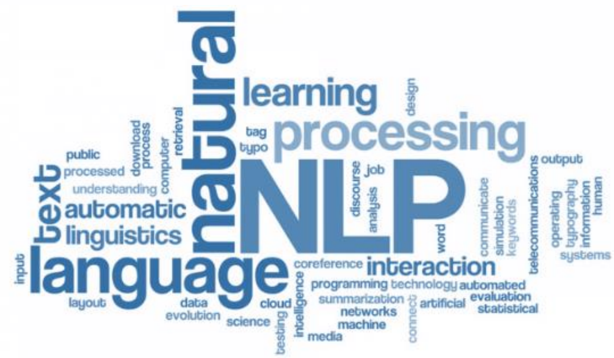
CENG 3526
Natural Language Processing

Lecture 2
Text Preprocessing & Representation

Instructor
Bekir Taner Dinçer

Teaching Assistant
Selahattin Aksoy



MUĞLA SITKI KOÇMAN ÜNİVERSİTESİ
COMPUTER ENGINEERING

1

Recap

MSKU NLP | CENG-3526 Natural Language Processing

2

Text Representation

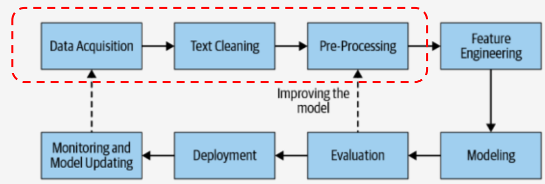
Text Representation in (NLP)

- is the process of **converting textual data into a numerical format** that can be understood and processed by machine learning algorithms.

Goal

- **transform raw text** into **a structured representation** so that the **semantic** and **syntactic information** contained within the text **is captured**.

Text Representation Process

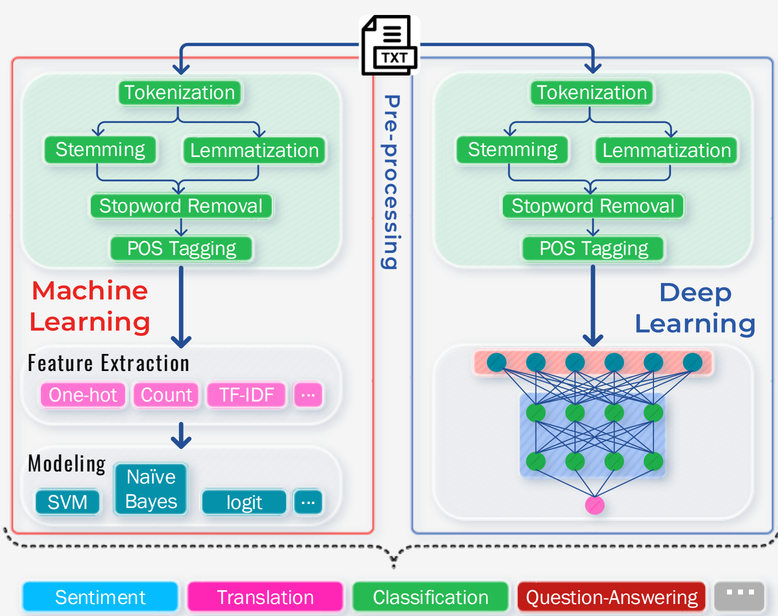


Common text representation techniques include:

- **Bag of Words** (BoW)
- **TF-IDF** (Term Frequency-Inverse Document Frequency):
- **Word Embeddings** (e.g., Word2Vec, GloVe)
- **Character-level Embeddings**
- **n-gram language models**
- **Document Embeddings** (Doc2Vec, or Deep Learning Models)

