LING530F: Deep Learning for Natural Language Processing (DL-NLP)

Muhammad Abdul-Mageed

muhammad.mageed@ubc.ca

Natural Language Processing Lab

The University of British Columbia

Table of Contents

Word Meaning: Zellig Harris (1954)

> “*words that are used and occur in the same contexts tend to purport similar meanings.*”

> “*oculist and eye-doctor . . . occur in almost the same environments.*”

> “*If A and B have almost identical environments. . . we say that they are synonyms.*”

Word Meaning: J. R. Firth (1957)

> “ *You shall know a word by the company it keeps.* ”

Apalachicola. . .

- Imagine you don't now the word *"apalachicola"*, and I gave you the following 4 sentences:
  1. *Apalachicola* offers terrific seafood.
  2. He was looking for things to do in *Apalachicola*.
  3. Downtown *Apalachicola* isn't busy.
  4. Many people like to visit *Apalachicola*.

Apalachicola. . .

- Imagine you don't now the word *"apalachicola"*, and I gave you the following 4 sentences:
  1. *Apalachicola* offers terrific seafood.
  2. He was looking for things to do in *Apalachicola*.
  3. Downtown *Apalachicola* isn't busy.
  4. Many people like to visit *Apalachicola*.

(Inspired by an example quoted in Jurafsky and Martin [2017] from [Nida, 1975, page 167])

- What does this tell us about the word?

What's in a Word?

- a city
- possibly with a beach
- either in or close to Florida
- attractive to tourists
- ...

What's in a Word?

1. *Apalachicola* offers terrific seafood.
2. He was looking for things to do in *Apalachicola*.
3. Downtown *Apalachicola* isn't busy.
4. Many people like to visit *Apalachicola*.

- Co-occuring words include:
  - seafood
  - offers
  - downtown
  - visit
  - Florida
  - ...
- Syntactically:
  - Can precede a verb ("offers", "is")
  - Occurs after a preposition ("in")
  - Occurs after a verb ("visit")
  - ...
- Similar words would include:
  - ...
  - ...
  - ...

Vector Space

- A vector space is:[1]
  - "a collection of objects called vectors, which may be added together and multiplied ("scaled") by numbers...
  - ...

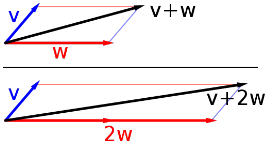

Figure: Vector addition and scalar multiplication.[Wikipedia]

[1]From Wikipedia.

Vectors and Words

- Vector space model of IR (Salton, 1971)
- Can we **identify which words are closer in meaning?**

Example sentences

1. Alex bought a bike
2. Susan rides a bike to school
3. Alex rides a car to work
4. Susan goes to work by car
5. Alex goes to meetings in a suite
6. Susan goes to parties in a suite

Words in Vector Space

Example sentences

1. Alex bought a bike
2. Susan rides a bike to school
3. Alex rides a car to work
4. Susan goes to work by car
5. Alex goes to meetings in a suite
6. Susan goes to parties in a suite

Vocabulary (a set)

V=*{'a', 'school', 'alex', 'in', 'susan', 'car', 'meetings', 'work', 'to', 'bike', 'goes', 'parties', 'suite', 'rides', 'by', 'bought'}*.

[car,bike,suite]

- Suppose we have the following contexts for the words [car,bike,suite]:

Context words

- bike= *{bought,ride,school}*
- car= *{goes,rides,work}*
- suite= *{goes,meetings,parties}*

[car,bike,suite]

- Suppose we have the following contexts for the words [car,bike,suite]:

Context words

- bike= *{bought,ride,school}*
- car= *{goes,rides,work}*
- suite= *{goes,meetings,parties}*

- Suppose we represent each with a vector:

Word vectors

- bike=[1,0,0,0,1,1,1,0]
- car=[0,1,0,0,1,0,0,1]
- suite=[0,1,1,1,0,0,0,0]

[car,bike,suite]

- Suppose we have the following contexts for the words [car,bike,suite]:

Context words

- bike= *{bought,ride,school}*
- car= *{goes,rides,work}*
- suite= *{goes,meetings,parties}*

- Suppose we represent each with a vector:

Word vectors

- bike=[1,0,0,0,1,1,1,0]
- car=[0,1,0,0,1,0,0,1]
- suite=[0,1,1,1,0,0,0,0]

- We can then measure similarity in vector space.

