



ALMA MATER STUDIORUM  
UNIVERSITÀ DI BOLOGNA  
CAMPUS DI FORLÌ

# 91258 / B0385

## Natural Language Processing

### Lesson 7. From Word Counts to Meaning

Alberto Barrón-Cedeño  
a.barron@unibo.it

17/10/2024

# Previously

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

# Previously

- Pre-processing
- BoW representation
- One rule-based sentiment model
- One statistical model (Naïve Bayes)
- *tf-idf* (+ Zipf's law)

# Table of Contents

## 1. Topic Vectors

## 2. Latent Semantic Analysis

Jumping from Chapter 3 to Chapter 4 of Lane et al. (2019)

# Topic Vectors

# Topic Vectors

What for?

“[...] using the correlation of normalized frequencies with each other to group words together in topics to define the dimensions of new topic vectors.” (Lane et al., 2019, p. 98)

# Topic Vectors

What for?

“[...] using the correlation of normalized frequencies with each other to group words together in topics to define the dimensions of new topic vectors.” (Lane et al., 2019, p. 98)

What can we achieve with this?

- Compare texts on the basis of *meaning* (not keywords)
- Search based on *meaning*

# Topic Vectors

What for?

“[...] using the correlation of normalized frequencies with each other to group words together in topics to define the dimensions of new topic vectors.” (Lane et al., 2019, p. 98)

What can we achieve with this?

- Compare texts on the basis of *meaning* (not keywords)
- Search based on *meaning*
- Represent the subject of a statement/document or corpus
- Extract keywords



# Topic Vectors

## Limitation of word vectors

$d_1$  Una macchina rossa

$d_2$  Le macchine blu

# Topic Vectors

## Limitation of word vectors

$d_1$  Una macchina rossa

$d_2$  Le macchine blu

↓

stopwording + stemming

↓

$d'_1$  macchin ross

$d'_2$  macchin blu

# Topic Vectors

## Limitation of word vectors

$d_1$  Una macchina rossa

$d_2$  Le macchine blu

↓

stopwording + stemming

↓

$d'_1$  macchin ross

$d'_2$  macchin blu

↓

vectorisation

↓

$\vec{d}_1$  [1, 1, 0]

$\vec{d}_2$  [1, 0, 1]

# Topic Vectors

## Limitation of word vectors

$d_1$  Una macchina rossa

$d_2$  Le macchine blu

↓

stopwording + stemming

↓

$d'_1$  macchin ross

$d'_2$  macchin blu

↓

vectorisation

↓

$\vec{d}_1$  [1, 1, 0]

$\vec{d}_2$  [1, 0, 1]

$$\cos(\vec{d}_1, \vec{d}_2) > 0$$

# Topic Vectors

## Limitation of word vectors

$d_1$    Un'automobile rosso  
 $d_2$    Le macchine blu

# Topic Vectors

## Limitation of word vectors

$d_1$	Un'automobile rosso
$d_2$	Le macchine blu
	↓
$d'_1$	automob ross
$d'_2$	macchinn blu

# Topic Vectors

## Limitation of word vectors

$d_1$  Un'automobile rosso

$d_2$  Le macchine blu



$d'_1$  automob ross

$d'_2$  macchinn blu



$\vec{d}_1$  [1, 1, 0, 0]

$\vec{d}_2$  [0, 0, 1, 1]

# Topic Vectors

## Limitation of word vectors

$d_1$  Un'automobile rosso

$d_2$  Le macchine blu



$d'_1$  automob ross

$d'_2$  macchinn blu



$\vec{d}_1$  [1, 1, 0, 0]

$\vec{d}_2$  [0, 0, 1, 1]

$$\cos(\vec{d}_1, \vec{d}_2) = 0$$



# Topic Vectors

- We need to infer what  $w \in d$  “means”
- Indeed, we need to infer what  $\{w_k, w_{k+1}, \dots\} \in d$  “mean”

# Topic Vectors

- We need to infer what  $w \in d$  “means”
- Indeed, we need to infer what  $\{w_k, w_{k+1}, \dots\} \in d$  “mean”
- We need a *different* kind of vector

# Topic Vectors

- We need to infer what  $w \in d$  “means”
- Indeed, we need to infer what  $\{w_k, w_{k+1}, \dots\} \in d$  “mean”
- We need a *different* kind of vector

