ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
CAMPUS DI FORLÌ

91258 / B0385
Natural Language Processing

Lesson 3. Vector Space Model

Alberto Barrón-Cedeño
a.barron@unibo.it

09/10/2024

Table of Contents

1. Current Status

2. Representations Revisited

3. More Basic Algebra

Current Status

Current Status

You know...

- what is natural language processing

On your own, you have...

On your own, you (could) have...

You can...

Current Status

You know...

- what is natural language processing
- there are two main paradigms: rule-based and statistical

On your own, you have...

On your own, you (could) have...

You can...

Current Status

You know...

- what is natural language processing
- there are two main paradigms: rule-based and statistical

On your own, you have...

- setup a Python development environment
  1. command line
  2. PyCharm or any other option (e.g., Eclipse)
  3. Google's Colab
- played with spacy and nltk

On your own, you (could) have...

You can...

Current Status

You know...

- what is natural language processing
- there are two main paradigms: rule-based and statistical

On your own, you have...

- setup a Python development environment
  1. command line
  2. PyCharm or any other option (e.g., Eclipse)
  3. Google's Colab
- played with spacy and nltk

On your own, you (could) have...

- played with pandas (tutorato)
- found out what is **git** (and perhaps LaTeX as well!)

You can...

Current Status

You know...

- what is natural language processing
- there are two main paradigms: rule-based and statistical

On your own, you have...

- setup a Python development environment
  1. command line
  2. PyCharm or any other option (e.g., Eclipse)
  3. Google's Colab
- played with spacy and nltk

On your own, you (could) have...

- played with pandas (tutorato)
- found out what is **git** (and perhaps LaTeX as well!)

You can...

- open a text file (Python intro)

Current Status

You know...

- what is natural language processing
- there are two main paradigms: rule-based and statistical

On your own, you have...

- setup a Python development environment
  1. command line
  2. PyCharm or any other option (e.g., Eclipse)
  3. Google's Colab
- played with spacy and nltk

On your own, you (could) have...

- played with pandas (tutorato)
- found out what is **git** (and perhaps LaTeX as well!)

You can...

- open a text file (Python intro)
- tokenise and normalise text

Current Status

You know...

- what is natural language processing
- there are two main paradigms: rule-based and statistical

On your own, you have...

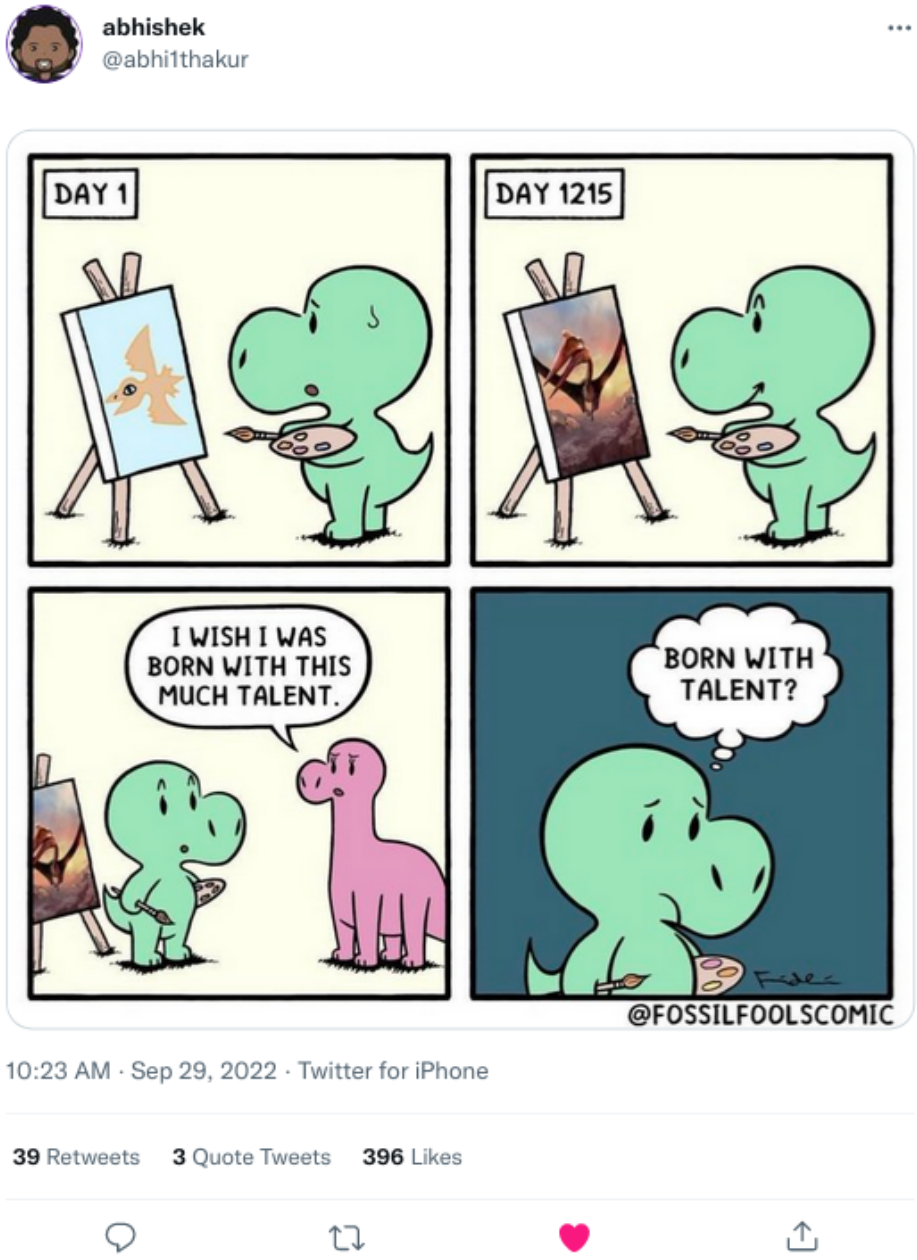
- setup a Python development environment
  1. command line
  2. PyCharm or any other option (e.g., Eclipse)
  3. Google's Colab
- played with spacy and nltk