Word Similarity Example

```
from scipy.spatial.distance import cosine
# spatial.distance.cosine computes the distance,
# and not the similarity. So we subtract the
# value from 1 to get the similarity.
#vocab=[bought,goes,meetings,parties,rides,suite,school,work]
bike=[1,0,0,0,1,1,1,0]
car= [0,1,0,0,1,0,0,1]
suite=[0,1,1,1,0,0,0,0]
#-----------------------------------
bike_car = 1 - cosine(bike, car)
bike_suite = 1 - cosine(bike, suite)
print("bike_car",round(bike_car, 2))
print("bike_suit",bike_suite)
```

```
('bike_car', 0.29)
('bike_suit', 0.0)
```

$$cosine(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \times \|\mathbf{v}\|}$$

Word Co-Occurrence Matrix

- Suppose we want to identify word co-occurence.
- We need a corpus to form a matrix
- We can then use, e.g., pointwise mutual information (see next slide)

sugar, a sliced lemon, a tablespoonful of	**apricot**	preserve or jam, a pinch each of,
their enjoyment. Cautiously she sampled her first	**pineapple**	and another fruit whose taste she likened
well suited to programming on the digital	**computer**.	In finding the optimal R-stage policy from
for the purpose of gathering data and	**information**	necessary for the study authorized in the

	aardvark	...	**computer**	**data**	**pinch**	**result**	**sugar**	...
apricot	0	...	0	0	1	0	1	
pineapple	0	...	0	0	1	0	1	
digital	0	...	2	1	0	1	0	
information	0	...	1	6	0	4	0	

Figure 15.4 Co-occurrence vectors for four words, computed from the Brown corpus, showing only six of the dimensions (hand-picked for pedagogical purposes). The vector for the word *digital* is outlined in red. Note that a real vector would have vastly more dimensions and thus be much sparser.

Figure: [From Jurafsky and Martin, 2017]

(Pointwise) Mutual Information

- Mutual information between two random variables X and Y:

1: Mutual Information

$$I(X, Y) = \sum_x \sum_y P(\frac{X}{Y}) \log_2 \frac{x, y}{P(x)P(y)}$$

- Pointwise Mutual information between two random variables X and Y:

2: Pointwise Mutual Information

$$PMI(x, y) = \log_2 \frac{x, y}{P(x)P(y)}$$

PMI Between Words I

- Pointwise Mutual information between target word w and context c tells us how much more w and c co-occur than we expect by chance:

3: Pointwise Mutual Information

$$PMI(w, c) = \log_2 \frac{w, c}{P(w)P(c)}$$

Pointwise Mutual Information (PMI) Between Two Words

4: Pointwise Mutual Information

$$PMI(w,c) = \log_2 \frac{w,c}{P(w)P(c)}$$

Note:

- PMI values range from **negative to positive infinity**.
- **Negative values** (which imply co-occurrence is less often than we would expect by chance) are **unreliable except** with an enormous-sized corpus.
- **Solution:** Use Positive PMI (PPMI), replacing negative values with zero.
- For other smoothing methods, see Jurafsky & Martin, 2017).

So What?

- PMI is **still a score**. . .
- The word co-occurrence matrix is **only based on counts**.
- We want to **learn, not count**. . .
- What do we still have in our pocket??

- PMI is **still a score**...
- The word co-occurrence matrix is **only based on counts**.
- We want to **learn, not count**...
- What do we still have in our pocket??
- # Language Models!