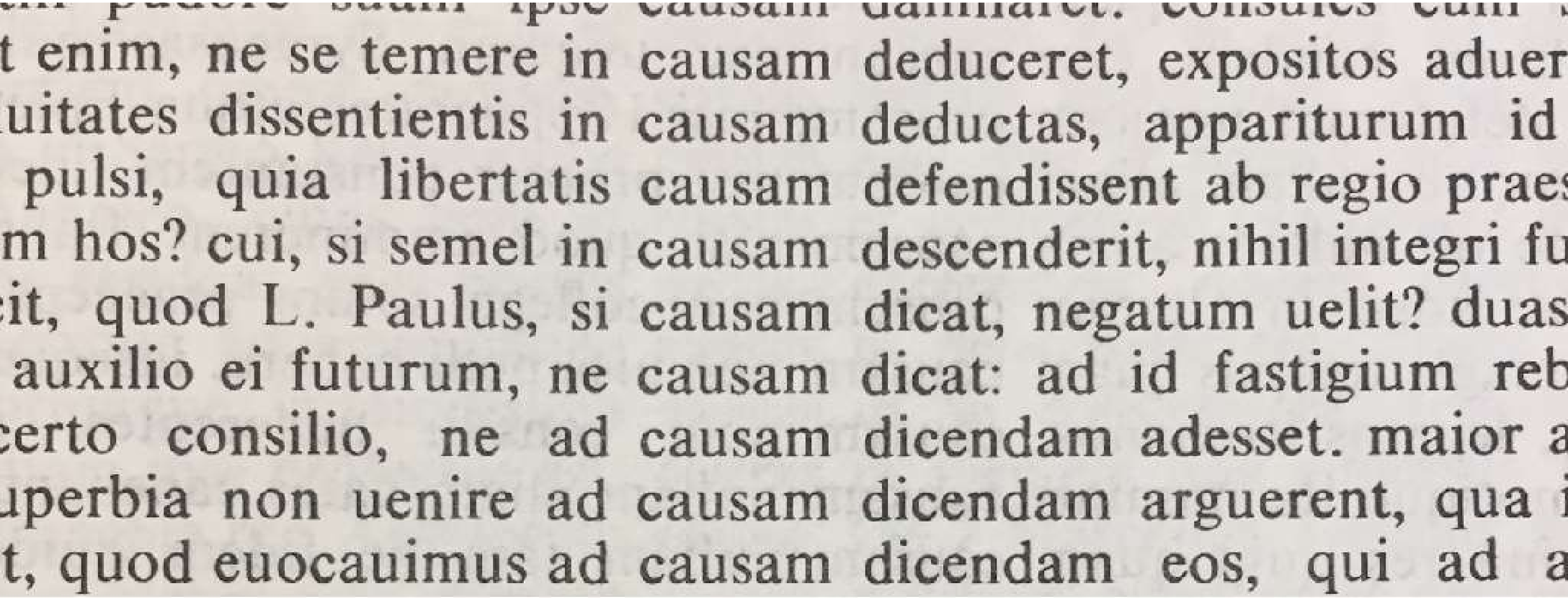

t enim, ne se temere in causam deduceret, expositos aduer
uitates dissentientis in causam deductas, apparitururm id
pulsi, quia libertatis causam defendissent ab regio praes
m hos? cui, si semel in causam descenderit, nihil integri fu
it, quod L. Paulus, si causam dicat, negatum uelit? duas
auxilio ei futurum, ne causam dicat: ad id fastigium reb
certo consilio, ne ad causam dicendam adesset. maior a
uperbia non uenire ad causam dicendam arguerent, qua i
t, quod euocauimus ad causam dicendam eos, qui ad a

David Packard, *A Concordance to Livy* (1968)

Natural Language Processing

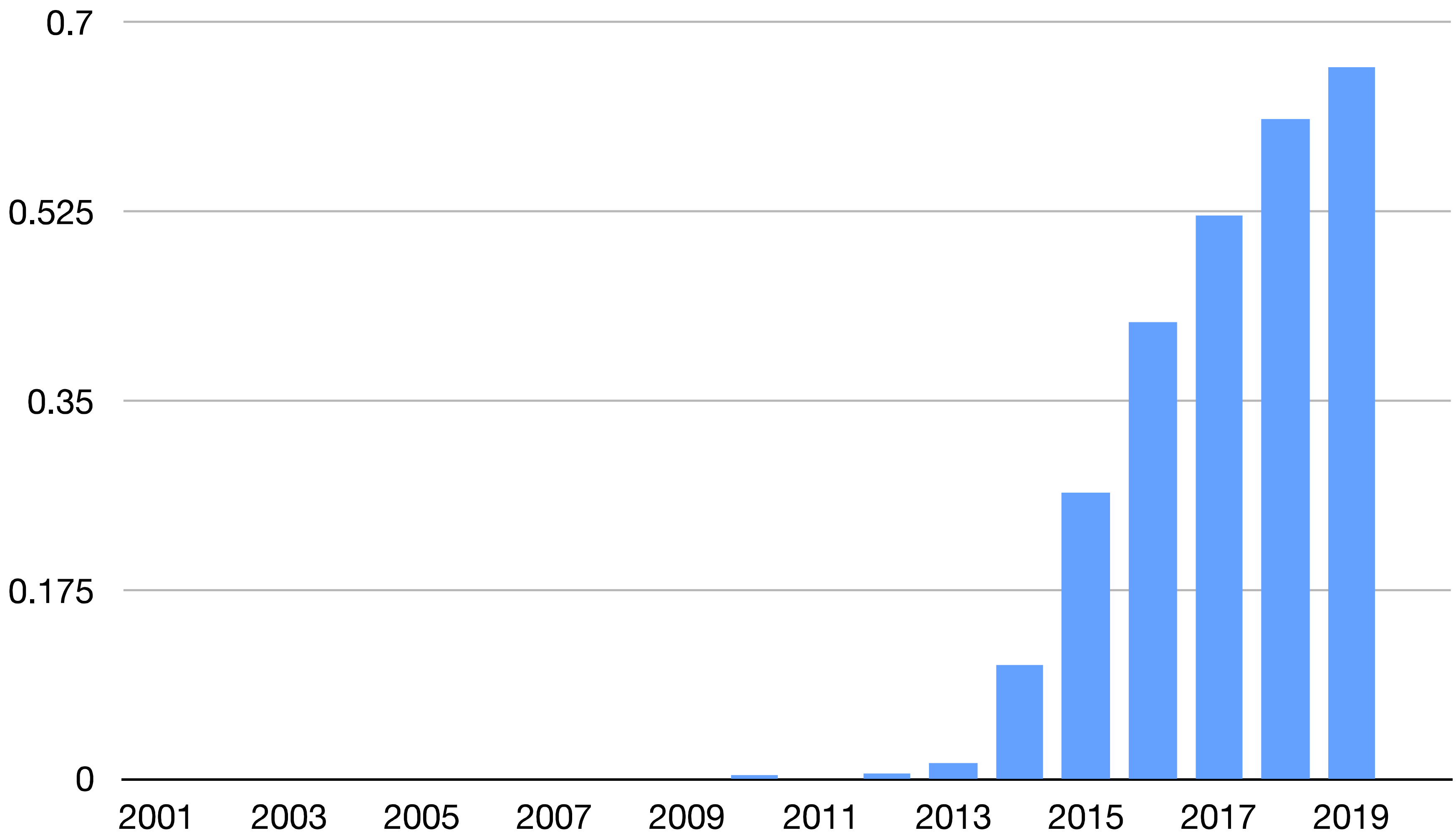
Info 159/259

Lecture 8: Vector semantics and word embeddings

(Feb 13, 2020)

