ANCHORMEN
data activators

HPP – NATURAL LANGUAGE PROCESSING

14-JUN-21

NLP

MOTIVATION

NLP - APPLICATIONS

Question answering	“What year was Live Aid?”
Information Extraction	Extracting relevant information from news reports for investors.
Sentiment analysis	Determining the polarity of a review: positive, negative or neutral.
Machine translation	Translate “Where is the nearest coffee shop” to Dutch.
Profiling	What is the gender and age of an author? Or even, personality type?
Dialogue systems	Chatbots that performs many of the applications above!

NLP – STATE OF THE ART

Mostly solved	Making good progress	Remaining tough nuts to crack
Spam detection	Sentiment analysis	Question answering
Part Of Speech (POS) tagging	Coreference resolution	Paraphrasing
Name Entity Recognition (NER)	Word sense disambiguation	Summarization
	Machine translation	Dialogue
	Information extraction	
	Generating text	

THE PROBLEM

COMPLEXITY

Linguistic data is

- Symbolic
- Multidimensional
- Hierarchical
- Recursive

COMPLEXITY

- Symbolic: dog, hond, kalb, 개, หมา
- Multidimensional: Think of the multiple senses of star. Some ‘dimensions of meaning’ stay fixed, while others move depending on the context.
- Hierarchical: Overall theme -> subthemes
- Recursive: *Dorothy thinks witches are dangerous.* What follows the verb ‘think’, is another sentence. In principle, we could embedd another sentence in the embedded sentence. *Dorothy thinks that witches are, just as we would expect, dangerous.*

MORE COMPLEXITY

- Language is ambiguous

  Zij zwemt in een **meer**. ‘She swims in a **lake**.’

  Hij wil altijd **meer**. ‘He always wants **more**.’

  Visiting relatives can be **boring**.

- Language is varied (spoken vs. written, regional, sub-groups)

  A duck musta been at the door wae a parcel this morning.

- Language is idiomatic/idiosyncratic

  Hang in there!

- Language understanding is heavily reliant on context and world knowledge.

LINGUISTICS

The scientific study of language:

Goal: Build a model that explains what we know when we know a language

LINGUISTICS

- A computational model that is made up of the atoms of the language (sounds, words)
- Operations that combine these atoms
- Constraints on the combinatorics

e.g. in ensemble speech recognition models.

Also: neuroscientific research:

- What happens in a EEG, when we misinterpret an ambiguous sentence?
- How do we learn synthetic languages, constructed by a researcher?

PREPROCESSING

PREPROCESSING AND NORMALIZATION

For a lot of problems, we don’t need things like:

- Capitals
- Html tags
- Punctuation

So we often make everything lowercase, stripped of html and punctuation.

PREPROCESSING AND NORMALIZATION

For some problems, it helps to modify the words to make things more uniform.

- Tokenization
- Case folding
- Stemming
- Lemmatization

PREPROCESSING AND NORMALIZATION

- Tokenization
  - Corpora (collections of text) -> list of documents -> list of sentences -> list of words -> list of characters
- Case folding:
  - Mother, mother -> mother
- Stemming (removing prefixes and suffixes and keeping the word stem)
  - Sometimes the stem isn't a word: scientific -> scientif; unstructured -> structur
- Lemmatization:
  - is, was, were, are, am -> be
  - goed, beter, best -> goed

STOPWORDS

- The most frequent words are function words (not content words). We call them stopwords.
- Stopwords are often **removed for bag-of-word** approaches to get at the actual content
- Beware: stopwords are an integral part of phrasal verbs, idiomatic expressions:
  - Ik sta voor vrijheid
  - I like the current incarnation of Dr. Who

Word	Frequency
the	69971
of	36412
and	28853
to	26158
a	23195
in	21337
that	10594
is	10109
was	9815
he	9548
for	9489
it	8760
with	7289
as	7253
his	6996
on	6741
be	6377
at	5372
by	5306
i	5164

PREPROCESSING AND NORMALIZATION

For things like stemming, most people use libraries like:

- https://www.nltk.org
- https://spacy.io

They have out-of-the-box solutions for things like stemming, tagging, tokenizing, etc.

