Word Embeddings

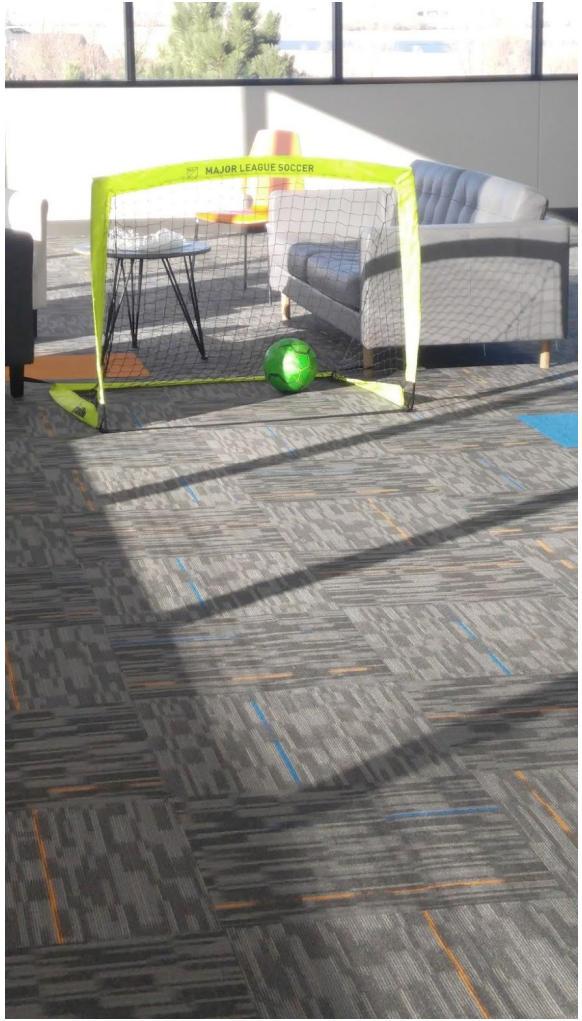
Max Lei and Grehg Hilston

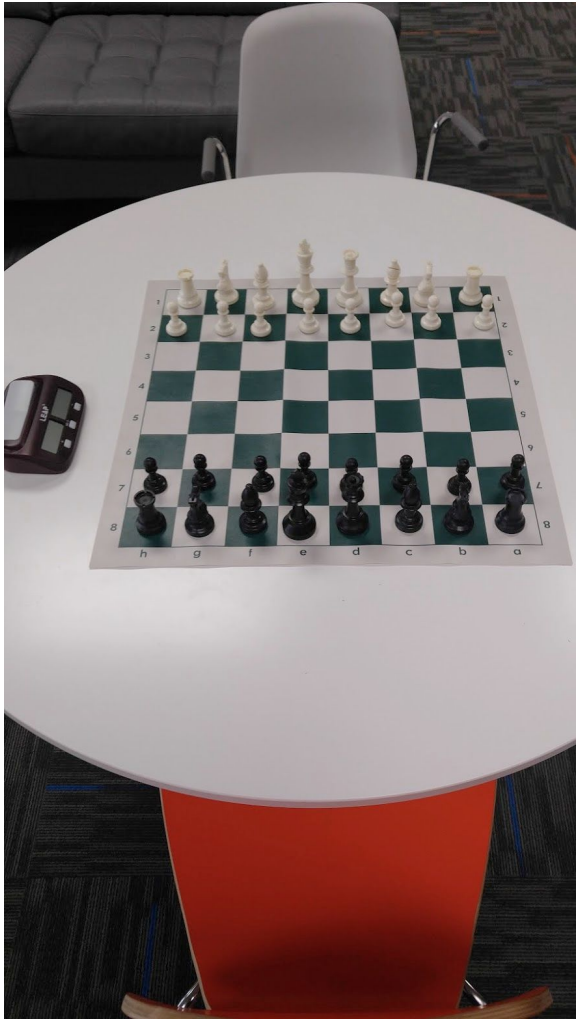
Sponsored By


EXIT


the AMAZING
SPIDER-MAN
14600
SPIDER-MAN
STAR SERIES
80
D. Gottlieb & Co.


