Natural Language Processing

Info 159/259

Lecture 2: Lexical Semantics, Word Embeddings

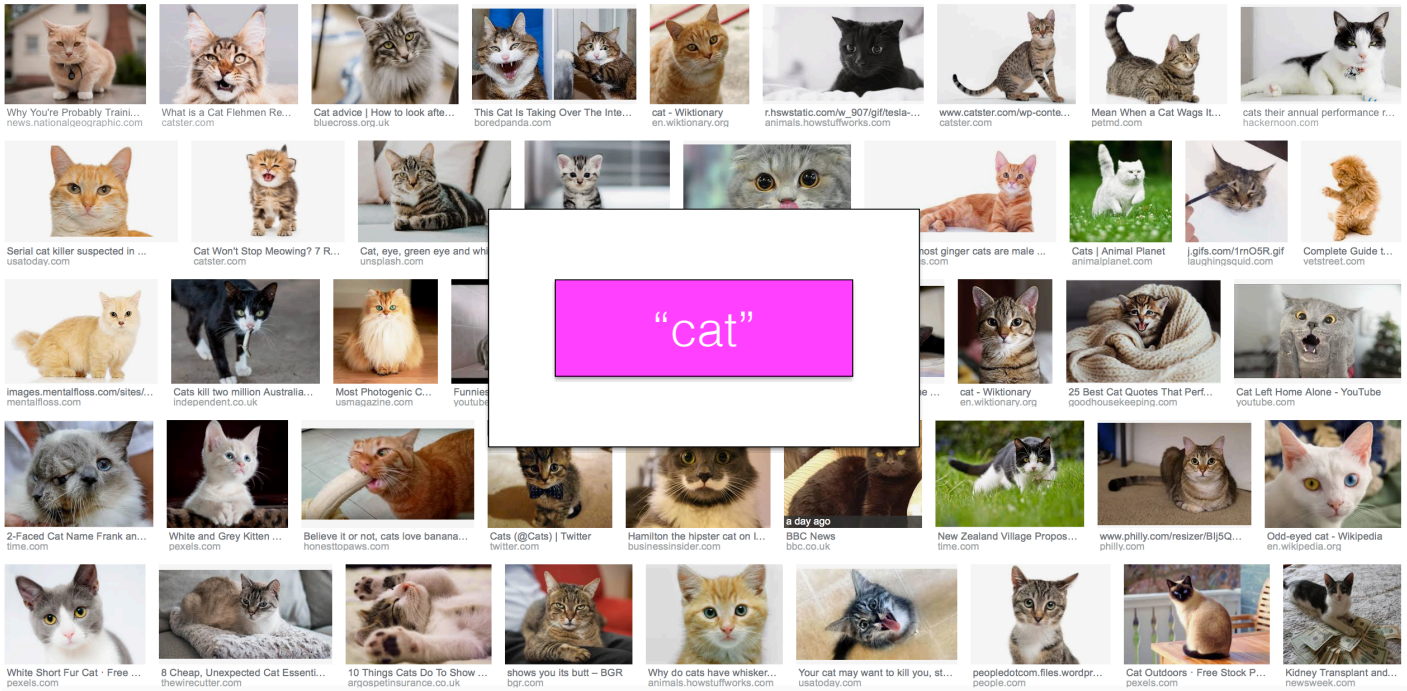
Many slides & instruction ideas borrowed from:

David Bamman, Mohit Iyyer & Kemal Oflazar

Logistics

- Office Hours to start tomorrow (Jan 23)
- Quiz 1:Out this Friday (due Mon 29th).
- Homework 1: Out next Monday

Words as dimensionality reduction



https://brenocon.com/blog/2012/07/the-60000-cat-deep-belief-networks-make-less-sense-for-language-than-vision/

Words: types and tokens

- Type = abstract descriptive concept
- Token = instantiation of a type

To be or not to be

6 tokens (to, be, or, not, to, be)
4 types (to, be, or, not)

- Types = the vocabulary; the unique tokens.

Words: types and tokens

- Type = abstract descriptive concept
- Token = instantiation of a type

How can we use types and tokens to measure vocabulary richness?

