Data Wrangling and Data Analysis
Text mining #2
Sentiment analysis & Embeddings

DL Oberski

Dept of Methodology & Statistics
Utrecht University

With slides by **Dr. Dong Nguyen**

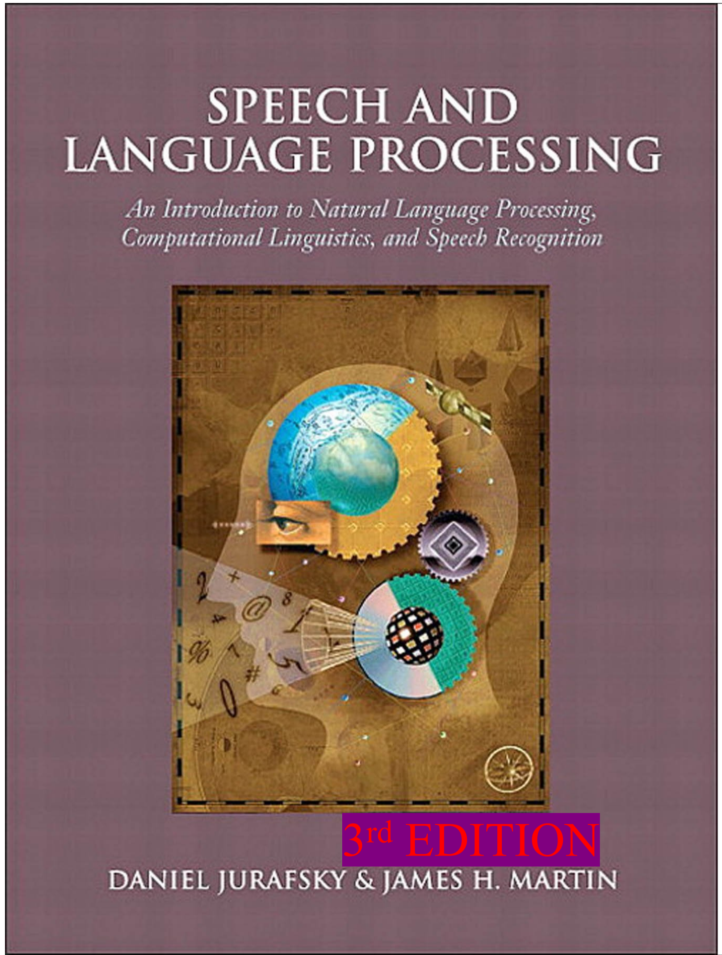
Dept of Computing Sciences
Utrecht University

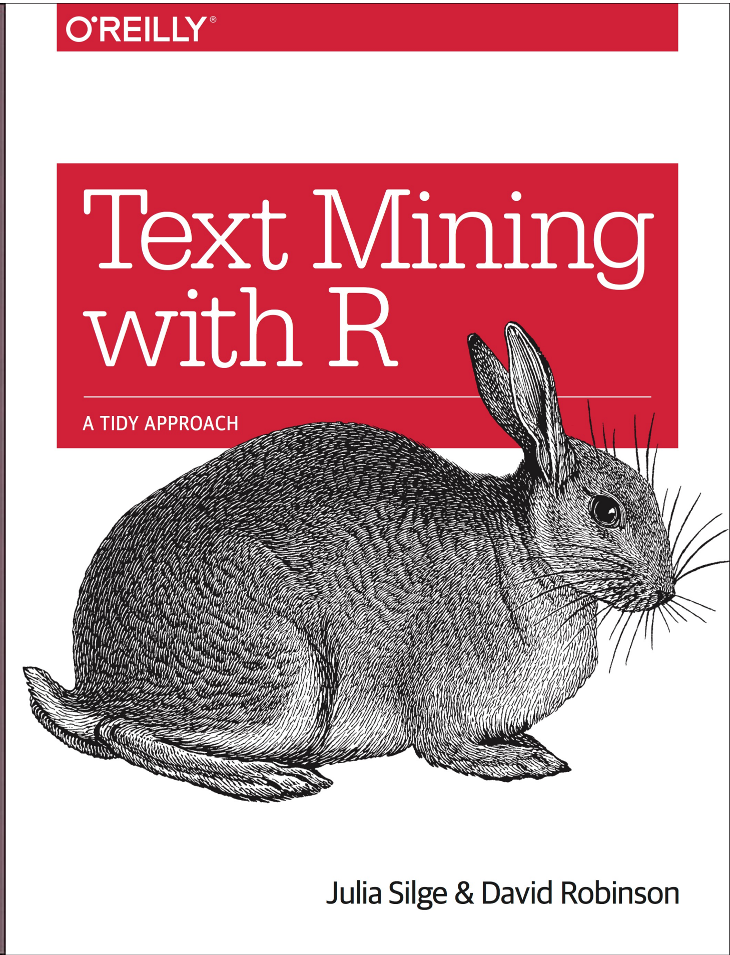
This week

- Day 1: Clustering #2: Model-based clustering
- Day 2: Text mining #1: regular expressions, BoW, TF-IDF
- **Day 3: Text mining #2: sentiment analysis, embeddings**

Readings about text mining

- Jurafsky & Martin (2021). *Speech and language processing (3rd ed draft)* https://web.stanford.edu/~jurafsky/slp3/
  - Sections
  - 2.1, 2.4, (regular expressions)
  - 6.2, 6.3, 6.4, 6.5, 6.8
- *Silge & Robinson (2021). Text mining with R: A tidy approach.* https://www.tidytextmining.com/
  - Chapter 3
- More accessible (?) intro to regular expressions:
  - **R4 data science** ch. 14
  - https://r4ds.had.co.nz/strings.html#matching-patterns-with-regular-expressions





Sentiment analysis

“task of classifying the polarity of a given text.”