MSKU NLP | CENG-3526 Natural Language Processing

3



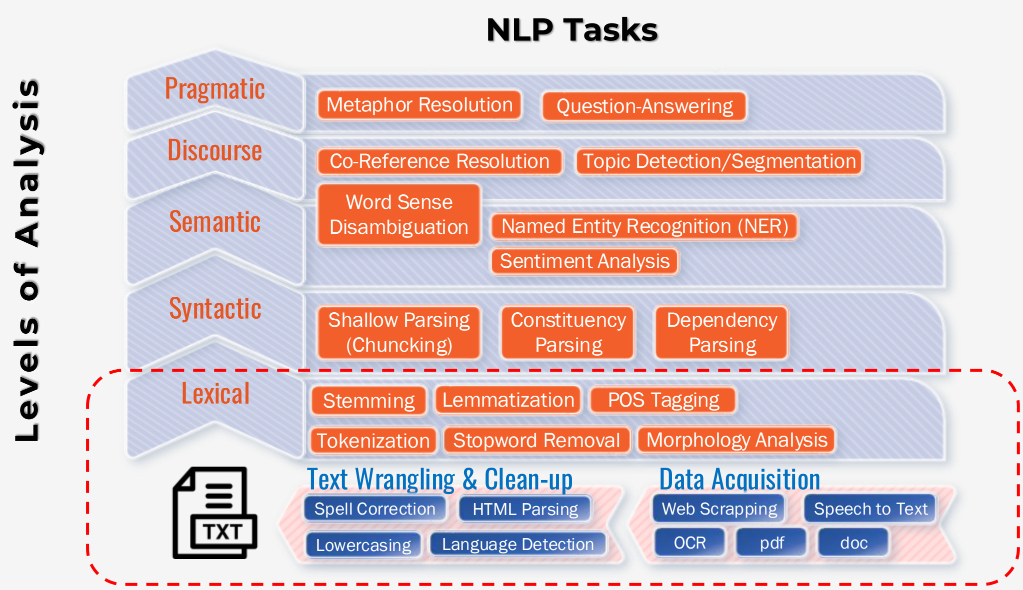
Common NLP Tasks

1. **Tokenization**
2. **Text Normalization**
3. **Text Cleaning**

=> **Text Representation**

MSKU CENG

4



MSKU CENG

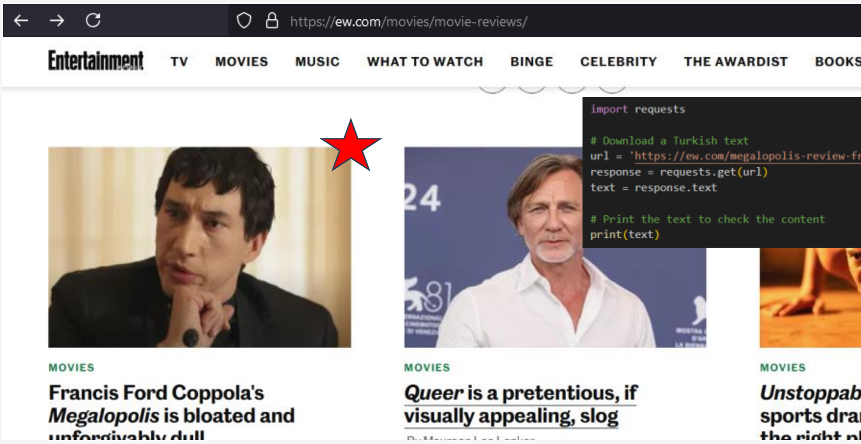
5

Text Representation

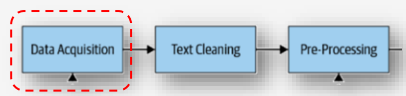
MSKU NLP | CENG-3526 Natural Language Processing

6

Data Acquisition: a Movie Review from the Internet



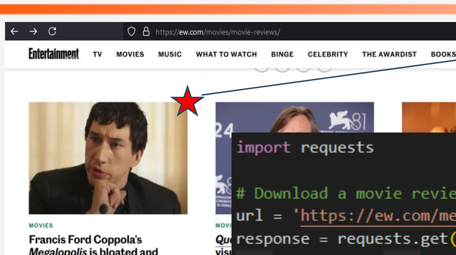

```python
import requests

# Download a Turkish text
url = 'https://ew.com/megalopolis-review-francis-ford-coppola-film-bloated-dull-8710050'
response = requests.get(url)
text = response.text

# Print the text to check the content
print(text)
```



