NLP and Deep Learning 1: Human Language & Word Vectors

Christopher Manning

Stanford University

@chrmanning

2015 Deep Learning Summer School, Montreal

My plan

Morning:

Learn a little about language

Word embeddings

Afternoon:

Input-dependent Tree Recursive Neural Networks

The nature of human language

What's special about human language?

Until now, deep learning has concentrated on the analysis of natural, raw signals:

- Object recognition in images
- Bird calls
- Detecting cancer

Principally, any meaning is in the eye of the beholder, based on their analysis of the signal

What's special about human language?

A human language is instead a system specifically **constructed to convey the speaker/writer's meaning**

A human language is a **discrete/symbolic/categorical signaling system**

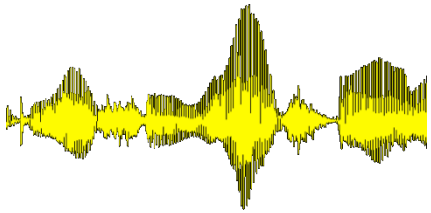
- With very minor exceptions for expressive signaling ("I loooove it." "Whoomppaaa")
- Presumably because of greater information-theoretic signaling reliability
- Symbols are not just an invention of classical AI!

What's special about human language?

The categorical symbols of a language can be encoded as a signal for communication in several ways:

- Sound
- Gesture
- Images (writing)

The symbol is invariant is invariant across different encodings!

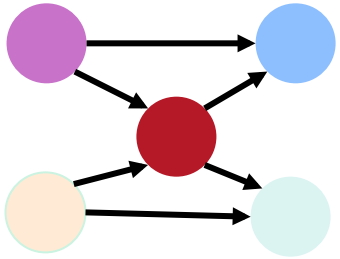


What's special about human language?

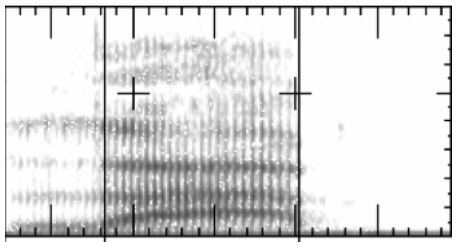
A human language is a **discrete/symbolic/categorical signaling system**

- Despite brain encoding as a continuous pattern of activation and transmission via continuous signals of sound/vision

High-dimensional, symbolic encoding of words creates a problem for neural network processing – sparsity!

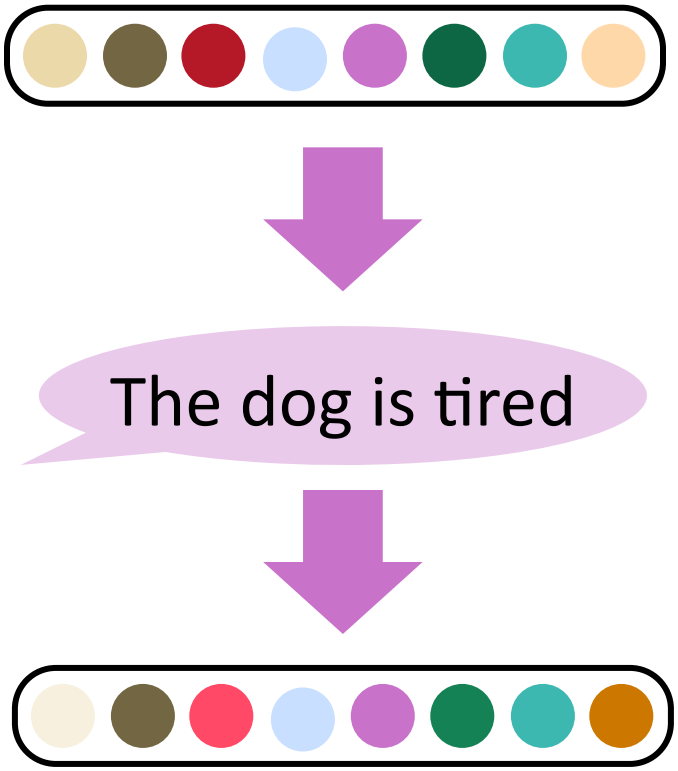


lab

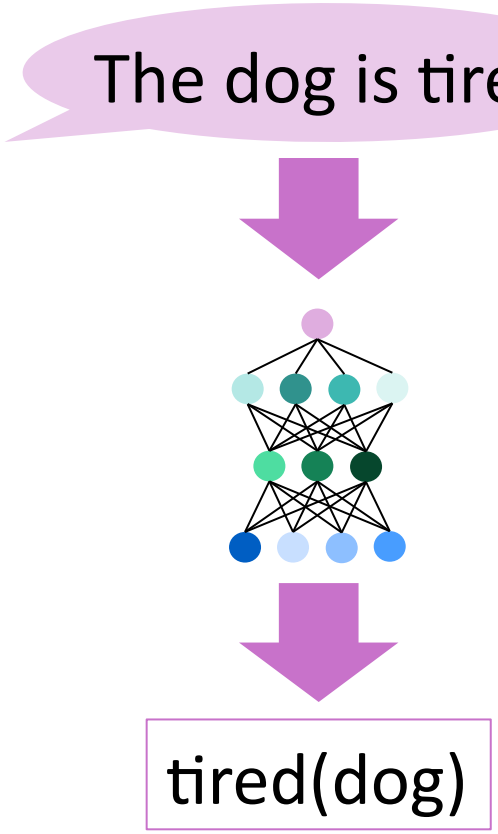


From symbolic to distributed and back

Working of our brains



Usual working of our systems



Distributional word representations: Introduction

From symbolic to distributed representations

The vast majority of rule-based **and** statistical NLP work regarded words as atomic symbols: hotel, conference, walk

In vector space terms, this is a vector with one 1 and a lot of zeroes

[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

We call this a "one-hot" representation.

From symbolic to distributed representations

Its problem:

- If user searches for [Dell notebook battery size], we would like to match documents with "Dell laptop battery capacity"
- If user searches for [Seattle motel], we would like to match documents containing "Seattle hotel"

But

motel [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]^T
hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0] = 0

Our query and document vectors are **orthogonal**

There is no natural notion of similarity in a set of one-hot vectors

Distributional similarity-based representations

You can get a lot of value by representing a word by means of its neighbors

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

(J. R. Firth 1957: 11)

One of the most successful ideas of modern NLP

government debt problems turning into	banking	crises as has happened in
saying that Europe needs unified	banking	regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

Basic idea of learning neural word embeddings

We define some model that aims to predict a word based on other words in its context

Choose $\text{argmax}_w\ w \cdot ((w_{j-1} + w_{j+1})/2)$

which has a loss function, e.g.,

$J = 1 - w_j \cdot ((w_{j-1} + w_{j+1})/2)$

Unit norm vectors

We look at many samples from a big language corpus

We keep adjusting the vector representations of words to minimize this loss

With distributed, distributional representations, syntactic and semantic similarity is captured

$$take = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

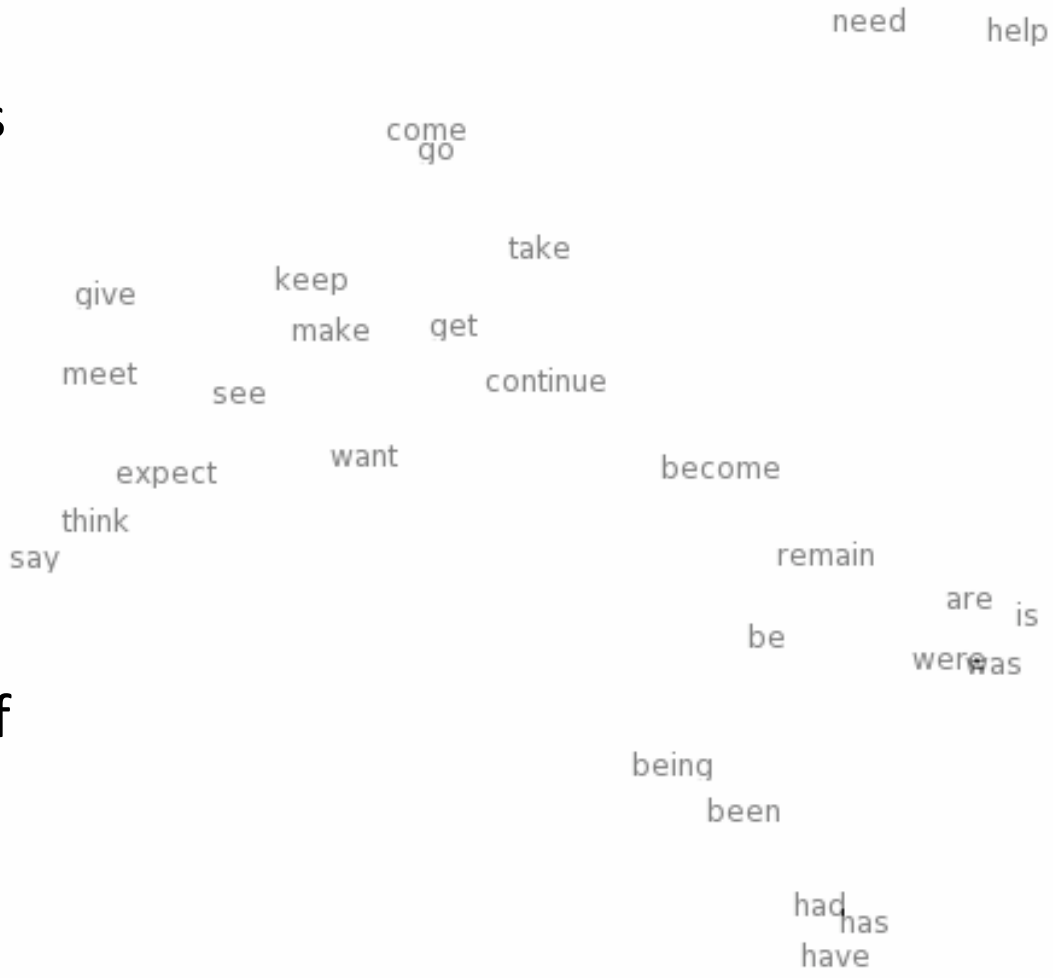
With distributed, distributional representations, syntactic and semantic similarity is captured

