COMP90042
Web search and text analysis

Workshop Week 2

Your tutor

- Xudong Han
- Master of Data Science
- xudong.han@unimelb.edu.au

Workshops

- Please post questions to LMS
- Workshop slides: https://github.com/HanXudong/COMP90042_Workshops
- Review and discussion

Python 3

- Familiarise yourself with Python 3
- https://trevorcohn.github.io/comp90042/workshops/week1-python-01.pdf
- Canopy: https://store.enthought.com/downloads/
- NLTK: https://www.nltk.org/install.html
  using Canopy Command Prompt

Web Search & Text Analysis

- Natural Language Processing(NLP)
- Web search
  E.g. Information retrieval(IR)
- Text analysis
  E.g. Structure learning
- Larger tasks:
  E.g. Information extraction, question answering, translation.

```python
from nltk.tokenize import word_tokenize
```

```python
sentence = "Hello Aswathi How are you doing today"
sentence_token = word_tokenize(sentence)
sentence_token
```

```
['Hello', 'Aswathi', 'How', 'are', 'you', 'doing', 'today']
```

Tokenization

http://blog.xnextcon.com/?p=233

Tokenization

- A token is an instance of a sequence of characters in some particular document that are grouped together as a useful semantic unit for processing.
- A type is the class of all tokens containing the same character sequence.
- E.g. You for me and me for you.

```python
from nltk.stem.porter import PorterStemmer
stem = PorterStemmer()
```

```python
word = "mulitplying"
stem.stem(word)
```

```
'mulitpli'
```

```python
from nltk.stem.wordnet import WordNetLemmatizer
lem = WordNetLemmatizer()
```

```python
word = "multiplying"
lem.lemmatize(word,"v")
```

```
'multiply'
```

Stemming and Lemmatisation

http://blog.xnextcon.com/?p=233

Stemming and Lemmatisation

- https://semanticsmorphology.weebly.com/inflectional-and-derivational-morphemes.html
- Inflectional morphology
  - Do not really alter the meaning
- Derivational morphology
  - Alter the meaning
  - One class to another e.g. Verb (teach) -> Noun (teacher)