FEATURE GENERATION

METHODS

Simple models

- Tfidf
- N-grams
- Bag of words
- Term-context matrix

Deep models

- Word2vec
- Embeddings

TERM FREQUENCY – INVERSE DOCUMENT FREQUENCY

Intuition:

- If you have 10 documents, and a word that occurs in all 10 documents, it does not tell you much.

- If you have a word, that occurs in only 3 documents, it is probably a word that is more specific to that type of documents and will probably have a higher predictive values.

How do you discern if an email is spam? There are often typical ”signal” words, that we only see in spam messages.

We formalize this with TFIDF.

TERM FREQUENCY – INVERSE DOCUMENT FREQUENCY

This is composed of:

- N: total number of documents D in corpus
- $|\{d \in D: t \in d\}|$: number of documents d in which the term t appears.
- $idf(t, D) = \log \frac{N}{|\{d \in D: t \in d\}|}$: inverse document frequency.
- $tf(t, D)$: either
  - **raw count** of a term in a document $tf(t, d) = f_{t,d}$
  - **Boolean** (1 if t occurs, 0 otherwise)
  - **log scaled** $tf(t, d) = \log(1 + f_{t,d})$

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D)$$

N-GRAMS

A contiguous sequence of *n* items.

- Unigram (1-gram)

we	should	start	back	gared	urged	as	the	woods	began	to	grow	dark	around	them

- Bigram (2-gram)

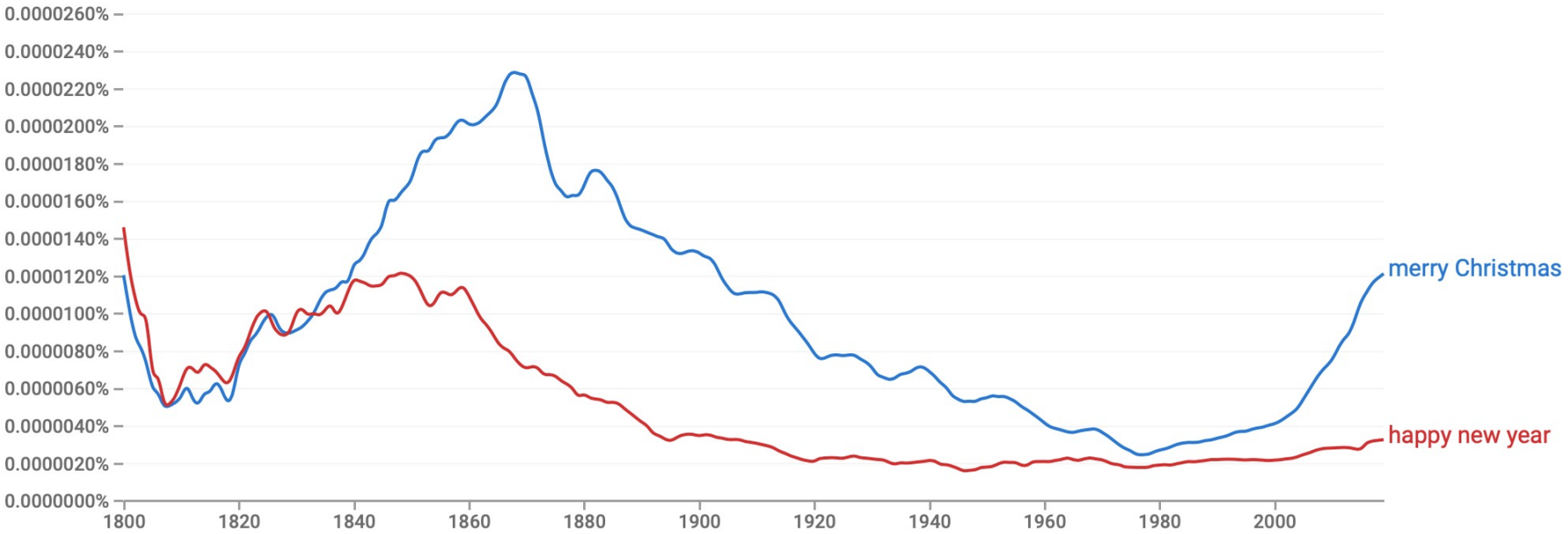
we should	should start	start back	back gared	gared urged	urged as	as the	the woods	woods began	began to	to grow	grow dark	dark around	around them