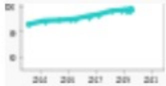
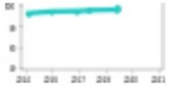
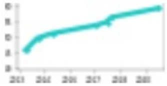
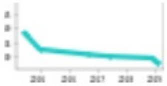
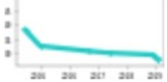
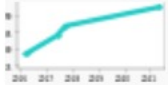
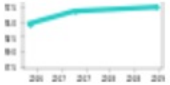
Classify the following Google reviews of UU into

☺, 😐, or ☹:

1. “Great university and great campus”
2. “Overrated university. The facilities for the humanities studies are severely outdated and really poor quality.”
3. “Good school but hideous building”

https://paperswithcode.com/task/sentiment-analysis

Benchmarks

Trend	Dataset	Best Model
	SST-2 Binary classification	🏆 SMART-RoBERTa Large
	IMDb	🏆 NB-weighted-BON + dv-cosine
	SST-5 Fine-grained classification	🏆 RoBERTa-large+Self-Explaining
	Yelp Binary classification	🏆 XLNet
	Yelp Fine-grained classification	🏆 XLNet
	Yelp Fine-grained classification	🏆 XLNet
	MR	🏆 EFL
	Amazon Review Polarity	🏆 BERT large
	Amazon Review Full	🏆 BERT large
	User and product information	🏆 BiLSTM+CHIM
	CR	🏆 EFL

Show all 32 benchmarks

Old-school sentiment analysis

- **Algorithm.** Start with a list of “positive” words and “negative” words, the “*lexicon*”. Then count them.

  Sentiment = Total no. positive words – Total no. negative words.

- Popular lexicons are: LIWC, FINN, bing, NRC, …
- Tidytext has AFINN, bing, and nrc
- There are also domain-specific sentiment lexicons, and lexicons for languages that are not English

Old-school sentiment analysis

`AFINN` lexicon (Finn Årup Nielsen):

- assigns words with a score that runs between -5 and 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment
- terms manually labelled for valence by Finn Årup Nielsen between 2009 and 2011.
- Specifically created for sentiment analysis of microblogs such as Twitter

```
get_sentiments("afinn")
```

```
## # A tibble: 2,477 x 2
##    word       value
##    <chr>      <dbl>
##  1 abandon       -2
##  2 abandoned     -2
##  3 abandons      -2
##  4 abducted      -2
##  5 abduction     -2
##  6 abductions    -2
##  7 abhor         -3
##  8 abhorred      -3
##  9 abhorrent     -3
## 10 abhors        -3
## # ... with 2,467 more rows
```

Old-school sentiment analysis

`bing` lexicon (Bing Liu and collaborators):

- categorizes words into positive and negative categories
- Developed for mining and summarizing customer reviews
- First, adjective words were identified using a natural language processing method. Second, for each opinion word, semantic orientation was determined

```
## # A tibble: 6,786 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 2-faces     negative
##  2 abnormal    negative
##  3 abolish     negative
##  4 abominable  negative
##  5 abominably  negative
##  6 abominate   negative
##  7 abomination negative
##  8 abort       negative
##  9 aborted     negative
## 10 aborts      negative
## # ... with 6,776 more rows
```

Old-school sentiment analysis

`nrc` lexicon (Saif Mohammad and Peter Turney):

- categorizes words into categories of positive, negative, anger, anticipation, disgust, fear, joy, sadness, surprise, and trust
- annotations were manually done by crowdsourcing

```
## # A tibble: 13,901 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 abacus      trust
##  2 abandon     fear
##  3 abandon     negative
##  4 abandon     sadness
##  5 abandoned   anger
##  6 abandoned   fear
##  7 abandoned   negative
##  8 abandoned   sadness
##  9 abandonment anger
## 10 abandonment fear
## # ... with 13,891 more rows
```

