Exploring Lexical Semantics with Word Embeddings

Aniello De Santo

aniellodesanto.github.io
aniello.desanto@stonybrook.edu

San Jose State
Feb 6, 2020

Get the slides!

Word Meanings

- What is a word meaning?

Word Meanings

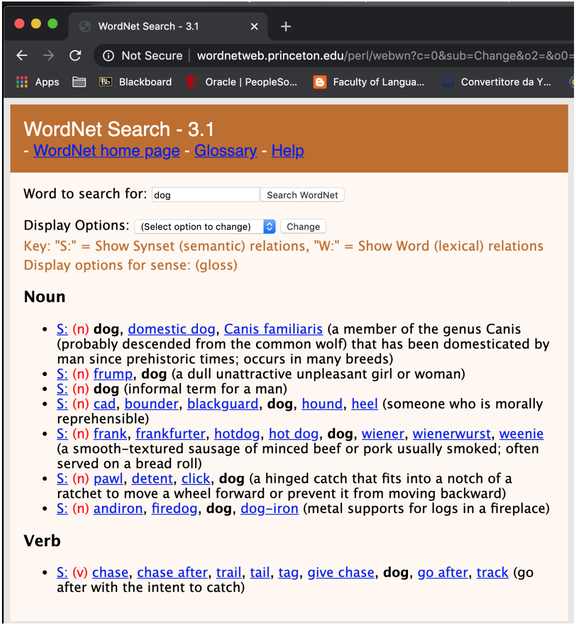
- What is a word meaning?

Example: Dictionary Approach

dog

- is a **mammal**,
- descended from **wolf**,
- is commonly a **pet**,
- subtypes are **poodle**, **bulldog**, . . .
- has **fur**,
- . . .

Wordnet



WordNet Search - 3.1

- WordNet home page - Glossary - Help

Word to search for: dog

Display Options: (Select option to change)

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss)

Noun

- S: (n) **dog**, domestic dog, Canis familiaris (a member of the genus Canis (probably descended from the common wolf) that has been domesticated by man since prehistoric times; occurs in many breeds)
- S: (n) frump, **dog** (a dull unattractive unpleasant girl or woman)
- S: (n) **dog** (informal term for a man)
- S: (n) cad, bounder, blackguard, **dog**, hound, heel (someone who is morally reprehensible)
- S: (n) frank, frankfurter, hotdog, hot dog, **dog**, wiener, wienerwurst, weenie (a smooth-textured sausage of minced beef or pork usually smoked; often served on a bread roll)
- S: (n) pawl, detent, click, **dog** (a hinged catch that fits into a notch of a ratchet to move a wheel forward or prevent it from moving backward)
- S: (n) andiron, firedog, **dog**, dog-iron (metal supports for logs in a fireplace)

Verb

- S: (v) chase, chase after, trail, tail, tag, give chase, **dog**, go after, track (go after with the intent to catch)

Why the Dictionary Approach is Problematic

- Such dictionaries have been tried for computers.

  e.g. WordNet
- They must be created by hand, which is a big problem:
  - expensive
  - only available for some languages
  - many new words missing
- We need dictionaries that can be generated automatically.

Meaning as Word Use

- The philosopher **Ludwig Wittgenstein** said that a word's meaning is its use.

Computational Counterpart

A word's meaning is given by how often it occurs together with other words.

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-			
cat		-		
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2		
cat		-		
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	
cat		-		
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat		-		
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-		
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark			-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2		-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2	1	-	
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2	1	-	0
run				-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2	1	-	0
run	1			-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2	1	-	0
run	1	2		-

Counting Tokens for Word Meaning

Step 1: Record in how many sentences words **occur together**

Example

The dog barked at the cat. The cat ran away. The dog ran after the cat. The dog kept barking. He also kept running.

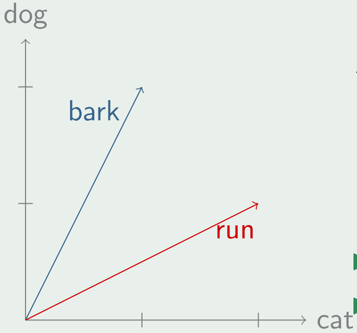
	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2	1	-	0
run	1	2	0	-

From Vectors to Vector Spaces

Step 2: Construct an n-dimensional vector space.

n is given by the number of word types in the text

2-Dimensional Vector Space with *dog* and *cat*



	dog	cat	bark	run
dog	-	2	2	1
cat	2	-	1	2
bark	2	1	-	0
run	1	2	0	-

- *bark* more closely related to *dog*
- *run* more closely related to *cat*

Problems?