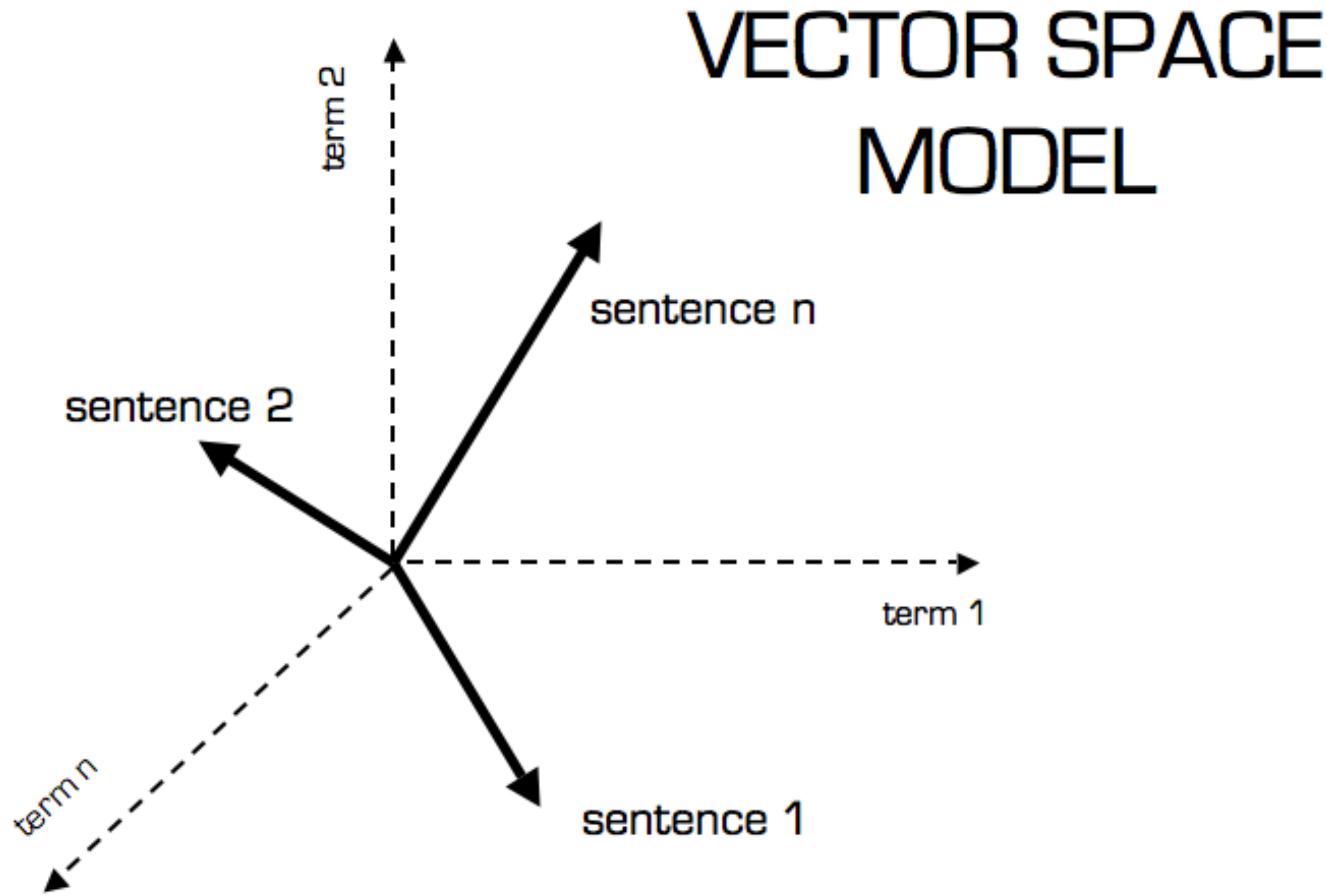
Stemming and Lemmatisation

English Inflectional Morphemes		Added to	Examples
-s	plural	*Nouns*	*She has got two guitar**s**.*
-'s	possessive	*Nouns*	*Zeynep**'s** hair is long.*
-er	comparative	*Adjectives*	*Zeynep has long**er** hair than Derya.*
-est	superlative	*Adjectives*	*Zeynep has the long**est** hair.*
-s	3rd person singular present tense	*Verbs*	*Zeynep play**s** the guitar.*
-ed	past tense	*Verbs*	*She play**ed** the guitar at the party.*
-ing	progressive	*Verbs*	*She is play**ing** the guitar at the party*
-en	past participle	*Verbs*	*She has tak**en** the guitar to the party.*

Stemming and Lemmatisation

Some English derivational affixes

Affix	Change	Examples
Suffixes		
-able	V → A	fix-able, do-able
-(at)ion	V → N	realiz-ation
-ing	V → N	the shoot-ing, the danc-ing
-ing	V → A	the sleep-ing giant
-ive	V → A	assert-ive
-al	V → N	refusal
-ment	V → N	treat-ment
-ful	N → A	hope-ful



Vector Space Model

http://bitsearch.blogspot.com/2011/01/vector-space-model-for-scoring.html

Term-document matrix(TDM)

doc1	Two	for	tea	and	tea	for	two
doc2	Tea	for	me	and	tea	for	you
doc3	You	for	me	and	me	for	you

	two	tea	me	you
doc1	2	2	0	0
doc2	0	2	1	1
doc3	0	0	2	2

Term-document matrix(TDM)

	two	tea	me	you
doc1	2	2	0	0
doc2	0	2	1	1
doc3	0	0	2	2

- Query 1: Tea me
- Query 2: Two

$$cos(a,b) = \frac{a \cdot b}{|a||b|}$$

Term-document matrix(TDM)

	two	tea	me	you
doc1	0.707	0.707	0	0
doc2	0	0.707	0.353	0.353
doc3	0	0	0.707	0.707