David Bamman, UC Berkeley

“Word embedding” in NLP papers



Data from ACL papers in the ACL Anthology
https://www.aclweb.org/anthology/

Lexical semantics

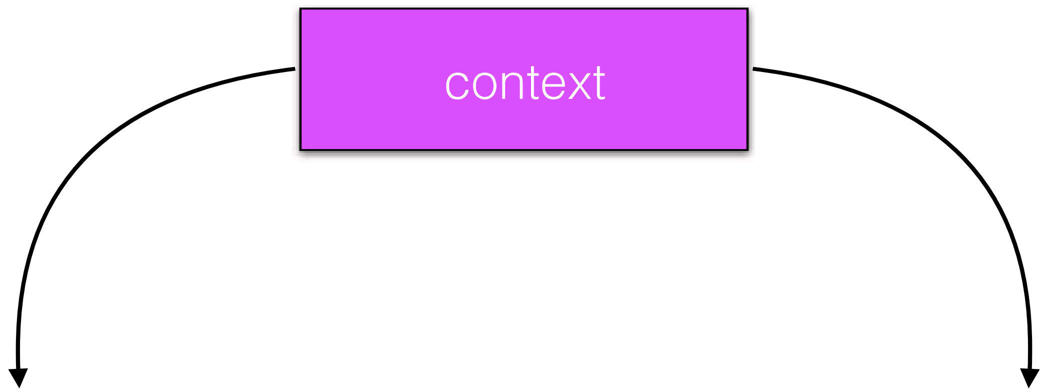
"You shall know a word by the company it keeps"

[Firth 1957]

Zellig Harris, "Distributional Structure" (1954)

Ludwig Wittgenstein, Philosophical Investigations (1953)

everyone likes	________________	
a bottle of	________________	is on the table
	________________	makes you drunk
a cocktail with	________________	and seltzer



everyone likes	______________	
a bottle of	______________	is on the table
	______________	makes you drunk
a cocktail with	______________	and seltzer

Distributed representation

- Vector representation that encodes information about the distribution of contexts a word appears in

- Words that appear in similar contexts have similar representations (and similar meanings, by the distributional hypothesis).

- We have several different ways we can encode the notion of “context.”

Term-document matrix

	Hamlet	Macbeth	Romeo & Juliet	Richard III	Julius Caesar	Tempest	Othello	King Lear
knife	1	1	4	2		2		10
dog				6	12	2		
sword	2	2	7	5		5		17
love	64		135	63		12		48
like	75	38	34	36	34	41	27	44

Context = appearing in the same document.

Vectors

knife	1	1	4	2		2		10

sword	2	2	7	5		5		17

Vector representation of the term; vector size = number of documents

Cosine Similarity

$$cos(x,y) = \frac{\sum_{i=1}^{F} x_i y_i}{\sqrt{\sum_{i=1}^{F} x_i^2}\sqrt{\sum_{i=1}^{F} y_i^2}}$$

- We can calculate the cosine similarity of two vectors to judge the degree of their similarity [Salton 1971]

- Euclidean distance measures the magnitude of distance between two points
- Cosine similarity measures their orientation

	Hamlet	Macbeth	R&J	R3	JC	Tempest	Othello	KL
knife	1	1	4	2		2		10
dog				6	12	2		
sword	2	2	7	5		5		17
love	64		135	63		12		48
like	75	38	34	36	34	41	27	44

cos(knife, knife)	1
cos(knife, dog)	0.11
cos(knife, sword)	0.99
cos(knife, love)	0.65
cos(knife, like)	0.61

Weighting dimensions

- Not all dimensions are equally informative

TF-IDF

- Term frequency-inverse document frequency
- A scaling to represent a feature as function of how frequently it appears in a data point but accounting for its frequency in the overall collection
- IDF for a given term = the number of documents in collection / number of documents that contain term

TF-IDF

- Term frequency ($tf_{t,d}$) = the number of times term t occurs in document d; several variants (e.g., passing through log function).

- Inverse document frequency = inverse fraction of number of documents containing (D_t) among total number of documents N

$$tfidf(t, d) = tf_{t,d} \times \log \frac{N}{D_t}$$

IDF

	Hamlet	Macbeth	Romeo & Juliet	Richard III	Julius Caesar	Tempest	Othello	King Lear
knife	1	1	4	2		2		2
dog	2		6	6		2		12
sword	17	2	7	12		2		17
love	64		135	63		12		48
like	75	38	34	36	34	41	27	44

IDF
0.12
0.20
0.12
0.20
0

IDF for the informativeness of the terms when comparing documents

PMI

- Mutual information provides a measure of how independent two variables (X and Y) are.

- Pointwise mutual information measures the independence of two outcomes (x and y)

PMI

$$\log_2 \frac{P(x,y)}{P(x)P(y)}$$

w = word, c = context

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

What's this value for w and c that never occur together?

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

	Hamlet	Macbeth	Romeo & Juliet	Richard III	Julius Caesar	Tempest	Othello	King Lear	total
knife	1	1	4	2		2		2	12
dog	2		6	6		2		12	28
sword	17	2	7	12		2		17	57
love	64		135	63		12		48	322
like	75	38	34	36	34	41	27	44	329
total	159	41	186	119	34	59	27	123	748

$$PMI(\text{love}, \text{R\&J}) = \frac{\frac{135}{748}}{\frac{186}{748} \times \frac{322}{748}}$$

Term-context matrix

- Rows and columns are both words; cell counts = the number of times word w_i and w_j show up in the same context (e.g., a window of 2 tokens).

Dataset

- the big dog ate dinner
- the small cat ate dinner
- the white dog ran down the street
- the yellow cat ran inside

Dataset

- the big dog ate dinner
- the small cat ate dinner
- the white dog ran down the street
- the yellow cat ran inside

DOG terms (window = 2)

the big ate dinner the white ran down

Dataset

- the big dog ate dinner
- the small cat ate dinner
- the white dog ran down the street
- the yellow cat ran inside

DOG terms (window = 2)

the big ate dinner the white ran down

CAT terms (window = 2)

the small ate dinner the yellow ran inside

Term-context matrix

contexts

term

	the	big	ate	dinner	…
dog	2	1	1	1	…
cat	2	0	1	1	…

- Each cell enumerates the number of times a context word appeared in a window of 2 words around the term.
- How big is each representation for a word here?