Distributed: A concept is represented as continuous activation levels in a number of elements

[Contrast: local]

Distributional: Meaning is represented by contexts of use

[Contrast: denotational]



Word Analogies: Word vectors capture dimensions of similarity as linear relations

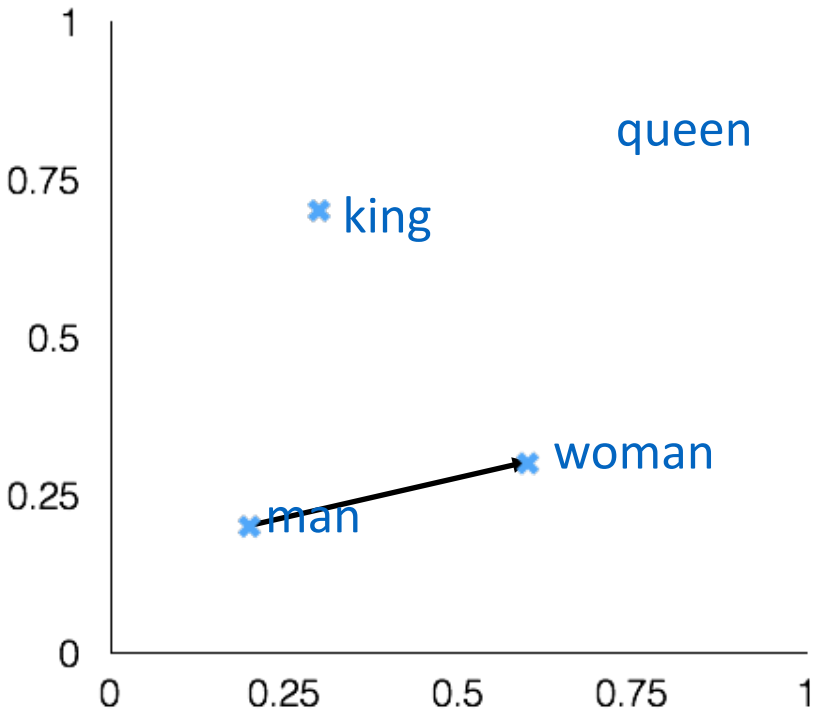
Test for linear relationships, examined by Mikolov, Yih & Zweig (2013)

a:b :: c:?

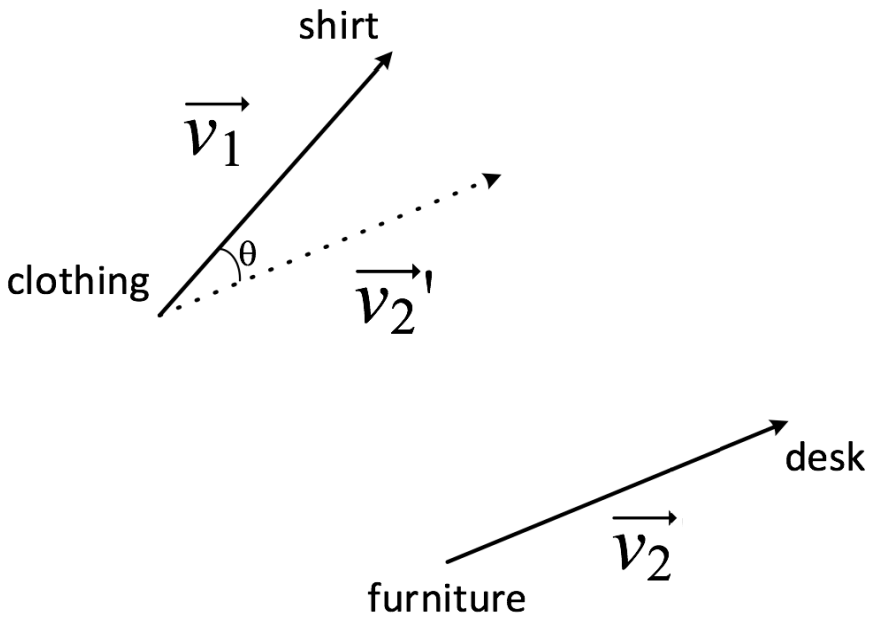
$$d = \arg\max_x \frac{(w_b - w_a + w_c)^T w_x}{||w_b - w_a + w_c||}$$

man:woman :: king:?

+	king	[0.30 0.70]
–	man	[0.20 0.20]
+	woman	[0.60 0.30]
	queen	[0.70 0.80]



Word Analogies: Word vectors capture dimensions of similarity as linear relations

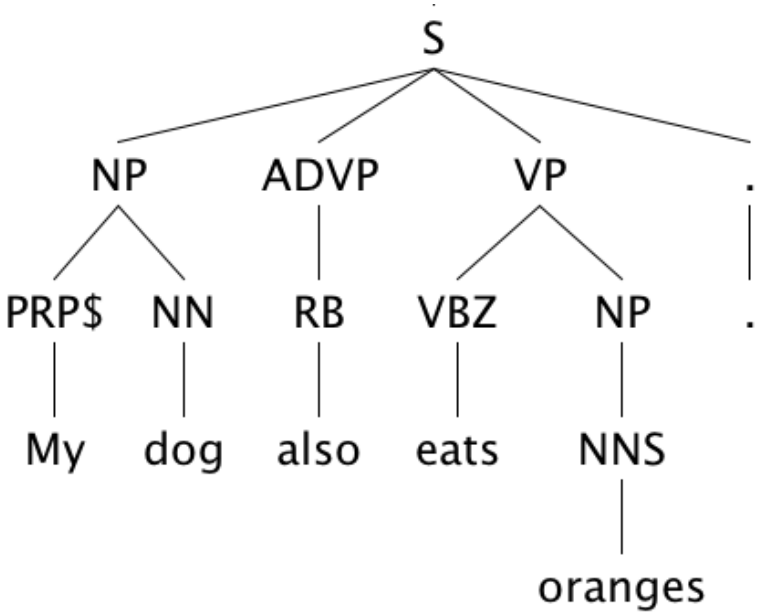


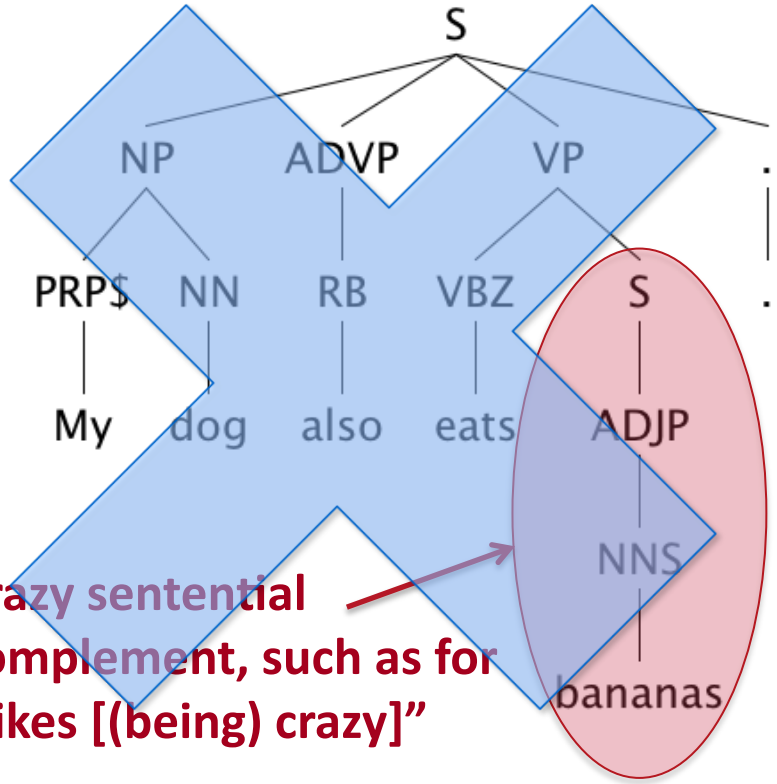
Method	Syntax % correct
LSA 320 dim	16.5 [previous best]
RNN 80 dim	16.2
RNN 320 dim	28.5
RNN 1600 dim	39.6
Method	**Semantics Spearm ρ**
UTD-NB (Rink & H. 2012)	0.230 [Semeval win]
LSA 640	0.149
RNN 80	0.211
RNN 1600	0.275 [new SOTA]