midwire


MAJOR LEAGUE SOCCER




SWAT


JOOLA

Overview (Madwire)

1. Theory Presentation (30 minutes)
    a. Why / how this stuff works
2. Implementation Presentation (30 minutes)
    a. How to use this stuff in a project

Overview (DS Meetup)

1. Theory Presentation (20 minutes)
    a. Why / how this stuff works
2. Implementation Presentation (20 minutes)
    a. How to use this stuff in a project
3. Personal Exploration (20 minutes)
    a. Practice using this stuff in code

Vocab

- Document:
  - A single collection of words
    - Examples:
      - Books
      - Blog posts
      - CNN article
- Corpus:
  - A collection of documents

Now For Some Jokes

- I like my coffee like I like my wars, **cold**
- I like my boys like I like my sectors, **bad**
- I like my relationships like I like my source, **open**

Word Embeddings

- Definition = “words from the vocabulary are mapped to vectors of real numbers”
- By converting words to a vector representation, we can apply many mathematical operations
- There exists premade word embeddings online, created from millions of sources

Usages

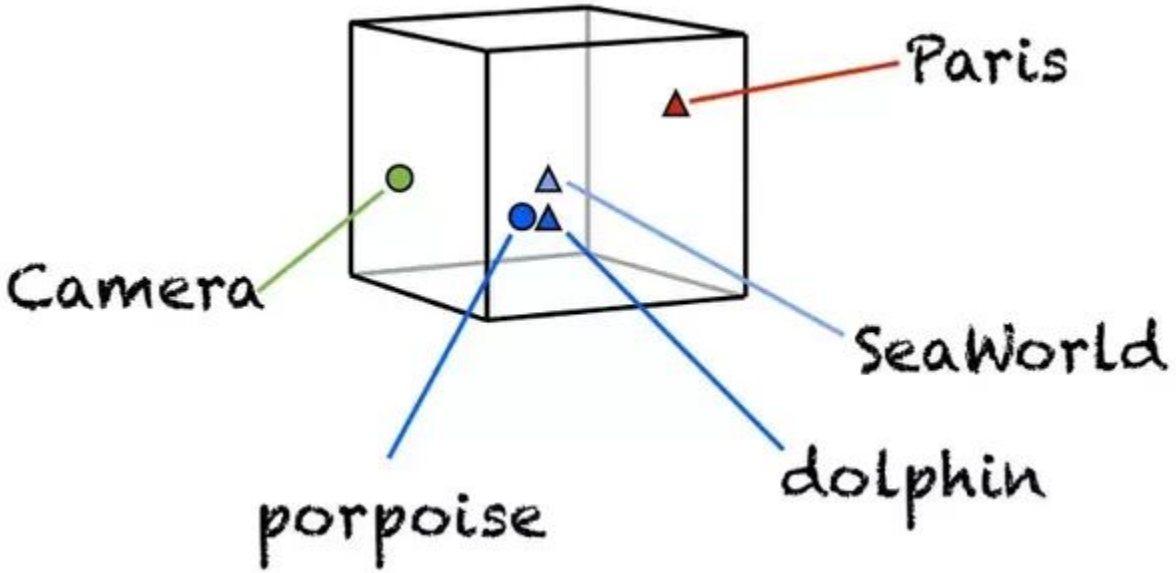
- Programmatically generating contextually similar words
- Creating simple filters
- Judging how similar two documents are
- Sentiment analysis
- Document Classification

Potential Problems

- Word ambiguity
  - Pool: billiards games
  - Pool: swimming pool
- Vectors only as good as the data you use

