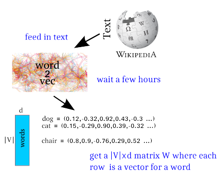
word embeddings
what, how and whither

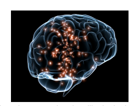
Yoav Goldberg
Bar Ilan University

understanding word2vec

word2vec

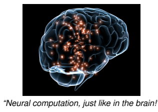


Seems magical.



"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

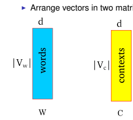
- Continuous Bag of Words (CBOW)
- **Skip-grams**

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 ].`

$c_1 \quad c_2 \quad c_3 \quad w \quad c_4 \quad c_5 \quad c_6$

- 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 ].`

$c_1 \quad c_2 \quad c_3 \quad w \quad c_4 \quad c_5 \quad c_6$

- 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:

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

$c_1 \quad c_2 \quad c_3 \quad w \quad c_4 \quad c_5 \quad c_6$

- 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**

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

`[ a cow or **comet** close to calving ]`

$c_1 \quad c_2 \quad c_3 \quad w' \quad c_4 \quad c_5 \quad c_6$

- 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 **low**

How does word2vec work?

The training procedure 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.

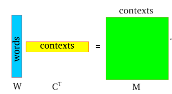
At the end, word2vec throws away C and returns W.

Reinterpretation

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

Reinterpretation

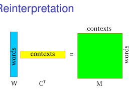
Imagine we didn't throw away C. Consider the product $WC^\top$



The result is a matrix M in which:

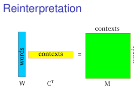
- Each row corresponds to a word.
- Each column corresponds to a context.
- Each cell correspond to $w \cdot c$, an association measure between a word and a context.

Reinterpretation



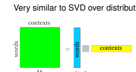
Does this remind you of something?

Reinterpretation



Does this remind you of something?

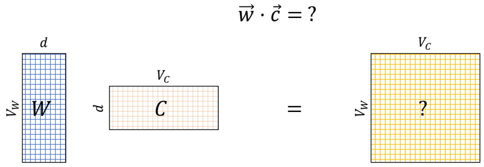
Very similar to SVD over distributional representation:



What is SGNS learning?

- A $V_W \times V_C$ matrix
- Each cell describes the relation between a specific word-context pair

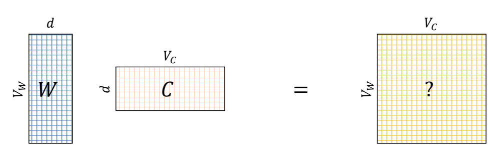
$$\vec{w} \cdot \vec{c} = ?$$



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

What is SGNS learning?

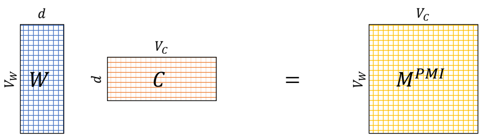
- We **prove** that for large enough d and enough iterations



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

What is SGNS learning?

- We **prove** that for large enough d and enough iterations
- We get the word-context PMI matrix

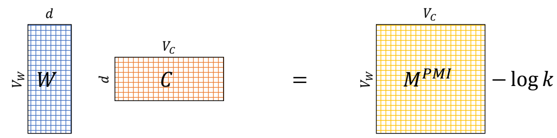


"Neural Word Embeddings as Implicit Matrix Factorization"
Levy & Goldberg, NIPS 2014

What is SGNS learning?

- We **prove** that for large enough d and enough iterations
- We get the word-context PMI matrix, shifted by a global constant

$$Opt(\vec{w} \cdot \vec{c}) = PMI(w, c) - \log k$$



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

What is SGNS learning?

- SGNS is doing something very similar to the older approaches
- SGNS is factorizing the traditional word-context PMI matrix
- So does SVD!
- Do they capture the same similarity function?

SGNS vs SVD

Target Word	SGNS	SVD
cat	dog rabbit cats poodle pig	dog rabbit pet monkey pig

SGNS vs SVD

Target Word	SGNS	SVD
wine	wines grape grapes winemaking tasting	wines grape grapes varietal vintages

SGNS vs SVD

Target Word	SGNS	SVD
November	October December April January July	October December April June March

But `word2vec` is still better, isn't it?

- Plenty of evidence that `word2vec` outperforms traditional methods
  - In particular: "Don't count, predict!" (Baroni et al., 2014)
- How does this fit with our story?

The Big Impact of "Small" Hyperparameters

Hyperparameters

- `word2vec` is more than just an algorithm…
- Introduces many **engineering tweaks** and **hyperpararameter settings**
  - May seem minor, but **make a big difference** in practice
  - Their impact is often more significant than the embedding algorithm's
- These modifications can be ported to distributional methods!