A then stunning new result by Mikolov, Yih & Zweig at NAACL 2013

Distributional representations can solve the fragility of NLP tools

Current NLP systems – here, the Stanford Parser – are incredibly fragile because of symbolic representations





Distributional word representations give great performance gains

Word representations built from contexts of use capture meaning similarity and greatly help all NLP applications

- +1.4% F_1 Dependency Parsing

  15.2% error reduction (Koo & Collins 2008)

- +3.4% F_1 Named Entity Recognition

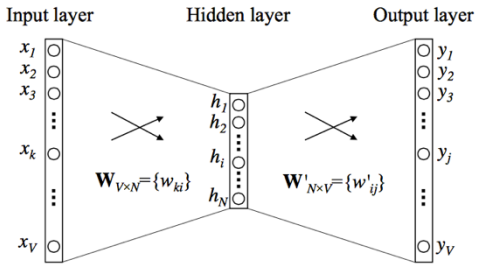
  23.7% error reduction (Stanford NER, exchange clustering)

Neural embeddings

- The first stage of most work in Deep Learning NLP is mapping word symbols to distributed vector representations
- This is often either so useful or so easy that people stop there
  - The use of word vectors was **everywhere** at the NAACL 2015 and ACL 2015 conferences this year
- But simply using word vectors is **not** deep learning
- Neural embeddings aren't that different to other distributed representations of words traditionally used in NLP for lexical semantics
- Let's try to understand both what they are and how they relate

word2vec

Word2vec recipe



word2vec

Supply text

Wait a few hours

Get a $d\times|V|$ matrix

A vector for each word

word2vec

Instead of capturing co-occurrence counts directly, predict between every word and its context

Two algorithms

1. **Skip-grams (SG)**

   Predict context words given target (position independent)

2. Continuous Bag of Words (CBOW)

   Predict target word from bag-of-words context

Two (moderately efficient) training methods

1. Hierarchical softmax
2. **Negative sampling**

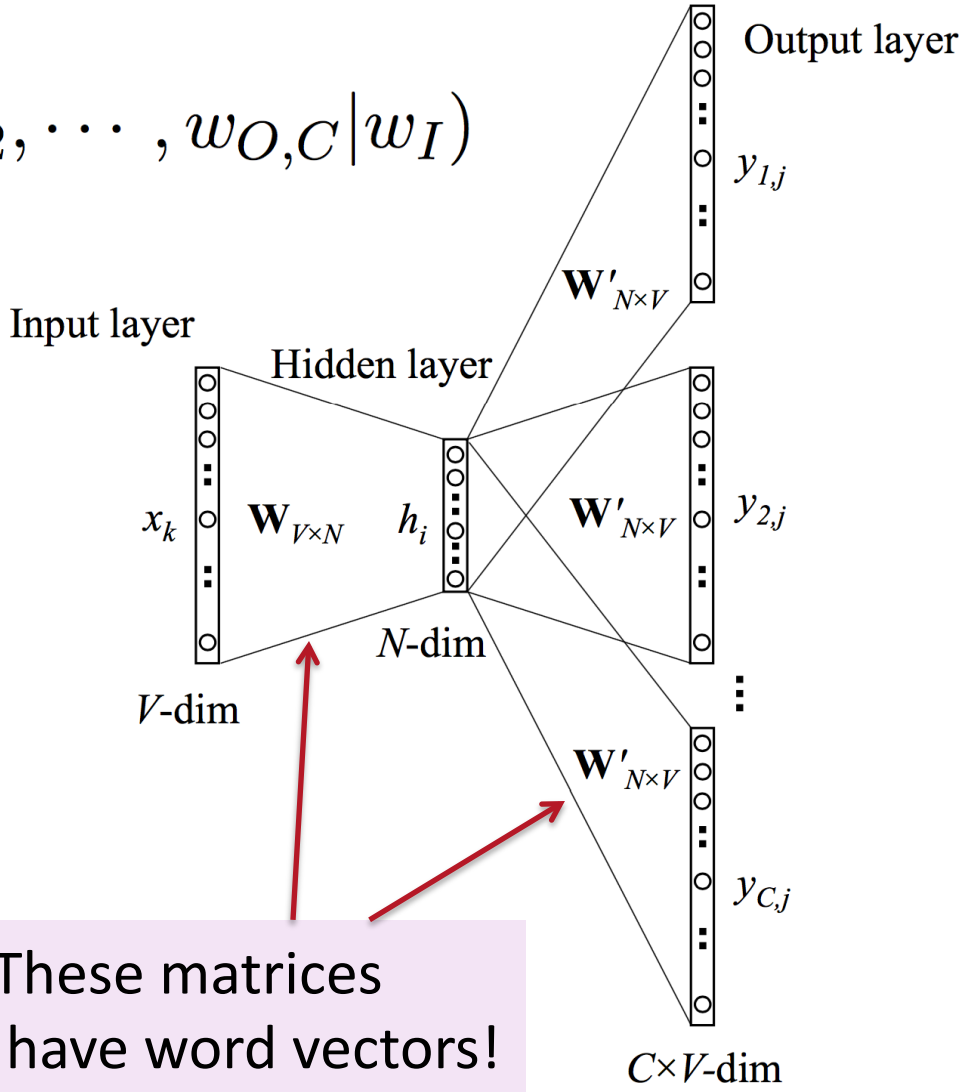
Word2vec (SkipGram)

$$E = -\log p(w_{O,1}, w_{O,2}, \cdots, w_{O,C} | w_I)$$

- Loss function is negative probability of predictions of context words
- The hidden layer simply selects a row of **W** based on the input word

$$\mathbf{h} = \mathbf{x}^T \mathbf{W} = \mathbf{W}_{(k,\cdot)}$$

- This is then mapped by another matrix to an output probability



Naïve Algorithm

$$
\begin{aligned}
\text{minimize } J &= -\log P(w^{(i-C)},\ldots,w^{(i-1)},w^{(i+1)},\ldots,w^{(i+C)}|w^{(i)}) \\
&= -\log \prod_{j=0,j\neq C}^{2C} P(w^{(i-C+j)}|w^{(i)}) \\
&= -\log \prod_{j=0,j\neq C}^{2C} P(v^{(i-C+j)}|u^{(i)}) \\
&= -\log \prod_{j=0,j\neq C}^{2C} \frac{\exp(v^{(i-C+j)T}h)}{\sum_{k=1}^{|V|}\exp(v^{(k)T}h)}
\end{aligned}
$$

Skip Gram Negative Sampling

$$P(D = 1|w, c, \theta) = \frac{1}{1 + e^{(-v_c^T v_w)}}$$

- Get positives (w,c) from data relative to their count in the data. So we will see them with P(w,c)
- Get negatives from random words. Since these are encountered relative to the frequency of w and c independently, you will see them with P(w)P(c)

Skip Gram Negative Sampling

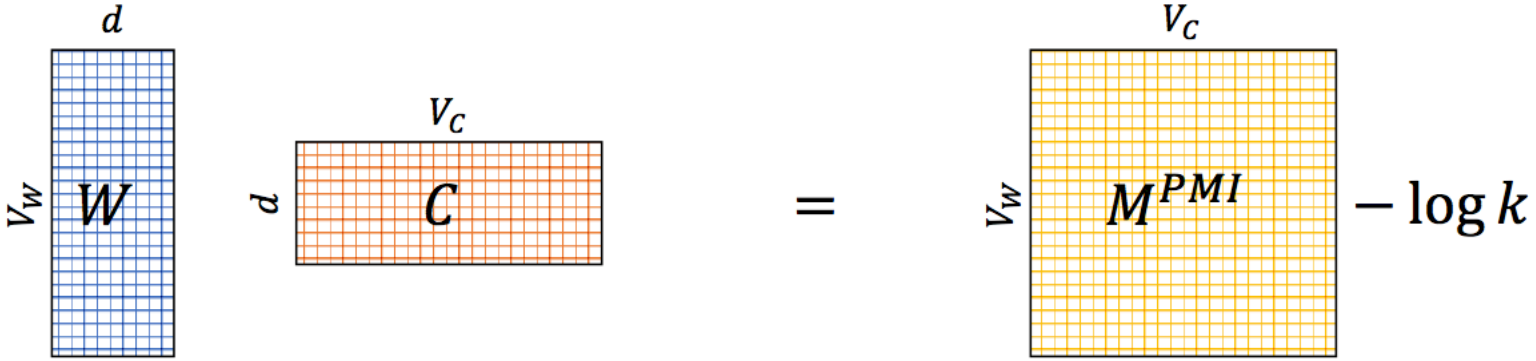
$$
\begin{aligned}
\theta &= \underset{\theta}{\operatorname{argmax}} \prod_{(w,c)\in D} P(D=1|w,c,\theta) \prod_{(w,c)\in \tilde{D}} P(D=0|w,c,\theta) \\
&= \underset{\theta}{\operatorname{argmax}} \prod_{(w,c)\in D} P(D=1|w,c,\theta) \prod_{(w,c)\in \tilde{D}} (1-P(D=1|w,c,\theta)) \\
&= \underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log P(D=1|w,c,\theta) + \sum_{(w,c)\in \tilde{D}} \log(1-P(D=1|w,c,\theta)) \\
&= \underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log \frac{1}{1+\exp(-v_c^T v_w)} + \sum_{(w,c)\in \tilde{D}} \log(1-\frac{1}{1+\exp(-v_c^T v_w)}) \\
&= \underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log \frac{1}{1+\exp(-v_c^T v_w)} + \sum_{(w,c)\in \tilde{D}} \log(\frac{1}{1+\exp(v_c^T v_w)})
\end{aligned}
$$

word2vec is a matrix factorization

- So we are basically doing binary logistic regression

$$-\log\sigma(v^{(i-C+j)}\cdot h)+\sum_{k=1}^{K}\log\sigma(\tilde{v}^{(k)}\cdot h)$$

