turian et al., 2010
- Distributed Representations of Words and Phrases and their Compositionality
  - Word2Vec; Mikolov et al., 2013

Word2Vec

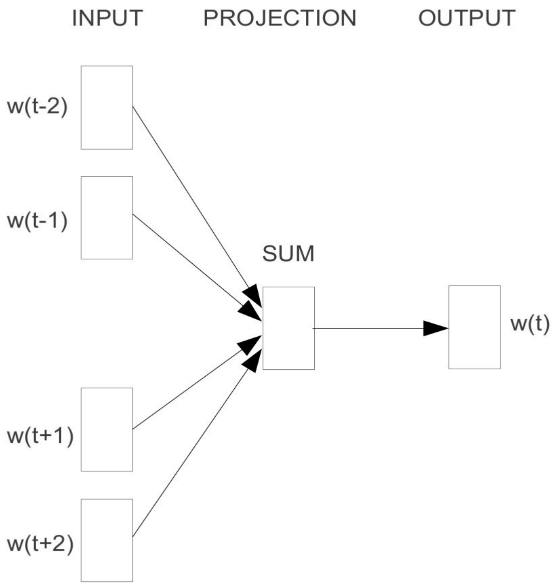
- Popular embedding method
- Very fast to train
- Code available on the web
- Idea: predict rather than count



Word2Vec



Skip-gram



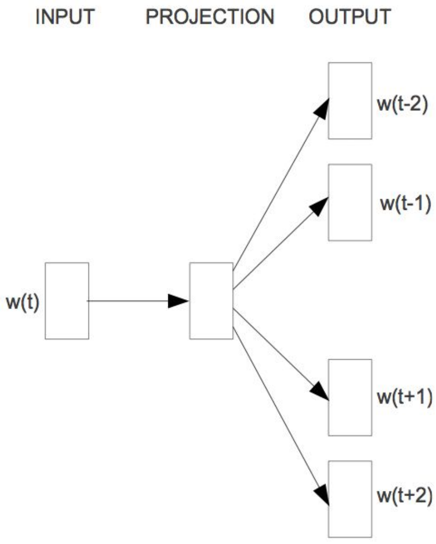
CBOW

- [Mikolov et al.' 13]