Whitespace

```
text.split(“ ”)
```

- As much mud in the streets as if the waters had but newly retired from the face of the earth, and it would not be wonderful to meet a Megalosaurus, forty feet long or so, waddling like an elephantine lizard up Holborn Hill.

Whitespace

```
text.split(“ ”)
```

- As much mud in the streets as if the waters had but newly retired from the face of the earth, and it would not be wonderful to meet a Megalosaurus, forty feet long or so, waddling like an elephantine lizard up Holborn Hill.

what do we lose with whitespace tokenization?

368	earth
135	earth,
68	earth.
26	earth
24	earth.
18	earth."
16	earth;
14	earth,
9	earth's
5	earth!"
5	earth!
4	earth;
4	earth,"
3	earth.”
3	earth?
3	earth!"

2	earth--to
2	earth--if
2	earth--and
2	earth:
2	earth,'
1	earth-worms,
1	earth-worm.
1	earth--which
1	earth--when
1	earth--something
1	earth-smeared,
1	earth-scoops,
1	earth's
1	earth--oh,

“earth” in sample of 100 books (Project Gutenberg)

Punctuation

- We typically don't want to just strip all punctuation
  - Punctuation signals boundaries (sentence, clausal boundaries, parentheticals)
  - Some punctuation communicate emotions, tone, like exclamation points (!) and question marks (?)
  - Emoticons are strong signals of e.g. sentiment

Regular expressions

- Most tokenization algorithms (for languages typically delimited by whitespace) use regular expressions to segment a string into discrete tokens.

Exploring Tokenization

tinyurl.com/5n8nvhfm

```
import nltk
tokens=nltk.word_tokenize(text)
```

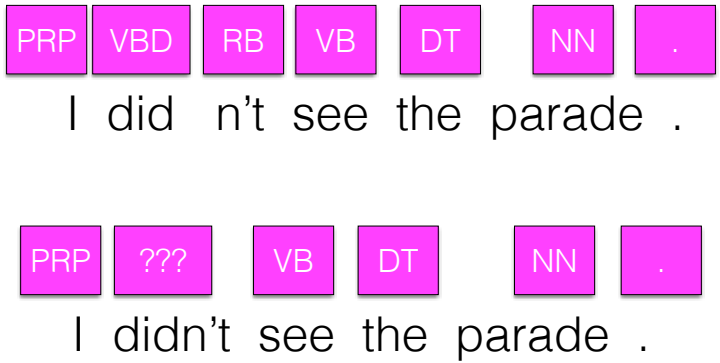
Tokenizes following the conventions of the Penn Treebank:

- punctuation split from adjoining words
- double quotes (“) changes to forward/backward quotes based on on their location in word (``the”)
- verb contractions + ’s split into separate tokens: (`did_n’t`, `children_’s`)

ftp://ftp.cis.upenn.edu/pub/treebank/public_html/tokenization.html

```
import nltk
tokens=nltk.word_tokenize(text)
```

Penn Treebank tokenization is important because a lot of downstream NLP is trained on annotated data that uses Treebank tokenization!



Sentence segmentation

- Word tokenization presumes a preprocessing step of sentence segmentation — identifying the boundaries between sentences.
- Lots of NLP operates at the level of the sentence (POS tagging, parsing), so really important to get it right.
- Harder to write regexes to delimit these, since there are many cases where the usual delimiters (periods, question marks) serve double duty.

Sentence segmentation

- “Do you want to go?” said Jane.
- Mr. Collins said he was going.
- He lives in the U.S. John, however, lives in Canada.

Sentence segmentation

- NLTK: Punkt sentence tokenizer — unsupervised method to learn common abbreviations, collocations, sentence-initial words. Can be trained on data from new domain.
  [Kiss, Tibor and Strunk, Jan (2006): Unsupervised Multilingual Sentence Boundary Detection (*Computational Linguistics*)]

- spaCy: Relies on dependency parsing to find sentence boundaries.

```
import spacy
nlp = spacy.load('en_core_web_sm')
doc=nlp(text)
for sent in doc.sents:
    for token in sent:
        print(token.text)
```

Stemming and lemmatization

- Many languages have some inflectional and derivational morphology, where similar words have similar forms:

  organizes, organized, organizing

- Stemming and lemmatization reduce this variety to a single common base form.