We can also define "context" to be directional ngrams (i.e., ngrams of a defined order occurring to the left or right of the term)

Dataset

- the big dog ate dinner
- the small cat ate dinner
- the white dog ran down the street
- the yellow cat ran inside

DOG terms (window = 2)

L: the big, R: ate dinner, L: the white, R: ran down

CAT terms (window = 2)

L: the small, R: ate dinner, L: the yellow, R: ran inside

Term-context matrix

contexts

term

	L: the big	R: ate dinner	L: the small	L: the yellow	…
dog	1	1	0	0	…
cat	0	1	1	1	…

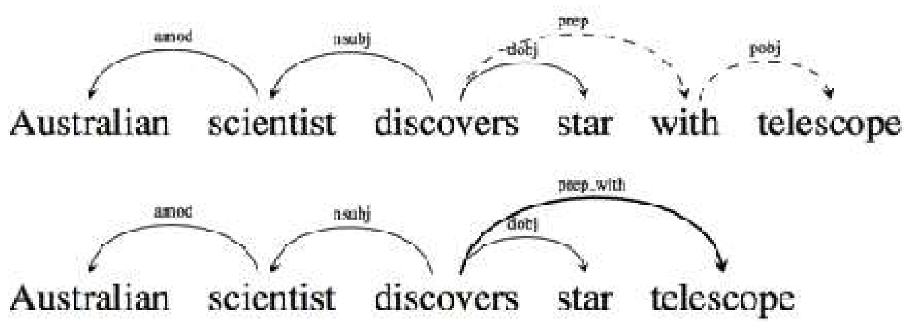
- Each cell enumerates the number of time a directional context phrase appeared in a specific position around the term.

write a book
write a poem

- First-order co-occurrence (syntagmatic association): write co-occurs with book in the same sentence.

- Second-order co-occurrence (paradigmatic association): book co-occurs with poem (since each co-occur with write)

Syntactic context



WORD	CONTEXTS
australian	scientist/amod^{-1}
scientist	australian/amod, discovers/nsubj^{-1}
discovers	scientist/nsubj, star/dobj, telescope/prep_with
star	discovers/dobj^{-1}
telescope	discovers/prep_with^{-1}

Target Word	BoW5	BoW2	DEPS
batman	nightwing aquaman catwoman superman manhunter	superman superboy aquaman catwoman batgirl	superman superboy supergirl catwoman aquaman
hogwarts	dumbledore hallows half-blood malfoy snape	evernight sunnydale garderobe blandings collinwood	sunnydale collinwood calarts greendale millfield
turing	nondeterministic non-deterministic computability deterministic finite-state	non-deterministic finite-state nondeterministic buchi primality	pauling hotelling heting lessing hamming
florida	gainesville fla jacksonville tampa lauderdale	fla alabama gainesville tallahassee texas	texas louisiana georgia california carolina
object-oriented	aspect-oriented smalltalk event-driven prolog domain-specific	aspect-oriented event-driven objective-c dataflow 4gl	event-driven domain-specific rule-based data-driven human-centered
dancing	singing dance dances dancers tap-dancing	singing dance dances breakdancing clowning	singing rapping breakdancing miming busking

Lin 1998; Levy and Goldberg 2014

Evaluation

Intrinsic Evaluation

- Relatedness: correlation (Spearman/Pearson) between vector similarity of pair of words and human judgments

word 1	word 2	human score
midday	noon	9.29
journey	voyage	9.29
car	automobile	8.94
…	…	…
professor	cucumber	0.31
king	cabbage	0.23

WordSim-353 (Finkelstein et al. 2002)

Intrinsic Evaluation

- Analogical reasoning (Mikolov et al. 2013). For analogy Germany : Berlin :: France : ???, find closest vector to v("Berlin") - v("Germany") + v("France")

			target
possibly	impossibly	certain	uncertain
generating	generated	shrinking	shrank
think	thinking	look	looking
Baltimore	Maryland	Oakland	California
shrinking	shrank	slowing	slowed
Rabat	Morocco	Astana	Kazakhstan

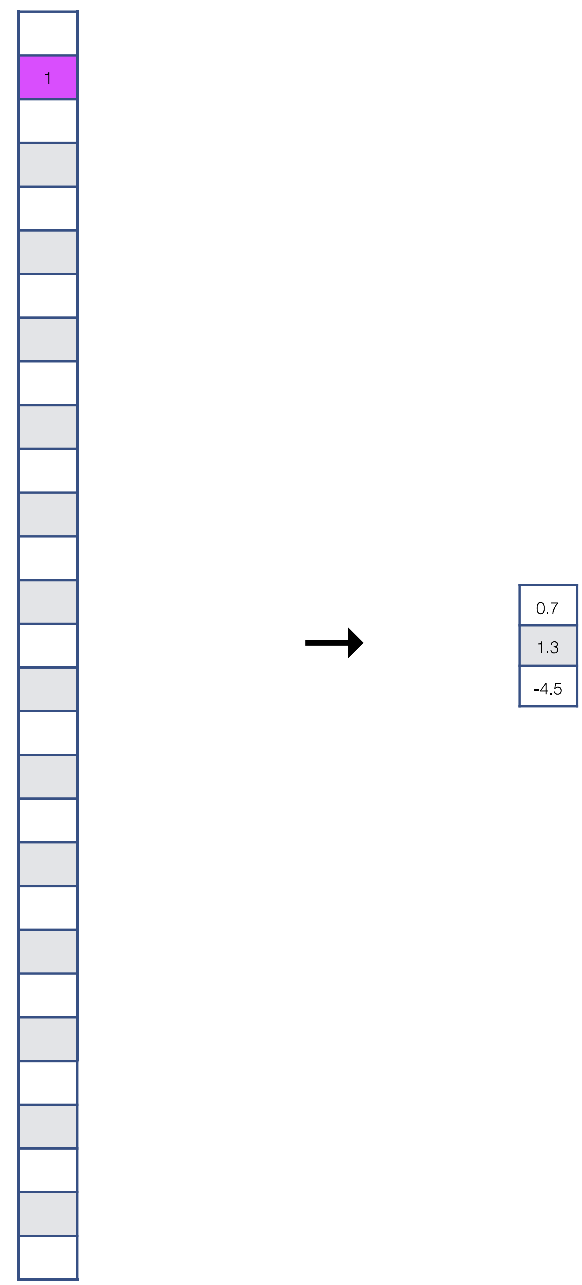
Sparse vectors

“aardvark”

