COMP90042 Web Search & Text Analysis

Workshop Week 2

Zenan Zhai

March 12, 2019

University of Melbourne

Timetable

Mon	13:15	207-221-Bouverie-St-B113	Andrei
Mon	17:15	207-221-Bouverie-St-B116	Andrei
Thu	14:15	Alice-Hoy-211	Ekaterina
Thu	15:15	Alice-Hoy-211	Ekaterina
Mon	18:15	Old-Engineering-EDS4	Navnita
Wed	11:00	Elec.-Engineering-121	Navnita
Fri	10:00	221-Bouverie-St-B117	Nitika
Thu	17:15	221-Bouverie-St-B132	Nitika
Fri	15:15	Alice-Hoy-210	Shivashankar
Tue	18:15	Old-Engineering-EDS4	Shivashankar
Mon	11:00	Elec.-Engineering-121	Winn
Mon	9:00	Doug-McDonell-502	Xudong
Mon	17:15	221-Bouverie-St-B132	Xudong
Tue	16:15	221-Bouverie-St-B113	Zenan
Tue	17:15	221-Bouverie-St-B132	Zenan

Contact

- LMS - Discussion Board

- Subject Coordinator
  - A/Prof. Trevor Cohn
  - t.cohn@unimelb.edu.au
  - https://trevorcohn.github.io/comp90042/

- Me
  - Zenan Zhai
  - zenan.zhai@unimelb.edu.au
  - Workshop slides available at https://zenanz.github.io/comp90042-2019/

Programming

- Python 3
  - Virtualenv
  - Ana/mini conda3
- Canopy EPD
  - Search ”canopy” in the start menu of your lab computer
  - Open ”Editor”
  - Have fun!
  - License available when register with Unimelb Email.
- Packages
  - NLTK, gensim
  - Matplotlib, Numpy, Scipy
  - Scikit-learn

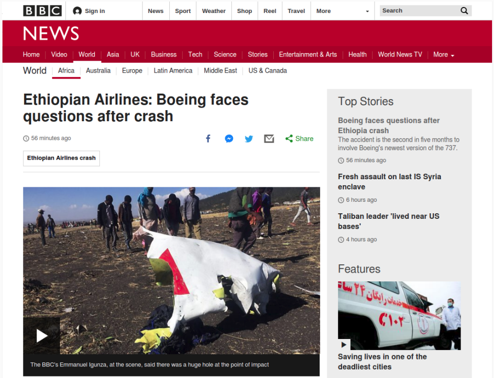
Outline

- Pre-processing
    - Pipeline
    - Lemmatisation/Stemming
- Vector Space Model
    - Document-Term Matrix/Inverted Index
    - TF-IDF/BM25
    - Exercise

Pre-processing Pipeline

- Formatting
- Sentence Segmentation
- Tokenisation
- Normalisation
  - Lemmatisation
  - Stemming
- Remove Stopwords
  - May varies in different toolkit

Formatting




```
         <div id="page" class="configurable story " data-story-id="world-africa-47519467">      <div role="main"> <div
class="container-width-only">       <span class="index-title index-title--redundant " id="comp-index-title" data-index-title-
meta="{&quot;id&quot;:&quot;comp-index-title&quot;,&quot;type&quot;:&quot;index-
title&quot;,&quot;handler&quot;:&quot;indexTitle&quot;,&quot;deviceGroups&quot;:null,&quot;opts&quot;:
{&quot;alwaysVisible&quot;:false,&quot;onFrontPage&quot;:false},&quot;template&quot;:&quot;index-title&quot;}">
        <span class="index-title__container">
            <a href="/news/world/africa">Africa</a>
        </span>
    </span>
    </div>      <div class="container">      <div class="container--primary-and-secondary-columns column-clearfix">
<div class="column--primary">

<div class="story-body">
    <h1 class="story-body__h1">Ethiopian Airlines: Boeing faces questions after crash</h1>

        <div class="with-extracted-share-icons">
        <div class="story-body__mini-info-list-and-share">
            <div class="story-body__mini-info-list-and-share-row">
                <div class="mini-info-list-wrap">
                                            <ul class="mini-info-list">
    <li class="mini-info-list__item"><div class="date date--v2" data-seconds="1552267449" data-datetime="11 March 2019">11 March
2019</div></li>
</ul>
                                    </div>
                                    <div class="share-tools--event-tag">
```

Sentence Segmentation & Tokenisation

'Ethiopian Airlines: Boeing faces questions after crash.'