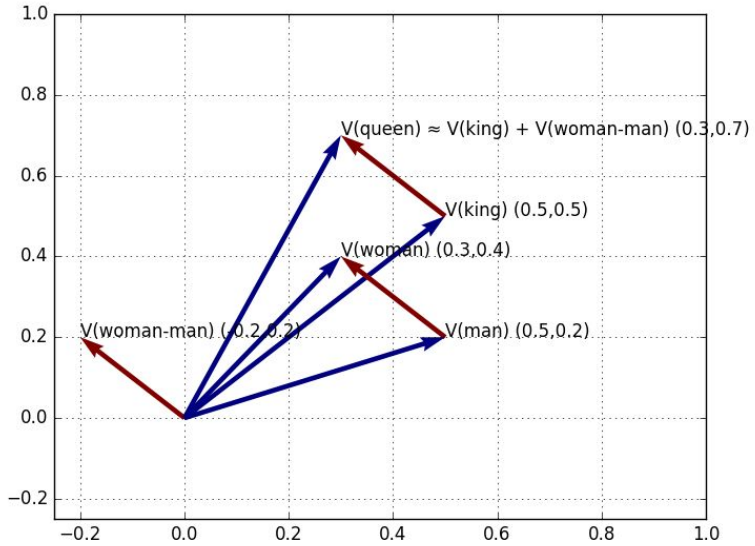
Why Are Word Embeddings Cool?

- Words with similar meanings can be clustered



Why Are Word Embeddings Cool? Part 2

- You can perform addition or subtraction on word embeddings
- Example:
  - king - man = ?
  - Vectors will produce queen



Dimension Reduction

- One can reduce dimensions of a data set
  - You'll lose some data
  - But keep the jist of what is going on
- This actual process is rather complicated

What Is A Word Embeddings?

- ELI5: Turning text into numbers
  - This process is necessary as many machine learning algorithms operate on vectors of continuous values and won’t work on strings
- Bag of Words
  - Used for huge, very space vectors
- Word Embeddings
  - Used for semantic parsing
  - Extracts meaning from text, enabling natural language understanding

Why Does This Work?

- Context clues are incredibly powerful for determining what a word is:
  - “A small, fluffy roosety climbed a tree.”
  - Q: What’s a “roosety”?
- “a word is characterized by the company it keeps” - John Rupert Firth
- Like mentioned before, a bag of words might not be able to discern this
  - However, word embeddings may be able to

Other Simple NLP Procedures

- Bag of Words
  - One vector per document
- TF-IDF
- Simple Co-Occurrence
  - One vector

Bag of Words

- We will create a dictionary for each document
  - Keys = any word that occurs at least once in any document
  - Values = count of this word in this given document
- Example:
  - Corpus
    - Document 1: “The quick brown fox jumped over the lazy dog”
    - Document 2: “The lazy dog jumped over the friendly cat”

	the	**quick**	**brown**	**fox**	**jumped**	**over**	**lazy**	**dog**	**friendly**	**cat**
d1	2	1	1	1	1	1	1	1	0	0
d2	2	0	0	0	1	1	1	1	1	1

Bag of Words Continued

Problems:

- Syntactic and semantic accuracy isn’t as high as it should because of the fact that context is king.
  - For instance; ‘Chicago’ means one thing and ‘Bulls’ means another, but ‘Chicago Bulls’ means a completely different thing.
  - Counting word-frequencies doesn’t take this into account.

TF-IDF Vector

- TF-IDF
  - Term Frequency - Inverse Document Frequency
    - Ensure that you understand this is hyphen and not a minus sign
- Term Frequency
  - the number of times a term occurs in a document
- Inverse Document Frequency
  - The inverse function of the number of documents in which the term occurs
  - Used to
    - diminish the weight of terms that occur very frequently
    - Increase the weight of the terms that occur rarely

$$tf * (\frac{1}{DF})$$

Simple Co-occurrence Vectors

- Take into account neighboring words
- Example: “I love Programming. I love Math. I tolerate Biology.”