V-dimensional vector, single 1 for the identity of the element

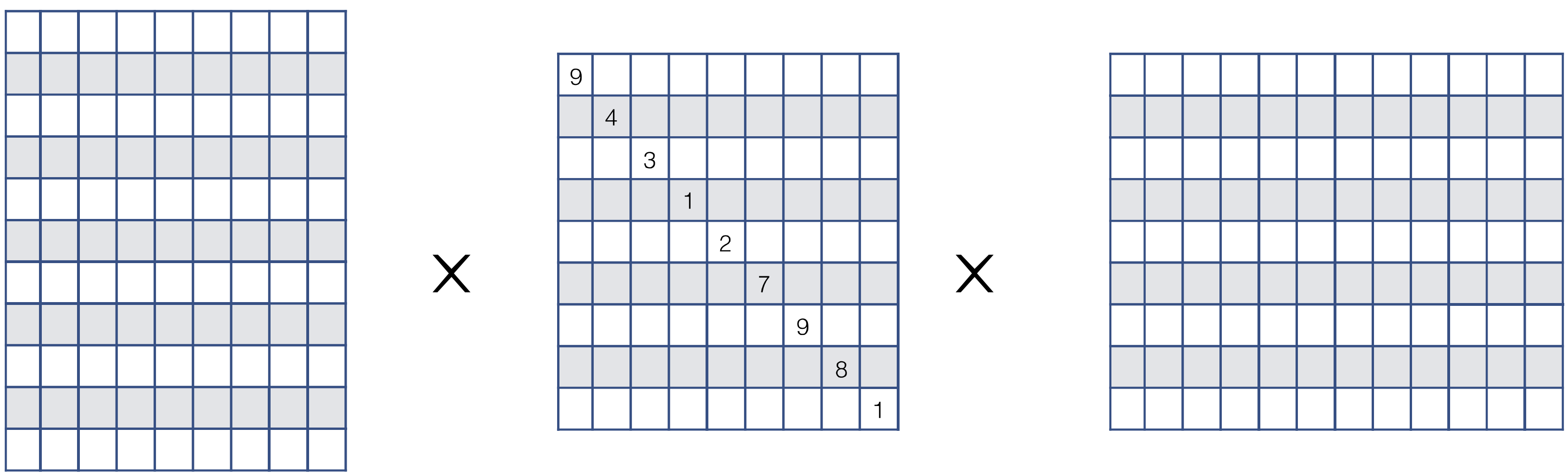
A	0
a	0
aa	0
aal	0
aalii	0
aam	0
Aani	0
aardvark	1
aardwolf	0
...	0
zymotoxic	0
zymurgy	0
Zyrenian	0
Zyrian	0
Zyryan	0
zythem	0
Zythia	0
zythum	0
Zyzomys	0
Zyzzogeton	0

Dense vectors



Singular value decomposition

- Any nxp matrix X can be decomposed into the product of three matrices (where m = the number of linearly independent rows)



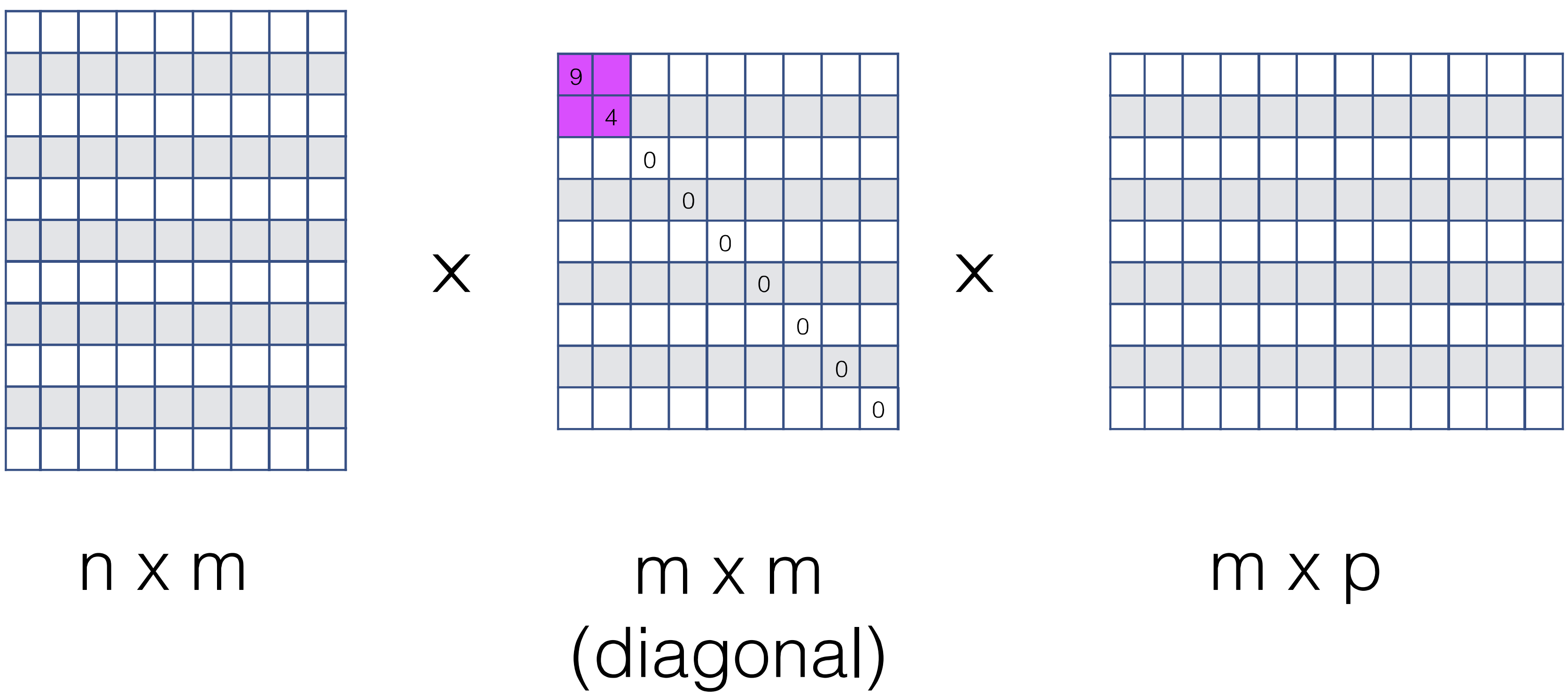
n x m

m x m
(diagonal)