Levy, Goldberg, Dagan (In submission)

the magic of cbow

the magic of cbow

- Represent a sentence / paragraph / document as a (weighted) average vectors of its words.
- Now we have a single, 100-dim representation of the text.
- Similar texts have similar vectors!
- Isn't this magical? (no)

the math of cbow

the math of cbow

$$\mathrm{doc}A = A + B + C$$
$$\mathrm{doc}B = X + y + Z$$
$$\cos(\mathrm{doc}A, \mathrm{doc}B) = \frac{\mathrm{doc}A \cdot \mathrm{doc}B}{\|\mathrm{doc}A\| \cdot \|\mathrm{doc}B\|}$$

the math of cbow

$$\frac{\mathrm{doc}A \cdot \mathrm{doc}B}{\|\mathrm{doc}A\| \cdot \|\mathrm{doc}B\|} = \frac{(A+B+C)\cdot(X+y+Z)}{\|A+B+C\| \cdot \|X+y+Z\|}$$

the math of cbow

$$(A+B+C)(X+y+Z) =$$
$$A \cdot X + A \cdot y + A \cdot Z$$
$$+ B \cdot X + B \cdot y + B \cdot Z$$
$$+ C \cdot X + C \cdot Y + C \cdot Z$$

the magic of cbow

- It's all about (weighted) all-pairs similarity
  - ... done in an efficient manner.
- That's it. no more, no less.
- I'm amazed by how few people realize this.

  (the math is so simple... even I could do it)

this also explains king-man+woman

this also explains king-man+woman

$$\operatorname*{argmax}_x \cos(x, k - m + w) =$$
$$\operatorname*{argmax}_x \frac{x \cdot (k - m + w)}{\|x\| \; \|k - m + w\|}$$

1

constant

$$= \operatorname*{argmax}_x \; x \cdot k - x m + x w$$

similarity arith!!

and once we understand
we can improve

and once we understand
we can improve

$xk - xm + xw$

additive.
one term can dominate.

and once we understand
we can improve

$xk - xm + xw$

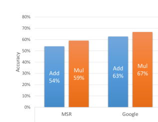
additive.
one term can dominate.

$\Downarrow$

$$\frac{(x \cdot k) \cdot (x \cdot w)}{(x \cdot m)}$$

multiplicative.
much more balanced.

Multiplication > Addition



math > magic

which brings me to:


Manaal Faruqui
@manaalfar
Follow

@yoavgo how to best compare two diff vector models? abolish (toy?) tasks like word-sim, word-analogy? how to standardize evaluation?

can we improve analogies
even further?

which brings me to:

> Manaal Faruqui @manaalfar
>
> @yoavgo how to best compare two diff vector models? abolish (toy?) tasks like word-sim, word-analogy? how to standardize evaluation?

- **Yes. Please stop evaluating on word analogies.**
- It is an artificial and useless task.
- Worse, it is just a proxy for (a very particular kind of) word similarity.
- Unless you have a good use case, don't do it.
- **Alternatively:** show that it correlates well with a real and useful task.

let's take a step back

- We don't really care about the vectors.
- We care about the **similarity function** they induce.
  - (or, maybe we want to use them in an external task)
- We want similar words to have similar vectors.
- So evaluating on word-similarity tasks is great.
- **But what does similar mean?**

many faces of similarity

- dog -- cat
- dog -- poodle
- dog -- animal
- dog -- bark
- dog -- leash

many faces of similarity

- dog -- cat
- dog -- poodle
- dog -- animal
- dog -- bark
- dog -- leash

- dog -- chair
- dog -- dig
- dog -- god
- dog -- fog
- dog -- 6op

many faces of similarity

- dog -- cat
- dog -- poodle
- dog -- animal
- dog -- bark
- dog -- leash

- dog -- chair — same POS
- dog -- dig — edit distance
- dog -- god — same letters
- dog -- fog — rhyme
- dog -- 6op — shape

some forms of similarity look more useful than they really are

- Almost every algorithm you come up with will be good at capturing:
  - countries
  - cities
  - months
  - person names

some forms of similarity look more useful than they really are

- Almost every algorithm you come up with will be good at capturing:
  - countries
  - cities
  - months
  - person names

useful for tagging/parsing/NER

some forms of similarity look more useful than they really are

- Almost every algorithm you come up with will be good at capturing:
  - countries
  - cities
  - months
  - person names

useful for tagging/parsing/NER

but do we really want "John went to China in June" to be similar to "Carl went to Italy in February" ??

there is no single downstream task

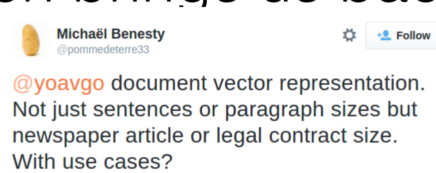
- Different tasks require different kinds of similarity.
- Different vector-inducing algorithms produce different similarity functions.
- **No single representation for all tasks.**
- If your vectors do great on task X, I don't care that they suck on task Y.

"but my algorithm works great for all these different word-similarity datasets! doesn't it mean something?"

"but my algorithm works great for all these different word-similarity datasets! doesn't it mean something?"

- Sure it does.
- It means these datasets are not diverse enough.
- They should have been a single dataset.
- (**alternatively**: our evaluation metrics are not discriminating enough.)

which brings us back to:



- This is really, really il-defined.
- What does it mean for legal contracts to be similar?
- What does it mean for newspaper articles to be similar?
- Think about this before running to design your next super-LSTM-recursive-autoencoding-document-embedder.
- **Start from the use case!!!!**

so how to evaluate?