MSKU NLP | CENG-3526 Natural Language Processing

7

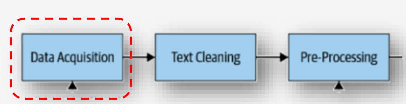
Data Acquisition: a Movie Review from the Internet



Download using *request* library

```python
import requests

# Download a movie review
url = 'https://ew.com/megalopolis-review-francis-ford-coppola-film-bloated-dull-8710050'
response = requests.get(url)
text = response.text

# Print the text to check the content
print(text)
```

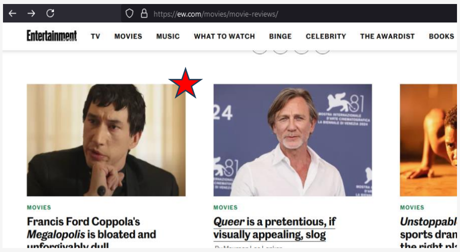
CENG3526 - Week 2 - Text Preprocessing and Representation.ipynb



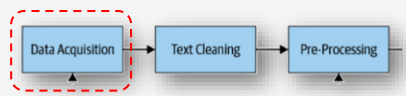
MSKU NLP | CENG-3526 Natural Language Processing

8

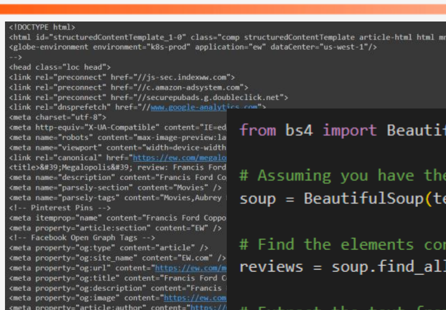
Data Acquisition: a Movie Review from the Internet




```html
<!DOCTYPE html>
<html id="structuredContentTemplate_1-0" class="comp structuredContentTemplate article-html html mn
<globe-environment environment="k8s-prod" application="ew" dataCenter="us-west-1"/>
-->
<head class="loc head">
<link rel="preconnect" href="//js-sec.indexww.com">
<link rel="preconnect" href="//c.amazon-adsystem.com">
<link rel="preconnect" href="//securepubads.g.doubleclick.net">
<link rel="dnsprefetch" href="//www.google-analytics.com">
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="robots" content="max-image-preview:large, NOODP, NOYDIR" />
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="canonical" href="https://ew.com/megalopolis-review-francis-ford-coppola-film-bloated-dul
<title>&#39;Megalopolis&#39; review: Francis Ford Coppola film is bloated and dull</title>
<meta name="description" content="Francis Ford Coppola finally debuts his ‘Megalopolis,’ but it’s n
<meta name="parsely-section" content="Movies" />
<meta name="parsely-tags" content="Movies,Aubrey Plaza,Adam Driver,Lionsgate,TIFF,Francis Ford Copp
<!-- Pinterest Pins -->
<meta itemprop="name" content="Francis Ford Coppola&#39;s Megalopolis is bloated and unforgivably d
<meta property="article:section" content="EW" />
<!-- Facebook Open Graph Tags -->
<meta property="og:type" content="article" />
<meta property="og:site_name" content="EW.com" />
<meta property="og:url" content="https://ew.com/megalopolis-review-francis-ford-coppola-film-bloate
<meta property="og:title" content="Francis Ford Coppola&#39;s &#39;Megalopolis&#39; is bloated and 
<meta property="og:description" content="Francis Ford Coppola finally debuts his ‘Megalopolis,’ but
<meta property="og:image" content="https://ew.com/thmb/Kjg3kM2B673MxPNwk-1SCRzapqU=/1500x0/filters:
<meta property="article:author" content="https://www.facebook.com/entertainmentweekly" />
<!-- Twitter Cards -->
```