- If no dimensionality reduction v·h = PMI(w,c)



Levy & Goldberg, NIPS 2014: Neural Word Embeddings as Implicit Matrix Factorization

Other neural embeddings work

Use distributed representations to give a language model

- Bengio (2003) NPLM
- Mnih & Hinton (2008) Hierarchical Log Bilinear Model (HLBL)

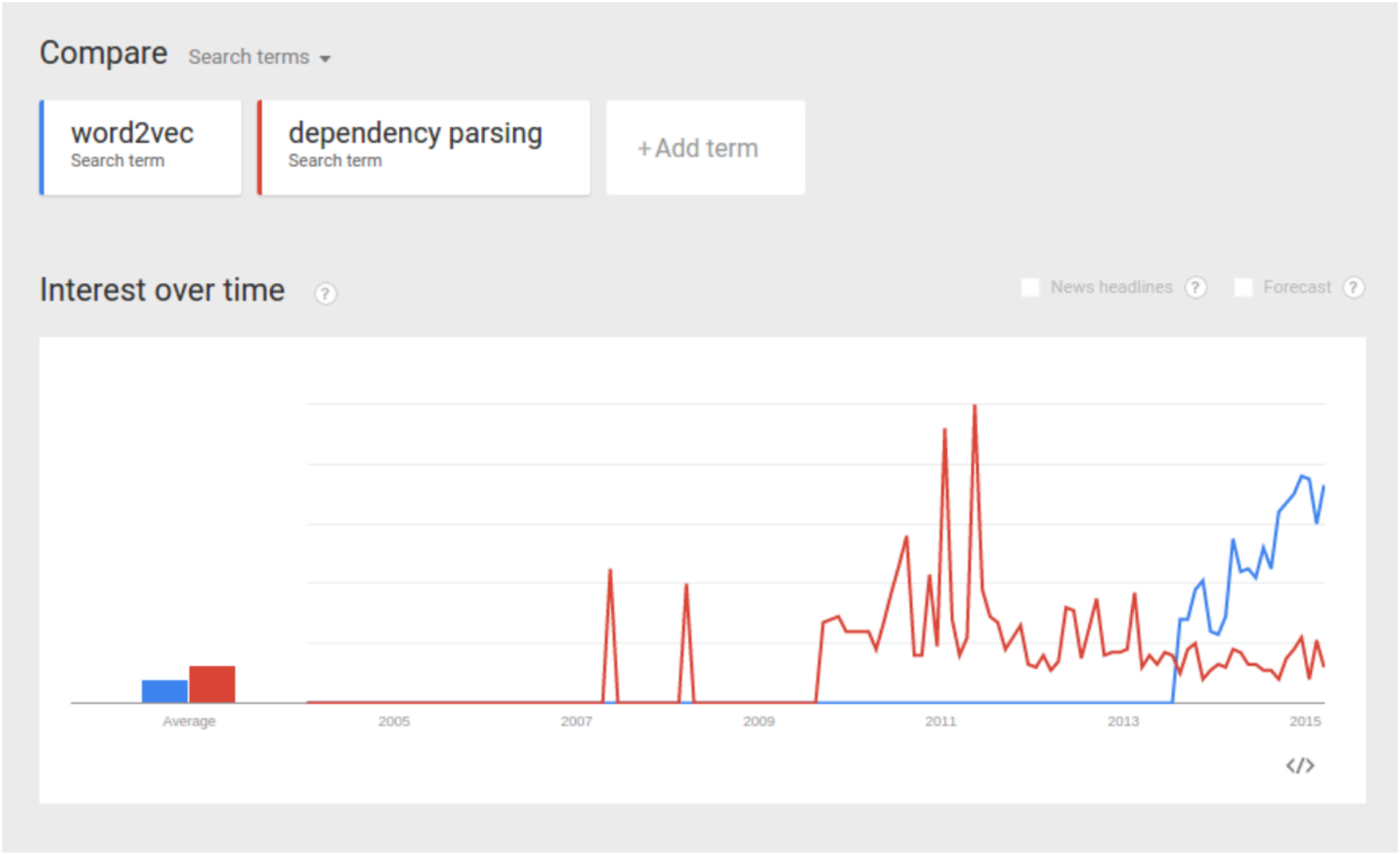
Learn representations without worrying about normalizing to probabilities

- Collobert & Weston (2008; 2011)

Make it much faster and bigger by simplifying to bilinear models:

- Mikolov et al. (2013)
- Mnih et al. (2013)

The word2vec code is fast, produces good results – and it's from Google ☺ – so everyone uses it.

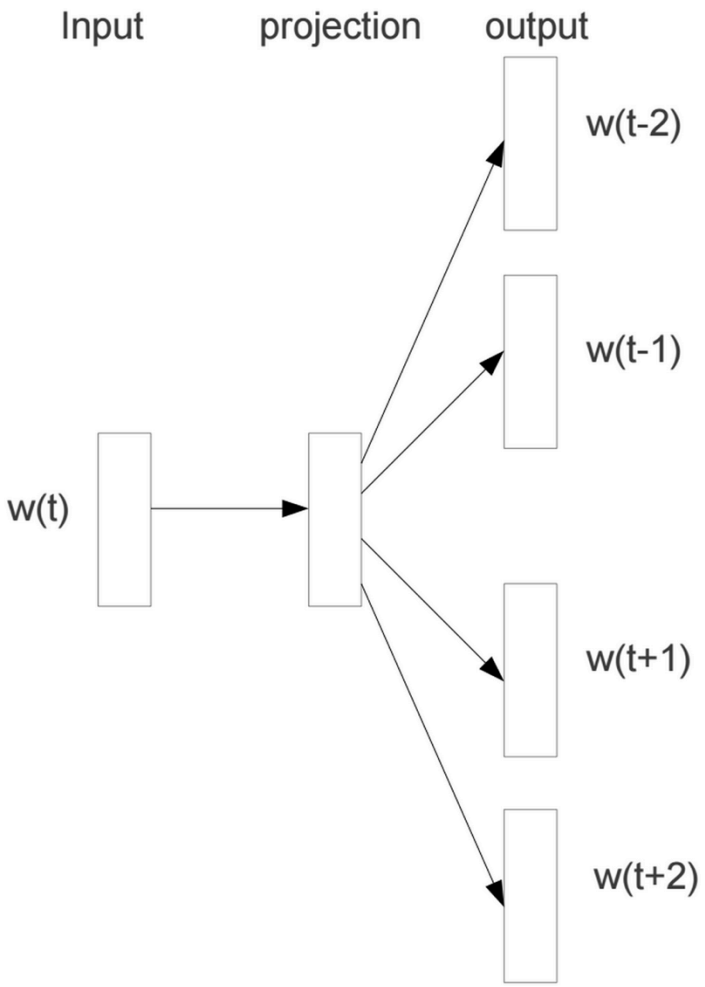

Compare
Search terms
word2vec
Search term
dependency parsing
Search term
+ Add term
Interest over time
News headlines
Forecast
Average
2005
2007
2009
2011
2013
2015