Stemming

- Heuristic process for chopping off the inflected suffixes of a word

organizes, organized, organizing → organ

- Porter Stemmer: Sequence of rules for removing suffixes from words

Lemmatization

- Using morphological analysis to return the dictionary form of a word (the entry in a dictionary you'd find all forms under)

organizes, organized, organizing → organize

```
import spacy
nlp = spacy.load('en_core_web_sm')
lemmas=[token.lemma_ for token in nlp(text)]
```

"TOM!" No answer. "TOM!" No answer. "What's gone with that boy, I wonder? You TOM!" No answer. The old lady pulled her spectacles down and looked over them about the room; then she put them up and looked out under them. She seldom or never looked *through* them for so small a thing as a boy; they were her state pair, the pride of her heart, and were built for "style," not service--she could have seen through a pair of stove-lids just as well. She looked perplexed for a moment, and then said, not fiercely, but still loud enough for the furniture to hear: "Well, I lay if I get hold of you I'll--" She did not finish, for by this time she was bending down and punching under the bed with the broom, and so she needed breath to punctuate the punches with. She resurrected nothing but the cat. "I never did see the beat of that boy!" She went to the open door and stood in it and looked out among the tomato vines and "jimpson" weeds that constituted the garden. No Tom. So she lifted up her voice at an angle calculated for distance and shouted: "Y-o-u-u TOM!" There was a slight noise behind her and she turned just in time to seize a small boy by the slack of his roundabout and arrest his flight. "There! I might 'a' thought of that closet. What you been doing in there?" "Nothing." "Nothing! Look at your hands. And look at your mouth. What *is* that truck?" "I don't know, aunt."

You TOM!" No answer. The old lady pulled her spectacles down and looked over them about the room; then she put them up and looked out under them. She seldom or never looked *through* them for so small a thing as a boy; they were her state pair, the pride of her heart, and were built for "style," not service--she could have seen through a pair of stove-lids just as well. She looked perplexed for a moment, and then said, not fiercely, but still loud enough for the furniture to hear: "Well, I lay if I get hold of you I'll--" She did not finish, for by this time she was bending down and punching under the bed with the broom, and so she needed breath to punctuate the punches with. She resurrected nothing but the cat. "I never did see the beat of that boy!" She went to the open door and stood in it and looked out among the tomato vines and "jimpson" weeds that constituted the garden. No Tom. So she lifted up her voice at an angle calculated for distance and shouted: "Y-o-u-u TOM!" There was a slight noise behind her and she turned just in time to seize a small boy by the slack of his roundabout and arrest his flight. "There! I might 'a' thought of that closet. What you been doing in there?" "Nothing." "Nothing! Look at

Lexical semantics

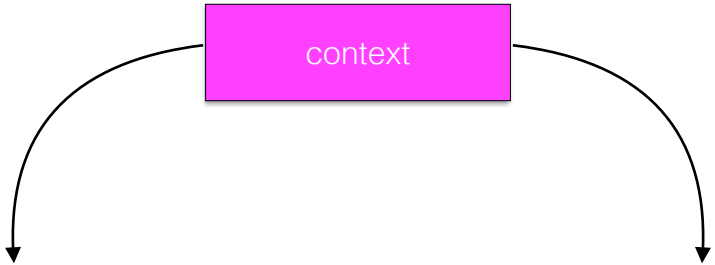
“You shall know a word by the company it keeps”

[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
- There are many other similarity metrics: Jaccard, Dice, etc.