m x p

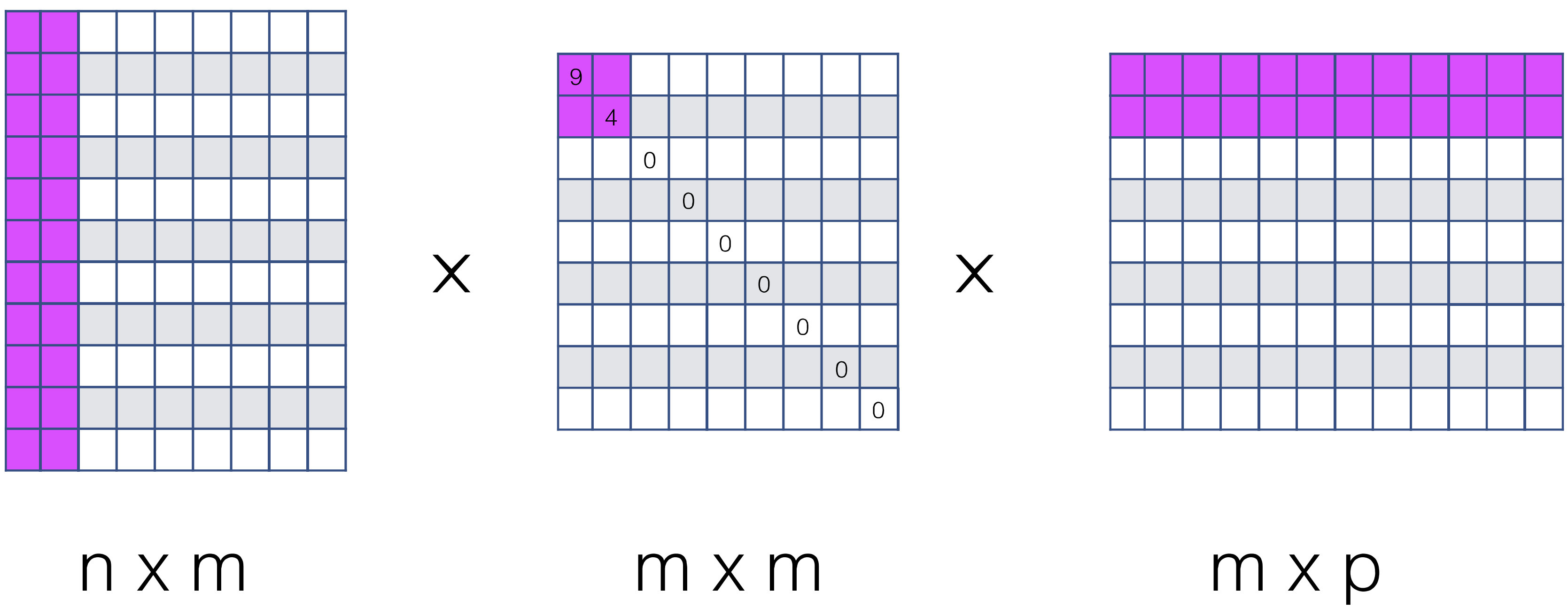
Singular value decomposition

- We can approximate the full matrix by only considering the leftmost k terms in the diagonal matrix



Singular value decomposition

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



	Hamlet	Macbeth	Romeo & Juliet	Richard III	Julius Caesar	Tempest	Othello	King Lear
knife	1	1	4	2		2		2
dog	2		6	6		2		12
sword	17	2	7	12		2		17
love	64		135	63		12		48
like	75	38	34	36	34	41	27	44

knife		
dog		
sword		
love		
like		

Hamlet	Macbeth	Romeo & Juliet	Richard III	Julius Caesar	Tempest	Othello	King Lear

Low-dimensional representation for terms (here 2-dim)

Low-dimensional representation for documents (here 2-dim)

knife		
dog		
sword		
love		
like		

Hamlet	Macbeth	Romeo & Juliet	Richard III	Julius Caesar	Tempest	Othello	King Lear

Latent semantic analysis

- Latent Semantic Analysis/Indexing (Deerwester et al. 1998) is this process of applying SVD to the term-document co-occurence matrix

- Terms typically weighted by tf-idf

- This is a form of dimensionality reduction (for terms, from a D-dimensionsal sparse vector to a K-dimensional dense one), $K << D$.

Dense vectors from prediction

- Learning low-dimensional representations of words by framing a predicting task: using context to predict words in a surrounding window

- Transform this into a supervised prediction problem; similar to language modeling but we're ignoring order within the context window

Dense vectors from prediction

Skipgram model (Mikolov et al. 2013): given a single word in a sentence, predict the words in a context window around it.

a cocktail with gin and seltzer

x	y
gin	a
gin	cocktail
gin	with
gin	and
gin	seltzer

Window size = 3

Dimensionality reduction

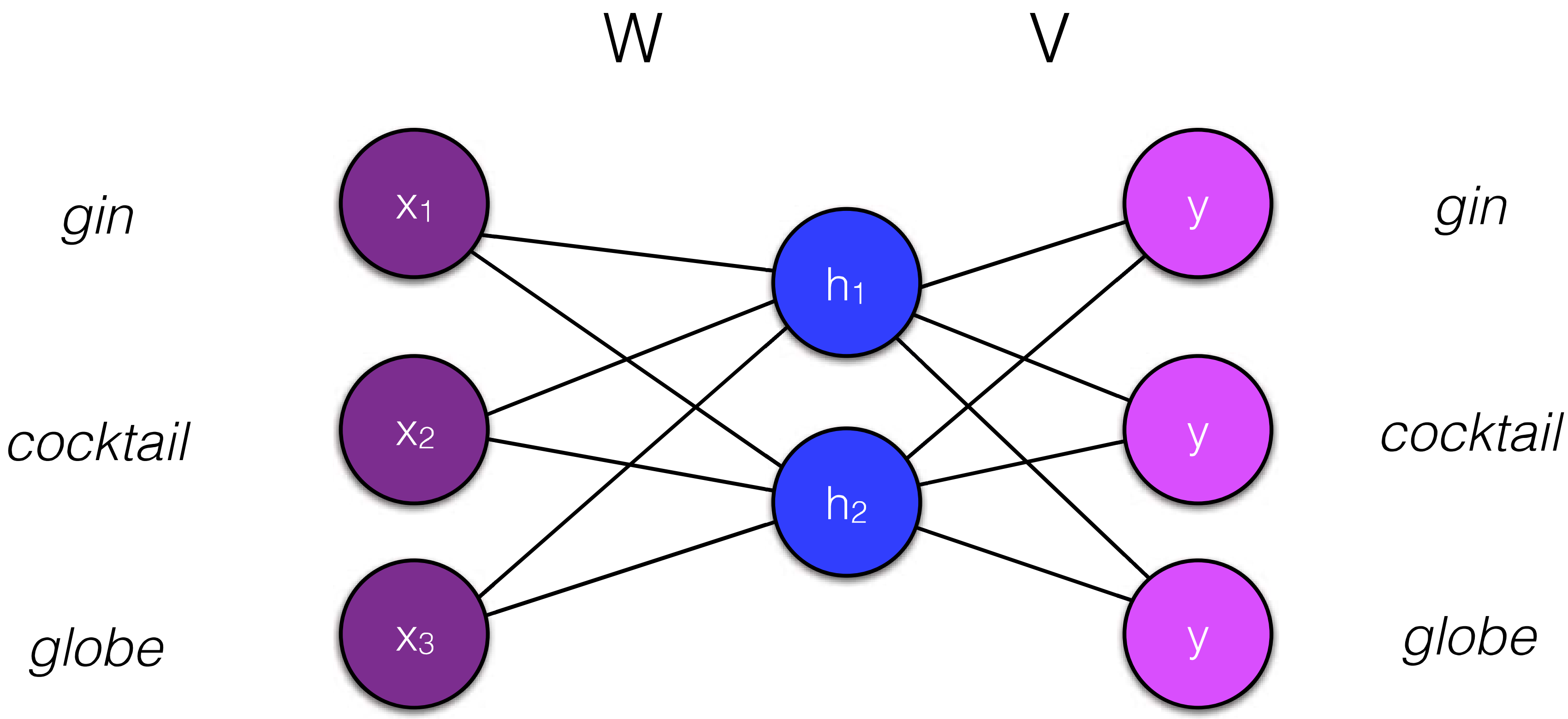
…	…
the	1
a	0
an	0
for	0
in	0
on	0
dog	0
cat	0
…	…