- **Conceptual Concerns**
  - Is word meaning really just a bunch of numbers?
- **(More) Practical Concerns**
  - In a real-word model, the vector space will have thousands of dimensions (thousands of unique words)
  - most of the words in the vocabulary will not co-occur in the same sentence (or document!)
    $\Rightarrow$ results in vectors with mostly empty (zeros) slots.
  - Will similar words have similar vectors?

Word Embedding Methods

Method	Paper
LSA	Landauer & Dumais (1997)
Word2Vec	Mikolov et al. (2013)
ELMo	Peters et al. (2018)
BERT	Devlin et al. (2019, arxiv)

Word2Vec

- Word2Vec is **predictive model** for learning word embeddings from raw text
- a shallow, two-layer neural networks trained to reconstruct linguistic contexts of words
- words that share common contexts in the corpus are located in close proximity to one another in the space

Word Embedding Methods

Method	Paper
LSA	Landauer & Dumais (1997)
Word2Vec	Mikolov et al. (2013)
ELMo	Peters et al. (2018)
BERT	Devlin et al. (2019, arxiv)

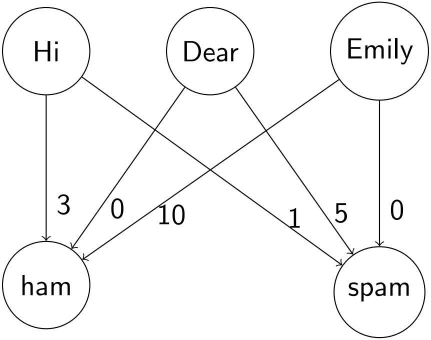
Word2Vec

- Word2Vec is **predictive model** for learning word embeddings from raw text
- a shallow, two-layer neural networks trained to reconstruct linguistic contexts of words
- words that share common contexts in the corpus are located in close proximity to one another in the space

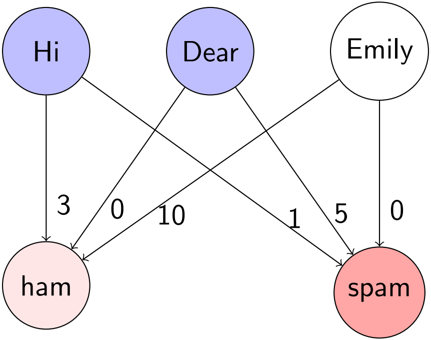
A Quick Excursus: The Perceptron

The Perceptron: A Mini-Version of a Neural Network

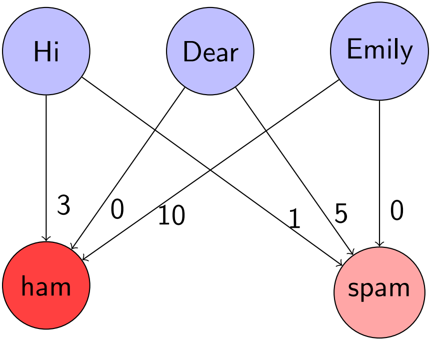
- **input layer:** neurons that are sensitive to input
- **output layer:** neurons that represent output values
- **connections:** weighted links between input and output layer
- most activated output neuron represents decision