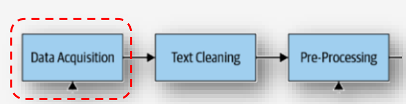


Extract the Text from the HTML

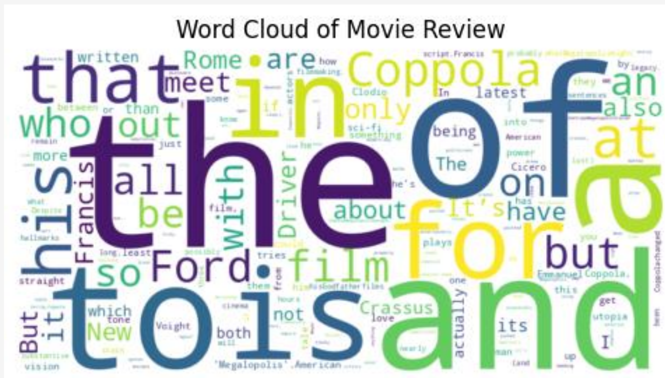


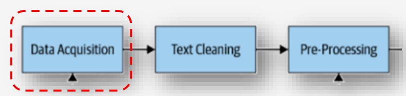
Use browsers' developer tools to identify

```python
from bs4 import BeautifulSoup

# Assuming you have the downloaded HTML content in the 'text' variable
soup = BeautifulSoup(text, 'html.parser')

# Find the elements containing the reviews (you might need to adjust the selector)
reviews = soup.find_all('div', class_='loc article-content')

# Extract the text from firts review element
review_text = [review.get_text(strip=True) for review in reviews][0]

# Now you have a list containing the extracted review texts.
print(review_text)
```



The text so far

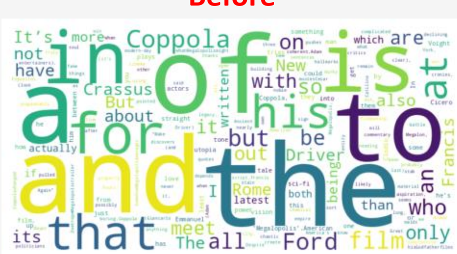




Normalization

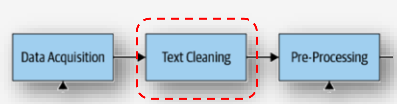
Converting text to a consistent format: lowercase, removing diacritics, handling contractions, typos, and other inconsistencies

Before



After

lowercased & punct removal



Most Frequent Words

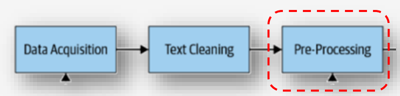
The most frequent terms are mostly the terms that are used due to grammatical necessity

Word	Frequency
the	32
of	28
and	27
a	26
to	23
is	16
in	12
for	12
that	12
his	11
at	8

Word	Frequency
at	8
Coppola	7
Ford	6
film	6
but	6

The key terms
or
content bearing terms
also have high frequencies
but not more than those
non-content bearing terms

MSKU NLP | CENG-3526 Natural Language Processing



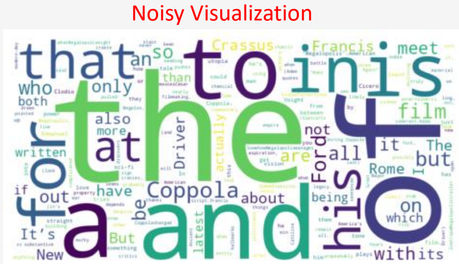
13

Stopwords

Stop words are common words in a language that are often removed from text data before processing.

- These words, such as "*the*“, "*and*“, "*a*“, "*in*" and "*it*" typically do not carry significant semantic meaning and can add noise to text analysis tasks.

Noisy Visualization



Word	Frequency
the	32
of	28
and	27
a	26
to	23
is	16
in	12
for	12
that	12
his	11
at	8

Word	Frequency
at	8
Coppola	7
Ford	6
film	6
but	6

The key terms
or
content bearing terms
also have high frequencies
but not more than those
non-content bearing terms

MSKU NLP | CENG-3526 Natural Language Processing

14

Stopword Removal

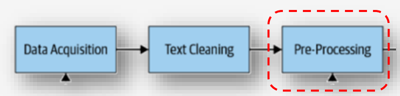
Why remove stop words?