On your own, you (could) have...

- played with pandas (tutorato)
- found out what is **git** (and perhaps LaTeX as well!)

You can...

- open a text file (Python intro)
- tokenise and normalise text
- build some text representations



https://twitter.com/abhi1thakur/status/1575400771541155842

Representations Revisited

Representations Revisited

[1] https://www.nltk.org/
[2] https://spacy.io

Representations Revisited

1. Use NLTK[1] or Spacy[2] to tokenise

[1] https://www.nltk.org/
[2] https://spacy.io

Representations Revisited

1. Use NLTK[1] or Spacy[2] to tokenise
2. Use `.lower()` to casefold (ignore capitalisation)

[1] `https://www.nltk.org/`
[2] `https://spacy.io`

Representations Revisited

1. Use NLTK[1] or Spacy[2] to tokenise
2. Use `.lower()` to casefold (ignore capitalisation)
3. Use Porter's stemmer to drop suffixes
   or use a lemmatiser to find the *actual* root of words

[1] `https://www.nltk.org/`
[2] `https://spacy.io`

Representations Revisited

1. Use NLTK[1] or Spacy[2] to tokenise
2. Use `.lower()` to casefold (ignore capitalisation)
3. Use Porter's stemmer to drop suffixes
   or use a lemmatiser to find the *actual* root of words
4. Discard stopwords from the text*

[1] `https://www.nltk.org/`
[2] `https://spacy.io`

Representations Revisited

1. Use NLTK[1] or Spacy[2] to tokenise
2. Use `.lower()` to casefold (ignore capitalisation)
3. Use Porter's stemmer to drop suffixes
   or use a lemmatiser to find the *actual* root of words
4. Discard stopwords from the text*
5. Build a vectorial representation*

[1] `https://www.nltk.org/`
[2] `https://spacy.io`

Stopwords

Common words in a language that occur with a high frequency, but carry much less substantive information about the meaning of a phrase (Lane et al., 2019, p. 51–54)

[3] For instance, from NLTK, sklearn, or `https://github.com/stopwords-iso`

Stopwords

Common words in a language that occur with a high frequency, but carry much less substantive information about the meaning of a phrase (Lane et al., 2019, p. 51–54)

Alternative 1 Consider the most frequent tokens in a reference corpus as stopwords (remember Genesis from P4P?)