	I	love	Programming	Math	tolerate	Biology	.
I	0	2	0	0	1	0	2
love	2	0	1	1	0	0	0
Programming	0	1	0	0	0	0	1
Math	0	1	0	0	0	0	1
tolerate	1	0	0	0	0	1	0
Biology	0	0	0	0	1	0	1
.	1	0	1	1	0	1	0

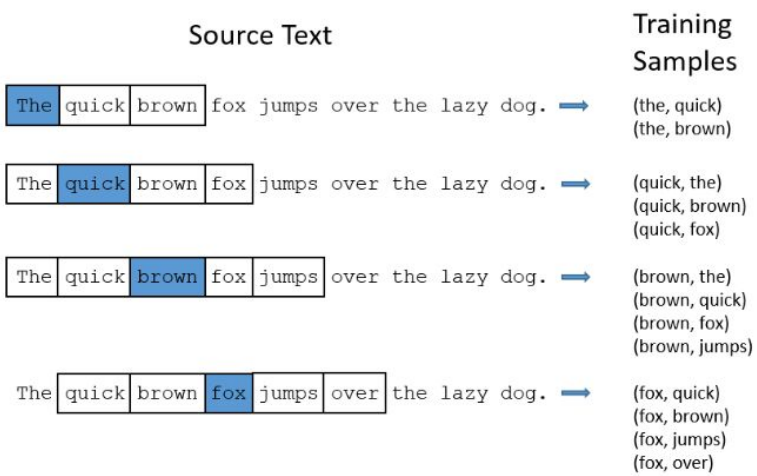
Simple Co-occurrence Vectors Continued

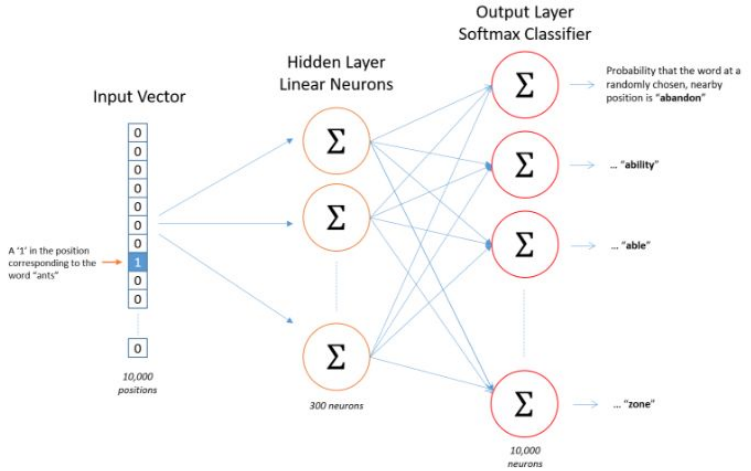
- Once we have the co-occurrence matrix filled we can plot its results into a multi-dimensional space.
- Since ‘Programming’ and ‘Math’ share the same co-occurrence values, they would be placed in the same place; meaning that in this context they mean the same thing (or ‘pretty much’ the same thing).
- The semantic and syntactic relationships generated by this technique are really powerful but it’s computationally expensive since we are talking about a very high-dimensional space.
    - Therefore, we need a technique that reduces dimensionality for us with the least data-loss possible.
        - Answer: Glove Vectors

How Are Word Vectors Made?

- Vectors are made by walking through data sets of text
- Vocab
  - Document = a single thing of related text
    - Examples:
      - A book
      - A CNN article
      - A Wikipedia page
  - Corpus = a set of documents
    - Examples:
      - An online library
      - A reddit user’s post history
- Each document in the corpus is traversed, building up the vectors by document

Creating Word Embeddings





Cosine Similarity

- Dictates how related two word embeddings are

Given two vectors of attributes, *A* and *B*, the cosine similarity, *cos(θ)*, is represented using a dot product and magnitude as

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

, where A_i and B_i are components of vector A and B respectively.