Normalisation

All document pre-normalised to unit length

- Query 1: Tea me
- Query 2: Two

$$cos(a,b) = \frac{a \cdot b}{|a||b|} = a \cdot b$$

Inverted index

	two	tea	me	you
doc1	0.707	0.707	0	0
doc2	0	0.707	0.353	0.353
doc3	0	0	0.707	0.707

- Query 1: Tea me
- Query 2: Two

two	1: 0.707;
tea	1:0.707; 2: 0.707
me	2: 0.353; 3:0.707
you	2: 0.353; 3:0.707

TF*IDF similarity score

$tf_{d,t} \times idf_t$

$tf_{d,t}$: term frequency of a document (count of a term t in a document d)

idf_t : inverse document frequency

df_t : document frequency (count of documents that contain the term t)

	two	tea	me	you
doc1	2	2	0	0
doc2	0	2	1	1
doc3	0	0	2	2
idf_t				

$$idf_t = log\frac{N}{df_t}$$

TF*IDF similarity score

	two	tea	me	you
doc1	2	2	0	0
doc2	0	2	1	1
doc3	0	0	2	2
idf_t	1.58	0.58	0.58	0.58

$$idf_t = log\frac{N}{df_t}$$

log_2 **or** log_e**?**

	two	tea	me	you
doc1	3.17	1.16	0	0
doc2	0	1.16	0.58	0.58
doc3	0	0	1.16	1.16

TF*IDF similarity score

Question 4

	apple	ibm	lemo	sun
D1	4	0	1	1
D2	5	0	5	0
D3	2	5	0	0
D4	1	0	1	7
D5	0	1	3	0
idf				

TF*IDF similarity score

	apple	ibm	lemo	sun
D1	4	0	1	1
D2	5	0	5	0
D3	2	5	0	0
D4	1	0	1	7
D5	0	1	3	0
idf				

	apple	ibm	lemon	sun
idf	$\log \frac{5}{4} = 0.22$	$\log \frac{5}{2} = 0.92$	$\log \frac{5}{4} = 0.22$	$\log \frac{5}{2} = 0.92$

TF*IDF similarity score

	apple	ibm	lemon	sun
D1	0.89	0	0.22	0.92
D2	1.12	0	1.12	0
D3	0.45	4.58	0	0
D4	0.22	0	0.22	6.41
D5	0	0.92	0.67	0

Query: apple ibm

skip the document normalisation

$$S_{TF-IDF}(d,Q) = \sum_{t \in Q} tf_{d,t} \times log \frac{N}{df_t}$$

TF*IDF similarity score

	apple	ibm	lemon	sun
D1	0.89	0	0.22	0.92
D2	1.12	0	1.12	0
D3	0.45	4.58	0	0
D4	0.22	0	0.22	6.41
D5	0	0.92	0.67	0

Query: apple ibm

What if do document normalisation?

$$S_{TF-IDF}(d,Q) = \sum_{t \in Q} tf_{d,t} \times log \frac{N}{df_t}$$

BM25

$$w_t = log \frac{N - df_t + 0.5}{df_t + 0.5} \times \frac{(K_1 + 1)tf_{d,t}}{k_1((1 - b) + b\frac{L_d}{L_{avg}}) + tf_{d,t}} \times \frac{(k_3 + 1)tf_{q,t}}{k_3 + tf_{q,t}}$$

where $0 \leq K_1 \leq \infty,\ 0 \leq K_3 \leq \infty,$ **and** $0 \leq b \leq 1$

What happens?

- K_1 = 0; K_1 = inf
- K_3 = 0; K_3 = inf
- b = 0; b = 1

- binary model or document tf?
- binary model or query tf?
- document length matters?

Very Useful Online Resources

- https://web.stanford.edu/~jurafsky/
- 2012 NLP MOOC w/Chris Manning(YouTube)