Perceptron Activation for *Hi Dear*

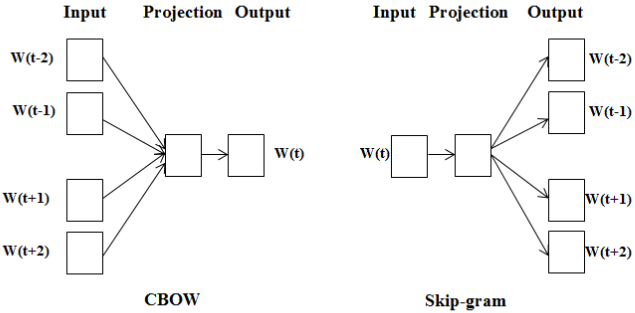


Perceptron Activation for *Hi Dear Emily*



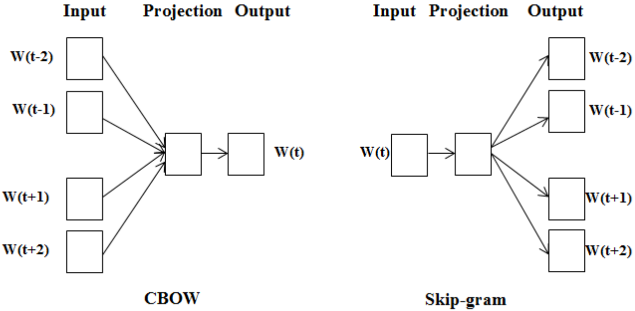
Back to Word2Vec

- A NN trained to reconstruct linguistic contexts of words
- Two learning algorithms:
  - the Continuous Bag-of-Words (CBOW)
  - the Skip-Gram model



Back to Word2Vec

- A NN trained to reconstruct linguistic contexts of words
- Two learning algorithms:
  - the Continuous Bag-of-Words (CBOW)
  - the Skip-Gram model

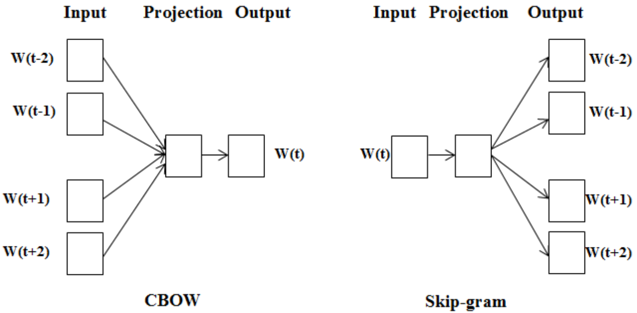


CBOW

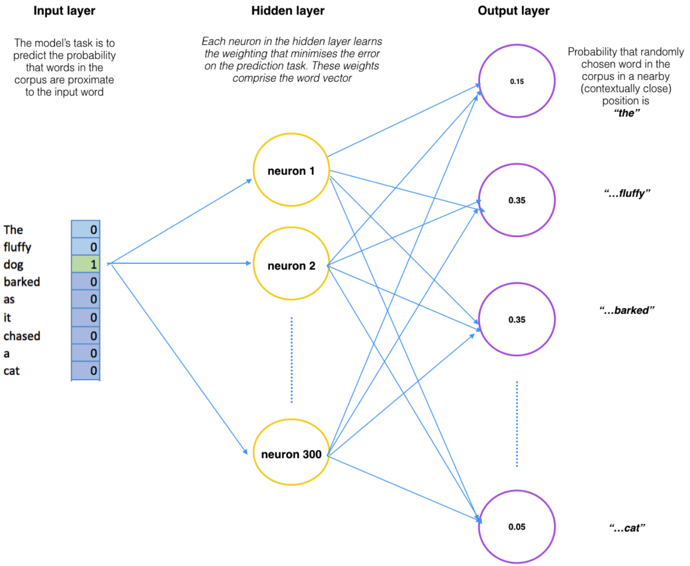
Skip-gram

Back to Word2Vec

- A NN trained to reconstruct linguistic contexts of words
- Two learning algorithms:
  - the Continuous Bag-of-Words (CBOW)
  - the Skip-Gram model: predict context based on target word.



The Skip-Gram Model: Architecture



Input layer

The model's task is to predict the probability that words in the corpus are proximate to the input word

Hidden layer

Each neuron in the hidden layer learns the weighting that minimises the error on the prediction task. These weights comprise the word vector

Output layer

Probability that randomly chosen word in the corpus in a nearby (contextually close) position is ***"the"***

The	0
fluffy	0
dog	1
barked	0
as	0
it	0
chased	0
a	0
cat	0

"...fluffy"

