# 92586 Computational Linguistics

## Lesson 4. More Math

Alberto Barrón-Cedeño

Alma Mater Studiorum-Università di Bologna
a.barron@unibo.it @_albarron_

10/03/2020

# Previously

- Pre-processing
- BoW representation

# Previously

- Pre-processing
- BoW representation
- One rule-based sentiment model
- One statistical model (Naïve Bayes)

# Table of Contents



These slides cover roughly chapter 3 of Lane et al. (2019)

## From BoW to tf

# Intuition

1. The frequency of a token in a document is an important factor of its **relevance**

# Intuition

1. The frequency of a token in a document is an important factor of its **relevance**
2. The relative frequency of a word in a document wrt **all other documents** in the collection provide better information

# Binary Bag of Words

We departed from a binary representation.

We are simply interested in the existence (or not) of a word in a document.

# Binary Bag of Words

We departed from a binary representation.

We are simply interested in the existence (or not) of a word in a document.

$$
\begin{array}{lcl}
d_1 & = & [\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ ] \\
d_2 & = & [\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ ]
\end{array}
$$

# “Counting” Bag of Words

The more times a word occurs, the more meaning it contributes to the document

# “Counting” Bag of Words

The more times a word occurs, the more meaning it contributes to the document

A document with many occurrences of “good”, “awesome”, “best” is more **positive** than one in which they occur only once.

# “Counting” Bag of Words

The more times a word occurs, the more meaning it contributes to the document

A document with many occurrences of “good”, “awesome”, “best” is more **positive** than one in which they occur only once.

$$
\begin{array}{lcl}
d_1 & = & [\ 0\ \ 1\ \ 0\ \ 0\ \ 2\ \ 0\ \ 1\ \ 3\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ ] \\
d_2 & = & [\ 2\ \ 3\ \ 5\ \ 0\ \ 0\ \ 0\ \ 0\ \ 4\ \ 0\ \ 0\ \ 0\ \ 4\ \ 2\ ]
\end{array}
$$

# "Counting" Bag of Words

The more times a word occurs, the more meaning it contributes to the document

A document with many occurrences of "good", "awesome", "best" is more **positive** than one in which they occur only once.

$$
\begin{array}{lcl}
d_1 & = & [\ 0\ 1\ 0\ 0\ 2\ 0\ 1\ 3\ 0\ 0\ 0\ 0\ 0\ ] \\
d_2 & = & [\ 2\ 3\ 5\ 0\ 0\ 0\ 0\ 4\ 0\ 0\ 0\ 4\ 2\ ]
\end{array}
$$

Let us see the difference...

# "Counting" Bag of Words

The more times a word occurs, the more meaning it contributes to the document

A document with many occurrences of "good", "awesome", "best" is more **positive** than one in which they occur only once.

$$
\begin{array}{lcl}
d_1 & = & [\; 0 \;\; 1 \;\; 0 \;\; 0 \;\; 2 \;\; 0 \;\; 1 \;\; 3 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;] \\
d_2 & = & [\; 2 \;\; 3 \;\; 5 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 4 \;\; 0 \;\; 0 \;\; 0 \;\; 4 \;\; 2 \;]
\end{array}
$$

Let us see the difference. . .

Already a useful representation to diverse tasks, such as detecting **spam** and computing **"sentiment"**

# *tf*: Term Frequency

The number of times a word occurs in a document

# *tf*: Term Frequency

The number of times a word occurs in a document

(In general!) the times a word occurs depends on the length of the document.

- Shorter document $\rightarrow$ lower frequencies
- Longer document $\rightarrow$ higher frequencies

# *tf*: Term Frequency

The number of times a word occurs in a document

(In general!) the times a word occurs depends on the length of the document.

- Shorter document $\rightarrow$ lower frequencies
- Longer document $\rightarrow$ higher frequencies

Ideally, our counting should be document-length independent.

# *tf*: Term Frequency

The number of times a word occurs in a document

(In general!) the times a word occurs depends on the length of the document.

- Shorter document $\rightarrow$ lower frequencies
- Longer document $\rightarrow$ higher frequencies

Ideally, our counting should be document-length independent.