Cosine Similarity Example

1. We have the following documents
    - D1 = “Julie loves me more than Lina loves me”
    - D2 = “Jane likes me more than Julie loves me”
2. We want to know how similar these texts are, purely in terms of word counts (and ignoring word order). We begin by making a list of the words from both texts:
    - [me Julie loves Linda than more likes Jane]

Cosine Similarity Example Continued

3. Now we count the number of times each of these words appears in each text:

a.

me	2	2
Jane	0	1
Julie	1	1
Linda	1	0
likes	0	1
loves	2	1
more	1	1
than	1	1

Cosine Similarity Example Continued

4. We are not interested in the words themselves though. We are interested only in those two vertical vectors of counts. For instance, there are two instances of 'me' in each text. We are going to decide how close these two texts are to each other by calculating one function of those two vectors, namely the cosine of the angle between them.

The two vectors are, again:

d1: [2, 0, 1, 1, 0, 2, 1, 1]

d2: [2, 1, 1, 0, 1, 1, 1, 1]

Cosine Similarity Example Continued

$$\frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

A . B = (2 * 2) + (0 * 1) + (1 * 1) + (1 * 0) + (0 * 1) + (2 * 1) + (1 * 1) + (1 * 1) = 9

A . A = sqrt((2 * 2) + (0 * 0) + (1 * 1) + (1 * 1) + (0 * 0) + (2 * 2) + (1 * 1) + (1 * 1)) = sqrt(12) = 3.46

B . B = sqrt((2 * 2) + (1 * 1) + (1 * 1) + (0 * 0) + (1 * 1) + (1 * 1) + (1 * 1) + (1 * 1)) = sqrt(10) = 3.16

9 / (3.46 * 3.16) = 0.822

These vectors are 8-dimensional. A virtue of using cosine similarity is clearly that it converts a question that is beyond human ability to visualise to one that can be.

In this case you can think of this as the angle of about 35 degrees which is some 'distance' from zero or perfect agreement.

Stop Words

- Stop words: words which are filtered out before or after processing of natural language text
  - Usually refers to the most common words in a language
  - No official list exists
- Examples:
  - The
  - Is
  - At
  - Which
  - On
- Potential Problems:
  - Searching for words or phrases which use stop words
    - “The Who”
    - “Take That”

Top Rated Local's Usage For Word Embeddings

- Originally Top Rated Local had only sixty categories
- A user was required to type in an exact match to make a valid search
- So if a category called "Brewery" existed, typing in "Beer" would be invalid
- Max and I iterated over every category and produced a list of similar words to improve searchability

Inappropriate Word Generation

- During the process of generating similar words, we found categories were generating inappropriate words
- This is due to our project using pre gathered word vectors, which could have been made from any sources on the internet
- The inappropriate words all were sexual
- At first we were puzzled how we were going to sensor the words
- Q: How would you filter inappropriate words?
    - In the end, we simply subtracted “sex” from each vector and got a list of clean similar words

Useful Tools

- Spacy
  - easy mode
  - Vectors, cosine similarity, entities and vocabulary
- Gensim
  - Harder, yet more powerful
  - Vectors, cosine similarity and more
- NLTK
  - Linguistics giving subjective value
- https://projector.tensorflow.org/
  - Supah cooool

References

- Image: https://qph.fs.quoracdn.net/main-qimg-3e812fd164a08f5e4f195000fecf988f
- http://p.migdal.pl/2017/01/06/king-man-woman-queen-why.html
- https://medium.com/ai-society/jkljlj-7d6e699895c4
- https://www.quora.com/What-is-word-embedding-in-deep-learning
- https://en.wikipedia.org/wiki/Stop_words
- https://www.quora.com/Is-cosine-similarity-effective
- https://stackoverflow.com/a/1750187/1983957