the is a point in V-dimensional space

the

4.1
-0.9

the is a point in 2-dimensional space

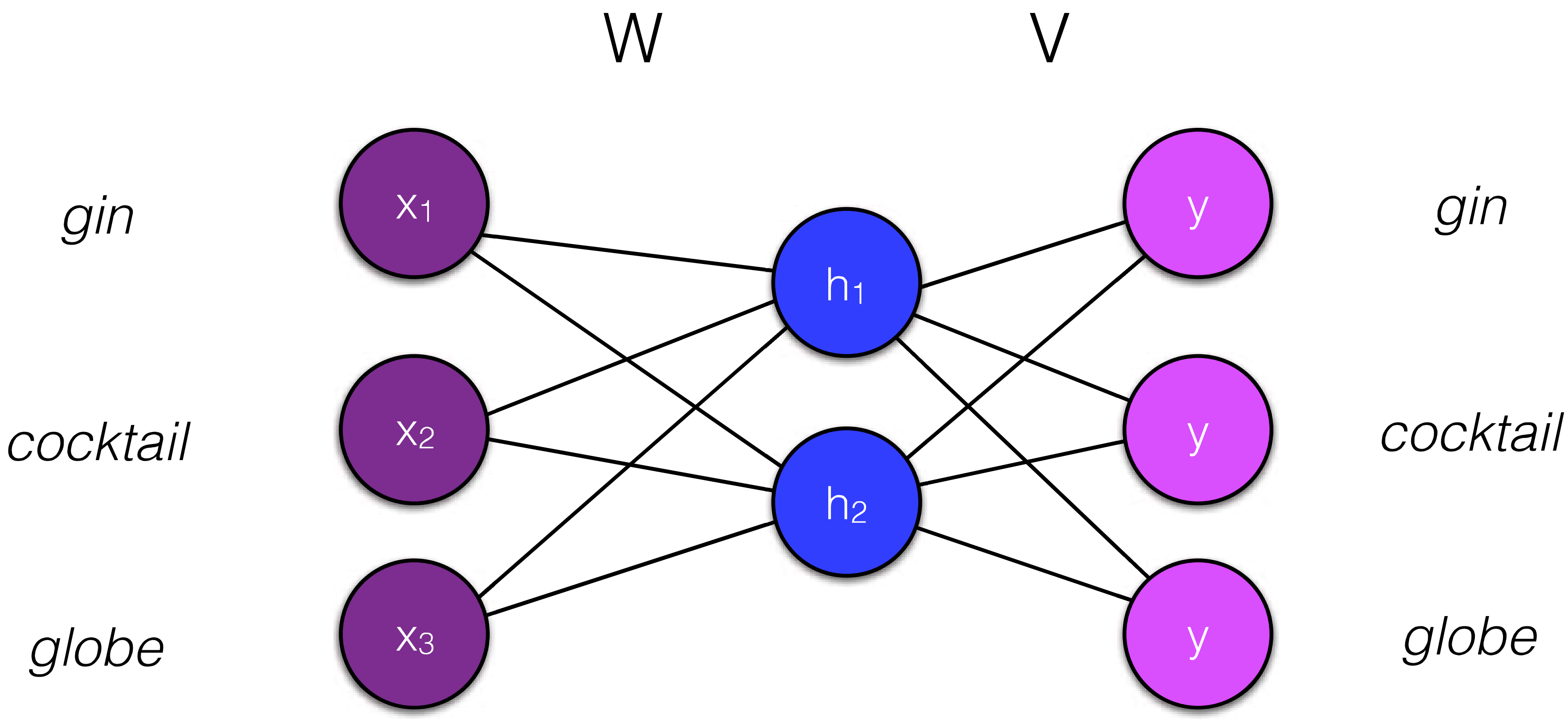


	x
gin	0
cocktail	1
globe	0

W	
-0.5	1.3
0.4	0.08
1.7	3.1

V		
4.1	0.7	0.1
-0.9	1.3	0.3

y
1
0
0



Only one of the inputs is nonzero.

= the inputs are really $W_{cocktail}$

W	
-0.5	1.3
0.4	0.08
1.7	3.1

V		
4.1	0.7	0.1
-0.9	1.3	0.3

x	W	
	0.13	0.56
	-1.75	0.07
	0.80	1.19
	-0.11	1.38
	-0.62	-1.46
	-1.16	-1.24
	0.99	-0.26
	-1.46	-0.85
	0.79	0.47
	0.06	-1.21
	-0.31	0.00
1	-1.01	-2.52
	-1.50	-0.14
	-0.14	0.01
	-0.13	-1.76
	-1.08	-0.56
	-0.17	-0.74
	0.31	1.03
	-0.24	-0.84
	-0.79	-0.18

$$x^\top W =$$

-1.01	-2.52

This is the embedding of the context

Word embeddings

- Can you predict the output word from a vector representation of the input word?

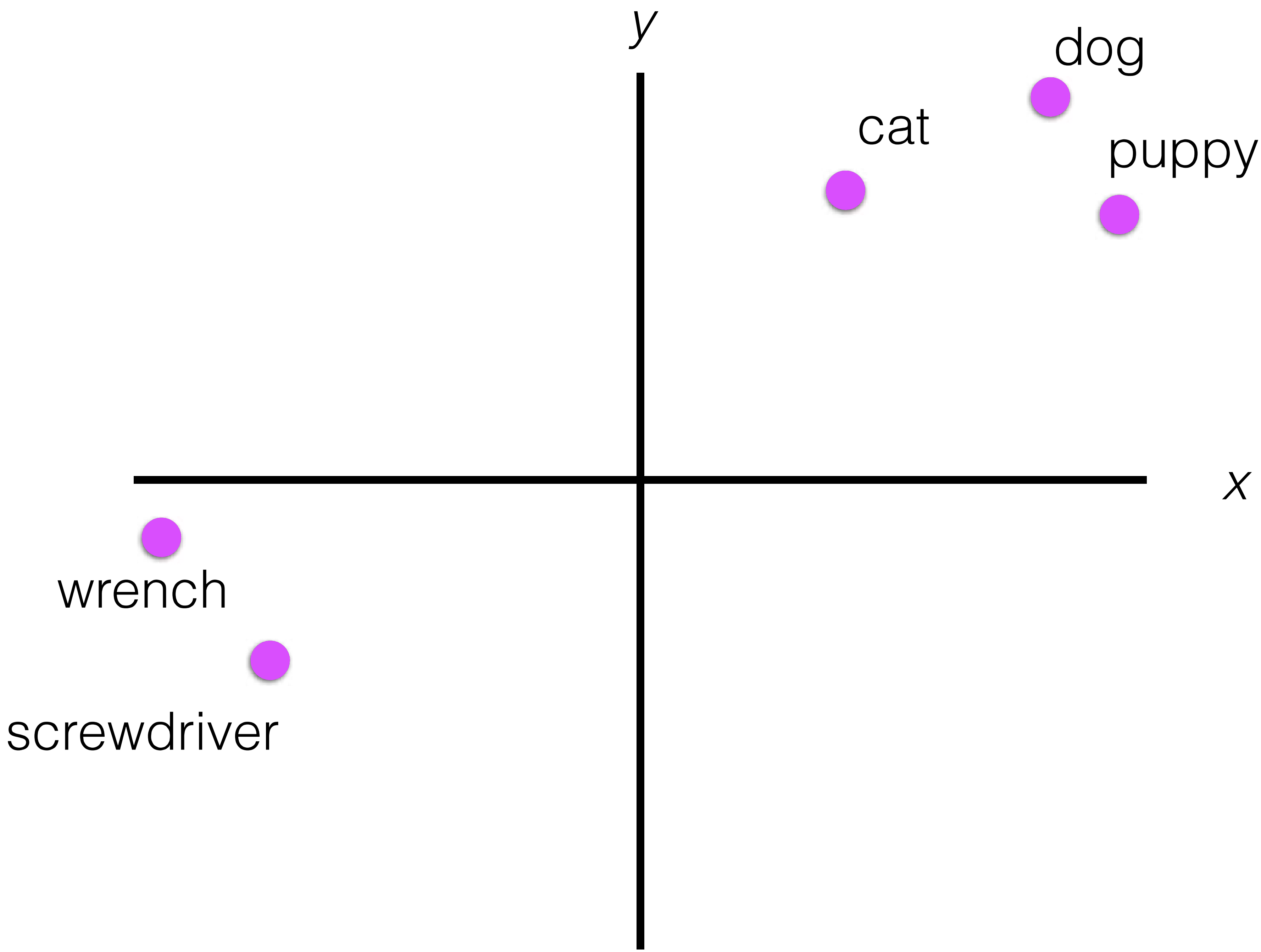
- Rather than seeing the input as a one-hot encoded vector specifying the word in the vocabulary we're conditioning on, we can see it as indexing into the appropriate row in the weight matrix W