Skip-gram Prediction

- Predict vs Count

the cat sat on the mat



Skip-gram

Skip-gram Prediction

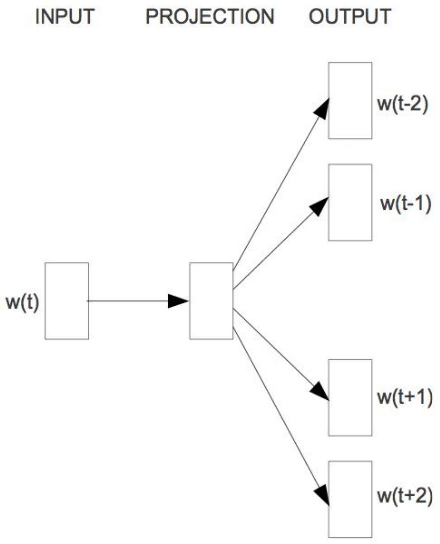
- Predict vs Count

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

Skip-gram Prediction

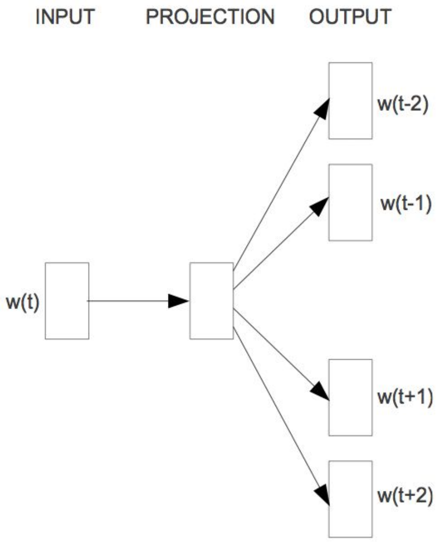
- Predict vs Count

the cat 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

Skip-gram Prediction

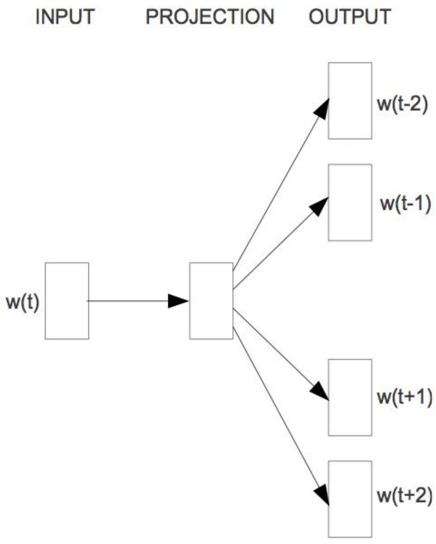
- Predict vs Count

the cat sat 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

Skip-gram Prediction

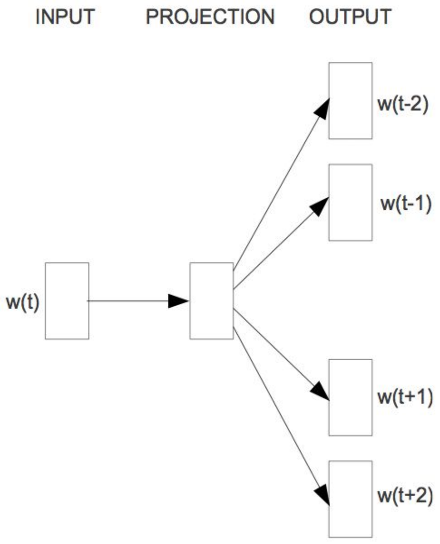
- Predict vs Count

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

Skip-gram Prediction

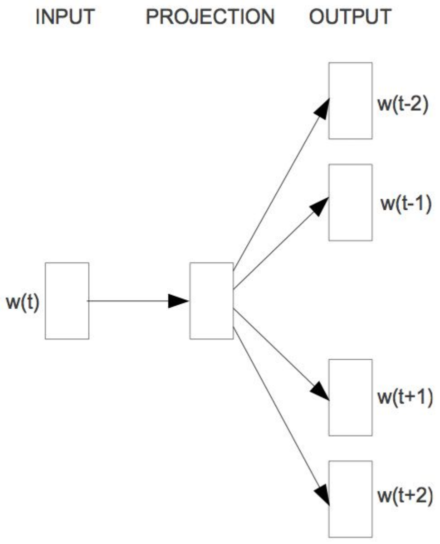