- Trigram (3-gram)

we should start	should start back	start back gared	back gared urged	gared urged as	urged as the	as the woods	the woods began	woods began to	began to grow	to grow dark	grow dark around	dark around them

N-GRAMS

- N-grams can be analyzed at the character level
- Fun with google books n-gram viewer:
  - https://books.google.com/ngrams



BAG OF WORDS

A bag-of-words is just an unstructured representation of the text without any consideration to structure or syntax.

```
['and', 'better', 'care', 'cat', 'document',
'good', 'is', 'my', 'of', 'one', 'please',
'second', 'take', 'taking', 'the', 'third',
'this', 'was', 'were', 'you']
```

BAG OF WORDS –DOCUMENT-TERM MATRIX

	GREAT	LOVE	RECOMMEND	LAUGH	HAPPY	…
Document 1	2	2	1	1	1	
Document 2	3	0	1	0	1	
Document 3	0	1	5	1	5	

Each document can be represented as:

- Doc1 = (2,2,1,1,1)
- Doc2 = (3,0,1,0,1)
- Doc3 = (0,1,5,1,5)

TERM-TERM (TERM-CONTEXT) MATRIX

You shall know a word by the company it keeps (Firth, J. R. 1957:11)

- Given the following small corpus:

Human machine interface for computer applications

User opinion of computer system response time

User interface management system

System engineering for improved response time

- We can build a co-occurrence matrix with window of size k=2.

	human	machine	system	for	...	user
human	0	1	0	1	...	0
machine	1	0	0	1	...	0
system	0	0	0	1	...	2
for	1	1	1	0	...	0
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
user	0	0	2	0	...	0

Example from Rana's Art of Vector Representations of Words

TERM-TERM (TERM-CONTEXT) MATRIX

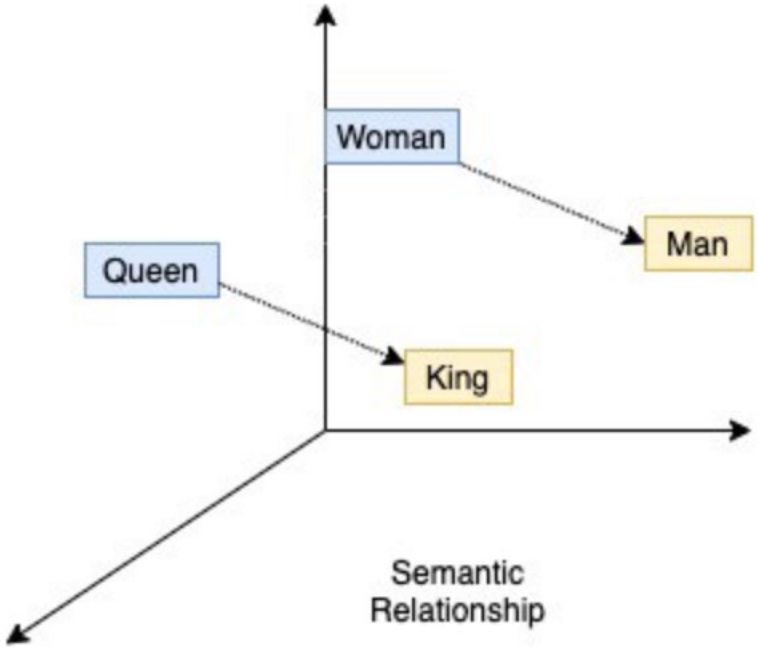
- Upsides:
  - Captures words' meaning and senses
  - Similar words have similar vectors
- Downside:
  - Sparse matrix – computationally inefficient
  - The length of the vector is the size of the vocabulary (10,000-50,000!)