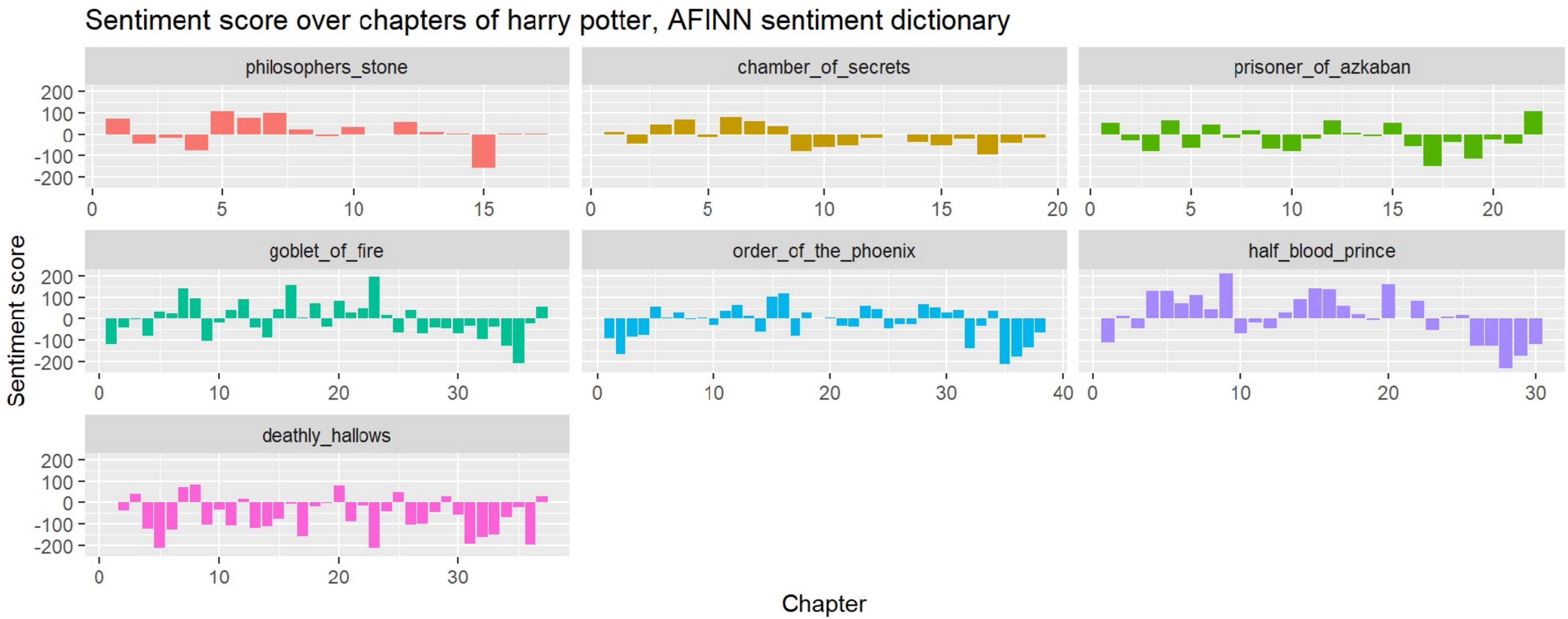
Example using NRC

Most common joy words in Harry Potter

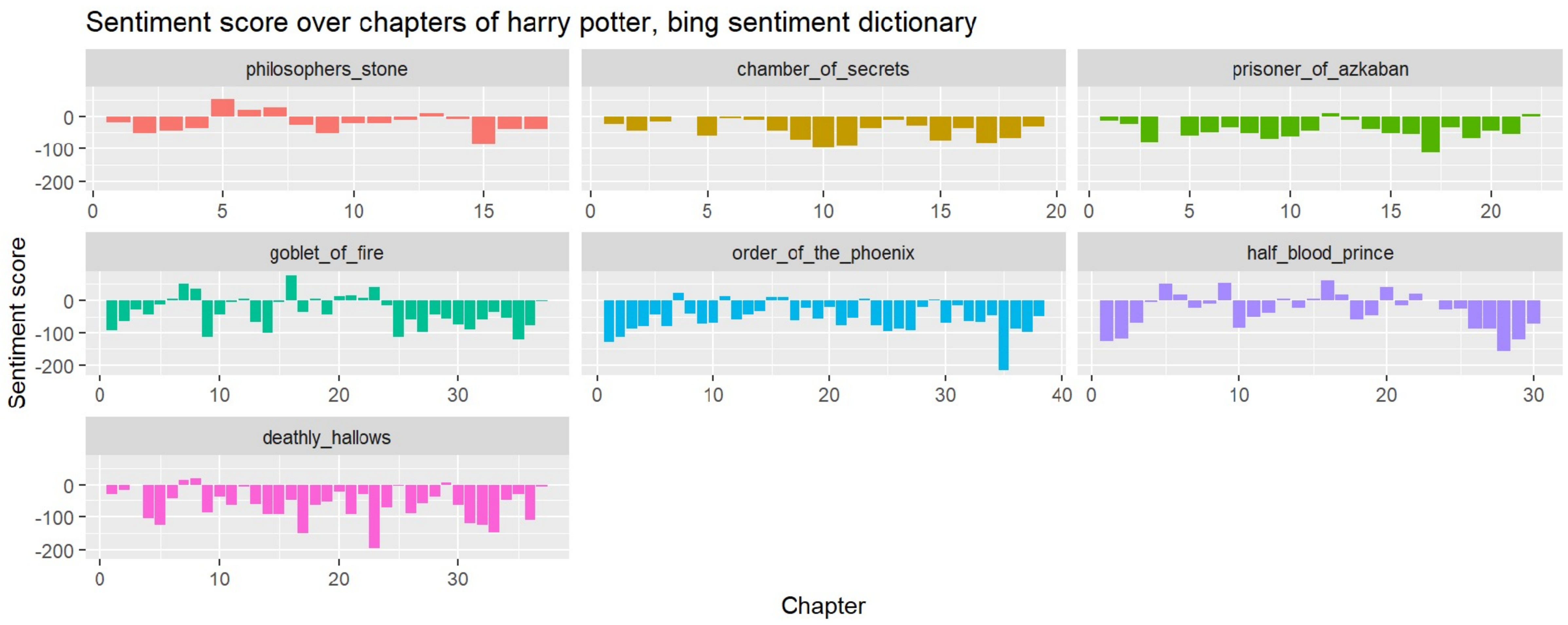
```
## # A tibble: 440 x 2
##    word         n
##    <chr>    <int>
##  1 good      1065
##  2 found      614
##  3 ministry   576
##  4 feeling    391
##  5 magical    380
##  6 white      331
##  7 green      294
##  8 mother     284
##  9 smile      244
## 10 hope       234
## # ... with 430 more rows
```

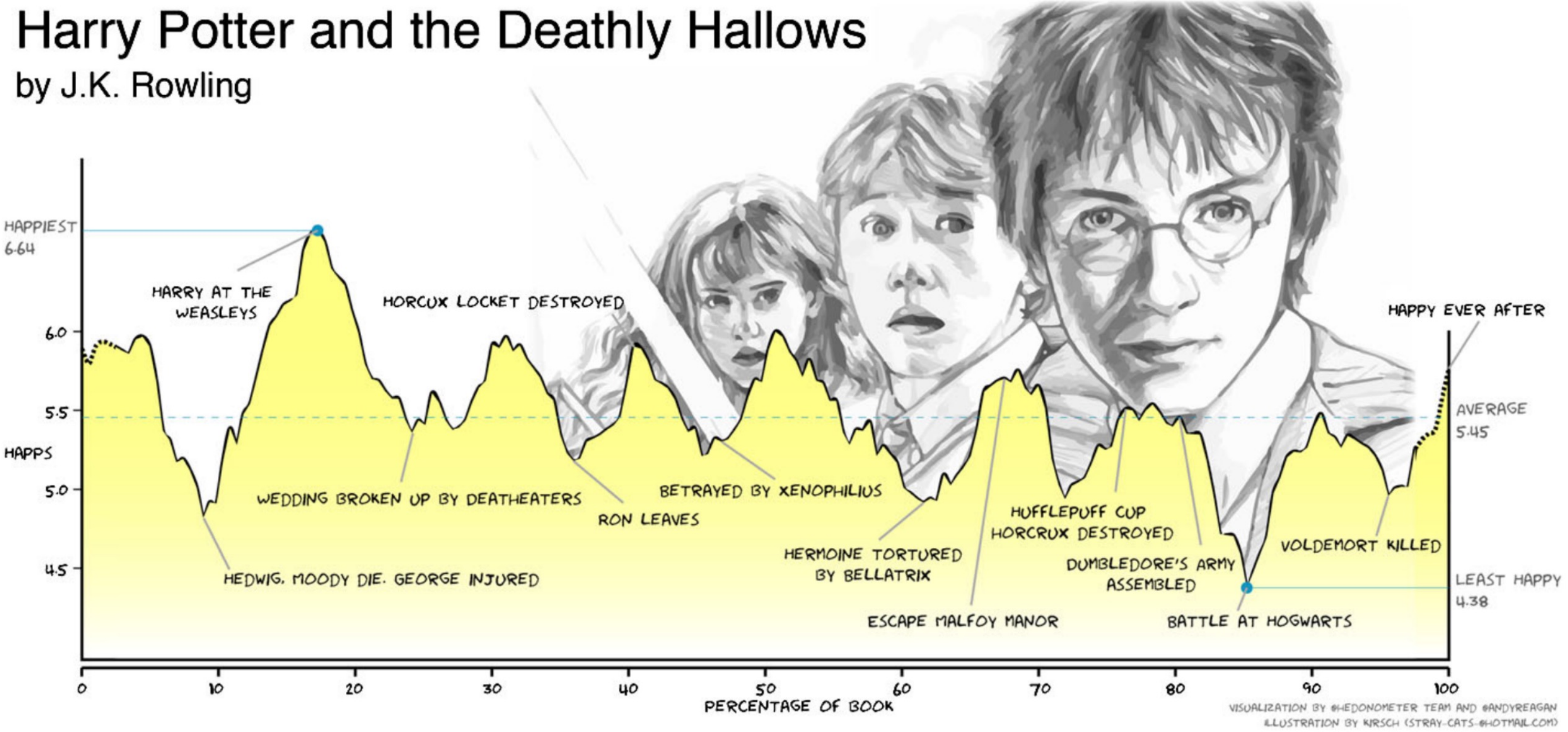
Most common fear words in Harry Potter

```
## # A tibble: 888 x 2
##    word         n
##    <chr>    <int>
##  1 death      757
##  2 feeling    391
##  3 fire       388
##  4 crouch     297
##  5 shaking    277
##  6 scar       276
##  7 mad        269
##  8 kill       267
##  9 elf        259
## 10 watch      256
## # ... with 878 more rows
```