- Predict vs Count

the cat sat on the 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

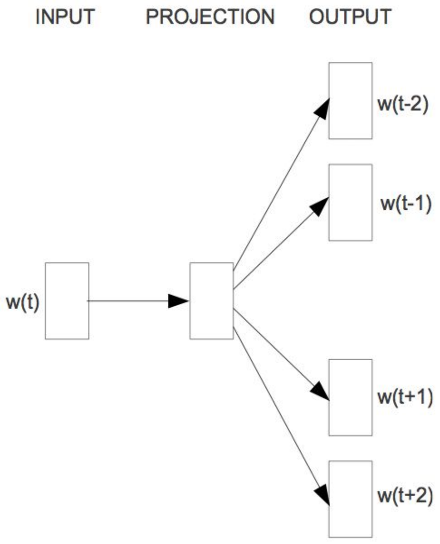
Skip-gram Prediction

- Predict vs Count

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

w_t = mat → **CLASSIFIER** →

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



Skip-gram

context size = 2

Skip-gram Prediction

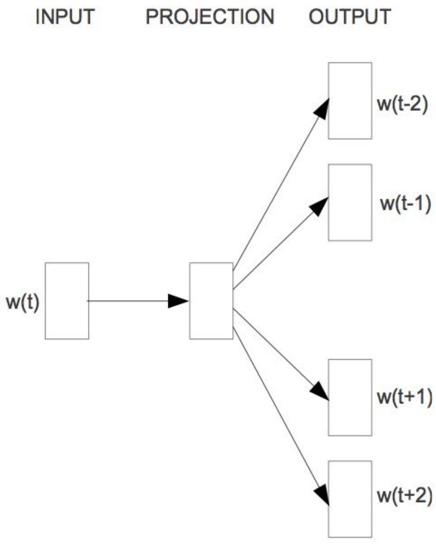
- Predict vs Count

w_t = the → **CLASSIFIER** →

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

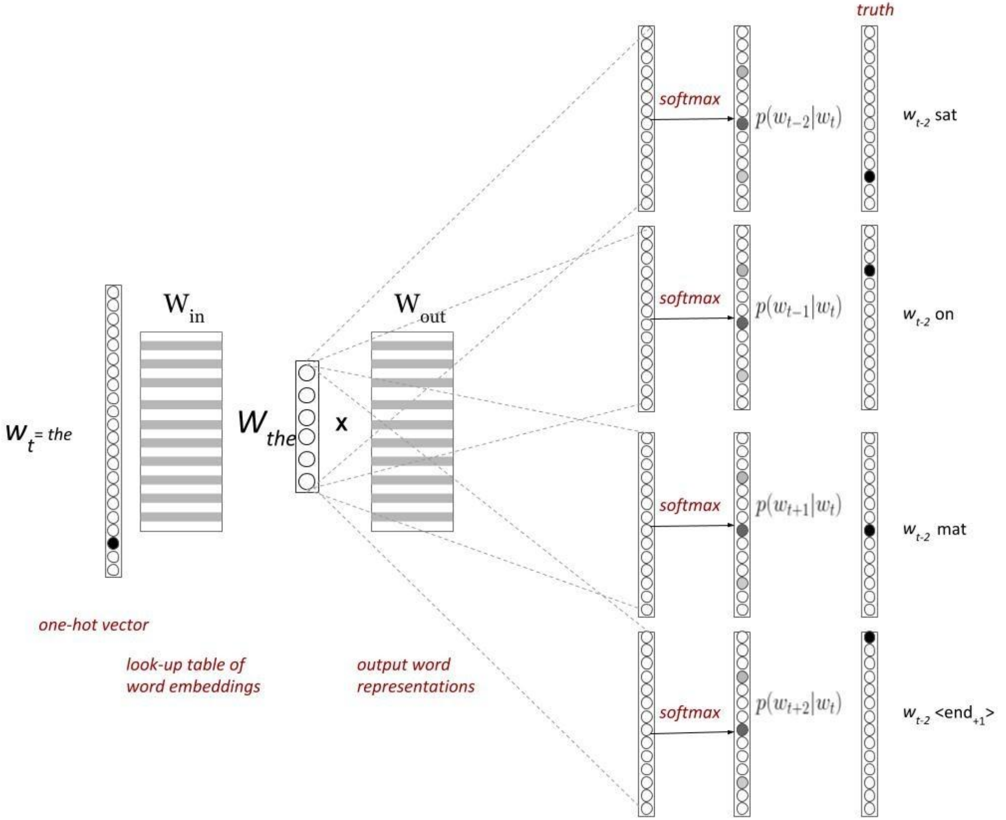
w_t = the → **CLASSIFIER** →

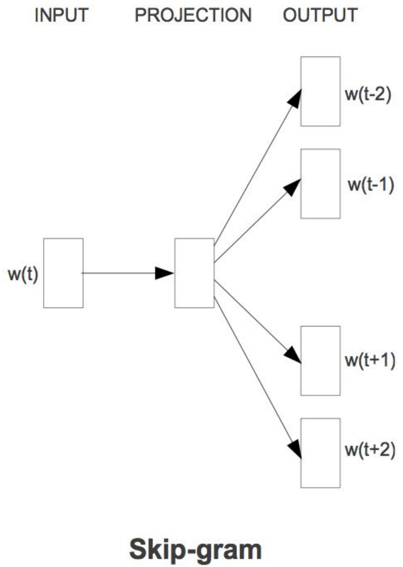
w_{t-2} = <start$_{-2}$>
w_{t-1} = <start$_{-1}$>
w_{t+1} = cat
w_{t+2} = sat