Comparison with LSA & GloVe

LSA vs. word2vec

word2vec CBOW/SkipGram: **Predict!**

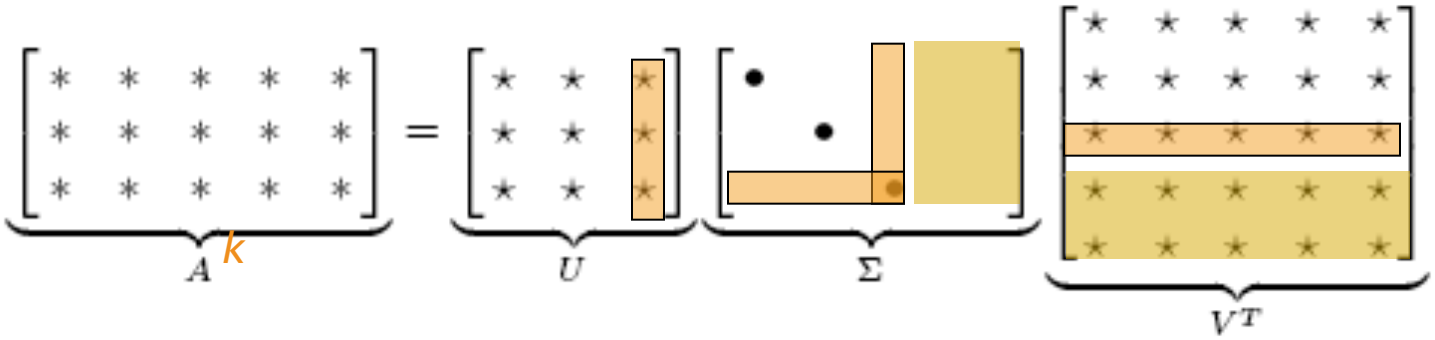
- Train word vectors to try to either:
  - Predict a word given its bag-of-words context (CBOW); or
  - Predict a context word (position-independent) from the center word
- Update word vectors until they can do this prediction well



LSA vs. word2vec

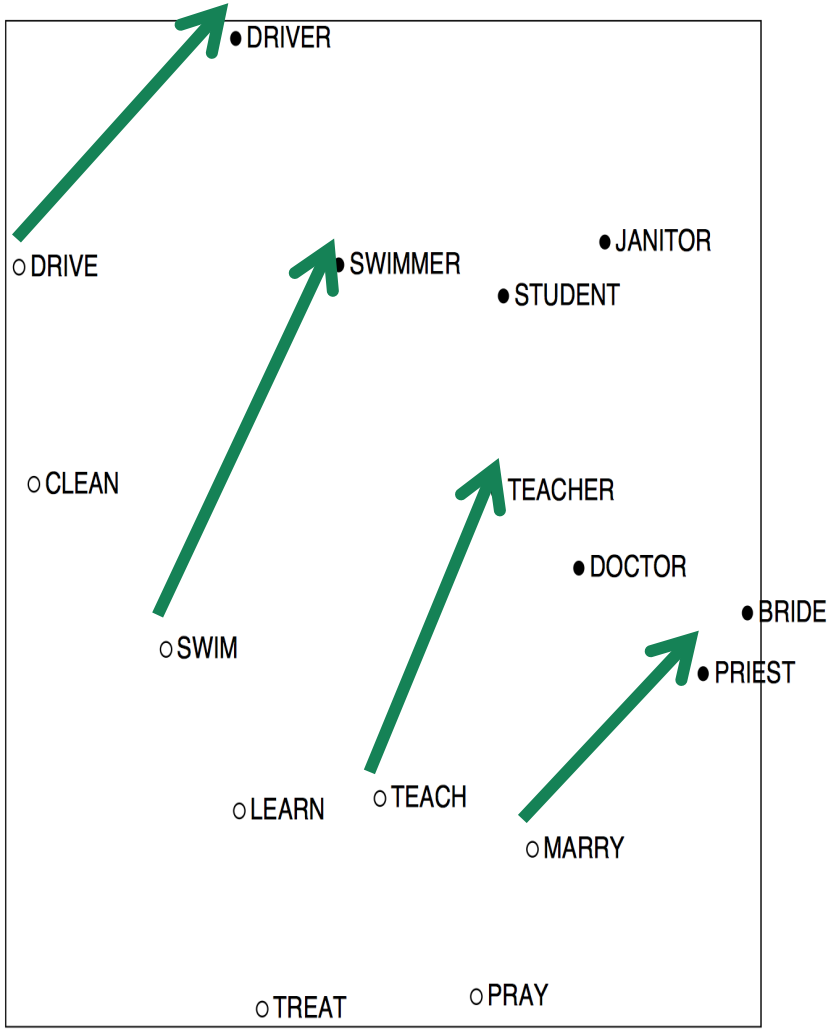
vs. LSA: **Count!**

- Factorize a (maybe weighted, maybe log scaled) term-document or word-context matrix (Schütze 1992) into $U\Sigma V^T$
- Retain only *k* singular values, in order to generalize



[Cf. Baroni: Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. ACL 2014]

COALS model (count-modified LSA)
[Rohde, Gonnerman & Plaut, ms., 2005]



Count based vs. direct prediction

LSA, HAL (Lund & Burgess),
COALS (Rohde et al),
Hellinger-PCA (Lebret & Collobert)

- Fast training
- Efficient usage of statistics
- Primarily used to capture word similarity
- Disproportionate importance given to small counts

- NNLM, HLBL, RNN, word2vec Skip-gram/CBOW, (Bengio et al; Collobert & Weston; Huang et al; Mnih & Hinton; Mikolov et al; Mnih & Kavukcuoglu)

- Scales with corpus size
- Inefficient usage of statistics
- Generate improved performance on other tasks
- Can capture complex patterns beyond word similarity

Encoding meaning in vector differences

[Pennington, Socher, and Manning, EMNLP 2014]

Crucial insight: Ratios of co-occurrence probabilities can encode meaning components

	x = solid	x = gas	x = water	x = random
$P(x\|\text{ice})$	large	small	large	small
$P(x\|\text{steam})$	small	large	large	small
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	large	small	~1	~1

Encoding meaning in vector differences

[Pennington, Socher, and Manning, EMNLP 2014]

Crucial insight: Ratios of co-occurrence probabilities can encode meaning components

	x = solid	x = gas	x = water	x = fashion
$P(x\|\text{ice})$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(x\|\text{steam})$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(x\|\text{ice})}{P(x\|\text{steam})}$	8.9	8.5×10^{-2}	1.36	0.96

Encoding meaning in vector differences

- Q: How can we capture ratios of co-occurrence probabilities as meaning components in a word vector space?

A: Log-bilinear model: $w_i \cdot w_j = \log P(i|j)$

with vector differences $w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$

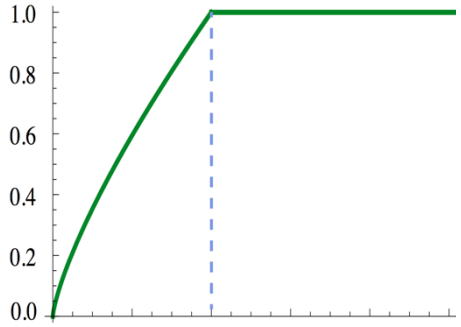
GloVe: A new model for learning representations [Pennington et al., EMNLP

$$w_i \cdot w_j = \log P(i|j)$$

$$w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$$

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2$$

$f \sim$



Word similarities

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

leptodactylidae

rana

eleutherodactylus

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

Word Analogies
[Mikolov et al., 2012, 2013]

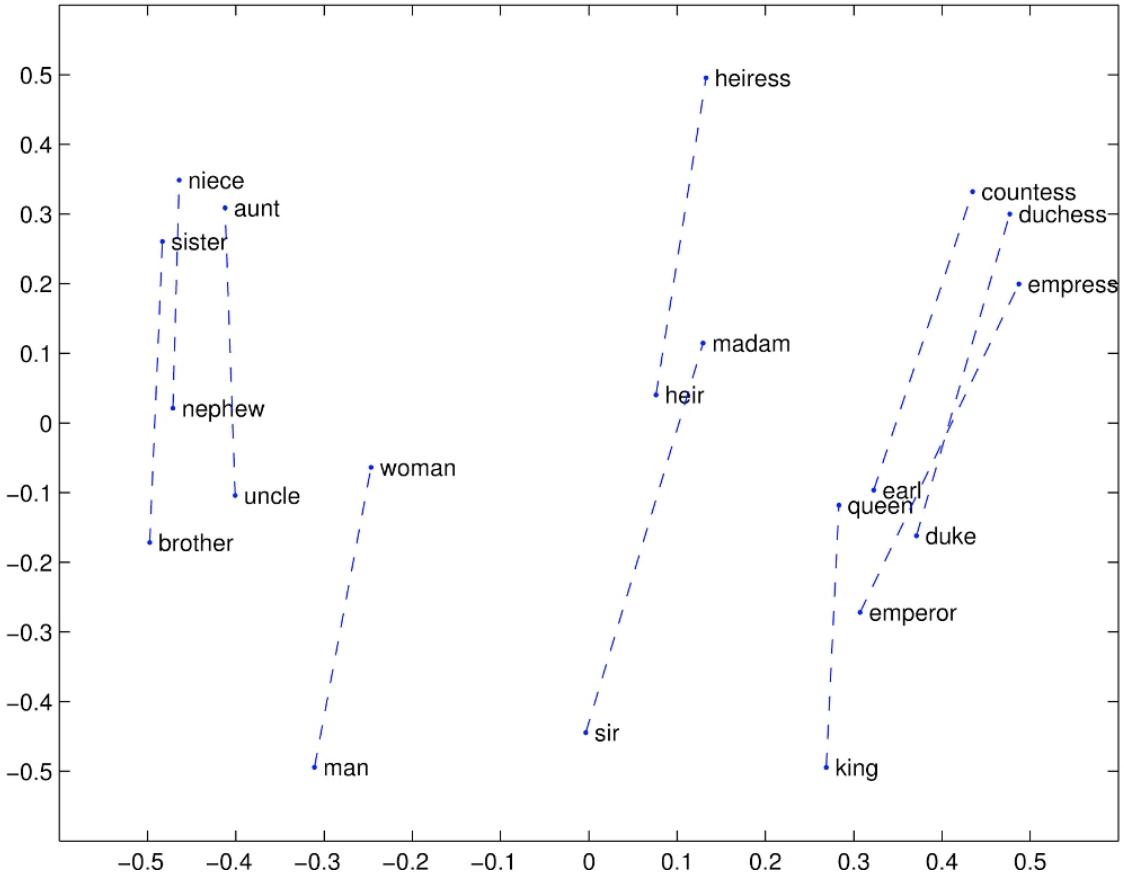
Task: predict the last column

Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

Word analogy task
[Mikolov, Yih & Zweig 2013a]