	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

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.

contexts

term	the	a	red	eats	stab	happy	in	cloud	for
knife	74	86	4		13		21		7
dog	65	58	1	6		7	17	1	3
sword	91	81	3		8		14		5
love	45	1		1		12	54	2	13
like	31	17				11	8	7	18

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

contexts

term

	the	a	red	eats	stab	happy	in	cloud	for
knife	74	86	4		13		21		7
dog	65	58	1	6		7	17	1	3
sword	91	81	3		8		14		5
love	45	1		1		12	54	2	13
like	31	17				11	8	7	18

IDF	0	0	0.51	0.92	0.92	0.51	0	0.51	0

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?

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

	the	a	red	eats	stab	happy	in	cloud	for	*total*
knife	74	86	4		13		21		7	205
dog	65	58	1	6		7	17	1	3	158
sword	91	81	3		8		14		5	202
love	45	1		1		12	54	2	13	128
like	31	17				11	8	7	18	92
total	306	243	8	7	21	30	114	10	46	*785*

$$\text{PMI}(w = \text{sword}, c = \text{stab}) = \log_2 \frac{P(w = \text{sword}, c = \text{stab})}{P(w = \text{sword})\ P(c = \text{stab})} = \log_2 \frac{\frac{8}{785}}{\frac{202}{785} \times \frac{21}{785}}$$

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

Extrinsic Evaluation

- Extrinsic evaluation is about down-stream impact.
- e.g., Evaluate word vectors on an NLP task and see its impact (compared with another model).

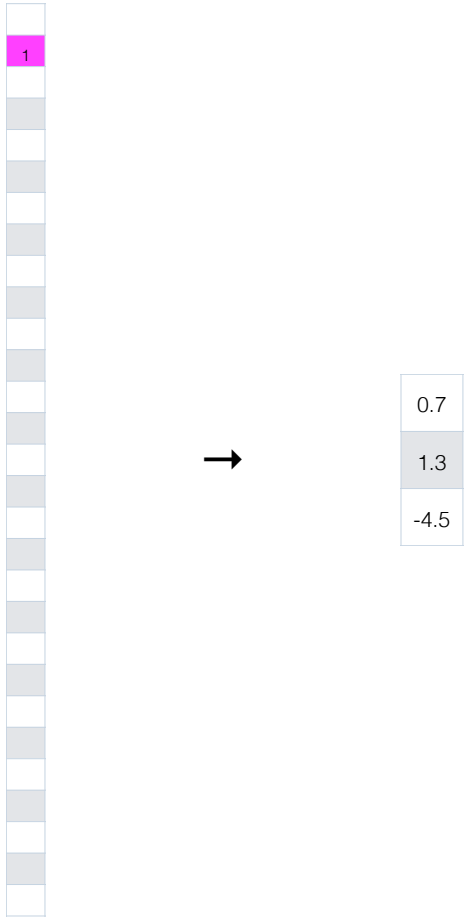
Sparse vectors

“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



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

Word2vec 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

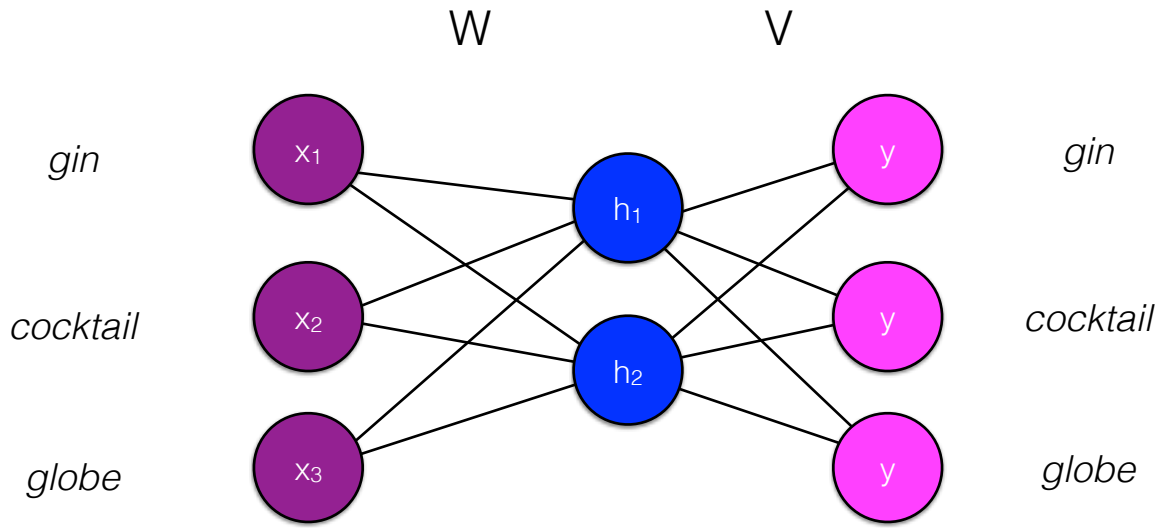
…	…
the	1
a	0
an	0
for	0
in	0
on	0
dog	0
cat	0
…	…