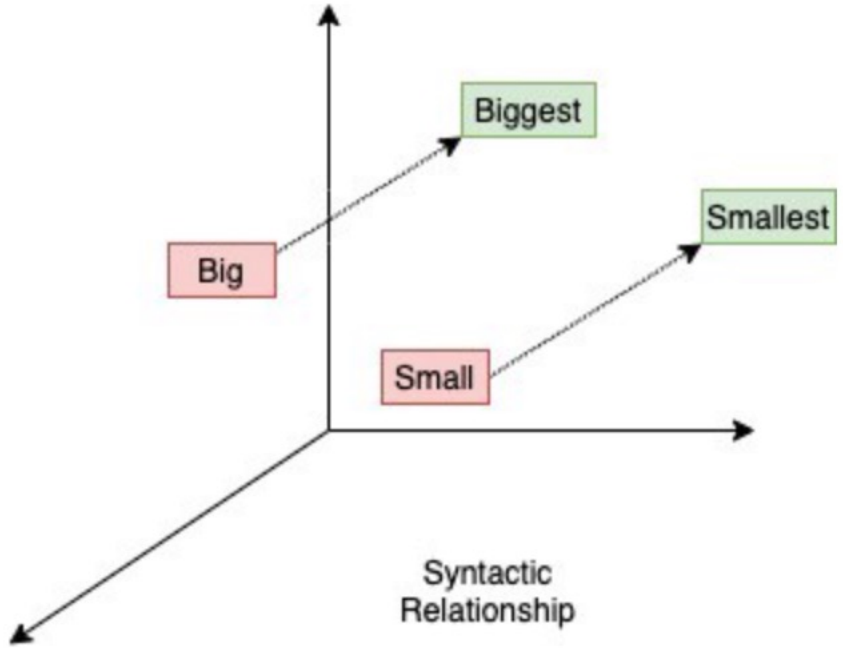
WORD2VEC

word2vec algorithm uses a neural network to learn semantic and syntactic relationships from a large corpus of text.

The vectors are chosen such that the cosine similarity indicates semantic similarity.

What embeddings do, is they simply *learn to map* the one-hot encoded categorical variables to vectors of floating point numbers of *smaller dimensionality* then the input vectors.