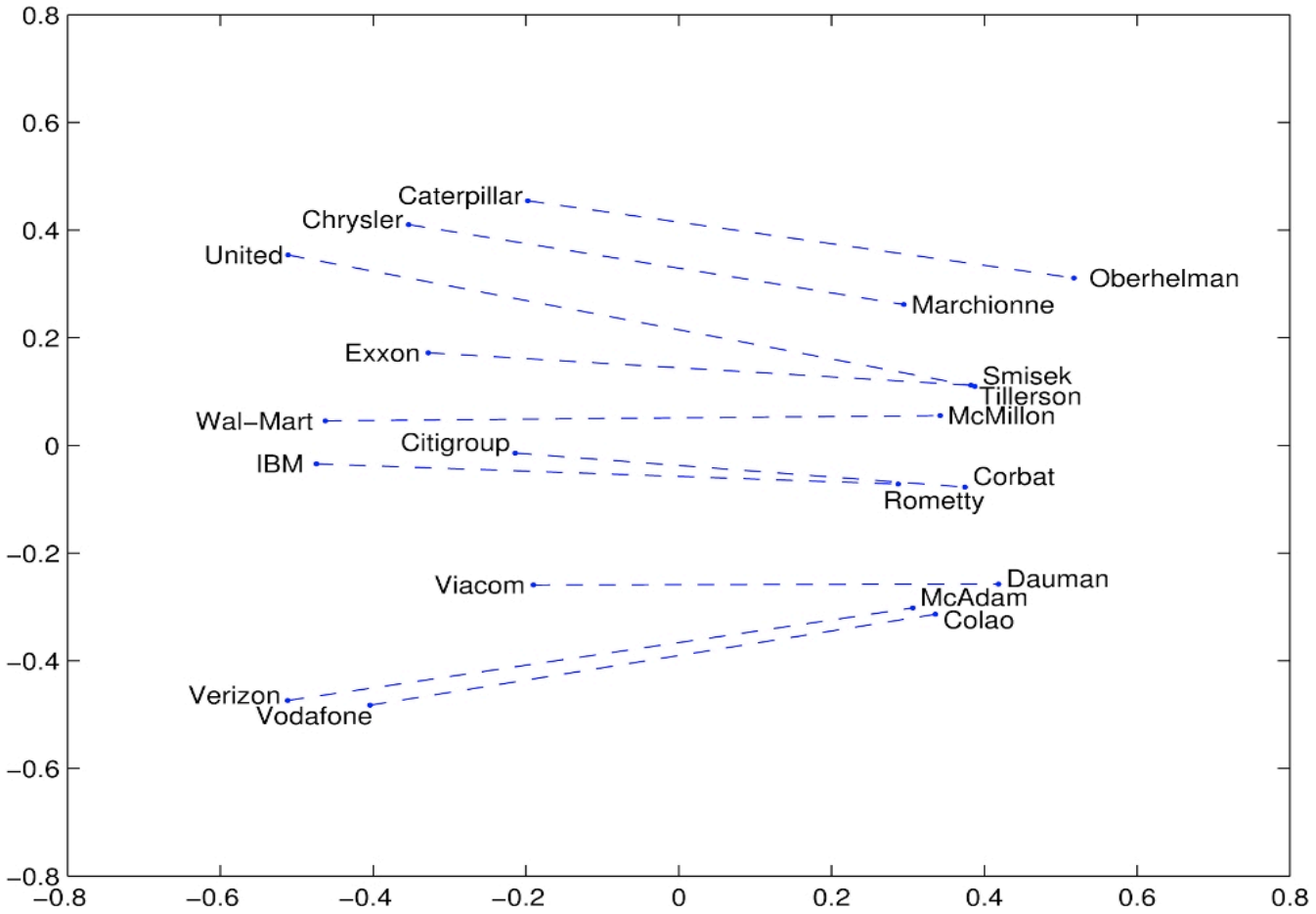
Model	Dimensions	Corpus size	Performance (Syn + Sem)
CBOW (Mikolov et al. 2013b)	300	1.6 billion	36.1