$\Downarrow$

['Ethiopian', 'Airlines', ':', 'Boeing', 'faces', 'questions', 'after', 'crash', '.']

- Sentence Segmentation / Tokenisation
  - Rule-based / Machine Learning
  - Varies in different languages/domains (e.g. Medicine Chemistry)
- Off-the-shelf implementations
  - NLTK
    `https://www.nltk.org/`
  - OpenNLP
    `https://opennlp.apache.org/`
  - StanfordNLP
    `https://stanfordnlp.github.io/stanfordnlp/`

Morphology

- Inflectional Morphology
  - Grammatical variants
- Derivational morphology
  - Another word with different meaning

Inflectional Morphology

airline $\rightarrow$ airlines

face $\rightarrow$ faces

question $\rightarrow$ questions

Derivational morphology

Ethiopia $\rightarrow$ Ethiopian

Lemmatisation & Stemming

Lemmatisation

Remove all inflections

Matches with lexicons

Product: Lemma

Stemming

Remove all suffixes

No matching required

Product: Stem

```
import nltk
nltk.download('wordnet')

sentence = ['Ethiopian', 'Airlines', ':', 'Boeing', 'faces', 'questions', 'after', 'crash', '.']
lemmatiser = nltk.stem.wordnet.WordNetLemmatizer()
stemmer = nltk.stem.porter.PorterStemmer()

# Code below from ...
def lemmatise(word):
    lemma = lemmatiser.lemmatize(word, 'v')
    if lemma == word:
        lemma = lemmatiser.lemmatize(word, 'n')
    return lemma
# End of copied code

lemmatised_sent = [lemmatise(word) for word in sentence ]
stemmed_sent = [stemmer.stem(word) for word in sentence ]

print('Sentence after lemmatisation: 'lemmatised_sent)
print('Sentence after stemming: ', stemmed_sent)
```

```
['Ethiopian', 'Airlines', ':', 'Boeing', 'face', 'question', 'after', 'crash', '.']
['ethiopian', 'airlin', ':', 'boe', 'face', 'question', 'after', 'crash', '.']
```

Sparsity

More word types $\Rightarrow$ Larger sparsity

- { 'apple': 1, 'apples':1, 'Apple': 1 }
- { 'apple': 3 }

- Stemming creates less sparsity than lemmatisation.
- When do we prefer smaller sparsity?
- Can we increse sparsity?

Removing unwanted tokens

- Stopword
    - Examples (NLTK): me, what, by, with, into, above …
- Punctuation
    - Examples: , . : ! ' " …

Outline

- Pre-processing
  - Pipeline
  - Lemmatisation/Stemming
- Vector Space Model
  - Document-Term Matrix/Inverted Index
  - TF-IDF/BM25
  - Exercise

Document-Term Matrix V.S. Inverted Index

Document-Term Matrix

DocID	apple	pear	banana	peach
doc_1	3	0	4	1
doc_2	0	3	0	0
doc_3	0	0	4	4
doc_4	1	2	0	0

Inverted Index

apple $\rightarrow$ [doc_1:3, doc_4:1]
pear $\rightarrow$ [doc_2:3, doc_4:2]
banana $\rightarrow$ [doc_1:4, doc_3:4]
peach $\rightarrow$ [doc_1:1, doc_3:4]

Consider time complexity when query is 'banana apple'.

TF-IDF

$$W_{d,t} = tf_{d,t} \times \log \frac{N}{df_t}$$

(TF) (IDF)

$$Score_{d,Q} = \frac{1}{\sqrt{|d|}} \times \sum_{q \in Q} tf_{d,q} \times \log \frac{N}{df_q}$$

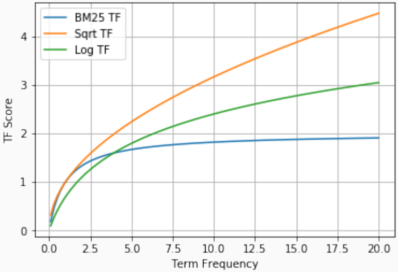
Okapi BM25

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

$$Score_{d,Q} = \sum_{q \in Q} W_{d,q}$$

(Document TF, document length) (IDF) (Query TF)