Word embeddings

- Similarly, V has one H-dimensional vector for each element in the vocabulary (for the words that are being predicted)

V			
gin	cocktail	cat	globe
4.1	0.7	0.1	1.3
-0.9	1.3	0.3	-3.4

This is the embedding of the word


y
dog
cat
puppy
x
wrench
screwdriver

- Why this behavior? *dog*, *cat* show up in similar positions

the	black	cat	jumped	on	the	table
the	black	dog	jumped	on	the	table
the	black	puppy	jumped	on	the	table
the	black	skunk	jumped	on	the	table
the	black	shoe	jumped	on	the	table

- Why this behavior? *dog*, *cat* show up in similar positions

the	black	[0.4, 0.08]	jumped	on	the	table
the	black	[0.4, 0.07]	jumped	on	the	table
the	black	puppy	jumped	on	the	table
the	black	skunk	jumped	on	the	table
the	black	shoe	jumped	on	the	table

To make the same predictions, these numbers need to be close to each other.

Dimensionality reduction

…	…
the	1
a	0
an	0
for	0
in	0
on	0
dog	0
cat	0
…	…

the is a point in V-dimensional space;
representations for all words are completely independent

the
4.1
-0.9

the is a point in 2-dimensional space
representations are now structured

Analogical inference

- Mikolov et al. 2013 show that vector representations have some potential for analogical reasoning through vector arithmetic.

$$\text{apple} - \text{apples} \approx \text{car} - \text{cars}$$

$$\text{king} - \text{man} + \text{woman} \approx \text{queen}$$

Mikolov et al., (2013), “Linguistic Regularities in Continuous Space Word Representations” (NAACL)

Science AAAS

REPORT

Semantics derived automatically from language corpora contain human-like biases

Aylin Caliskan[1,*], Joanna J. Bryson[1,2,*], Arvind Narayanan[1,*]

+ See all authors and affiliations

Science 14 Apr 2017:
Vol. 356, Issue 6334, pp. 183-186
DOI: 10.1126/science.aal4230

Low-dimensional distributed representations

- Low-dimensional, dense word representations are extraordinarily powerful (and are arguably responsible for much of gains that neural network models have in NLP).

- Lets your representation of the input share statistical strength with words that behave similarly in terms of their distributional properties (often synonyms or words that belong to the same class).

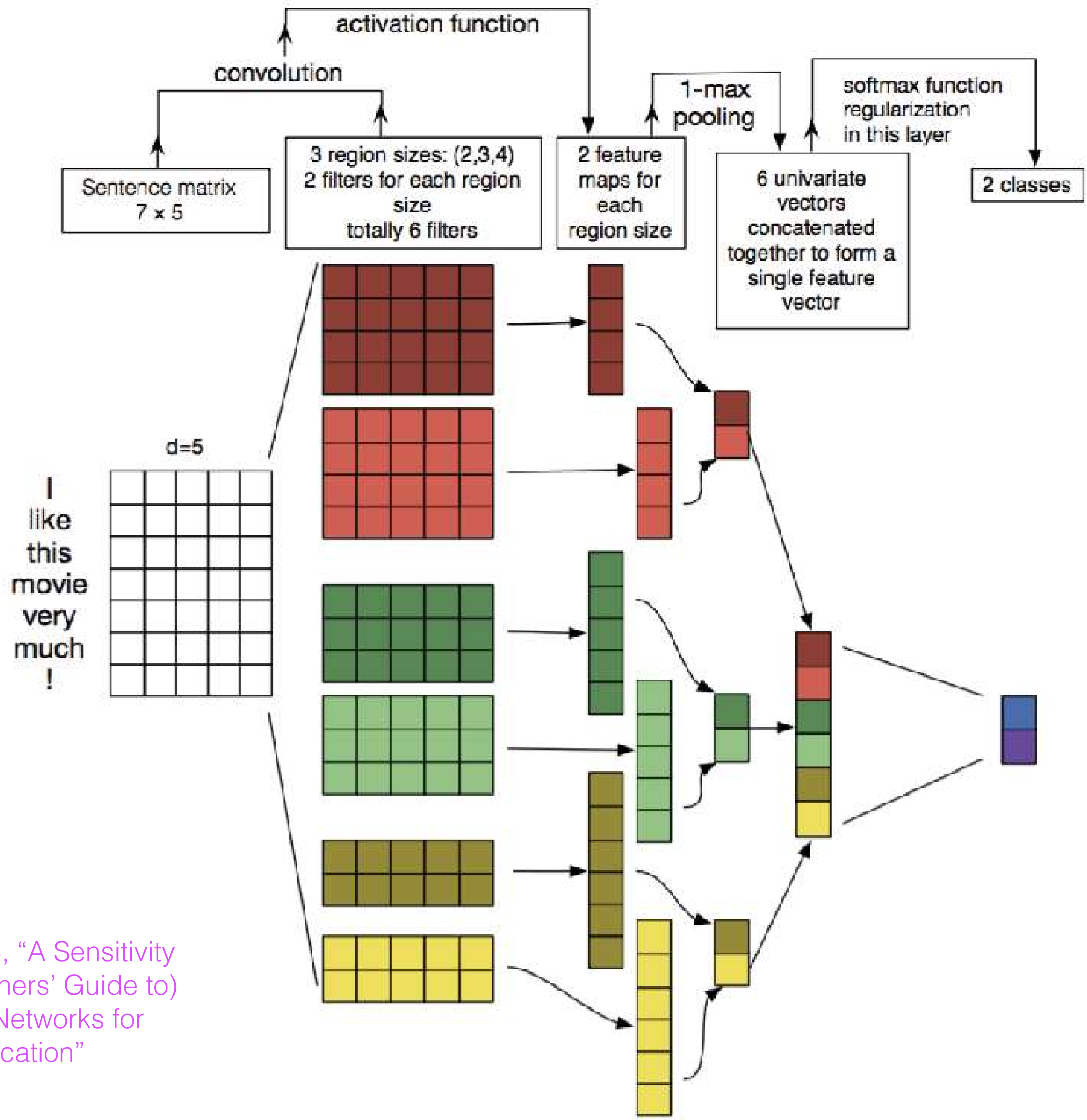
Two kinds of “training” data

- The labeled data for a specific task (e.g., labeled sentiment for movie reviews): ~ 2K labels/reviews, ~1.5M words → used to train a supervised model

- General text (Wikipedia, the web, books, etc.), ~ trillions of words → used to train word distributed representations

Using dense vectors

- In neural models (CNNs, RNNs, LM), replace the V-dimensional sparse vector with the much smaller K-dimensional dense one.
- Can also take the derivative of the loss function with respect to those representations to optimize for a particular task.