Glove Visualizations

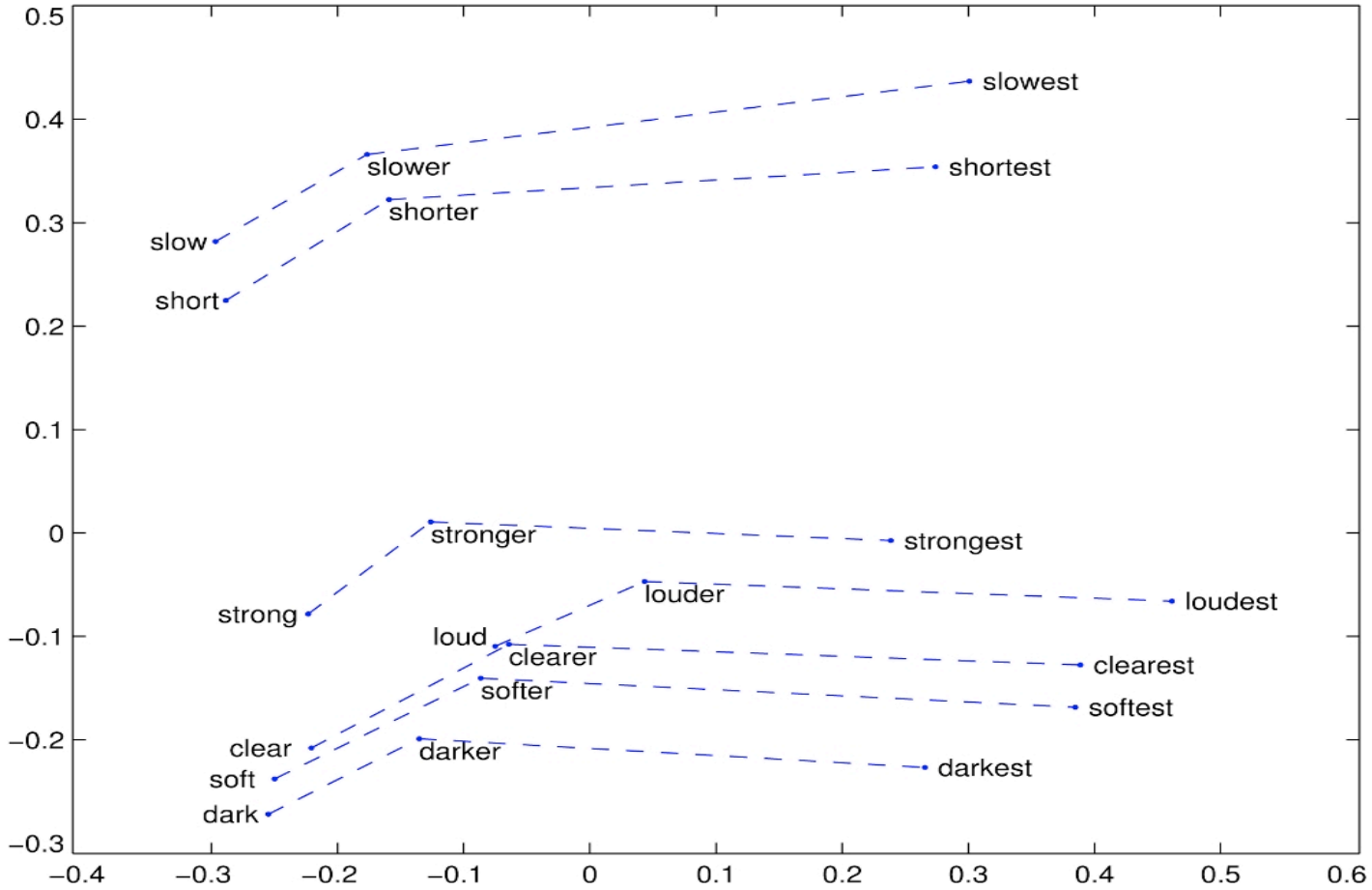


http://nlp.stanford.edu/projects/glove/

Glove Visualizations: Company - CEO

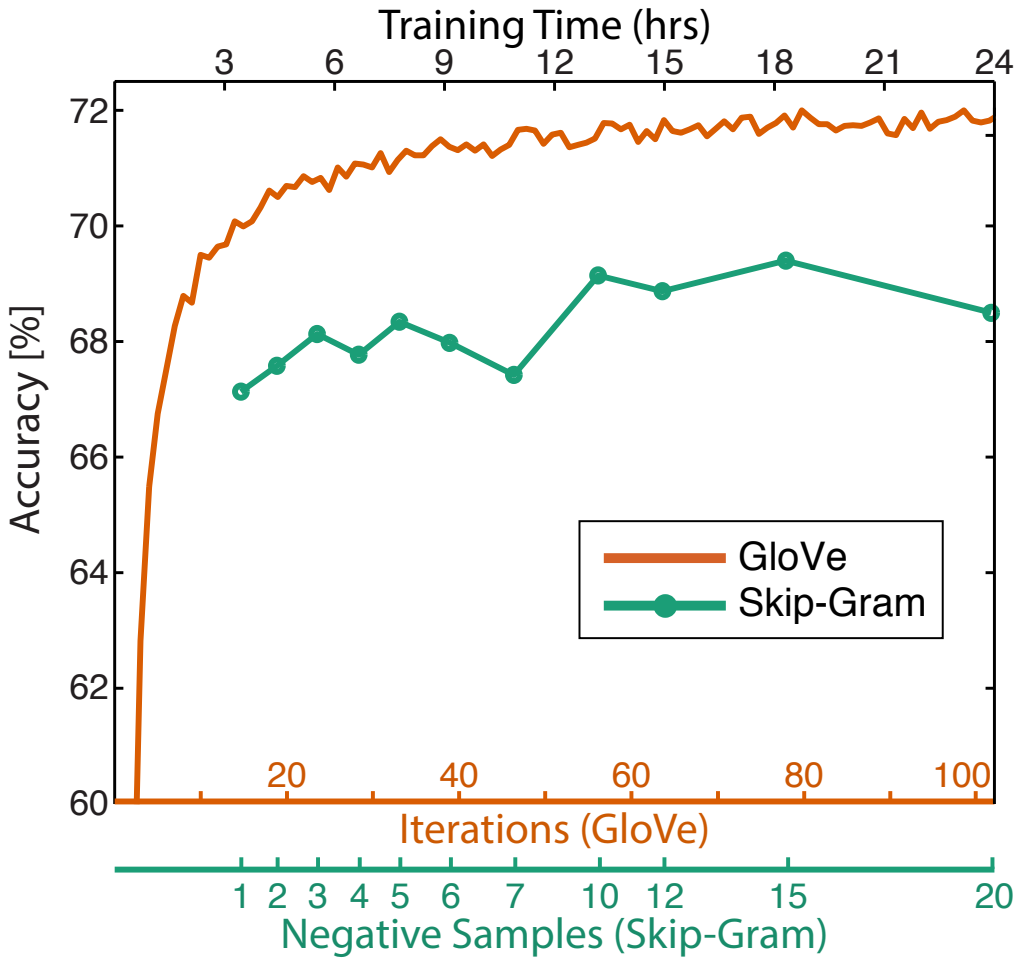


Glove Visualizations: Superlatives

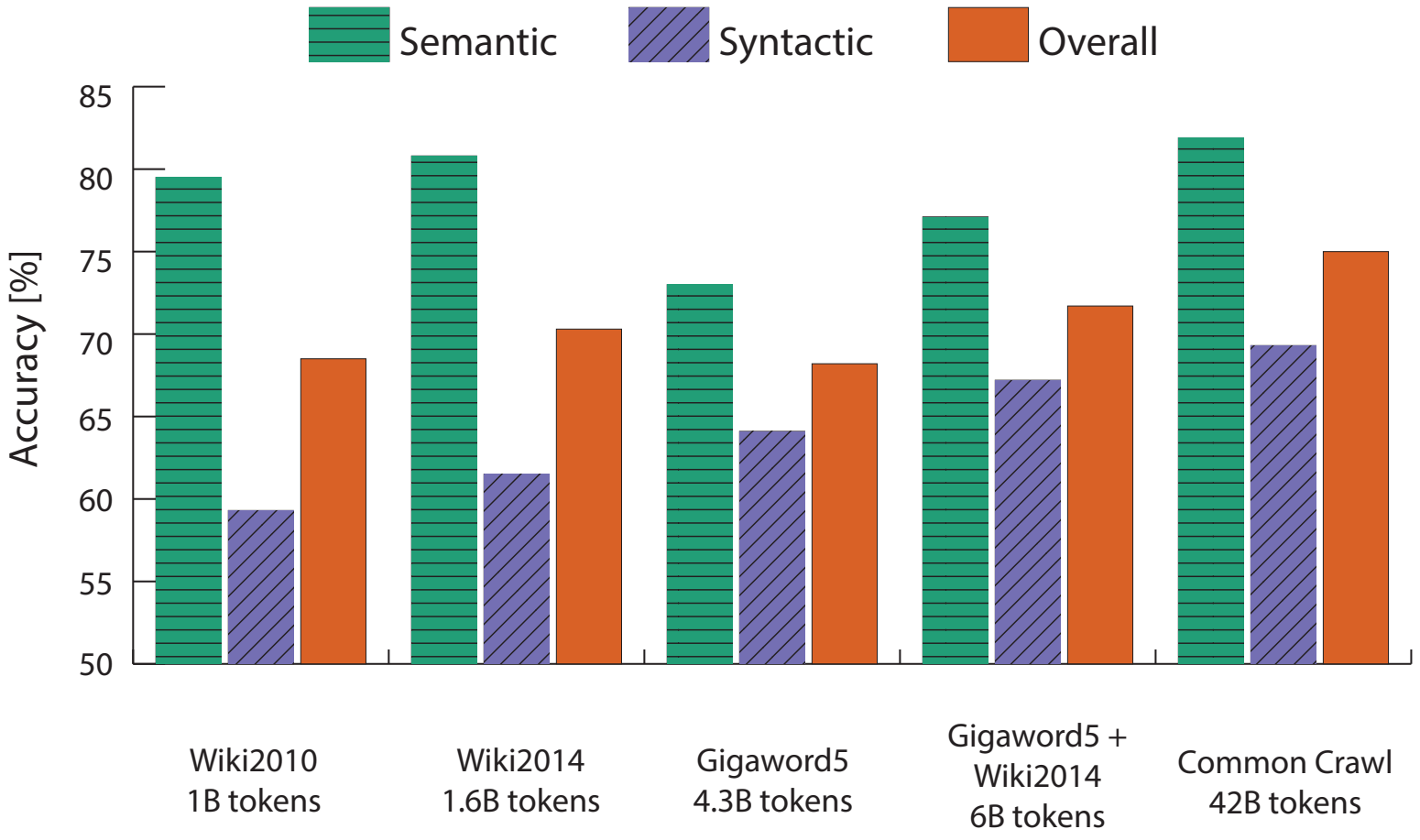


Analogy evaluation and hyperparameters

NB! ➔



Analogy evaluation and hyperparameters



Word Embeddings: Summary

We have developed models that can translate meaningful relationships between words into linear relations in the word vector space

Both Levy & Goldberg and GloVe shows the connection between **Count!** work and **Predict!** work – appropriate scaling of counts gives the properties and performance of **Predict!** Models

Word Embeddings: Other work

Can one better explain word2vec's linear structure?

See Arora, Li, Liang, Ma, & Risteski (2015) Random Walks on Context Spaces: Towards an Explanation of the Mysteries of Semantic Word Embeddings. [Develops a generative model.]

Word Embeddings: Word senses

- Most words have multiple senses
  - E.g., *pike*
- Most current models just give one vector per string
  - Vector is (weighted) average ("superposition") of all the senses
- Just a little work has tried to model multiple senses per word
  - Huang, Socher, Manning, & Ng 2012
  - Trask, Gilmore, & Russell 2015

- Of course, sentences can also be ambiguous

Word Embeddings: Multi-words

- There are many multi-word expressions (MWEs) which only have one meaning but are written and formed as several words
  - They are non-compositional
  - New York
  - make up
  - neural network
- May just be added to the vocabulary

Representations and levels in linguistics

Phonetics and phonology

- Phonetics is the sound stream – uncontroversial
- Phonology posits a small set or sets of distinctive, categorical units: phonemes or distinctive features
  - Perhaps universal typology but language-particular realization
  - Some of the best evidence of categorical perception comes from phonology

CONSONANTS (PULMONIC) © 2005 IPA

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	ɴ		
Trill	ʙ			r					ʀ		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Where symbols appear in pairs, the one to the right represents a voiced consonant. Shaded areas denote articulations judged impossible.

Phonetics and phonology

- Speech systems normally treat phonemes somewhat loosely – as "phones"
- They actually model triphones – a phone in context, since realization of phone varies greatly according to context
  - These are then automatically clustered to reduce their number
- Above is mainly true of traditional or DL speech systems
  - Though system may map directly to words or letter sequences rather than explicitly having a phone representation level
- Less depth of representation than traditional linguistics!