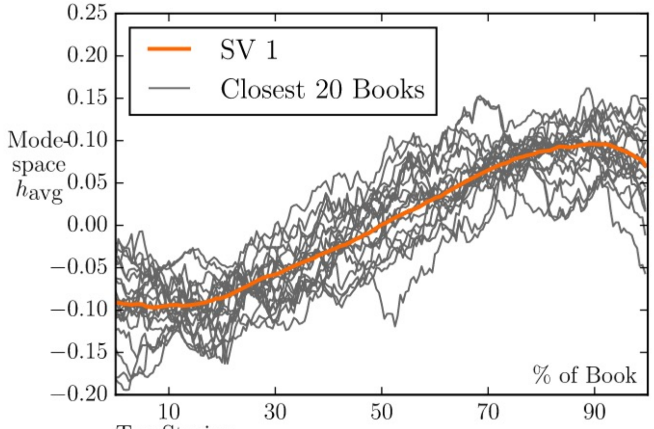


Plot of novel four to six changes towards a negative sentiment towards the end, while the seventh novel has a quite negative sentiment overall.


Sentiment score over chapters of harry potter, bing sentiment dictionary
philosophers_stone
chamber_of_secrets
prisoner_of_azkaban
goblet_of_fire
order_of_the_phoenix
half_blood_prince
deathly_hallows
Sentiment score
Chapter
0
-100
-200

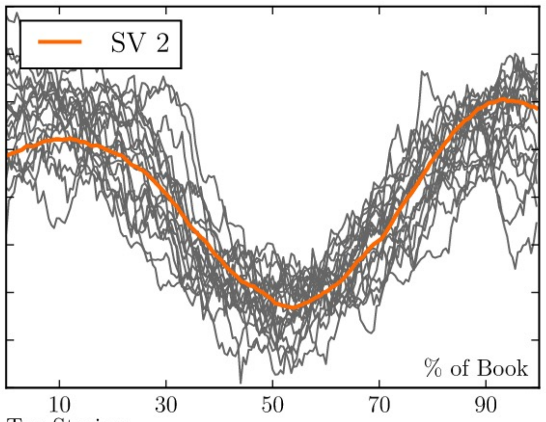


Reagan et al. (2016). The emotional arcs of stories are dominated by six basic shapes.
http://doi.org/10.1140/epjds/s13688-016-0093-1



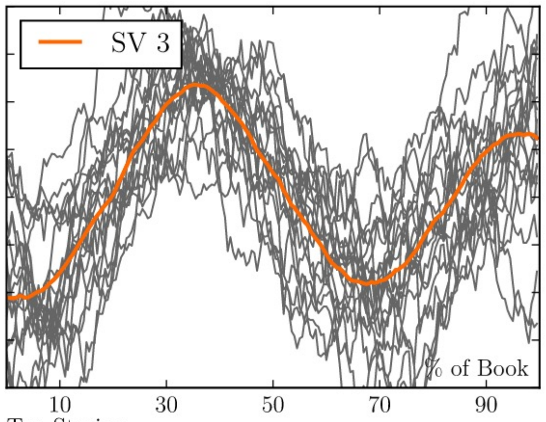
Top Stories:

1: The Winter's Tale (1539, 73)
`http://hedonometer.org/books/v3/1539/`
2: Oscar Wilde, Art and Morality: A... (33689, 88)
`http://hedonometer.org/books/v3/33689/`
3: The Terror: A Mystery (35617, 61)
`http://hedonometer.org/books/v3/35617/`
4: The Pilgrim's Progress in Words ... (7088, 55)
`http://hedonometer.org/books/v3/7088/`
5: The Road to Oz (26624, 68)
`http://hedonometer.org/books/v3/26624/`



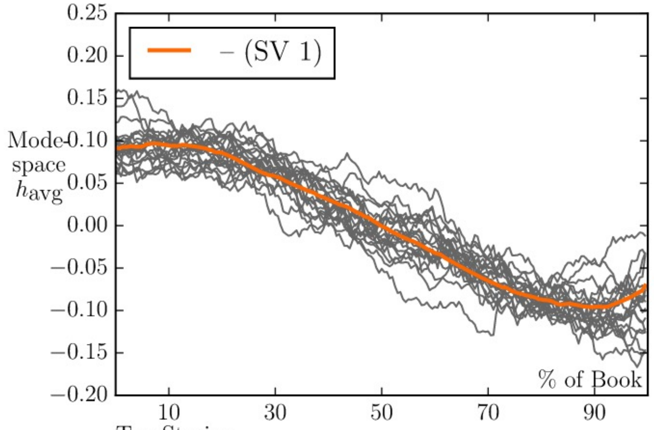
Top Stories:

1: The Magic of Oz (419, 186)
`http://hedonometer.org/books/v3/419/`
2: Children of the Frost (10736, 82)
`http://hedonometer.org/books/v3/10736/`
3: Tamburlaine the Great — Part 1 (1094, 474)
`http://hedonometer.org/books/v3/1094/`
4: The Life and Adventures of Santa... (520, 76)
`http://hedonometer.org/books/v3/520/`
5: Justice (2911, 50)
`http://hedonometer.org/books/v3/2911/`



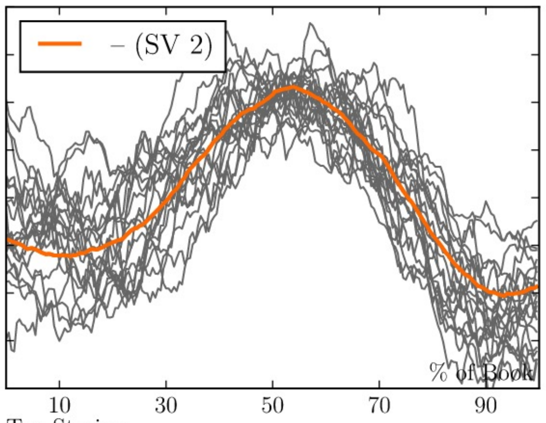
Top Stories:

1: The Mystery of the Hasty Arrow (17763, 93)
`http://hedonometer.org/books/v3/17763/`
2: Through the Magic Door (5317, 81)
`http://hedonometer.org/books/v3/5317/`
3: After London; Or, Wild England (13944, 146)
`http://hedonometer.org/books/v3/13944/`
4: The Shadow of the Rope (12590, 75)
`http://hedonometer.org/books/v3/12590/`
5: That Affair at Elizabeth (35247, 62)
`http://hedonometer.org/books/v3/35247/`



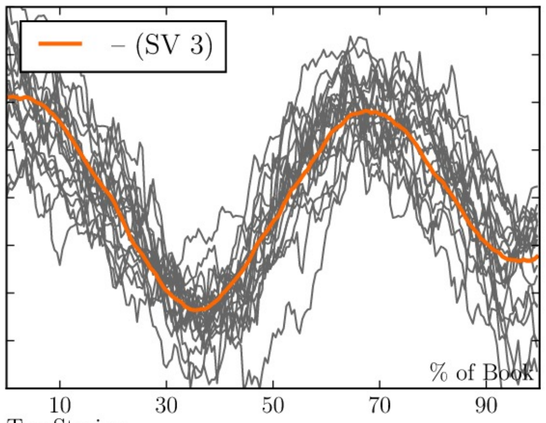
Top Stories:

1: Lady Susan (946, 894)
`http://hedonometer.org/books/v3/946/`
2: Warlord of Kor (17958, 70)
`http://hedonometer.org/books/v3/17958/`
3: The House of the Vampire (17144, 188)
`http://hedonometer.org/books/v3/17144/`
4: Tom Sawyer, Detective (93, 112)
`http://hedonometer.org/books/v3/93/`
5: The Island of Doctor Moreau (159, 1083)
`http://hedonometer.org/books/v3/159/`



Top Stories:

1: Shadowings (34215, 63)
`http://hedonometer.org/books/v3/34215/`
2: Battle-Pieces and Aspects of the... (12384, 194)
`http://hedonometer.org/books/v3/12384/`
3: The Slayer of Souls (36281, 63)
`http://hedonometer.org/books/v3/36281/`
4: The Bobbsey Twins : Or, Merry Day... (17412, 69)
`http://hedonometer.org/books/v3/17412/`
5: Allan's Wife (2727, 128)
`http://hedonometer.org/books/v3/2727/`



Top Stories:

1: This World Is Taboo (18172, 64)
`http://hedonometer.org/books/v3/18172/`
2: Old Indian Days (339, 139)
`http://hedonometer.org/books/v3/339/`
3: The Evil Guest (10377, 93)
`http://hedonometer.org/books/v3/10377/`
4: Pariah Planet (29448, 96)
`http://hedonometer.org/books/v3/29448/`
5: The Wind in the Willows (289, 1475)
`http://hedonometer.org/books/v3/289/`

Sentiment analysis

- The “old-school” (lexicon-based) method is not great with:
  - Longer texts (*why?*)
  - Negation
  - Context-dependency in general
- You can also just consider this a **classification task**, where the input data is the text and the target is categorical
- Could use BoW and TF-IDF, sometimes better
- Sometimes BoW similar problems as lexicon-based method
- Big disadvantage is that you will need (partly) labeled data

Word embeddings

based on slides by **dr. Dong Nguyen**

Word representations

How can we represent the *meaning* of words?

Word representations

How can we represent the *meaning* of words?

So we can ask:

- How similar is *cat* to *dog*, or *Paris* to *London*?
- How similar is *document A* to *document B*?

Word representations

How can we represent the *meaning* of words?

So we can ask:

- How similar is *cat* to *dog*, or *Paris* to *London*?
- How similar is *document A* to *document B*?

And use such representations for:

- various NLP tasks: translation, classification, etc.
- studying linguistic questions

Word as vectors

Key idea: Can we represent words as vectors?

The vector representations should:

- capture semantics
  - similar words should be close to each other in the vector space
  - relation between two vectors should reflect the relationship between the two words
- be efficient (vectors with fewer dimensions are easier to work with)
- be interpretable

Word as vectors

Key idea: Can we represent words as vectors?

The vector representations should:

- capture semantics
  - similar words should be close to each other in the vector space
  - relation between two vectors should reflect the relationship between the two words
- be efficient (vectors with fewer dimensions are easier to work with)
- be interpretable

How similar are *smart* and *intelligent?* (not similar 0–10 very similar):
How similar are *easy* and *big* (not similar 0–10 very similar):

Word as vectors

Key idea: Can we represent words as vectors?

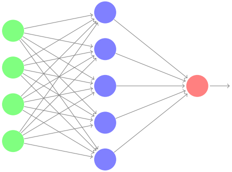
The vector representations should:

- capture semantics
  - similar words should be close to each other in the vector space
  - relation between two vectors should reflect the relationship between the two words
- be efficient (vectors with fewer dimensions are easier to work with)
- be interpretable

How similar are *smart* and *intelligent?* (not similar 0–10 very similar): 9.2
How similar are *easy* and *big* (not similar 0–10 very similar): 1.12
(*SimLex-999 dataset*)

How are they used?

How are they used?



In neural networks (text classification, sequence tagging, etc..)

cat	0.52	0.48	-0.01	⋯	0.28
dog	0.32	0.42	-0.09	⋯	0.78

As research objects

Properties

We can use cosine similarity to find similar words in the vector space.

- **dog**: *dogs, cat, man, cow, horse*
- **car**: *driver, cars, automobile, vehicle, race*
- **amsterdam**: *netherlands, rotterdam, dutch, centraal, paris*
- **chocolate**: *candy, beans, caramel, butter, liquor*

Exercise (5 min)

- Go to `https://projector.tensorflow.org/`. The site should load 'Word2Vec 10K' vectors by default (see left panel).
- What are the 5 nearest words to '*cat*'?
- What are the 5 nearest words to '*computer*'?

Words as vectors

One hot encoding

Map each word to a unique identifier
e.g. *cat (3)* and *dog (5)*.
→ Vector representation: all zeros, except 1 at the ID

cat	0	0	1	0	0	0	0
dog	0	0	0	0	1	0	0
car	0	0	0	0	0	0	1

One hot encoding

Map each word to a unique identifier
e.g. *cat (3)* and *dog (5)*.
→ Vector representation: all zeros, except 1 at the ID

cat	0	0	1	0	0	0	0
dog	0	0	0	0	1	0	0
car	0	0	0	0	0	0	1

What are limitations of one hot encodings?

One hot encoding

Map each word to a unique identifier
e.g. *cat (3)* and *dog (5)*.
→ Vector representation: all zeros, except 1 at the ID

cat	0	0	1	0	0	0	0
dog	0	0	0	0	1	0	0
car	0	0	0	0	0	0	1

Even related words have distinct vectors!

High number of dimensions

😔

Distributional hypothesis

some believe that	**wampos**	scales have medicinal qualities
approach to fighting	**wampos**	(and general wildlife) trafficking
Even though	**wampos**	scales are made of exactly the

Distributional hypothesis

some believe that	**wampos**	scales have medicinal qualities
approach to fighting	**wampos**	(and general wildlife) trafficking
Even though	**wampos**	scales are made of exactly the

What is a **wampos**?

Distributional hypothesis

Figure: Photo by Piekfrosch; CC-BY-SA-3.0

some believe that	**wampos**	scales have medicinal qualities
approach to fighting	**wampos**	(and general wildlife) trafficking
Even though	**wampos**	scales are made of exactly the

wampos = *pangolin*

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

Distributional hypothesis

Figure: Photo by Piekfrosch; CC-BY-SA-3.0

some believe that	**wampos**	scales have medicinal qualities
approach to fighting	**wampos**	(and general wildlife) trafficking
Even though	**wampos**	scales are made of exactly the

wampos = pangolin

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

The distributional hypothesis: Words that occur in similar contexts tend to have similar meanings

Word vectors based on co-occurrences

documents as context
word-document matrix

	doc_1	doc_2	doc_3	doc_4	doc_5	doc_6	doc_7
cat	5	2	0	1	4	0	0
dog	7	3	1	0	2	0	0
car	0	0	1	3	2	1	1

Word vectors based on co-occurrences

documents as context
word-document matrix

	doc_1	doc_2	doc_3	doc_4	doc_5	doc_6	doc_7
cat	5	2	0	1	4	0	0
dog	7	3	1	0	2	0	0
car	0	0	1	3	2	1	1

neighboring words as context
word-word matrix

	cat	dog	car	bike	book	house	tree
cat	0	3	1	1	1	2	3
dog	3	0	2	1	1	3	1
car	0	0	1	3	2	1	1

Word vectors based on co-occurrences

There are many variants:

- Context (words, documents, which window size, etc.)
- Weighting (raw frequency, etc.)

Vectors are sparse: Many zero entries.
Therefore: Dimensionality reduction is often used (e.g., SVD)

These methods are sometimes called **count-based** methods as they work directly on **co-occurrence** counts.

Word embeddings

Word embeddings

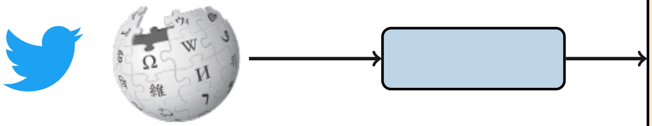
Word embeddings:

- Vectors are short; typically 50-1024 dimensions ☺
- Vectors are dense (mostly non-zero values)
- Very effective for many NLP tasks ☺
- Individual dimensions are less interpretable ☹

cat	0.52	0.48	-0.01	$\cdots$	0.28
dog	0.32	0.42	-0.09	$\cdots$	0.78

How do we learn word embeddings?

Learning word embeddings



cat =	**0.12**	. . .	**-0.2**
dog =	**0.92**	. . .	**-0.1**
tree =	**-0.12**	. . .	**0.1**
. . .	. . .	. . .	. . .

Learning word embeddings

Word2vec, GloVe, fastText

cat =	0.12	...	-0.2
dog =	0.92	...	-0.1
tree =	-0.12	...	0.1
...	...	...	...

Training data

How can we train a model to learn the meaning of words?
Which data can we use for supervised learning?

Training data

How can we train a model to learn the meaning of words?
Which data can we use for supervised learning?

Key idea:

Use text itself as training data for the model!

A form of *self-supervision*.

Training data

How can we train a model to learn the meaning of words?
Which data can we use for supervised learning?

Key idea:

Use text itself as training data for the model!

A form of *self-supervision*.

Example: Train a neural network to predict the next word given previous words.

A neural probabilistic language model. Bengio et al. (2003), JMLR [url]

Natural language processing (almost) from scratch, Collobert et al. (2011), JMLR, [url]

Exercise: Word prediction task

`yesterday I went to the` **?**

`A new study has highlighted the positive` **?**

Which word comes next?

Word2Vec

The domestic **cat** is a small, typically furry carnivorous mammal

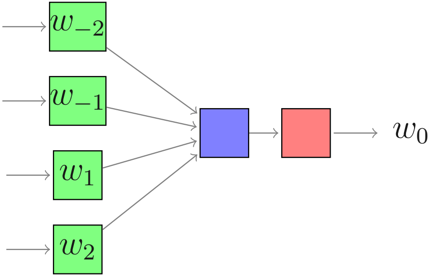
w_{-2} w_{-1} w_0 w_1 w_2 w_3 w_4 w_5

We have **target** words (*cat*) and **context** words (here: window=5).

Remember: distributional hypothesis

Word2Vec

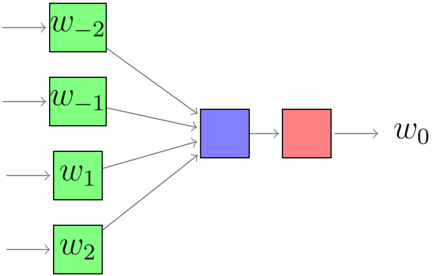
Continuous Bag-Of-Words (CBOW)



`one snowy ? she went`

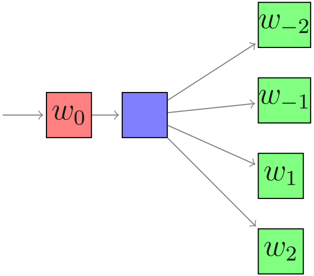
Word2Vec

Continuous Bag-Of-Words (CBOW)



one snowy **?** she went

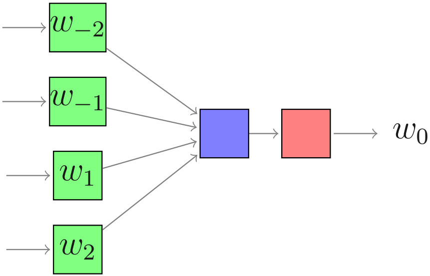
skipgram



? **?** day **?** **?**

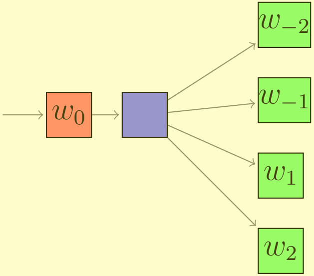
Word2Vec

Continuous Bag-Of-Words (CBOW)



`one snowy ? she went`

skipgram



`? ? day ? ?`

Word2Vec: skipgram overview

The domestic **cat** is a small, typically furry carnivorous mammal

word (w)	**context (c)**	**label**
cat	small	1
cat	furry	1
cat	car	0
...	...	...