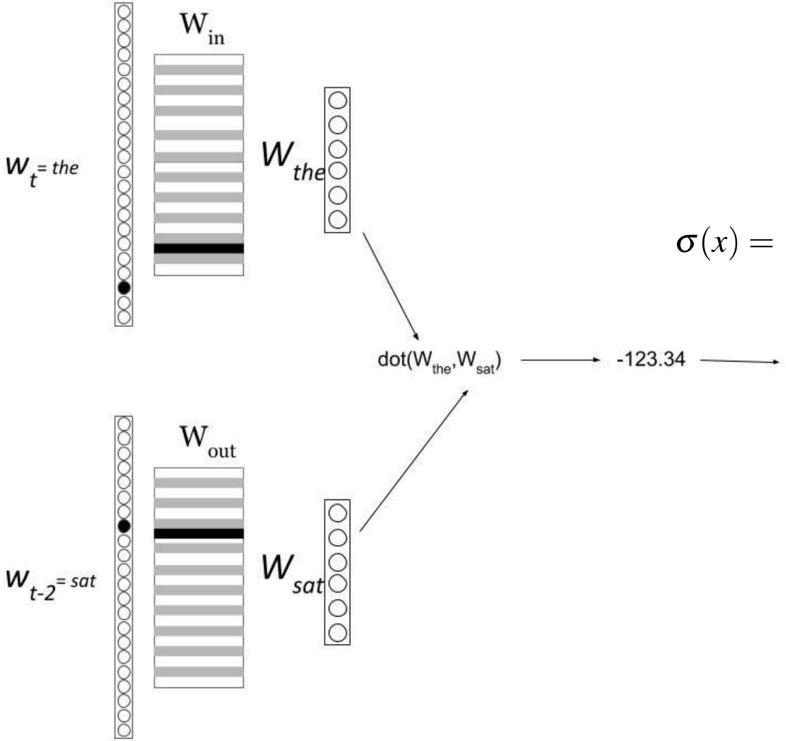
Skip-gram

Skip-gram Prediction

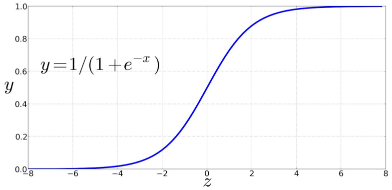




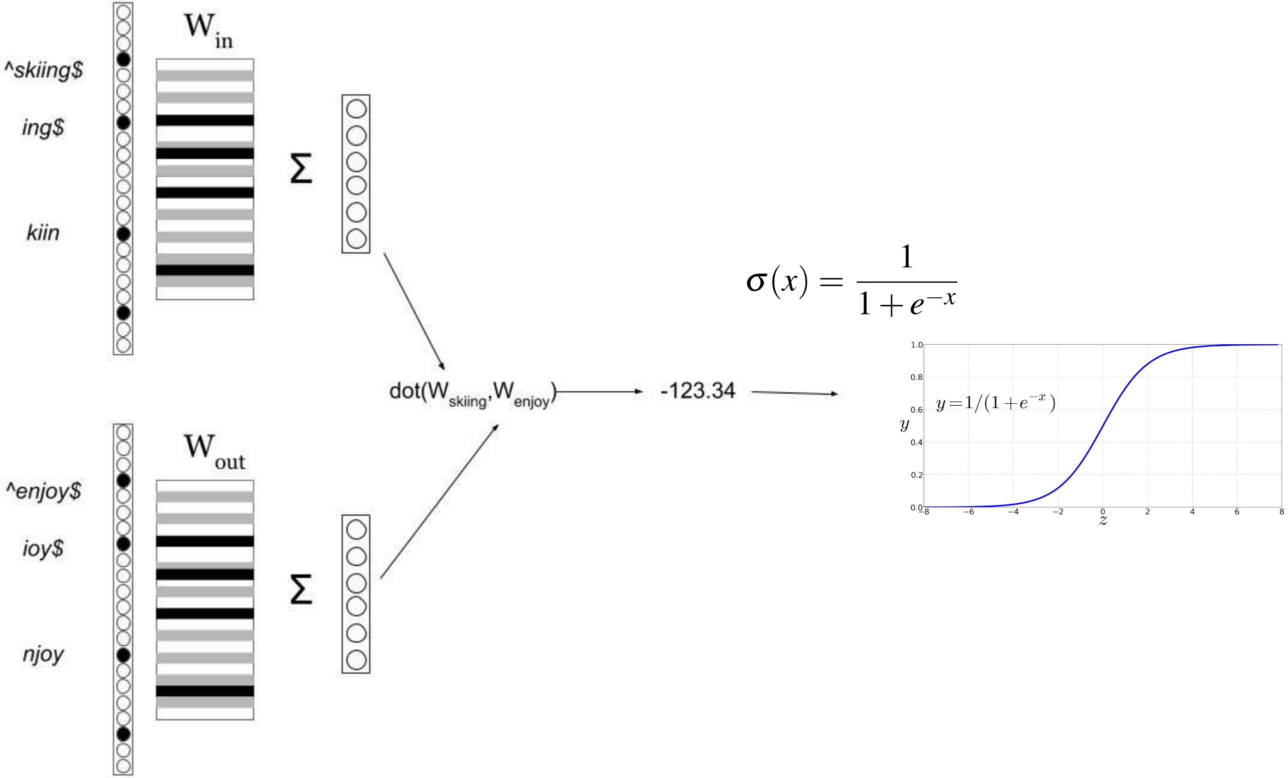
How to compute p(+|t,c)?



