# CPSC 532P / LING 530A: 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...
  - ...

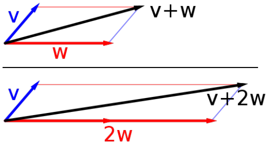


Figure: Vector addition and scalar multiplication.[Wikipedia]

[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

```python
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 Context Matrix

- Suppose we want to identify which words are similar
- We need a corpus to form a word-context matrix
- We can then use, e.g., pointwise mutual information to capture similarity (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]

# Mutual Information

- Mutual information between two random variables X and Y measures the mutual dependence between them.
- It quantifies the **amount of information** (e.g., in bits) acquired about one r.v. when we observe the other r.v.
- MI tells us ***how different the joint distribution of the two random variables is to the product of their marginal distributions***.
- MI is the expected value (the long-run avg value of repetitions of the same experiment) of the **pointwise mutual information** (in 2 slides). See Wikipedia [Link].

**1: Mutual Information**

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

# Mutual Information *Contd.*

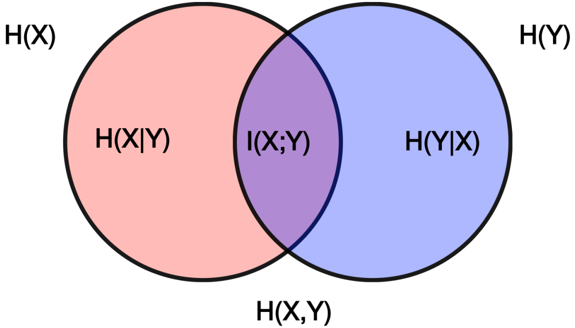


Figure: Venn diagram showing additive and subtractive relationships between various information measures associated with correlated variables X and Y. The area contained by both circles is the joint entropy H(Y,Y). The circle on the left (red and violet) is the individual entropy H(X), with the red being the conditional entropy H(X|Y). The circle on the right (blue and violet) is H(Y), with the blue being H(Y|X). The violet is the mutual information **I(X;Y)**. [From Wikipedia]

# Pointwise Mutual Information

- Pointwise Mutual information (PMI) is a measure of **association**.
- PMI between two random variables X and Y **quantifies the discrepancy between the probability of their coincidence given their joint distribution and their individual distributions, assuming independence**. See Wikipedia [Link].

**2: Pointwise Mutual Information**

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

# PMI Between Words

- 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: PMI Between Word w and Context c**

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

# Pointwise Mutual Information (PMI) Between Two Words

**4: Pointwise Mutual Information**

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

**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 (See Jurafsky and Martin, 2017).
- **Solution:** Use Positive PMI (PPMI), replacing negative values with zero.
- For other **smoothing methods**, see Jurafsky and Martin, 2017.

# PPMI From Term-Context Matrix I (For YourReference)

Matrix F with W rows (words) and C columns (contexts)

$f_{ij}$ is # of times $w_i$ occurs in context $c_j$

| | 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 |

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}} \qquad p_{i*} = \frac{\sum_{j=1}^{C} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}} \qquad p_{*j} = \frac{\sum_{i=1}^{W} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*}p_{*j}} \qquad ppmi_{ij} = \begin{cases} pmi_{ij} & \text{if } pmi_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Figure: [From Dan Jurafsky]

# PPMI From Term-Context Matrix II (For YourReference)

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

**Count(w,context)**

| | computer | data | pinch | result | sugar |
|---|---|---|---|---|---|
| apricot | 0 | 0 | 1 | 0 | 1 |
| pineapple | 0 | 0 | 1 | 0 | 1 |
| digital | 2 | 1 | 0 | 1 | 0 |
| information | 1 | 6 | 0 | 4 | 0 |

p(w=information,c=data) = 6/19 = .32

p(w=information) = 11/19 = .58

p(c=data) = 7/19 = .37

$$p(w_i) = \frac{\sum_{j=1}^{C} f_{ij}}{N} \qquad p(c_j) = \frac{\sum_{i=1}^{W} f_{ij}}{N}$$

**p(w,context)** **p(w)**

| | computer | data | pinch | result | sugar | p(w) |
|---|---|---|---|---|---|---|
| apricot | 0.00 | 0.00 | 0.05 | 0.00 | 0.05 | 0.11 |
| pineapple | 0.00 | 0.00 | 0.05 | 0.00 | 0.05 | 0.11 |
| digital | 0.11 | 0.05 | 0.00 | 0.05 | 0.00 | 0.21 |
| information | 0.05 | 0.32 | 0.00 | 0.21 | 0.00 | 0.58 |
| **p(context)** | 0.16 | 0.37 | 0.11 | 0.26 | 0.11 | |

Figure: [From Dan Jurafsky]

# PPMI From Term-Context Matrix III (For YourReference)

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*}p_{*j}}$$

| | p(w,context) | | | | | p(w) |
|---|---|---|---|---|---|---|
| | computer | data | pinch | result | sugar | |
| apricot | 0.00 | 0.00 | 0.05 | 0.00 | 0.05 | 0.11 |
| pineapple | 0.00 | 0.00 | 0.05 | 0.00 | 0.05 | 0.11 |
| digital | 0.11 | 0.05 | 0.00 | 0.05 | 0.00 | 0.21 |
| information | 0.05 | 0.32 | 0.00 | 0.21 | 0.00 | 0.58 |
| **p(context)** | 0.16 | 0.37 | 0.11 | 0.26 | 0.11 | |

pmi(information,data) = $\log_2$ ( .32 / (.37*.58) ) = .58

*(.57 using full precision)*

**PPMI(w,context)**

| | computer | data | pinch | result | sugar |
|---|---|---|---|---|---|
| apricot | - | - | 2.25 | - | 2.25 |
| pineapple | - | - | 2.25 | - | 2.25 |
| digital | 1.66 | 0.00 | - | 0.00 | - |
| information | 0.00 | 0.57 | - | 0.47 | - |

Figure: [From Dan Jurafsky]

# 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??

# 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??
- Language Models!