Word2Vec: skipgram overview

The domestic **cat** is a small, typically furry carnivorous mammal

word (w)	context (c)	label
cat	small	1
cat	furry	1
cat	car	0
…	…	…

1. **Create examples**
   - Positive examples: Target word and neighboring context
   - Negative examples: Target word and randomly sampled words from the lexicon (*negative sampling)*
2. Train a **logistic regression** model to distinguish between the positive and negative examples
3. The resulting **weights** are the embeddings!

Word2Vec: skipgram overview

The domestic **cat** is a small, typically furry carnivorous mammal

word (w)	**context (c)**	**label**
cat	small	1
cat	furry	1
cat	car	0
…	…	…

Embedding vectors are essentially a byproduct!

1. **Create examples**
   - Positive examples: Target word and neighboring context
   - Negative examples: Target word and randomly sampled words from the lexicon (*negative sampling)*
2. Train a **logistic regression** model to distinguish between the positive and negative examples
3. The resulting **weights** are the embeddings!

Word2Vec: skipgram

The domestic **cat** is a small, typically furry carnivorous mammal

$c1$ $c2$ w $c3$ $c4$ $c5$ $c6$ $c7$

We have **target** words (*cat*) and **context** words (here: window=5).

The probability that *c* is a real context word:

$$P(+|w,c)$$

The probability that *c* is not a real context word:

$$P(-|w,c)$$

See also: 6.8 of Speech and Language Processing (3rd ed. draft), Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

Word2Vec: skipgram

Intuition: A word c is likely to occur near the target if its embedding is similar to the target embedding.

$$\approx w \cdot c$$

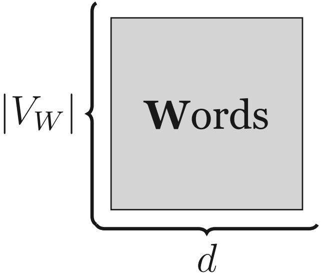
Turn this into a probability using the sigmoid function

$$P(+|w,c) = \frac{1}{1+e^{-w \cdot c}}$$

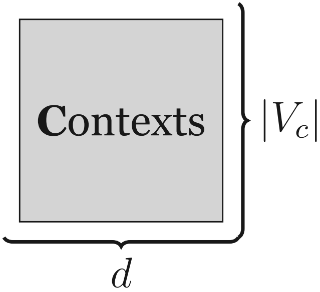
See also: 6.8 of Speech and Language Processing (3rd ed. draft), Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

Word2Vec

Words:
Each word w is represented as a d-dimensional vector.



Contexts:
Each word w is represented as a d-dimensional vector.



All vectors are initialized with random weights.

Word2vec: skipgram (learning)

We **start** with random embedding vectors.

Word2vec: skipgram (learning)

We **start** with random embedding vectors.

During training:

- *Maximize* the similarity between the embeddings of the target word and context words from the positive examples
- *Minimize* the similarity between the embeddings of the target word and context words from the negative examples

Word2vec: skipgram (learning)

We **start** with random embedding vectors.

During training:

- *Maximize* the similarity between the embeddings of the target word and context words from the positive examples
- *Minimize* the similarity between the embeddings of the target word and context words from the negative examples

After training:

- frequent word-context pairs in data: $w \cdot c$ high
- not word-context pairs in data: $w \cdot c$ low

So: Words occurring in same contexts are close to each other

Word2vec: skipgram (learning)

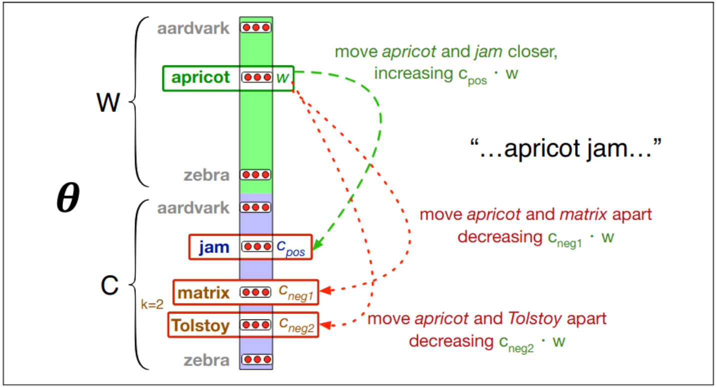


Figure: Figure 6.14 from Speech and Language Processing (3rd ed. draft), Dan Jurafsky and James H. Martin
https://web.stanford.edu/~jurafsky/slp3/

Pre-trained embeddings

- I want to build a system to solve a task (e.g. sentiment analysis)
  - Use pre-trained embedddings. Should I fine-tune?
    - Lots of data: yes
    - Just a small dataset: no
- Analysis (e.g. bias, semantic change)
  - Train embeddings from scratch

Properties of word embeddings

Properties of word embeddings

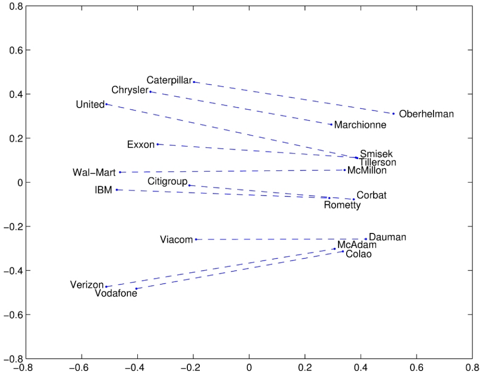


Figure: company - ceo

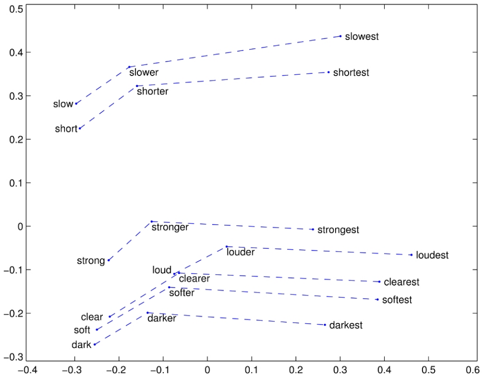


Figure: comparative - superlative

Source: https://nlp.stanford.edu/projects/glove/

Properties of word embeddings: analogies

We can look at analogies in the vector space, for example:
king - man + woman ≈ queen