the is a point in V-dimensional space

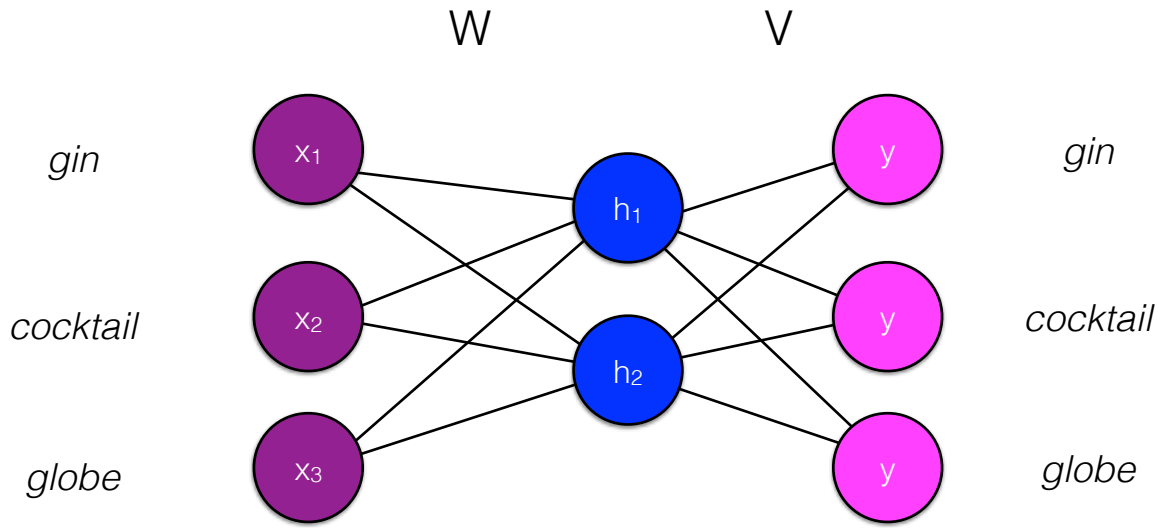
the

4.1
-0.9

the is a point in 2-dimensional space



	x	W		V			y
gin	0	-0.5	1.3	4.1	0.7	0.1	1
cocktail	1	0.4	0.08	-0.9	1.3	0.3	0
globe	0	1.7	3.1				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