Writing systems

Most deep learning NLP work begins with language in its written form – it's the easily processed, found data

But human language writing systems aren't one thing!

• Phonemic (maybe digraphs)	jiyawu ngabulu	Wambaya
• Fossilized phonemic system	thorough failure	English
• Syllabic/moraic	ᑐᖏᕐᕋᖏᓚᕐᖓᑦ	Inuktitut
• Ideographic writing	去年太空船二号坠毁	Chinese
• Combination of the above	インド洋の島	Japanese

Writing systems

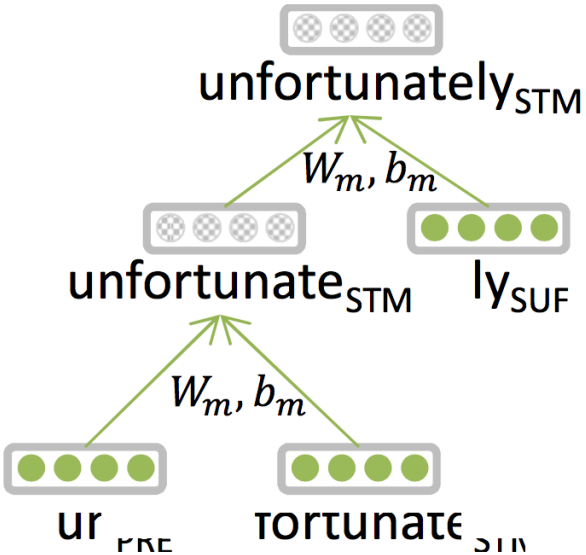
Written languages vary in how they represent words – or don't

- No word segmentation 美国关岛国际机场及其办公室均接获
- Words segmented
  - Clitics?
    - Separated **Je vous ai apporté** des bonbons
    - Joined ف+قال+ان+ها = so+said+we+it
  - Compounds?
    - Separated life insurance company employee
    - Joined Lebensversicherungsgesellschaftsangestellter

Morphology

- Traditional morpheme as smallest semantic unit
  - $[[\text{un}\ [[\text{fortunate}]_{\text{ROOT}}]_{\text{STEM}}]_{\text{STEM}}\ \text{ly}]_{\text{WORD}}$

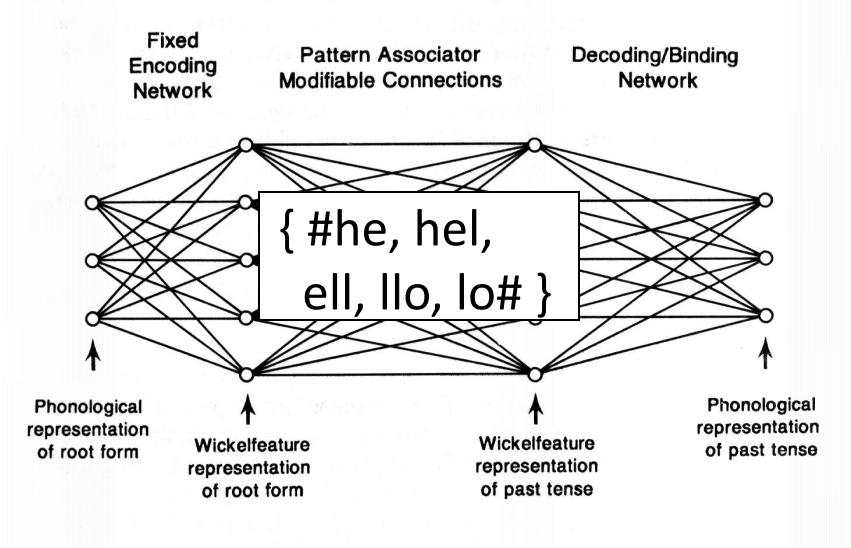
- Deep learning: Very little studied, though one attempt with recursive neural networks (Luong, Socher, & Manning 2013):



Possible way of dealing with a larger vocabulary – most unseen words are new morphological forms (or numbers)

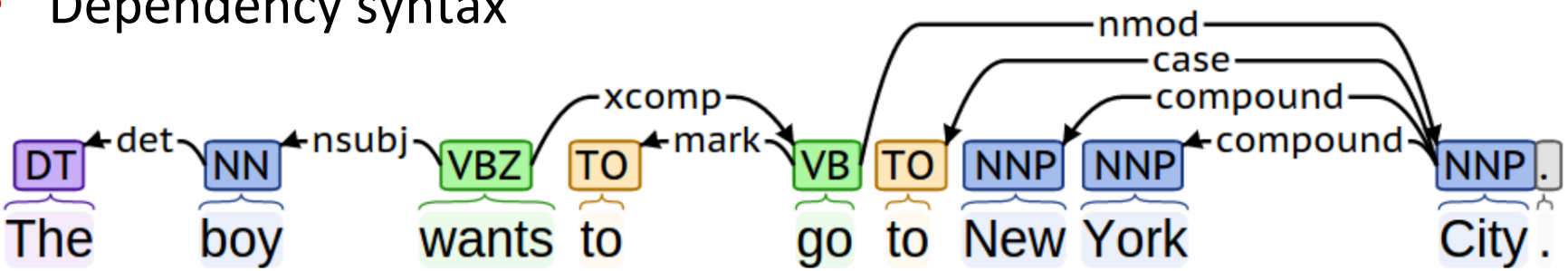
Morphology

- Alternative is to work with character n-grams
  - Wickelphones (Rumelhart & McClelland 1986)
  - Microsoft's DSSM (Huang, He, Gao, Deng, Acero, & Hect 2013)
- Can give many of the benefits of morphemes more easily?

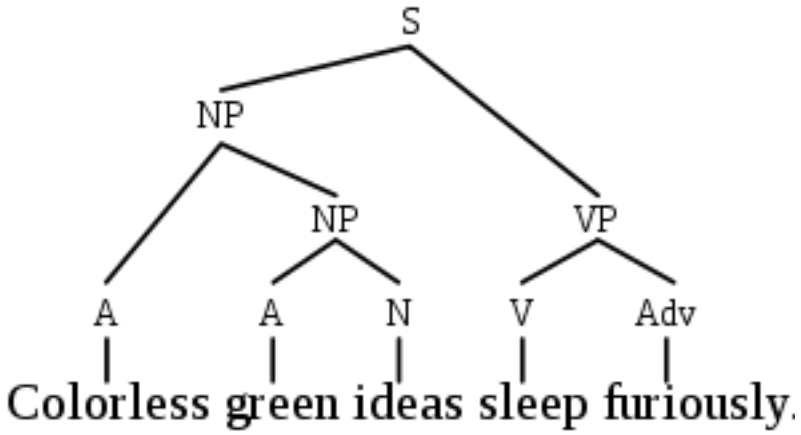


Syntax: Sentence structure

- Dependency syntax



- Constituency syntax



Sort of almost gives a shallow semantics

Semantics and Pragmatics

- Semantics: Inherent meaning of a linguistic unit
- Pragmatics: Interpretation of meaning in context
  - Many of our empirical models necessarily mix in pragmatics; not necessarily a bad thing

- Types of semantics:
  - Distributional semantics: *cold* is like *hot*
  - Frame semantics: *cold* takes a theme and optional experiencer
    - **That pipe** feels cold (to **me**)
  - Model-theoretic (denotational semantics)
    - Map language onto an executable model

Semantics

Frame semantics / semantic roles – more normalization than syntax

Cynthia	*sold*	*the bike*	*to*	*Bob*	*for*	*$200*
SELLER	**PREDICATE**	GOODS		BUYER		PRICE

Formal compositional semantics: building executable logical forms

What is the largest city in California?

$$\texttt{argmax}(\lambda x.\texttt{city}(x) \wedge \texttt{loc}(x, \texttt{CA}), \lambda x.\texttt{population}(x))$$

Reference/anaphora

The **dog** *chased the* **cat**, *which ran up a tree.* **It** *waited at the top.*

The **dog** *chased the* **cat**, *which ran up a tree.* **It** *waited at the bottom.*

"The Winograd Schema Challenge" (Levesque, 2011)

- Easy for humans, can't use surface-level patterns

Word embeddings: syntax vs. semantics

- Word embeddings can be biased to model either semantics or syntax
  - Either *carrying* is biased to be nearer to *transporting*
  - Or *carrying* is biased to be nearer to *carry*, *carried*
- Using bag of words contexts biases to semantic similarity
  - Good for word similarity tests, bad for linguistic analysis
    - Google word2vec or GloVe vectors
- Using linear order of bags of dependency-labeled context biases for syntactic similarity
  - Good for things like part-of-speech tagging and parsing
    - Collobert & Weston vectors, Levy & Goldberg dependency vectors

Envoi

- There are very good reasons to want to represent meaning with distributed representations
- So far, distributional learning has been most effective for this
  - But cf. [Young, Lai, Hodosh & Hockenmaier 2014] on denotational representations, using visual scenes
- A pressing question is modeling reference versus similarity, and that also suggests a denotational model