- Define the similarity / task you care about.
- **Score on this particular similarity / task**.
- Design your vectors to match this similarity
- ...and since the methods we use are distributional and unsupervised...
- ...design has less to do with the fancy math (= objective function, optimization procedure) and more with what you feed it.

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 <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

Bag of Words (BoW) Context

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

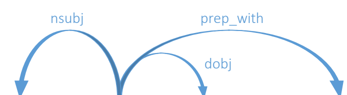
"Dependency-Based Word Embeddings"
Levy & Goldberg, ACL 2014

Syntactic Dependency Context

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

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

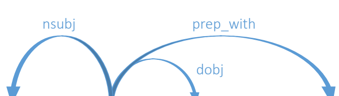
Syntactic Dependency Context



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

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

Syntactic Dependency Context



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

"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	singing
	dance	rapping
	dances	breakdancing
	dancers	miming
	tap-dancing	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.**

what's left to do?

- Pretty much nothing, and pretty much everything.
- Word embeddings are just a small step on top of distributional lexical semantics.
- All of the previous open questions remain open, including:
  - composition.
  - multiple senses.
  - multi-word units.

looking beyond words

- word2vec will easily identify that "hotfix" if similar to "hf", "hot-fix" and "patch"
- But what about "hot fix"?
- How do we know that "New York" is a single entity?
- Sure we can use a collocation-extraction method, but is it really the best we can do? can't it be integrated in the model?

what happens when we look outside of English?

- Things don't work nearly as well.
- Known problems from English become more extreme.
- We get some new problems as well.

a quick look at Hebrew

word senses

ספר

book(N). barber(N). counted(V). tell!(V). told(V).

חומה

brown (feminine, singular)
wall (noun)
her fever (possessed noun)

multi-word units

- עורך דין
- בית ספר
- שומר ראש
- יושב ראש
- ראש עיר
- בית שימוש

words vs. tokens

וכשמהבית

and when from the house

words vs. tokens

וכשמהבית

and when from the house

בצל

in shadow

בצל

onion

and of course: inflections

- nouns, pronouns and adjectives
  --> are inflected for *number* and *gender*
- verbs
  --> are inflected for *number*, *gender*, *tense*, *person*
- syntax requires *agreement* between
  - nouns and adjectives
  - verbs and subjects

and of course: inflections

she **saw** a **brown** fox

he **saw** a **brown** fence

and of course: inflections

[fem] **[masc]**
she **saw** a **brown** fox

he **saw** a **brown** fence
[masc] **[fem]**

and of course: inflections

[fem] **[masc]**
she **saw** a **brown** fox

היא **ראתה** שועל **חום**

הוא **ראה** גדר **חומה**

he **saw** a **brown** fence
[masc] **[fem]**

inflections and dist-sim

- More word forms -- more sparsity
- But more importantly: *agreement patterns affect the resulting similarities.*

adjectives

green [m,sg] ירוק	**green [f,sg]** ירוקה	**green [m,pl]** ירוקים
blue [m,sg]	gray [f,sg]	gray [m,pl]
orange [m,sg]	orange [f,sg]	blue [m,pl]
yellow [m,sg]	yellow [f,sg]	black [m,pl]
red [m,sg]	magical [f,g]	heavenly [m,pl]

verbs

(he) walked הלך	(she) thought חשבה	(they) ate אכלו
(they) walked	(she) is thinking	(they) will eat
(he) is walking	(she) felt	(they) are eating
(he) turned	(she) is convinved	(he) ate
(he) came closer	(she) insisted	(they) drank

nouns

Doctor [m,sg] רופא	Doctor [f, sg] רופאה
psychiatrist [m,sg]	student [f, sg]
psychologist [m, sg]	nun [f, sg]
neurologist [m, sg]	waitress [f, sg]
engineer [m, sg]	photographer [f, sg]

nouns

sweater סוודר	shirt חולצה
jacket	suit
down	robe
overall	dress
turban	helmet

nouns

sweater סוודר	shirt חולצה
jacket	suit
down	robe
overall	dress
turban	helmet
masculine	**feminine**

nouns

sweater סוודר	shirt חולצה
jacket	suit
down	robe
overall	dress
turban	helmet
masculine	**feminine**

completely arbitrary

inflections and dist-sim

- Inflections and agreement really influence the results.
- We get a mix of syntax and semantics.
- Which aspect of the similarity we care about? what does it mean to be similar?
- Need better control of the different aspects.

inflections and dist-sim

- Work with lemmas instead of words!!
- Sure, but where do you get the lemmas?
- ...for unknown words?
- And what should you lemmatize? everything? somethings? context-dependent?
- Ongoing work in my lab -- but still much to do.

to summarize

- Magic is bad. Understanding is good. Once you Understand you can control and improve.
- Word embeddings are just distributional semantics in disguise.
- Need to think of what you actually want to solve.
  --> focus on a specific task!
- Inputs >> fancy math.
- Look beyond just words.
- Look beyond just English.