- **Reduce dimensionality**: Removing stop words can significantly reduce the dimensionality of the feature space, making it easier to process and analyze text data.
- **Improve accuracy**: Stop words can often introduce noise into models, leading to reduced accuracy. Removing them can help improve the performance of NLP tasks like text classification, sentiment analysis, and information retrieval.
- **Focus on meaningful words**: By removing stop words, you can focus on the more informative words in the text, which can provide better insights.

Word	Frequency
the	32
of	28
and	27
a	26
to	23
is	16
in	12
for	12
that	12
his	11
at	8

Word	Frequency
at	8
Coppola	7
Ford	6
film	6
but	6

The key terms
or
content bearing terms
also have high frequencies
but not more than those
non-content bearing terms

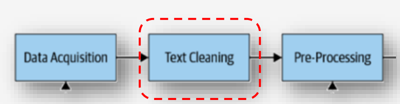


Stopword Removal

After

lowercased & punct removal & stopword removal

MSKU NLP CENG-3526 Natural Language Processing

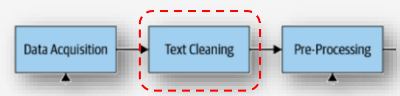


Stopword Removal: Important Note

While removing stop words can be beneficial in many cases, it's important to consider the specific requirements of your NLP task.

Sometimes,
stop words may contain important information
and removing them could lead to a loss of valuable context.

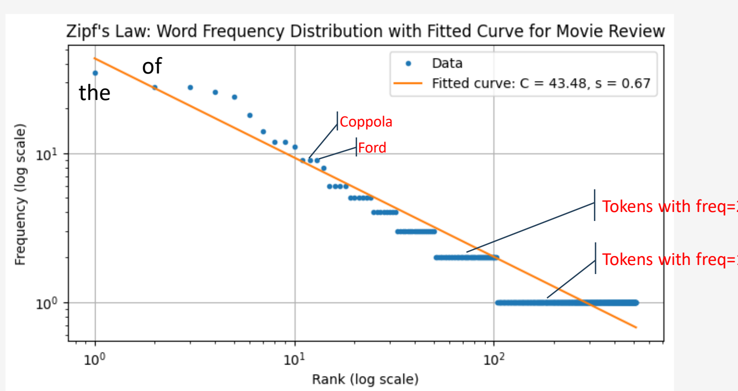
"to be or not to be that's the question"



17

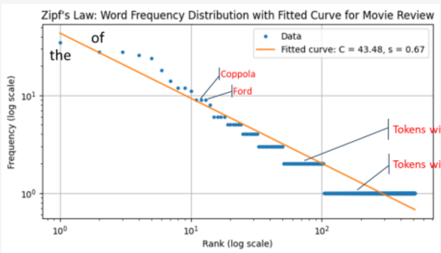
Low Frequency Tokens

Rank	Word	Frequency
...	...	...
85	somehow	2
86	just	2
87	performances	2
88	hes	2
89	while	2
90	plays	2
91	tries	2
...	...	...
132	chaotic	1
133	unspeakably	1
134	boringcoppola	1
135	melds	1
136	modernday	1
137	stab	1
138	commentary	1
139	america's	1
140	own	1
141	declining	1
142	empire	1
143	look	1
144	"make	1
145	great	1
146	again":	1
147	pointed	1
148	line	1



18

Hapax Legomena and Dis Legomena



Hapax legomenon:
A word that appears only once in a given text corpus.

Dis legomenon:
A word that appears twice in a given text corpus.

Significance of Hapax Legomena and Dis Legomena

Vocabulary richness

The number of hapax legomena can provide insights into the vocabulary richness of a text or author.
A high number of hapax legomena may indicate a diverse and specialized vocabulary.