[3] For instance, from NLTK, sklearn, or `https://github.com/stopwords-iso`

Stopwords

Common words in a language that occur with a high frequency, but carry much less substantive information about the meaning of a phrase (Lane et al., 2019, p. 51–54)

Alternative 1 Consider the most frequent tokens in a reference corpus as stopwords (remember Genesis from P4P?)

Alternative 2 Take an existing list of stopwords[3]

en	es	it
i	a	altri
me	ahora	certa
my	alli	della
it	cerca	nessuna
is	el	prima
do	es	quello
the	unas	solito
will	vez	va
other	yo	via

[3] For instance, from NLTK, sklearn, or `https://github.com/stopwords-iso`

Stopwords

Discarding stopwords

- They are the most frequent tokens in the documents
- Discarding them reduces the computational effort significantly

Stopwords

Discarding stopwords

- They are the most frequent tokens in the documents
- Discarding them reduces the computational effort significantly
- Typical size of a stopwords list: a few hundred words
- For some applications (e.g., topic clustering), they can be safely discarded
- For some others (e.g., dialogue) they cannot

Stopwords

Discarding stopwords

- They are the most frequent tokens in the documents
- Discarding them reduces the computational effort significantly
- Typical size of a stopwords list: a few hundred words
- For some applications (e.g., topic clustering), they can be safely discarded
- For some others (e.g., dialogue) they cannot

Stopwords have to be considered with a grain of salt
(as everything in NLP)

Vector representation

BoW

- A text is represented as the bag (set) of its words
- It disregards grammar
- It disregards word order
- It (can) consider frequency

From (Lane et al., 2019, p. 41)

More Basic Algebra

x and y

https://twitter.com/miniapeur/status/1710074831079690394

Dot product

Algebraically, it is the sum of the products of the corresponding entries of the two sequences of numbers $a \cdot b$

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

Dot product

Algebraically, it is the sum of the products of the corresponding entries of the two sequences of numbers $a \cdot b$

$$\begin{aligned} a \cdot b &= \sum_{i=1}^{n} a_i b_i \\ &= a_1 b_1 + a_2 b_2 + a_3 b_3 + \cdots + a_n b_n \end{aligned}$$

Dot product

Algebraically, it is the sum of the products of the corresponding entries of the two sequences of numbers $a \cdot b$

$$\begin{aligned} a \cdot b &= \sum_{i=1}^{n} a_i b_i \\ &= a_1 b_1 + a_2 b_2 + a_3 b_3 + \cdots + a_n b_n \end{aligned}$$

```
a = [1,2,3]
b = [3,4,6]
my_sum = 0
for i in range(len(a)):
    my_sum += a[i] * b[i]
```

There are better —more efficient— ways to compute the dot product!

Dot product

Algebraically, it is the sum of the products of the corresponding entries of the two sequences of numbers $a \cdot b$

$$\begin{aligned} a \cdot b &= \sum_{i=1}^{n} a_i b_i \\ &= a_1 b_1 + a_2 b_2 + a_3 b_3 + \cdots + a_n b_n \end{aligned}$$

```
a = [1,2,3]
b = [3,4,6]
my_sum = 0
for i in range(len(a)):
    my_sum += a[i] * b[i]
```

There are better —more efficient— ways to compute the dot product!

Now, we can use the dot product to compare two documents ($\sim$ similarity)

Vector space model

"[. . .] an algebraic model for representing text documents (or more generally, items) as vectors [. . .]" [4]

[4] https://en.wikipedia.org/wiki/Vector_space_model

Vector space model

"[. . .] an algebraic model for representing text documents (or more generally, items) as vectors [. . .]"[4]

Some applications

- Relevance rankings in keyword-based search
- Document clustering to "discover" structure and relations in a text collection

(not the SOTA for most tasks, but it's a *minimum viable product*)

[4] https://en.wikipedia.org/wiki/Vector_space_model

Vector space model

"[. . .] an algebraic model for representing text documents (or more generally, items) as vectors [. . .]"[4]

Some applications

- Relevance rankings in keyword-based search
- Document clustering to "discover" structure and relations in a text collection

(not the SOTA for most tasks, but it's a *minimum viable product*)

</> Let us see it working

[4] `https://en.wikipedia.org/wiki/Vector_space_model`

Tomorrow. . .

VADER

References

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