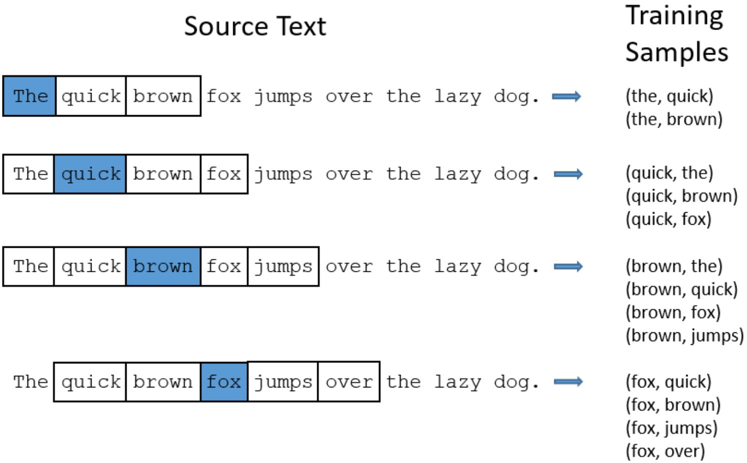
"...barked"

"...cat"

0.3	0.1	0.38	0.22	0.05	0.09	...	0.98	0.13	0.2

The Skip-Gram Model: Predicting Context

- Skip-Gram: predict context based on target word.



Source Text

Training Samples

The quick brown fox jumps over the lazy dog. ⟹

(the, quick)
(the, brown)

The quick brown fox jumps over the lazy dog. ⟹

(quick, the)
(quick, brown)
(quick, fox)

The quick brown fox jumps over the lazy dog. ⟹

(brown, the)
(brown, quick)
(brown, fox)
(brown, jumps)

The quick brown fox jumps over the lazy dog. ⟹

(fox, quick)
(fox, brown)
(fox, jumps)
(fox, over)

Let’s try it together!

shorturl.at/ADPSX

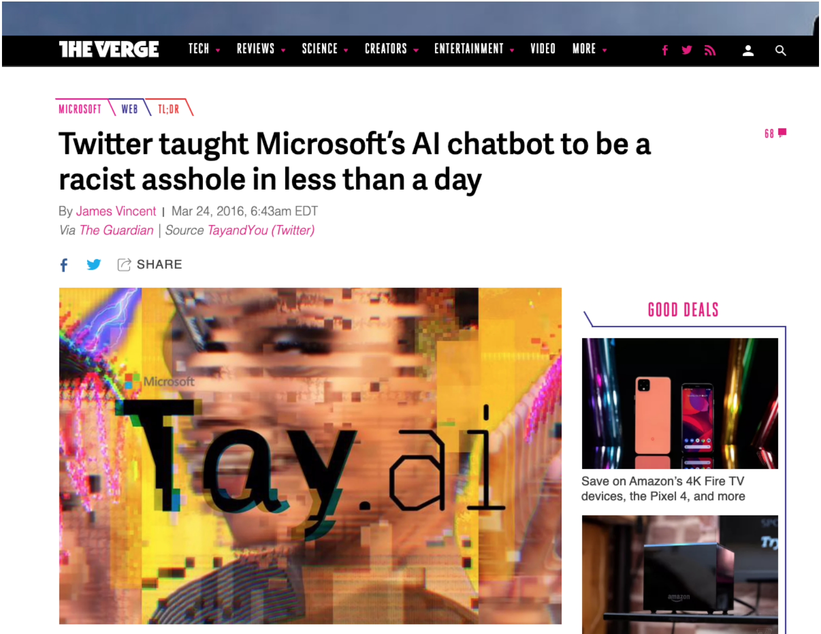
Some Recent Applications

- **Web Search**
  - construct meaning vector for every website
  - rank websites by vector similarity
- **Ad Sense**
  - associate every ad with a vector
  - pick ad that most closely matches website vector

Possible Concerns

- Watch out for **intrinsic biases**!

The Danger of Corpora



Twitter taught Microsoft's AI chatbot to be a racist asshole in less than a day

By James Vincent | Mar 24, 2016, 6:43am EDT

Via The Guardian | Source TayandYou (Twitter)

Save on Amazon's 4K Fire TV devices, the Pixel 4, and more

Is This Realistic?

- **Possible Concerns**
  - Is word meaning really just a bunch of numbers?
- But this might actually capture something psychologically real!

Psycholinguistic Experiments

- Word association tasks (Rubistein et al. 2015)
- ERP measures of context appropriateness (Broderik et al. 2018, Ettinger et al. 2016)
- Priming effects (Gunther et al. 2016)
  - Check it out: **Masked priming effects!**

Is This Realistic? [cont.]