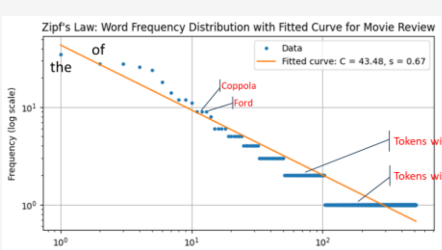
High Frequency vs Low Frequency Terms

Zipf's power law suggest that

The most frequent term appears *approximately* twice as frequent as (when *s=1.0*) the second most frequent term

freq("the") ~> 2 x freq("of")

35 ~> 2 x 28



Hapax legomenon:
A word that appears only once in a given text corpus.

Dis legomenon:
A word that appears twice in a given text corpus.

What if, in general for a language,

# of tokens with freq=1 **>** # of tokens with freq=2
(412) (53)

Growth Rate
412/53

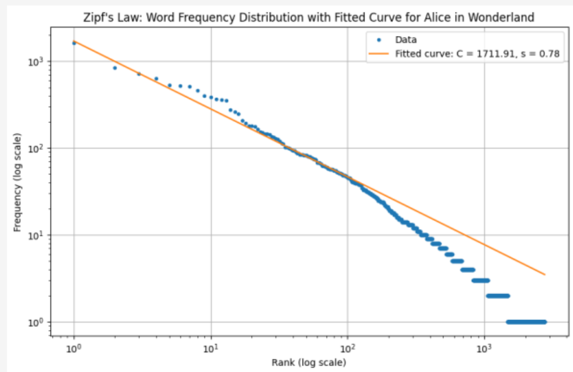
Open Vocabulary
(Unlimited Lexicon)

Example: Alice in Wonderland

Rank	Word	Frequency
1	the	1640
2	and	846
3	to	721
4	a	632
5	she	537
6	it	526
7	of	511
8	said	462
9	i	401
10	alice	385
11	in	369
12	you	360
13	was	357
14	that	276
15	as	262
16	her	248
17	at	209
18	on	193
19	with	181
20	all	179

freq(the) > 2 x freq(and)

1640 > 2 x 846

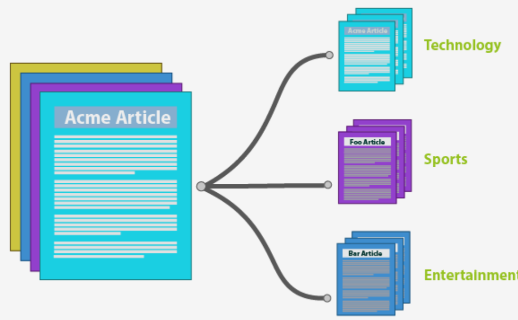


Text Classification

Text Classification

Classification tasks
involve categorizing text data
into **predefined classes or categories**.

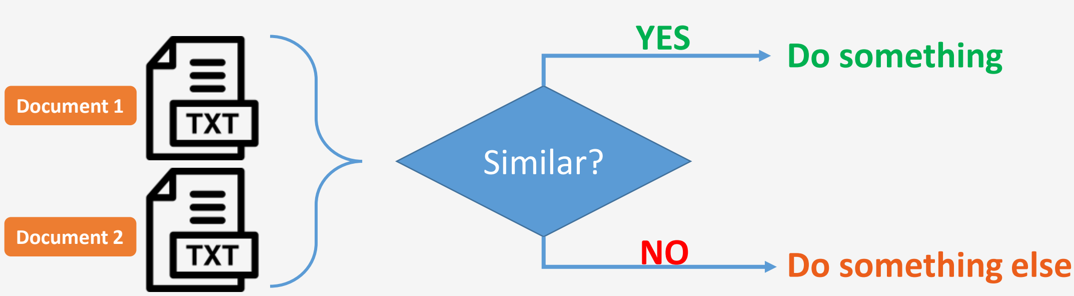
These tasks **aim**
to **assign labels or tags to text documents**
based on their content,
such as sentiment analysis, topic classification, or
intent detection.