**Normalisation!**

# *tf*: Term Frequency (Normalised)

Why should we normalise?

$d_1$ contains word `dog` 3 times
$d_2$ contains word `dog` 100 times

`dog` is way more important for $d_2$ than for $d_1$, right?

# *tf*: Term Frequency (Normalised)

Why should we normalise?

$d_1$ contains word `dog` 3 times
$d_2$ contains word `dog` 100 times

`dog` is way more important for $d_2$ than for $d_1$, right?

$d_1$ is an email by a veterinarian (30 words)
$d_2$ is *War & Peace* (580,000 words!)

# *tf*: Term Frequency (Normalised)

Why should we normalise?

$d_1$ contains word `dog` 3 times
$d_2$ contains word `dog` 100 times

`dog` is way more important for $d_2$ than for $d_1$, right?

$d_1$ is an email by a veterinarian (30 words)
$d_2$ is *War & Peace* (580,000 words!)

$$\begin{aligned} tf(\text{dog}, d_1) &= 3/30 &= 0.1 \\ tf(\text{dog}, d_2) &= 100/580000 &= 0.00017 \end{aligned}$$

# $tf$: Term Frequency (Normalised)

Why should we normalise?

$d_1$ contains word `dog` 3 times
$d_2$ contains word `dog` 100 times

`dog` is way more important for $d_2$ than for $d_1$, right?

$d_1$ is an email by a veterinarian (30 words)
$d_2$ is *War & Peace* (580,000 words!)

$$\begin{aligned} tf(\text{dog}, d_1) &= 3/30 &= 0.1 \\ tf(\text{dog}, d_2) &= 100/580000 &= 0.00017 \end{aligned}$$

**Remember:** normalised frequencies are indeed probabilities

# *tf*: Term Frequency (Normalised)

Playing with a longer text

https://en.wikipedia.org/wiki/Coronavirus_disease_2019

```
Coronavirus disease 2019 (COVID-19) is an infectious
disease caused by severe acute respiratory syndrome
coronavirus 2 (SARS coronavirus 2, or SARS-CoV-2), a
virus closely related to the SARS virus. The disease was
discovered and named during the 2019{20 coronavirus
outbreak. Those affected may develop a fever, dry cough,
fatigue, and shortness of breath. A sore throat, runny
nose or sneezing is less common. While the majority of
cases result in mild symptoms, some can progress to
pneumonia and multi-organ failure.
[...]
```

**Note.** The examples use NLTK. Nowadays, there are better tools. For instance, parsing with **spaCy** is faster and more accurate

# *tf*: Term Frequency (Normalised)

Playing with a longer text

- Loading frequencies into a dictionary
- Vectorising frequencies
- Normalising frequencies

# *tf*: Term Frequency

From a single to multiple documents

- The vectors have to be comparable across documents → **normalisation**

See https://en.wikipedia.org/wiki/Sparse_matrix

# *tf*: Term Frequency

From a single to multiple documents

- The vectors have to be comparable across documents → **normalisation**
- Each value in the vectors must represent **the same word**

See https://en.wikipedia.org/wiki/Sparse_matrix

# *tf*: Term Frequency

From a single to multiple documents

- The vectors have to be comparable across documents $\rightarrow$ **normalisation**
- Each value in the vectors must represent **the same word**

This is when representations become sparse: many values become 0

Sparse vector most of the elements are **zero**

Dense vector most of the elements are **nonzero**

See `https://en.wikipedia.org/wiki/Sparse_matrix`

# *tf*: Term Frequency

From a single to multiple documents

- The vectors have to be comparable across documents → **normalisation**
- Each value in the vectors must represent **the same word**

This is when representations become sparse: many values become 0

Sparse vector most of the elements are **zero**

Dense vector most of the elements are **nonzero**

Let us see

See `https://en.wikipedia.org/wiki/Sparse_matrix`

# Vectors of Term Frequency

**Vectors**

- Primary building blocks of linear algebra
- Ordered list of numbers, or coordinates, in a vector space

# Vectors of Term Frequency

**Vectors**

- Primary building blocks of linear algebra
- Ordered list of numbers, or coordinates, in a vector space
- They describe a location or position in that space...
- or identify a particular direction/magnitude/distance in that space