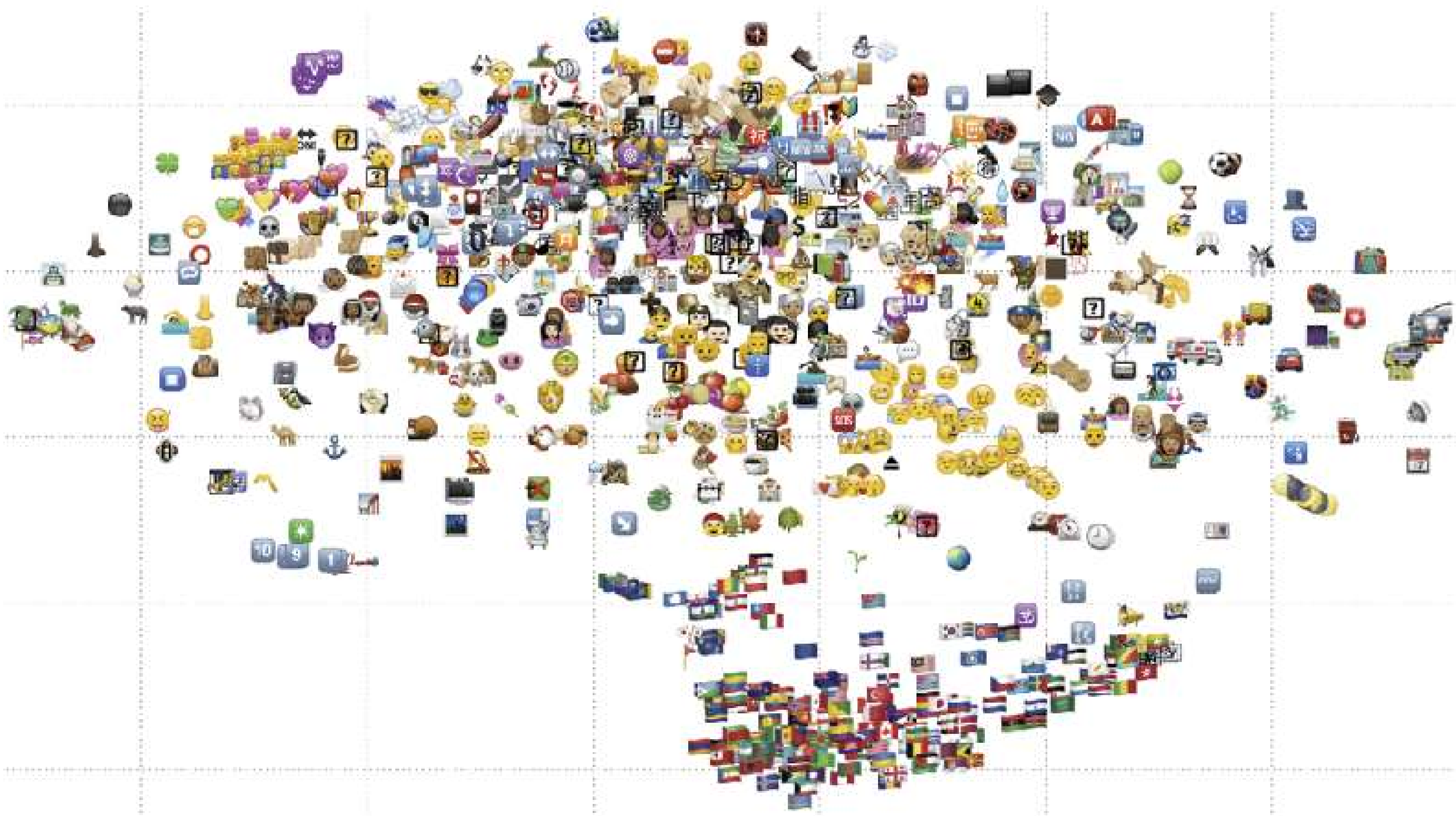
Zhang and Wallace 2016, "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification"

Using dense vectors

- (Short) document-level representation: coordinate-wise max, min or average; use directly in neural network.

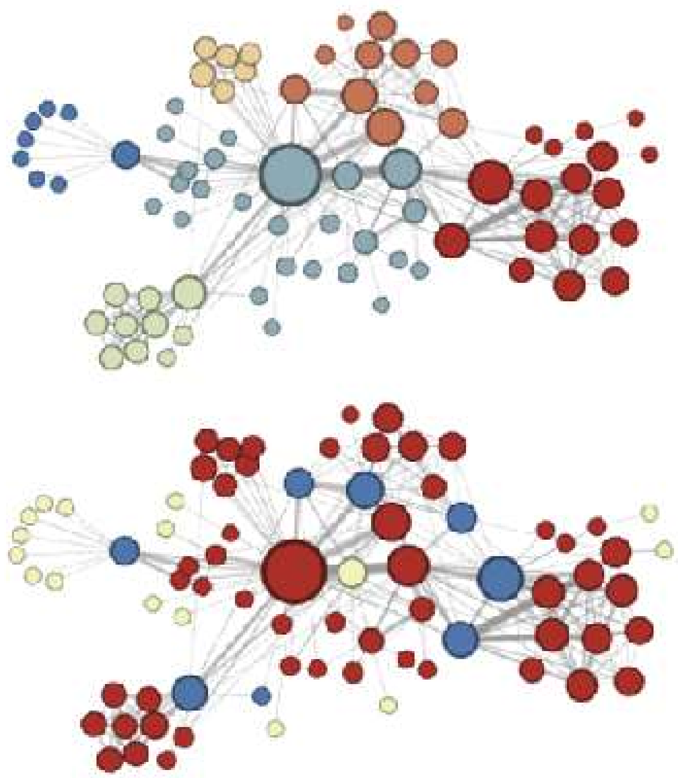
- K-means clustering on vectors into distinct partitions (though beware of strange geometry [Mimno and Thompson 2017])

emoji2vec



Eisner et al. (2016), “emoji2vec: Learning Emoji Representations from their Description”

node2vec



Grover and Leskovec (2016), “node2vec: Scalable Feature Learning for Networks”

Trained embeddings

- Word2vec
  https://code.google.com/archive/p/word2vec/

- Glove
  http://nlp.stanford.edu/projects/glove/

- Levy/Goldberg dependency embeddings
  https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings/