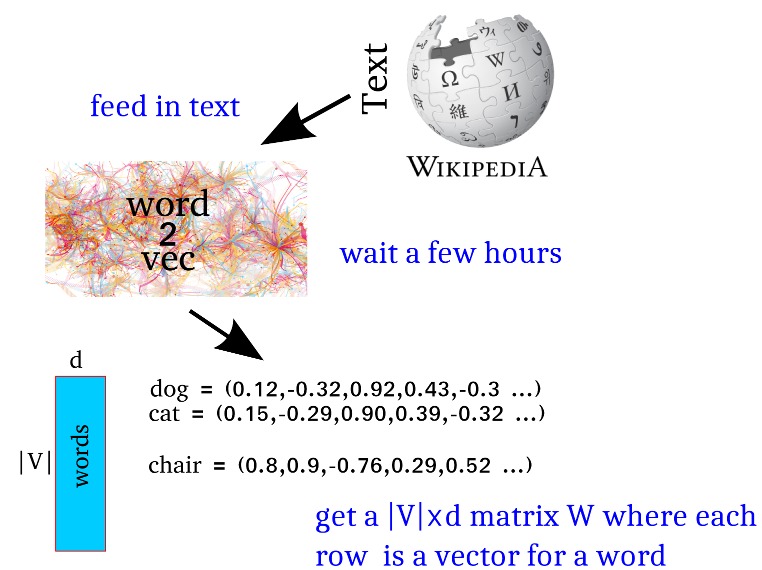
word embeddings what, how and whither

Yoav Goldberg
Bar Ilan University

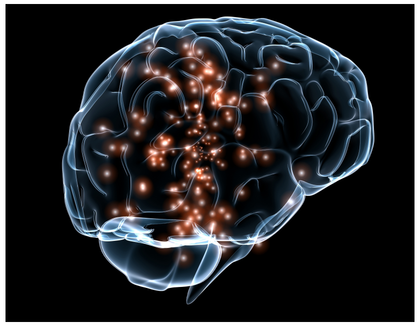
understanding word2vec

word2vec



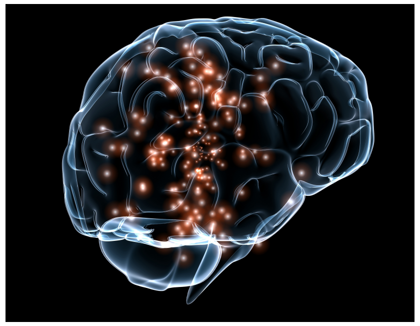
feed in text
Text
WIKIPEDIA
word
2
vec
wait a few hours
d
|V|
words
dog = (0.12,-0.32,0.92,0.43,-0.3 ...)
cat = (0.15,-0.29,0.90,0.39,-0.32 ...)
chair = (0.8,0.9,-0.76,0.29,0.52 ...)
get a |V|×d matrix W where each
row is a vector for a word

Seems magical.



"Neural computation, just like in the brain!"

Seems magical.



"Neural computation, just like in the brain!"

How does this actually work?

How does word2vec work?

word2vec implements several different algorithms:

Two training methods

- Negative Sampling
- Hierarchical Softmax

Two context representations

- Continuous Bag of Words (CBOW)
- Skip-grams

How does word2vec work?

word2vec implements several different algorithms:

Two training methods

- **Negative Sampling**
- Hierarchical Softmax

Two context representations

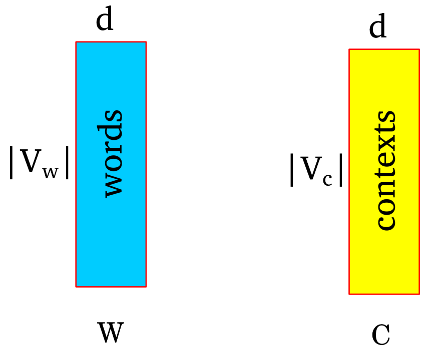
- Continuous Bag of Words (CBOW)
- **Skip-grams**

We'll focus on skip-grams with negative sampling.

intuitions apply for other models as well.

How does word2vec work?

- Represent each word as a *d* dimensional vector.
- Represent each context as a *d* dimensional vector.
- Initalize all vectors to random weights.
- Arrange vectors in two matrices, *W* and *C*.



How does word2vec work?

While more text:

- Extract a word window:

```
A springer is [  a    cow   or   heifer   close   to   calving  ].
```

$$\begin{array}{cccccc} c_1 & c_2 & c_3 & w & c_4 & c_5 & c_6 \end{array}$$

- w is the focus word vector (row in W).
- c_i are the context word vectors (rows in C).

How does word2vec work?

While more text:

- Extract a word window:

`A springer is [`	`a`	`cow`	`or`	**`heifer`**	`close`	`to`	`calving`	`].`
	c_1	c_2	c_3	w	c_4	c_5	c_6	

- Try setting the vector values such that:

  $$\sigma(w\cdot c_1)+\sigma(w\cdot c_2)+\sigma(w\cdot c_3)+\sigma(w\cdot c_4)+\sigma(w\cdot c_5)+\sigma(w\cdot c_6)$$

  is **high**

How does word2vec work?

While more text:

- Extract a word window:

$$\begin{array}{lcccccccc} \texttt{A springer is [} & \texttt{a} & \texttt{cow} & \texttt{or} & \textbf{heifer} & \texttt{close} & \texttt{to} & \texttt{calving} & \texttt{].} \\ & c_1 & c_2 & c_3 & w & c_4 & c_5 & c_6 & \end{array}$$

- Try setting the vector values such that:

$$\sigma(w\cdot c_1)+\sigma(w\cdot c_2)+\sigma(w\cdot c_3)+\sigma(w\cdot c_4)+\sigma(w\cdot c_5)+\sigma(w\cdot c_6)$$

  is **high**

- Create a corrupt example by choosing a random word w'

$$\begin{array}{ccccccccc} \texttt{[} & \texttt{a} & \texttt{cow} & \texttt{or} & \textbf{comet} & \texttt{close} & \texttt{to} & \texttt{calving} & \texttt{]} \\ & c_1 & c_2 & c_3 & w' & c_4 & c_5 & c_6 & \end{array}$$

- Try setting the vector values such that:

$$\sigma(w'\cdot c_1)+\sigma(w'\cdot c_2)+\sigma(w'\cdot c_3)+\sigma(w'\cdot c_4)+\sigma(w'\cdot c_5)+\sigma(w'\cdot c_6)$$

  is **low**

How does word2vec work?

The training procure results in:

- $w \cdot c$ for **good** word-context pairs is **high**.
- $w \cdot c$ for **bad** word-context pairs is **low**.
- $w \cdot c$ for **ok-ish** word-context pairs is **neither high nor low**.

As a result:

- Words that share many contexts get close to each other.
- Contexts that share many words get close to each other.

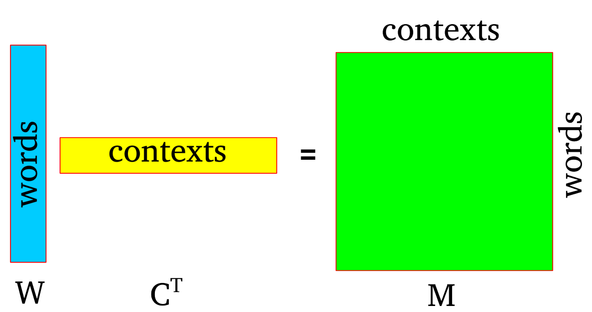
At the end, word2vec throws away C and returns W.

Reinterpretation

Imagine we didn't throw away C. Consider the product $WC^\top$

Reinterpretation

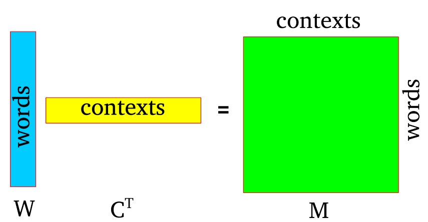
Imagine we didn't throw away C. Consider the product $WC^\top$



The result is a matrix M in which:

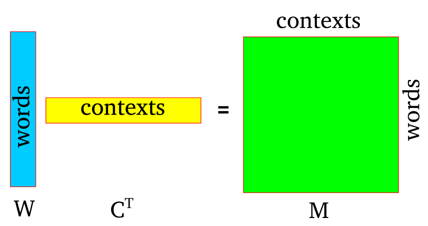
- Each row corresponds to a word.
- Each column corresponds to a context.
- Each cell correspond to $w \cdot c$, an association measure between a word and a context.

Reinterpretation



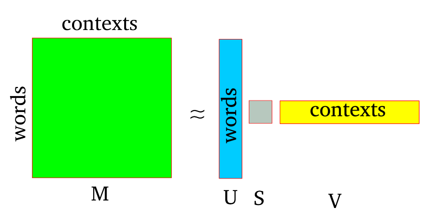
Does this remind you of something?

Reinterpretation



Does this remind you of something?

Very similar to SVD over distributional representation:



context matters

What’s in a Context?

- Importing ideas from embeddings improves distributional methods

- Can distributional ideas also improve embeddings?

- **Idea:** change SGNS’s default **BoW contexts** into **dependency contexts**

“Dependency-Based Word Embeddings”
Levy & Goldberg, ACL 2014

Example

Australian scientist discovers star with telescope

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Target Word

Australian scientist discovers star with telescope

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Bag of Words (BoW) Context

Australian scientist <u>discovers</u> star with telescope

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Bag of Words (BoW) Context

Australian scientist <u>discovers</u> star with telescope

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Bag of Words (BoW) Context

Australian scientist <u>discovers</u> star with telescope

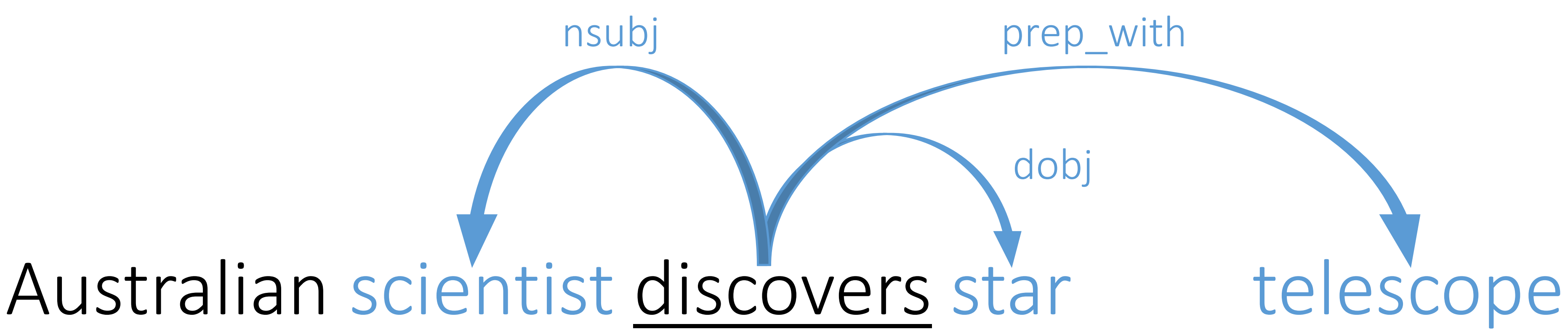
"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Syntactic Dependency Context

Australian scientist <u>discovers</u> star with telescope

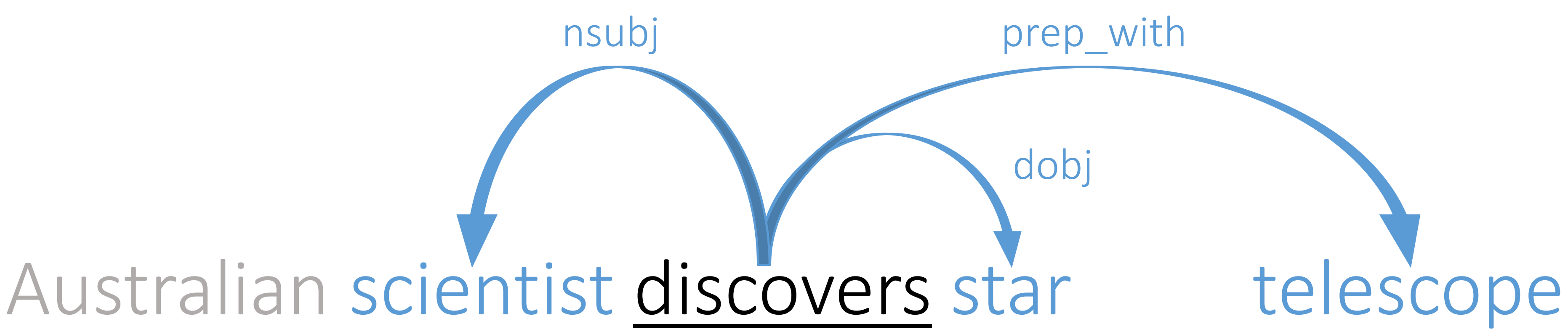
"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Syntactic Dependency Context



“Dependency-Based Word Embeddings”
Levy & Goldberg, ACL 2014

Syntactic Dependency Context



"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Embedding Similarity with Different Contexts

Target Word	Bag of Words (k=5)	Dependencies
Hogwarts (Harry Potter's school)	Dumbledore	Sunnydale
	hallows	Collinwood
	half-blood	Calarts
	Malfoy	Greendale
	Snape	Millfield
	Related to Harry Potter	**Schools**

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Embedding Similarity with Different Contexts

Target Word	Bag of Words (k=5)	Dependencies
Turing (computer scientist)	nondeterministic	Pauling
	non-deterministic	Hotelling
	computability	Heting
	deterministic	Lessing
	finite-state	Hamming
	Related to computability	Scientists

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Embedding Similarity with Different Contexts

Target Word	Bag of Words (k=5)	Dependencies
dancing (dance gerund)	singing dance dances dancers tap-dancing	singing rapping breakdancing miming busking
	Related to dance	**Gerunds**

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

What is the effect of different context types?

- Thoroughly studied in distributional methods
  - Lin (1998), Padó and Lapata (2007), and many others…

General Conclusion:

- Bag-of-words contexts induce ***topical*** similarities
- Dependency contexts induce ***functional*** similarities
  - Share the same semantic type
  - Cohyponyms

- Holds for **embeddings** as well

"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

- Same algorithm, different inputs -- very different kinds of similarity.

- Inputs matter much more than algorithm.

- **Think about your inputs.**