# Vectors of Term Frequency

**Vectors**

- Primary building blocks of linear algebra
- Ordered list of numbers, or coordinates, in a vector space
- They describe a location or position in that space. . .
- or identify a particular direction/magnitude/distance in that space

**A space** is the collection of all possible vectors that could appear in that space

$[1, 4] \rightarrow$ 2D vector space
$[1, 4, 9] \rightarrow$ 3D vector space

# Vectors of Term Frequency

**Vectors**

- Primary building blocks of linear algebra
- Ordered list of numbers, or coordinates, in a vector space
- They describe a location or position in that space...
- or identify a particular direction/magnitude/distance in that space

**A space** is the collection of all possible vectors that could appear in that space

$[1, 4] \rightarrow$ 2D vector space
$[1, 4, 9] \rightarrow$ 3D vector space

We have a 18D vectors space (we have seen 1kD and bigger ones!)

# Comparing Vectors

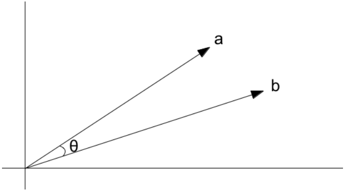
## Cosine similarity

The cosine of the angle between two vectors ($\theta$ theta)

# Comparing Vectors

Cosine similarity

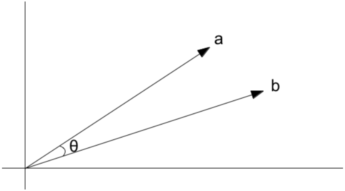
The cosine of the angle between two vectors ($\theta$ theta)



# Comparing Vectors

## Cosine similarity

The cosine of the angle between two vectors ($\theta$ theta)

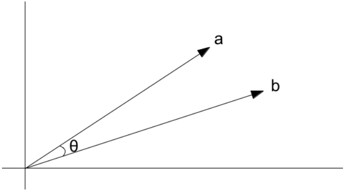



$$cos\,\theta = \frac{A \cdot B}{|A||B|} \qquad (1)$$

# Comparing Vectors

## Cosine similarity

The cosine of the angle between two vectors ($\theta$ theta)




$$cos\,\theta = \frac{A \cdot B}{|A||B|} \tag{1}$$

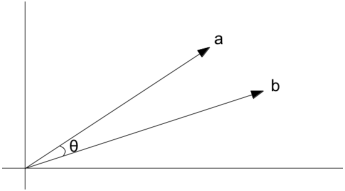
where

$A \cdot B$ is the **dot product** (we know it!)

# Comparing Vectors

Cosine similarity

The cosine of the angle between two vectors ($\theta$ theta)




$$cos\,\theta = \frac{A \cdot B}{|A||B|} \tag{1}$$

where

$A \cdot B$ is the **dot product** (we know it!)

$|A|$ is the **magnitude** of vector $A$

Let us see an implementation (but there are efficient libraries to do it)

# Comparing Vectors

## Cosine similarity

Properties of the cosine similarity

- It is ranged in $[-1, 1]$
- This is a very convenient range for ML

# Comparing Vectors

## Cosine similarity

Properties of the cosine similarity

- It is ranged in $[-1, 1]$
- This is a very convenient range for ML

- $cos = 1$ represents identical normalized vectors that point in exactly the same direction

# Comparing Vectors

## Cosine similarity

Properties of the cosine similarity

- It is ranged in $[-1, 1]$
- This is a very convenient range for ML

- $cos = 1$ represents identical normalized vectors that point in exactly the same direction
- $cos = 0$ represents two vectors that share no components (they are perpendicular in all dimensions)the closer the two vectors are in angle

# Comparing Vectors

## Cosine similarity

Properties of the cosine similarity

- It is ranged in $[-1, 1]$
- This is a very convenient range for ML

- $cos = 1$ represents identical normalized vectors that point in exactly the same direction
- $cos = 0$ represents two vectors that share no components (they are perpendicular in all dimensions)the closer the two vectors are in angle

- In NLP, at least for *tf*-like representation, it is ranged in $[0, 1]$ (frequencies are not negative)