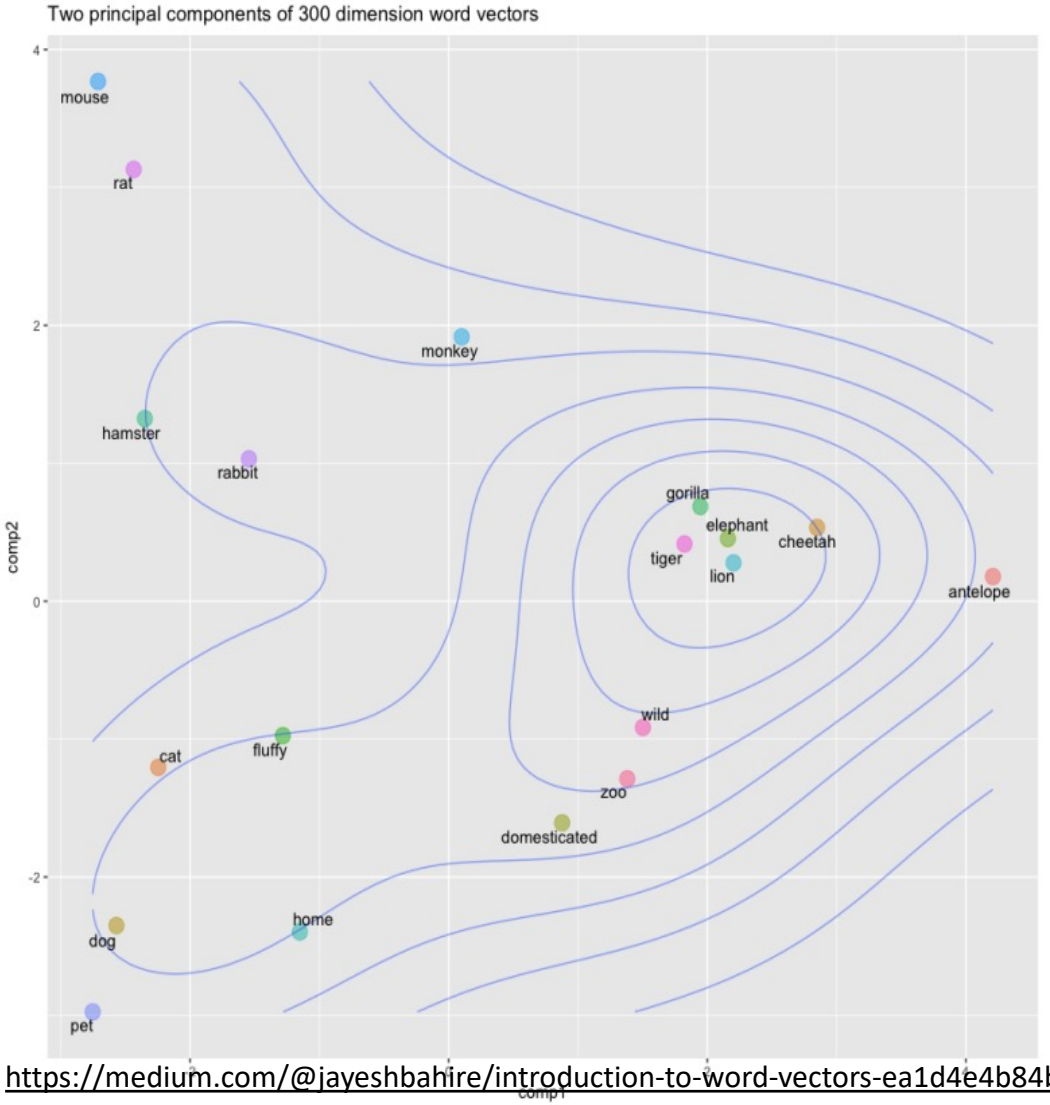


WORD EMBEDDINGS

Word2vec is trained to predict if word belongs to the context, given other words.

e.g. to tell if "milk" is a likely word given "The cat was drinking...".

With embeddings learned as a layer of a neural network, the network may be trained to predict whatever you want. For example, you can train your network to predict sentiment of a text



https://medium.com/@jayeshbahire/introduction-to-word-vectors-ea1d4e4b84bf

PREDICTING THE NEXT WORD

- Uses case: speech recognition, spelling and grammar, machine translation
- Language models: Models that assign probabilities to sentences and sequences of words
- The best language models on the market: ~~RNN~~ GPT-3

https://transformer.huggingface.co/doc/gpt2-large

PREDICTING THE NEXT WORD

https://transformer.huggingface.co/doc/gpt2-large

Story generated by GPT-2:

Legolas and Gimli advanced on the orcs, raising their weapons with a harrowing war cry. But it was only to be expected that they would be easily disarmed by their enemy. The two heroes began to swing their swords as their opponents drew their weapons , but in the end, the odds were against them. They had no way to deal with the massive creatures that swarmed them . The two heroes were forced to flee, and Gimli was forced to make the most crucial decision of his life . He began to yell the name of the one who had betrayed him.

ANCHORMEN
data activators

THANK YOU FOR YOUR ATTENTION

Pedro de Medinalaan 11,
1086 XK Amsterdam

020 - 773 1972

www.anchormen.nl