- For word meaning, the approach seems to work.
- But what about sentence/text meaning?

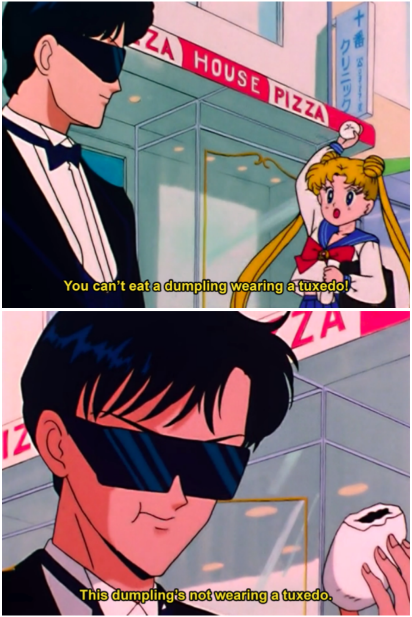
Example

The following two sentences receive the same vector:

(1) a. Dog bites man!
    b. Man bites dog!

Is This Realistic? [cont.]

- Meaning is not just about lexical representations.

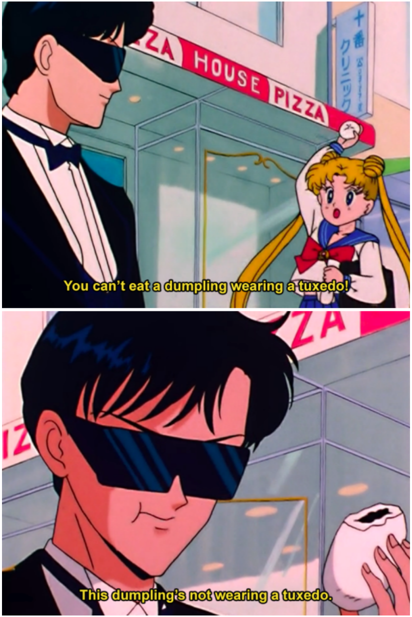

ZA HOUSE PIZZA
You can't eat a dumpling wearing a tuxedo!
This dumpling's not wearing a tuxedo.

Is This Realistic? [cont.]

- Meaning is not just about lexical representations.

You can't:

- [eat a dumpling] [wearing a tuxedo]
- eat a [dumpling wearing a tuxedo]


HOUSE PIZZA
十番 クリニック
You can't eat a dumpling wearing a tuxedo!
This dumpling's not wearing a tuxedo.

TL/DR

Word embeddings

- A computational implementation of a distributional semantics!

- useful in a variety of applications
  - Ad-sense, stylistic analysis
  - part-of-speech tagging, parsing, machine translation
- source of theoretical insights
  - diachronic change, semantic shifts, predictive processing, etc.
  - control for semantic similarity in psycholinguistic experiments
- cognitive parallels?

TL/DR

Word embeddings

- A computational implementation of a distributional semantics!

- useful in a variety of applications
  - Ad-sense, stylistic analysis
  - part-of-speech tagging, parsing, machine translation
- source of theoretical insights
  - diachronic change, semantic shifts, predictive processing, etc.
  - control for semantic similarity in psycholinguistic experiments
- cognitive parallels?

But: Meaning is more complex than simple distributional information!

Further Readings

1. Distributed Representations of Words and Phrases and their Compositionality
2. Efficient Estimation of Word Representations in Vector Space
3. A Neural Probabilistic Language Model
4. A nice series of blog posts by Chris McCormick
5. Evaluating distributional models of compositional semantics
6. A semi-technical tutorial (some of the pictures in this presentation are from there)
7. Exploring the Implications of Biases in Word2Vec
8. Debiasing Word Embeddings

Appendix

Word Embeddings

We saw sparse vectors:

0	1	0	0	0	0	...	0	0	0

- But word vectors can be dense:
  real numbers in a small number of dimensions
- Compress sparse matrices into smaller ones

0.3	0.1	0.38	0.22	0.05	0.09	...	0.98	0.13	0.2

Word Embeddings

We saw sparse vectors:

0	1	0	0	0	0	...	0	0	0

- But word vectors can be dense:
  real numbers in a small number of dimensions
- Compress sparse matrices into smaller ones