# Comparing Vectors

## Cosine similarity

Properties of the cosine similarity

- It is ranged in $[-1, 1]$
- This is a very convenient range for ML

- $cos = 1$ represents identical normalized vectors that point in exactly the same direction
- $cos = 0$ represents two vectors that share no components (they are perpendicular in all dimensions)the closer the two vectors are in angle

- In NLP, at least for *tf*-like representation, it is ranged in $[0, 1]$ (frequencies are not negative)

## Zipf's Law

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |
| 2nd | $k/2$ |

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |
| 2nd | $k/2$ |
| 3rd | $k/3$ |
| ... | ... |

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |
| 2nd | $k/2$ |
| 3rd | $k/3$ |
| ... | ... |

The system behaves **"roughly" exponentially**

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |
| 2nd | $k/2$ |
| 3rd | $k/3$ |
| ... | ... |

The system behaves **"roughly" exponentially**

**Examples** population dynamics, economic output

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |
| 2nd | $k/2$ |
| 3rd | $k/3$ |
| ... | ... |

The system behaves **"roughly" exponentially**

**Examples** population dynamics, economic output and **COVID-19**

[1] George K. Zipf; 1930s

# Zipf's Law

Given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table.[1]

| $pos(w)$ | $freq(w)$ |
|---|---|
| 1st | $k$ |
| 2nd | $k/2$ |
| 3rd | $k/3$ |
| ... | ... |

The system behaves **"roughly" exponentially**

**Examples** population dynamics, economic output and **COVID-19**

Let's see this in words

[1] George K. Zipf; 1930s

# Zipf's Law

Frequencies of the Brown corpus

| $w$ | $f_{exp}(w)$ | $f_{act}(w)$ |
|---|---|---|
| the | – | 69,971 |
| of | 34,985 | 36,412 |
| and | 23,323 | 28,853 |
| to | 17,492 | 26,158 |
| a | 13,994 | 23,195 |
| in | 11,661 | 21,337 |
| that | 9,995 | 10,594 |
| is | 8,746 | 10,109 |
| was | 7,774 | 9,815 |
| he | 6,997 | 9,548 |
| for | 6,361 | 9,489 |
| it | 5,830 | 8,760 |
| with | 5,382 | 7,289 |
| as | 4,997 | 7,253 |
| his | 4,664 | 6,996 |

- This distribution only holds with large volumes of data (not in a sentence, not in a couple of texts)

# Zipf's Law

Stats

- This distribution only holds with large volumes of data (not in a sentence, not in a couple of texts)
- By computing this distribution, we can obtain an a priori likelihood that a word *w* will appear in a document of the corpus

## Inverse Document Frequency

# *idf*–Inverse Document Frequency

There are two ways to count tokens

- Per document (*tf*)

# *idf*–Inverse Document Frequency

There are two ways to count tokens

- Per document (*tf*)
- Across the entire corpus: *idf*

# *idf*–Inverse Document Frequency

There are two ways to count tokens

- Per document (*tf*)
- Across the entire corpus: *idf*

Let's see...

# *idf* –Inverse Document Frequency

There are two ways to count tokens

- Per document (*tf*)
- Across the entire corpus: *idf*

Let's see. . .

**IDF** How strange is it that this token is in this document?

# *idf*–Inverse Document Frequency

There are two ways to count tokens

- Per document (*tf*)
- Across the entire corpus: *idf*

Let's see. . .

**IDF** How strange is it that this token is in this document?

If $w$ appears in $d$ a lot of, but rarely un any $d' \in D \mid d' \neq d$
$w$ is quite important for $d$!

# *idf*–Inverse Document Frequency

There are two ways to count tokens

- Per document (*tf*)
- Across the entire corpus: *idf*

Let's see...

**IDF** How strange is it that this token is in this document?

If $w$ appears in $d$ a lot of, but rarely un any $d' \in D \mid d' \neq d$
$w$ is quite important for $d$!