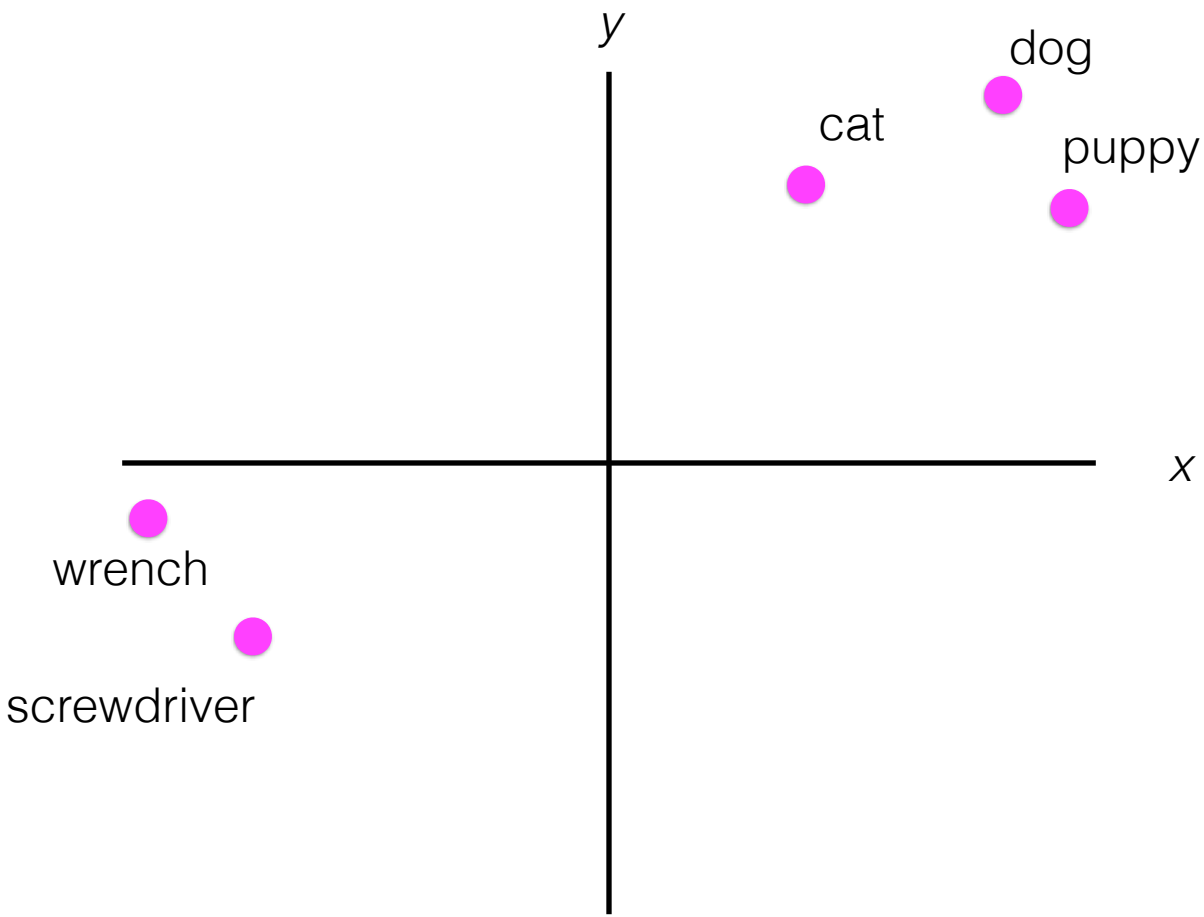
- Similarly, V has one H-dimensional vector for each element in the vocabulary (for the words that are being predicted)

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

	1	2	3	4	…	50
the	0.418	0.24968	-0.41242	0.1217	…	-0.17862
,	0.013441	0.23682	-0.16899	0.40951	…	-0.55641
.	0.15164	0.30177	-0.16763	0.17684	…	-0.31086
of	0.70853	0.57088	-0.4716	0.18048	…	-0.52393
to	0.68047	-0.039263	0.30186	-0.17792	…	0.13228
…	…	…	…	…	…	…
chanty	0.23204	0.025672	-0.70699	-0.04547	…	0.34108
kronik	-0.60921	-0.67218	0.23521	-0.11195	…	0.85632
rolonda	-0.51181	0.058706	1.0913	-0.55163	…	0.079711
zsombor	-0.75898	-0.47426	0.4737	0.7725	…	0.84014
sandberger	0.072617	-0.51393	0.4728	-0.52202	…	0.23096

https://nlp.stanford.edu/projects/glove/


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.