TF smoothing

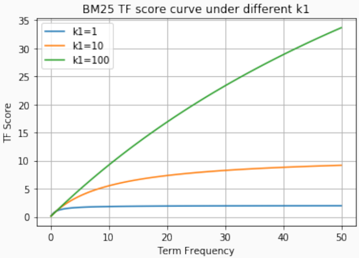


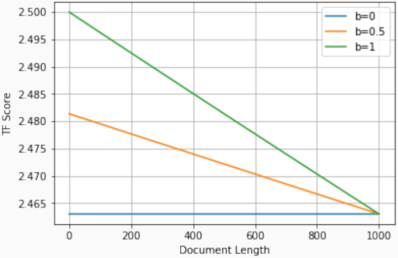
Raw TF	$TF_{score} = tf_{d,t}$
Square Root TF	$TF_{score} = \sqrt{tf_{d,t}}$
Log TF	$TF_{score} = \log(tf_{d,t})$

BM25 $\quad TF_{score} = \dfrac{(k+1)tf_{d,t}}{k + tf_{d,t}}$

$$\lim_{tf_{d,t} \to \infty} \frac{(k+1)tf_{d,t}}{k + tf_{d,t}} = k + 1$$

Document TF and Document length



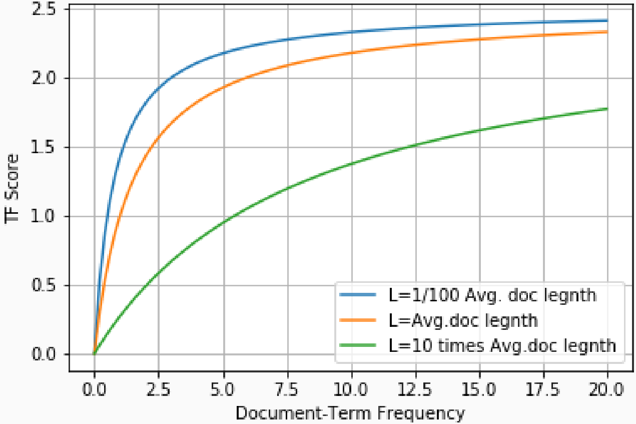


$TF_{score} = f(tf_{d,t})$ $TF_{score} = f(L)$

$$\frac{(k_1+1)tf_{d,t}}{k_1((1-b)+b(\frac{L}{L_{avg}}))+tf_{d,t}}$$

- What does k_1 controls?
- What happens when $b = 0$?

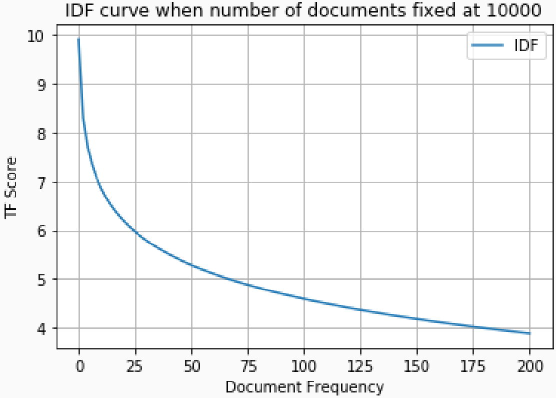
Document length and growth of TF



$$\frac{(k_1+1)tf_{d,t}}{k_1((1-b)+b(\frac{L}{L_{avg}}))+tf_{d,t}}$$

TF score grows faster when document length is short. Why?

Inverted Document Frequency



$$IDF_{score} = \log \frac{N - df_t + 0.5}{df_t + 0.5}$$

What are "stop-words" and why are they often discarded in information retrieval? (Final exam, 2015)

Exercise

- Workshop Sheet - Question 4
- Final exam, 2016

Consider the following "term-document matrix", where each cell shows the frequency of a given term in a document:

DocId	snipe	tax	tony	boats	malcolm	panama
doc_1	2	1	0	0	1	1
doc_2	0	0	3	2	1	0
doc_3	2	0	0	0	1	0
doc_4	0	3	4	0	2	0

a) Calculate the document ranking for the query *tax panama*, using the "TF*IDF" measure of similarity with the standard versions of TF (raw frequency) and IDF (logarithmic). Show your working. You do not need to simplify numerical values, and should use logarithms with base 2. [3 marks]