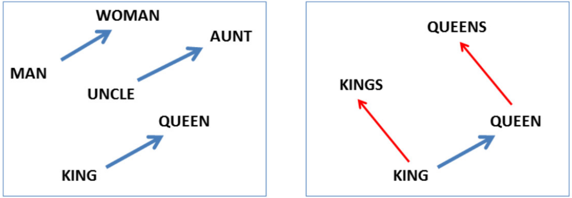
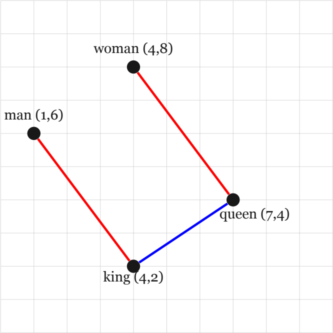


Figure: Figure 2 from Linguistic Regularities in Continuous Space Word Representations, Mikolov et al. NAACL 2013 [url]

Properties of word embeddings: analogies

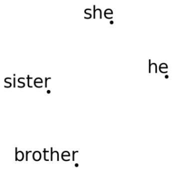
We can look at analogies in the vector space, for example:
king - man + woman ≈ queen




```
king-man = [4,2] - [1,6] = [3,-4]
king-man + woman = [3,-4] + [4,8] = [7,4]
```

Biases in word embeddings

Biases in word embeddings



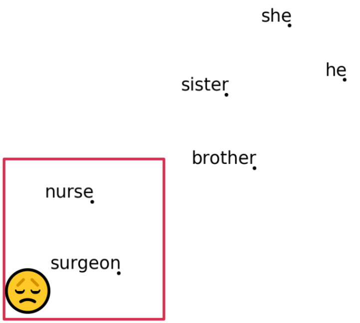
Measuring gender bias:

- To assess NLP models and investigate the impact of ‘bias mitigation’ techniques
- To study societal trends

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings, Bolukbasi, et al. NIPS 2016 [url]

Semantics derived automatically from language corpora contain human-like biases, Caliskan, Bryson, Narayanan, Science 2017 [url]

Biases in word embeddings



Pre-trained GloVe model on Twitter

Measuring gender bias:

- To assess NLP models and investigate the impact of 'bias mitigation' techniques
- To study societal trends

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings, Bolukbasi, et al. NIPS 2016 [url]

Semantics derived automatically from language corpora contain human-like biases, Caliskan, Bryson, Narayanan, Science 2017 [url]

Biases reflected in analogy tasks

Biases reflected in analogy tasks:

man is to *computer programmer* as *woman* is to ? : x = homemaker
father is to *doctor* as *mother* is to ? : x = nurse

Note: Input words are excluded as possible answers! (see also Nissim et al. 2020 [url])

Compare: gender-specific words (e.g., *brother*, *businesswoman*) vs. *gender-neutral* words (e.g. *nurse*, *teacher*).

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings, Bolukbasi, et al. NIPS 2016 [url]

Word-Embedding Association Test

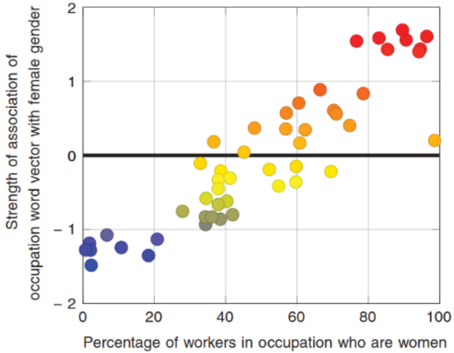


Fig. 1. Occupation-gender association. Pearson's correlation coefficient $\rho = 0.90$ with $P < 10^{-18}$.

Figure from: Semantics derived automatically from language corpora contain human-like biases, Caliskan, Bryson, Narayanan, Science 2017 [url]

Perpetuation of bias in sentiment analysis

> *"I had tried building an algorithm for sentiment analysis based on word embeddings [..]. When I applied it to restaurant reviews, I found it was ranking Mexican restaurants lower. The reason was not reflected in the star ratings or actual text of the reviews. It's not that people don't like Mexican food.* ***The reason was that the system had learned the word "Mexican" from reading the Web****."*

(emphasis mine)
http://blog.conceptnet.io/posts/2017/conceptnet-numberbatch-17-04-better-less-stereotyped-word-vectors/

Resources

Readings:

- *Contextual Word Representations: Putting Words into Computers*, Noah A. Smith, 2020 https://cacm.acm.org/magazines/2020/6/245162-contextual-word-representations/fulltext
- *Vector Semantics and Embeddings (Chapter 6)*, Speech and Language Processing (3rd ed. draft), Dan Jurafsky and James H. Martin, 2020 https://web.stanford.edu/~jurafsky/slp3/

Video's:

- *Stanford CS224N: NLP with Deep Learning | Winter 2019 | Lecture 1 – Introduction and Word Vectors* (and lecture 2): https://www.youtube.com/watch?v=8rXD5-xhemo
- video's by Jordan Boyd-Graber, e.g. *Understanding Word2Vec* https://www.youtube.com/watch?v=QyrUentbkvw and others

Resources: blogposts

- *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)* by Jay Alammar
  http://jalammar.github.io/illustrated-bert/ (2018)
- *The Illustrated Word2vec* by Jay Alammar
  http://jalammar.github.io/illustrated-word2vec/ (2019)
- *Generalized Language Models* by Lilian Weng
  https://lilianweng.github.io/lil-log/2019/01/31/generalized-language-models.html

Conclusion

- Lexicon-based sentiment analysis: just count ”positive” and “negative” words;
- Embeddings are SotA for many NLP tasks, including (but not limited to!) sentiment analysis;
- Key idea is the “distributional hypothesis”: “you will know a word by the company it keeps”;
- → Map each word into a low-dimensional vector space
- Or, in English: assign a bunch of numbers to each word, in such a way that “similar” words are closer together;
- Very fast-moving field.