Analogical inference

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

apple - apples ≈ car - cars

king - man + woman ≈ queen

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

Science AAAS

Home News Journals Topics Careers

Science Science Advances Science Immunology Science Robotics Science Signaling Science Translational Medicine

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

Peer Reviewed
see details

Bias

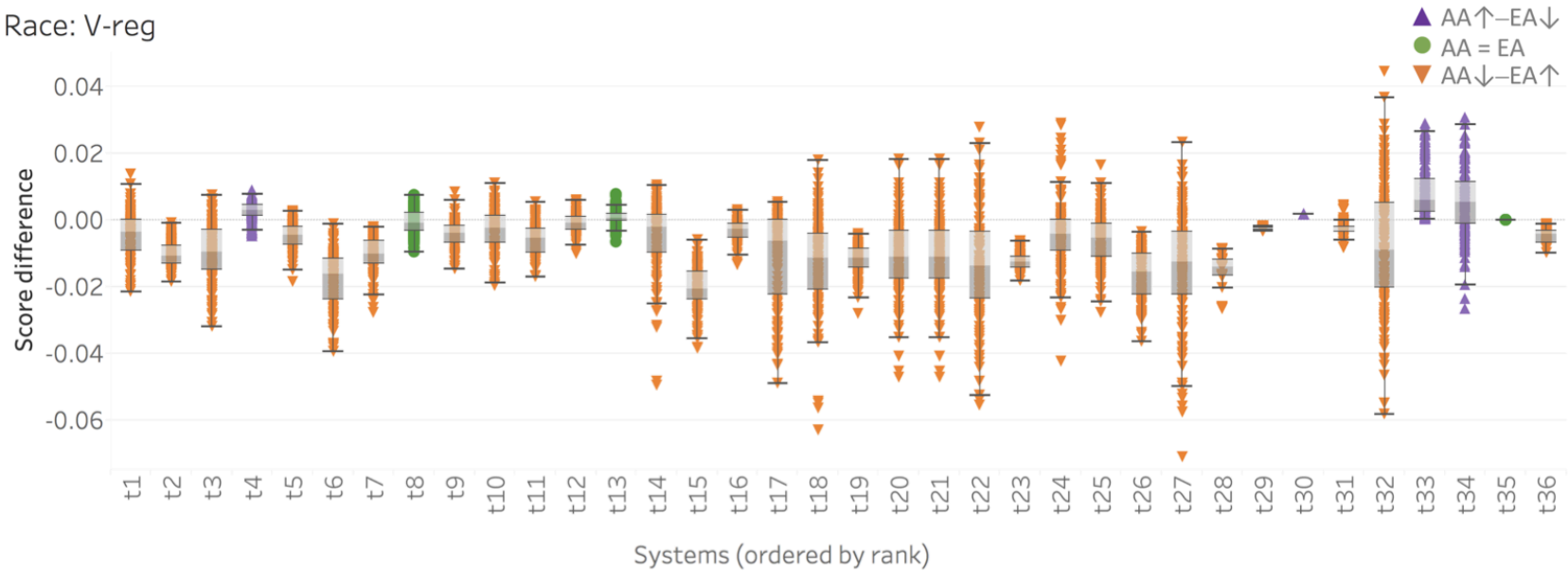
- Allocational harms: automated systems allocate resources unfairly to different groups (access to housing, credit, parole).
- Representational harms: automated systems represent one group less favorably than another (including demeaning them or erasing their existence).

Blodgett et al. (2020), “Language (Technology) is Power: A Critical Survey of “Bias” in NLP”

Representations

- **Pleasant**: caress, freedom, health, love, peace, cheer, friend, heaven, loyal, pleasure, diamond, gentle, honest, lucky, rainbow, diploma, gift, honor, miracle, sunrise, family, happy, laughter, paradise, vacation.
- **Unpleasant**: abuse, crash, filth, murder, sickness, accident, death, grief, poison, stink, assault, disaster, hatred, pollute, tragedy, bomb, divorce, jail, poverty, ugly, cancer, evil, kill, rotten, vomit.

- Embeddings for African-American first names are closer to “unpleasant” words than European names (Caliskan et al. 2017)



- Sentiment analysis over sentences containing African-American first names are more negative than identical sentences with European names

Kiritchenko and Mohammad (2018), "Examining Gender and Race Bias in Two Hundred Sentiment Analysis Systems"

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

Exploring Word Embeddings

tinyurl.com/4sea2tnv

Next time:

Text Classification/Sentiment Analysis
via Logistics Regression