23

Problem Definition: Similarity of Two Documents

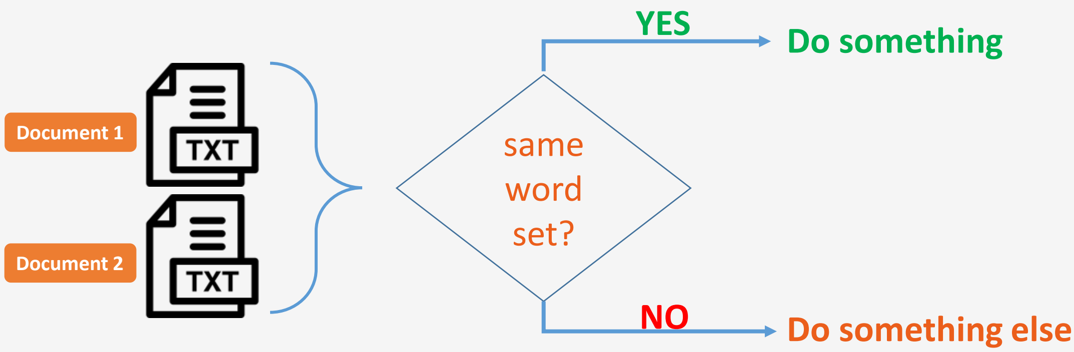
Given two or more **documents**,
determine
whether the they are **similar** to each other **in content**



24

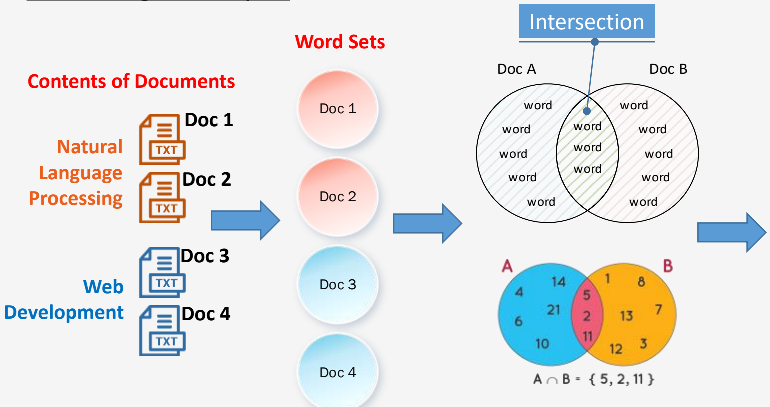
Trivial Approach to the Document Similarity Problem

if two documents are similar to each other,
they need to be composed of the same set of words



Trivial Approach: Results

Working Example



Results

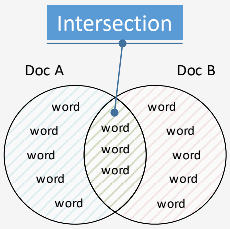
	Doc1	Doc2	Doc 3	Doc 4
Doc 1	222	54	38	45
Doc 2		178	29	37
Doc 3			216	79
Doc 4				231

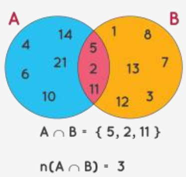
(Doc1, Doc2) < (Doc3, Doc4) : 54 > 79

(Doc1, Doc2) > (Doc1, Doc3) : 54 > 38
(Doc1, Doc2) > (Doc1, Doc4) : 54 > 45
(Doc1, Doc2) > (Doc2, Doc3) : 54 > 29
(Doc3, Doc4) > (Doc3, Doc1) : 79 > 38
(Doc3, Doc4) > (Doc3, Doc2) : 79 > 29
(Doc3, Doc4) > (Doc4, Doc2) : 79 > 37

Similarity Measure: Pros & Cons

Similarity Measure





Results

	Doc1	Doc2	Doc 3	Doc 4
Doc 1	222	54	38	45
Doc 2		178	29	37
Doc 3			216	79
Doc 4				231

Weakness:
Longer documents are favored!

Solution:
Document length normalization.