Let's see

# IDF and Zipf

Let us assume a corpus $D$, such that $|D| = 1M$

- 1 document $d \in D$ contains "cat"
  idf(cat) = 1,000,000 /1 = 1,000,000

# IDF and Zipf

Let us assume a corpus $D$, such that $|D| = 1M$

- 1 document $d \in D$ contains "cat"
  idf(cat) = 1,000,000 /1 = 1,000,000

- 10 document $\{d_1, d_2, \ldots, d_{10}\} \in D$ contain "dog"
  idf(dog) = 1,000,000 /10 = 100,000

# IDF and Zipf

Let us assume a corpus $D$, such that $|D| = 1M$

- 1 document $d \in D$ contains "cat"
  idf(cat) = 1,000,000 /1 = 1,000,000

- 10 document $\{d_1, d_2, \ldots, d_{10}\} \in D$ contain "dog"
  idf(dog) = 1,000,000 /10 = 100,000

According to Zipf's Law, when comparing $w_1$ and $w_2$, even if $f(w_1) \sim f(w_2)$, one will be **exponentially higher** than the other one!

# IDF and Zipf

Let us assume a corpus $D$, such that $|D| = 1M$

- 1 document $d \in D$ contains "cat"
  idf(cat) = 1,000,000 /1 = 1,000,000

- 10 document $\{d_1, d_2, \ldots, d_{10}\} \in D$ contain "dog"
  idf(dog) = 1,000,000 /10 = 100,000

According to Zipf's Law, when comparing $w_1$ and $w_2$, even if $f(w_1) \sim f(w_2)$, one will be **exponentially higher** than the other one!

We need the inverse of $exp()$ to mild the effect: $log()$

# IDF and Zipf

Let us assume a corpus $D$, such that $|D| = 1M$

- 1 document $d \in D$ contains "cat"
  idf(cat) = 1,000,000 /1 = 1,000,000

- 10 document $\{d_1, d_2, \ldots, d_{10}\} \in D$ contain "dog"
  idf(dog) = 1,000,000 /10 = 100,000

According to Zipf's Law, when comparing $w_1$ and $w_2$, even if $f(w_1) \sim f(w_2)$, one will be **exponentially higher** than the other one!

We need the inverse of $exp()$ to mild the effect: $log()$

$$\begin{aligned} idf &= log(1,000,000/1) = 6 \\ idf &= log(1,000,000/10) = 5 \end{aligned}$$

# *tf-idf*

$$tf(t, d) = \frac{count(t)}{count(d)} \tag{2}$$

# *tf-idf*

$$tf(t,d) = \frac{count(t)}{count(d)} \tag{2}$$

$$idf(t,D) = log \frac{\text{number of documents}}{\text{number of documents containing } t} \tag{3}$$

# *tf-idf*

$$tf(t,d) = \frac{count(t)}{count(d)} \tag{2}$$

$$idf(t,D) = log\,\frac{\text{number of documents}}{\text{number of documents containing } t} \tag{3}$$

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{4}$$

# *tf-idf*

$$tf(t,d) = \frac{count(t)}{count(d)} \tag{2}$$

$$idf(t,D) = log\,\frac{\text{number of documents}}{\text{number of documents containing } t} \tag{3}$$

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{4}$$

- The more often $t$ appears in $d$, the higher the TF (and hence the TF-IDF)

# *tf-idf*

$$tf(t,d) = \frac{count(t)}{count(d)} \tag{2}$$

$$idf(t,D) = log\,\frac{\text{number of documents}}{\text{number of documents containing } t} \tag{3}$$

$$tfidf(t,d,D) = tf(t,d) * idf(t,D) \tag{4}$$

- The more often $t$ appears in $d$, the higher the TF (and hence the TF-IDF)
- The higher the number of documents containing $t$, the lower the IDF (and hence the TF-IDF)

# *tf-idf*

**Outcome** The importance of a token in a specific document given its usage across the entire corpus.

# *tf-idf*

**Outcome** The importance of a token in a specific document given its usage across the entire corpus.

"TF-IDF, is the humble foundation of a simple search engine" (Lane et al., 2019, p. 90)

Let's see

# Coming Next

- Towards "semantics"

# References

Lane, H., C. Howard, and H. Hapkem
2019. *Natural Language Processing in Action*. Shelter Island, NY: Manning Publication Co.