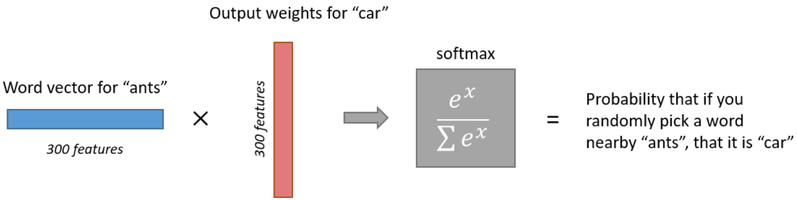
0.3	0.1	0.38	0.22	0.05	0.09	...	0.98	0.13	0.2

The Skip-Gram Model: Some Details

- some math:

$$\begin{bmatrix}0 & 0 & 0 & 1 & 0\end{bmatrix} \times \begin{bmatrix}17 & 24 & 1\\ 23 & 5 & 7\\ 4 & 6 & 13\\ 10 & 12 & 19\\ 11 & 18 & 25\end{bmatrix} = \begin{bmatrix}10 & 12 & 19\end{bmatrix}$$

- a high-level illustration of the architecture:



Output weights for "car"

Word vector for "ants"

300 features

×

300 features

softmax

$$\frac{e^x}{\sum e^x}$$

=

Probability that if you randomly pick a word nearby "ants", that it is "car"

Source: A nice technical tutorial

W2V: Problems?

Will similar words have similar vectors?

- Consider the following sentences:
  1. I like watching movies.
  2. I enjoy watching movies.
  3. I hate watching movie.
- What is the distance between *like*, *enjoy*, and *hate*?

W2V: Problems?

Will similar words have similar vectors?

- Consider the following sentences:
  1. I like watching movies.
  2. I enjoy watching movies.
  3. I hate watching movie.
- What is the distance between *like*, *enjoy*, and *hate*?
- How similar are the following sentences?

1. I like pancakes.
2. Steven enjoys cookies.

An Observation on Frequencies: Zipf's Law

- Word models care about word frequency.
- But there is a problem…

Zipf's Law

The frequency of a type is inversely proportional to its rank.

In Plain English

The most frequent word is

- 2 times as common as the 2nd most frequent word,
- 3 times as common as the 3rd most frequent word,
- and so on.

An Observation on Frequencies: Zipf's Law

- Word models care about word frequency.
- But there is a problem. . .

Zipf's Law

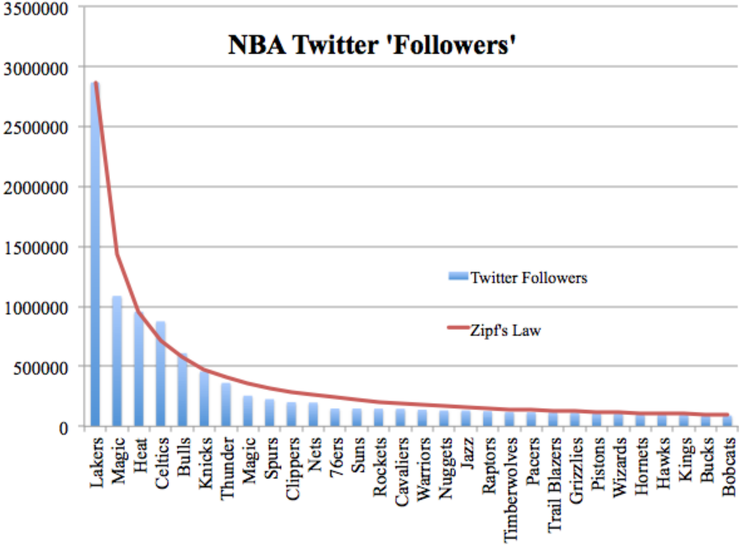
The frequency of a type is inversely proportional to its rank.