$$\sigma(x) = \frac{1}{1+e^{-x}}$$



FastText



$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Typical traits of these embeddings

Automatically learn some analogies pretty well

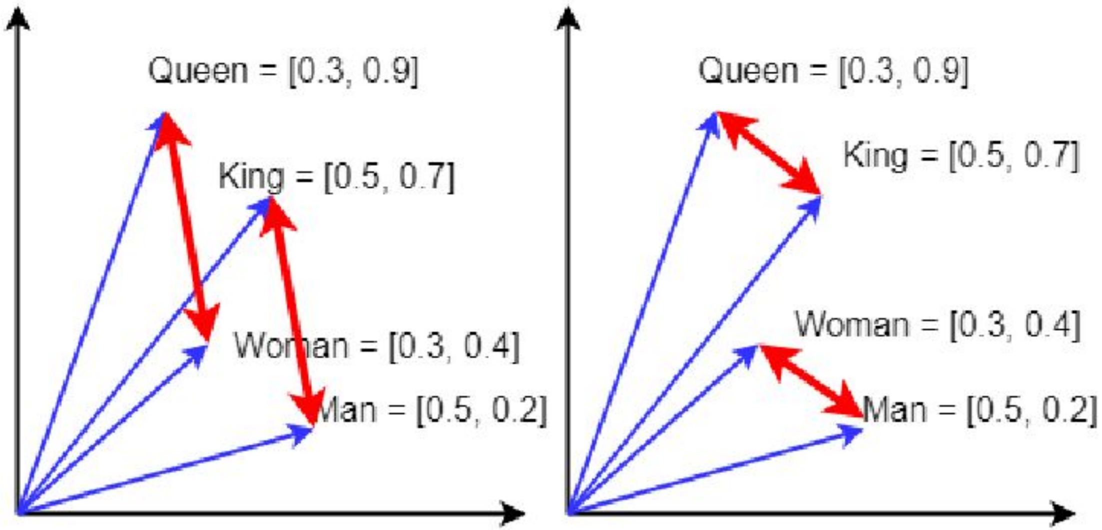


Figure from Sutor et al. MIPR 2019

Takeaways

What we’ve learned

- The contexts in which a word typically appears (i.e., the tokens that typically appear around it) tell us a lot about that word
- We can use those contexts to automatically learn more powerful representations of words than just a one-hot encoding
- These “word embeddings” can plug in as parameters in models of your choice

Next class

Neural Networks I (Feedforward networks and LSTMs)