**Word-topic vector** One vector represents one word

**Document-topic vector** One vector represents one document (by combining its word-topic vectors)

# Topic Vectors

- We need to infer what  $w \in d$  “means”
- Indeed, we need to infer what  $\{w_k, w_{k+1}, \dots\} \in d$  “mean”
- We need a *different* kind of vector

**Word-topic vector** One vector represents one word

**Document-topic vector** One vector represents one document (by combining its word-topic vectors)

These models can deal with polysemy (e.g., homonyms) at some extent

# Common-Sense Topic Modeling

## Scenario

- We are processing sentences about pets, Central Park, and New York
- Three topics: petness, animalness, cityness

Example from (Lane et al., 2019, p. 101–102)

# Common-Sense Topic Modeling

## Scenario

- We are processing sentences about pets, Central Park, and New York
- Three topics: petness, animalness, cityness
- cat and dog should contribute similarly to petness
- NYC should contribute negatively to animalness
- apple should contribute mildly to cityness

Example from (Lane et al., 2019, p. 101–102)

# Common-Sense Topic Modeling

## Scenario

- We are processing sentences about pets, Central Park, and New York
- Three topics: petness, animalness, cityness
- cat and dog should contribute similarly to petness
- NYC should contribute negatively to animalness
- apple should contribute mildly to cityness

topic	score		
	high	medium	low
Petness	cat, dog		NYC, apple
Cityness	NYC	apple	cat, dog

Example from (Lane et al., 2019, p. 101–102)

# Common-Sense Topic Modeling

## Scenario

- We are processing sentences about pets, Central Park, and New York
- Three topics: petness, animalness, cityness
- cat and dog should contribute similarly to petness
- NYC should contribute negatively to animalness
- apple should contribute mildly to cityness

topic	score		
	high	medium	low
Petness	cat, dog		NYC, apple
Cityness	NYC	apple	cat, dog

 Let us see

Example from (Lane et al., 2019, p. 101–102)



# Common-Sense Topic Modeling

We have a  $3 \times 6$  matrix: 3 topic vectors

	cat	dog	apple	lion	NYC	love
petness	[ .3	.3	0	0	-.2	.2 ]
animalness	[ .1	.1	-.1	.5	.1	-.1 ]
cityness	[ 0	-.1	.2	-.1	.5	.1 ]

The relationships between words and topics can be “flipped”: **transposing** the  $3 \times 6$  matrix to produce topic weights for each word

# Common-Sense Topic Modeling

We have a  $3 \times 6$  matrix: 3 topic vectors

	cat	dog	apple	lion	NYC	love
petness	[ .3	.3	0	0	-.2	.2 ]
animalness	[ .1	.1	-.1	.5	.1	-.1 ]
cityness	[ 0	-.1	.2	-.1	.5	.1 ]

The relationships between words and topics can be “flipped”: **transposing** the  $3 \times 6$  matrix to produce topic weights for each word

	petness	animalness	cityness
cat	[ .3	.1	0 ]
dog	[ .3	.1	-.1 ]
apple	[ 0	-.1	.2 ]
lion	[ 0	.5	-.1 ]
NYC	[ -.2	.1	.5 ]
love	[ .2	-.1	.1 ]

# Common-Sense Topic Modeling

Given:

- A new 6D *tf-idf* vector
- Our  $3 \times 6$ D matrix

# Common-Sense Topic Modeling

Given:

- A new 6D *tf-idf* vector
- Our  $3 \times 6$ D matrix

Multiply: 6D vector  $\times$   $[3 \times 6]$ D matrix

# Common-Sense Topic Modeling

Given:

- A new 6D *tf-idf* vector
- Our  $3 \times 6$ D matrix

Multiply: 6D vector  $\times$   $[3 \times 6]$ D matrix

→ 3D document vector

 Let us see

# Common-Sense Topic Modeling

Given:

- A new 6D *tf-idf* vector
- Our  $3 \times 6$ D matrix

Multiply: 6D vector  $\times$   $[3 \times 6]$ D matrix

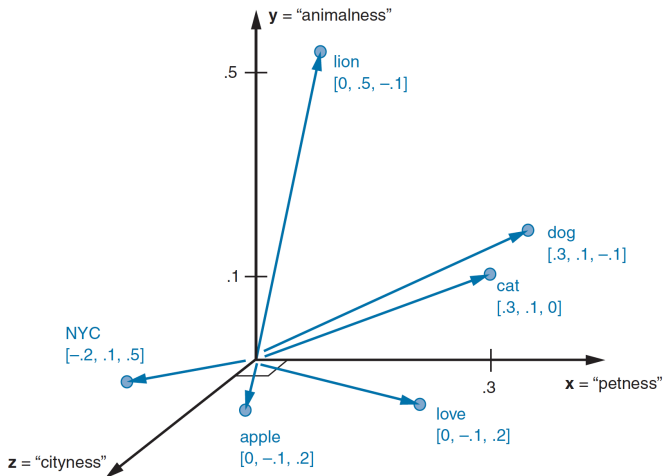
→ 3D document vector

 Let us see

## Advantages

- We can visualise 3D vectors
- A 3D vector space is convenient for classification: it can be sliced with a hyperplane to divide it into classes

# Common-Sense Topic Modeling



Borrowed from (Lane et al., 2019, p. 104)

# Common-Sense Topic Modeling

In summary...

$\vec{d}$  is a *tf-idf* vector of size  $|V|$



# Common-Sense Topic Modeling

In summary...

$\vec{d}$  is a *tf-idf* vector of size  $|V|$

$M$  is a  $3 \times V$  weight matrix

# Common-Sense Topic Modeling

In summary...

$\vec{d}$  is a *tf-idf* vector of size  $|V|$

$M$  is a  $3 \times V$  weight matrix



$\vec{d}_t$  becomes a topic vector of size 3

From one vector space to another

high-dimensional *tf-idf* space  $\rightarrow$  low-dimensional **topic** vector space

# Common-Sense Topic Modeling

In summary...

$\vec{d}$  is a *tf-idf* vector of size  $|V|$

$M$  is a  $3 \times V$  weight matrix



$\vec{d}_t$  becomes a topic vector of size 3

From one vector space to another

high-dimensional *tf-idf* space  $\rightarrow$  low-dimensional **topic** vector space

How can we **learn** the “transformation” matrix?

# Towards a Topic Space

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

# Towards a Topic Space

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

- We have corpora
- We have pre-processors
- We can produce *tf-idf* matrices

# Towards a Topic Space

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

- We have corpora
- We have pre-processors
- We can produce *tf-idf* matrices

We can count co-occurrences → the company of a word

# Latent Semantic Analysis

# Latent Semantic Analysis

- An algorithm to gather words (*tf-idf* matrix) into topics
- It (somehow) captures the meaning of words
- It is a **dimension reduction** technique (sparse  $\rightarrow$  dense vectors)



# Latent Semantic Analysis

- An algorithm to gather words (*tf-idf* matrix) into topics
- It (somehow) captures the meaning of words
- It is a **dimension reduction** technique (sparse  $\rightarrow$  dense vectors)

## AKA

- Principal Component Analysis (PCA)
- Latent Semantic Indexing (LSI, in IR)

# Latent Semantic Analysis

## Linear discriminant analysis (LDA)

A supervised algorithm (it requires labeled data)

# Latent Semantic Analysis

## Linear discriminant analysis (LDA)

A supervised algorithm (it requires labeled data)

### Algorithm

1. Compute the centroid of the vectors in the class
2. Compute the centroid of the vectors not in the class
3. Compute the vector difference between the centroids

# Latent Semantic Analysis

## Linear discriminant analysis (LDA)

A supervised algorithm (it requires labeled data)

### Algorithm

1. Compute the centroid of the vectors in the class
2. Compute the centroid of the vectors not in the class
3. Compute the vector difference between the centroids

**Centroid:** average in a vector space

### Basic algebra!

 Let us see

# Latent Semantic Analysis

## Linear discriminant analysis (LDA)

- We are not relying on individual words
- We are gathering up words with similar “semantics”

# Latent Semantic Analysis

## Linear discriminant analysis (LDA)

- We are not relying on individual words
- We are gathering up words with similar “semantics”

LDA has learned the spaminess of words and documents

# Coming Next

- Training and Evaluation in Machine Learning
- More LSA (from 4.2, p 111)

# References

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