In Plain English

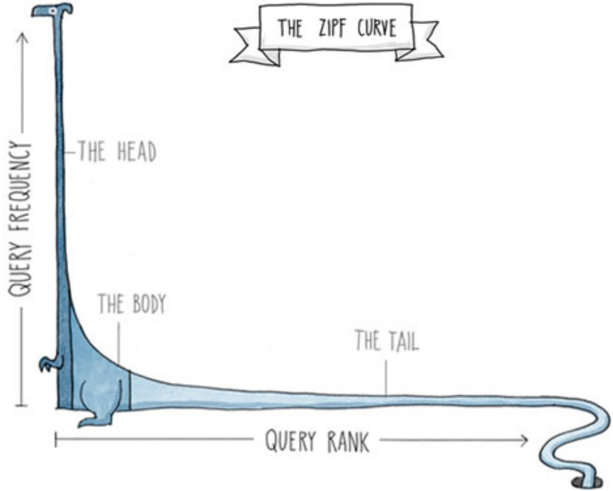
The most frequent word is

- 2 times as common as the 2nd most frequent word,
- 3 times as common as the 3rd most frequent word,
- and so on.

An Example from. . . the NBA?



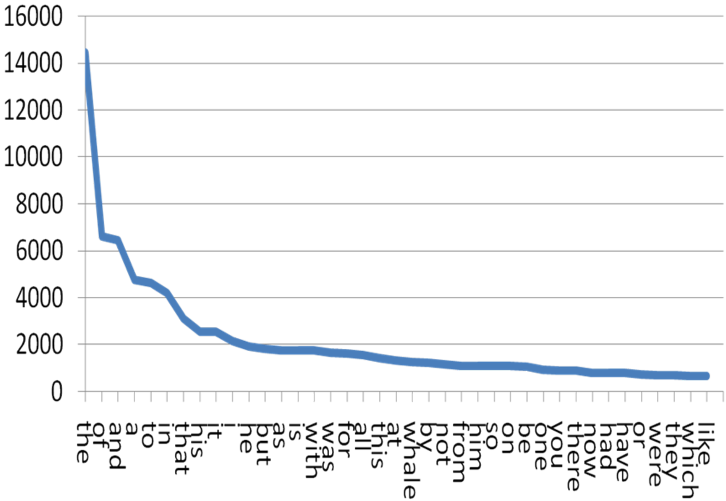
Visualizing Zipf Distributions



Zipf's Law is Everywhere...

- A distribution is probably Zipfian if
  - there is a **long neck**:
    a few types make up the majority of tokens,
  - there is a **long tail**:
    most types only have 1 token (**hapax legomenon**)
- Surprisingly, Zipf's Law shows up in tons of places:
  - size of large cities in a country
  - citations for academic papers
  - frequencies of last names
  - frequencies of weekdays in text

. . . Even in Language!



Stop Words

- About 150 words make up 50% of all English texts: *the*, *of*, *and*, *a*, . . .
- These are called **stop words**.
- Stop words are not very informative for many applications.
- So they are usually discarded after the tokenization step.
- Failure to do so can greatly reduce the model's performance.

Steps of Word Counting Model (Revised)

1. collect corpus
2. remove stop words
3. tokenize strings
4. count tokens for each type

Stop Words

- About 150 words make up 50% of all English texts: *the*, *of*, *and*, *a*, . . .
- These are called **stop words**.
- Stop words are not very informative for many applications.
- So they are usually discarded after the tokenization step.
- Failure to do so can greatly reduce the model's performance.

Steps of Word Counting Model (Revised)

1. collect corpus
2. remove stop words
3. tokenize strings
4. count tokens for each type

Example: A Text Without (Non)-Stop Words

- Stop words are much less informative than non-stop words.
- Just check the example below.

Stop Words only

The having no on the

Example: A Text Without (Non)-Stop Words

- Stop words are much less informative than non-stop words.
- Just check the example below.

Stop Words and Non-Stop Words

The sun shone having no alternative on the nothing new

Example: A Text Without (Non)-Stop Words

- Stop words are much less informative than non-stop words.
- Just check the example below.

Non-Stop Words only

sun shone alternative nothing new

An Important Consequence of Zipf’s Law

- Texts mostly consist of stop words.
- Hence it can be difficult to get representative counts for non-stop words.

Sparse Data Problem

- Most of the data is not informative.
- You need tons of data to have enough useful data.

An Important Consequence of Zipf's Law

- Texts mostly consist of stop words.
- Hence it can be difficult to get representative counts for non-stop words.

Sparse Data Problem

- Most of the data is not informative.
- You need tons of data to have enough useful data.

Example

- Most models require corpora with at least a few million sentences.
- Really good models (e.g. Google translate) use billions of data points.