Improved Similarity Measure

$$similarity(doc_1, doc_2) = \frac{intersection}{union} = \frac{doc_1 \cap doc_2}{doc_1 \cup doc_2}$$

Pros:
- Magnitudes are comparable.
- 0 <= similarity <= 1

Previous Results

	Doc1	Doc2	Doc 3	Doc 4
Doc 1	222	54	38	45
Doc 2		178	29	37
Doc 3			216	79
Doc 4				231

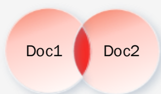
New Results

	Doc1	Doc2	Doc 3	Doc 4
Doc 1	1	**0.16**	0.10	0.11
Doc 2		1	0.08	0.10
Doc 3			1	**0.21**
Doc 4				1

Analysis of the Results: Within Theme

Natural Language Processing

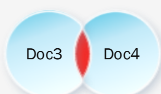
54



'human', 'this', 'while', 'is', 'several', 'may', 'as', 'in', 'also', 'development', 'to', 'other', 'used', 'computer', 'for', 'are', 'science,', 'subfield', 'a', 'processing', 'how', 'Natural', 'be', 'issue', 'with', 'over', 'and', 'computers', 'As', 'people', 'even', 'that', 'English', 'more', 'linguistics', 'NLP', 'intelligence', 'interactions', 'language.', '(NLP)', 'linguistics,', 'the', 'between', 'at', 'artificial', 'has', 'understanding', 'not', 'language', 'concerned', '(e.g.,', 'process', 'of', 'own'

Web Development

79



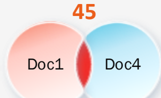
'complex', 'where', 'services.', 'user', 'to', 'other', 'used', 'range', 'In', 'involved', 'network', 'page', 'example,', 'which', 'goal', 'often', 'an', 'way', 'specific', 'it', 'Web', 'this', 'is', 'make', 'then', 'in', 'businesses,', 'if', 'website', 'plain', 'text', 'be', 'products', 'simple', 'some', 'these', 'the', 'private', 'also', 'work', 'for', 'hosting', 'a', 'by', 'one', 'Internet', 'can', 'they', 'Web)', 'not', 'become', 'developing', 'or', 'most', 'static', 'addition', 'social', 'as', 'development', 'are', 'network).', 'with', 'electronic', 'and', 'familiar', 'intranet', 'from', 'developers', 'that', 'such', 'web', '(a', 'applications,', 'Wide', 'part', '(World', 'single', 'of', 'very'

MSKU NLP | CENG-3526 Natural Language Processing

29

Analysis of the Results: Between Themes

45



'understand', 'this', 'social', 'is', 'may', 'as', 'in', 'also', 'development', 'them', 'to', 'other', 'used', 'computer', 'In', 'range', 'for', 'The', 'are', 'such,', 'a', 'be', 'with', 'thought', 'and', 'As', 'systems', 'can', 'providing', 'which', 'goal', 'an', 'that', 'tools.', 'such', 'through', 'way', 'technologies', 'the', 'using', 'learning', 'not', 'it', 'of', 'own'

23
$(Doc1 \cap Doc2) \cap Doc4$

'this', 'is', 'may', 'as', 'in', 'also', 'development', 'to', 'other', 'used', 'computer', 'for', 'are', 'a', 'be', 'with', 'and', 'As', 'that', 'the', 'not', 'of', 'own'

21
$(Doc1 \cap Doc2) \cap Doc3$

'this', 'is', 'as', 'in', 'also', 'development', 'to', 'other', 'used', 'for', 'are', 'a', 'how', 'be', 'with', 'and', 'that', 'more', 'the', 'not', 'of'

19
$Doc1 \cap Doc2 \cap Doc3 \cap Doc4$

'to', 'are', 'also', 'this', 'other', 'a', 'of', 'that', 'for', 'be', 'with', 'and', 'used', 'development', 'not', 'the', 'in', 'is', 'as'

MSKU NLP | CENG-3526 Natural Language Processing

30

Analysis of the Results: Observations

1. Stop-word elimination may help in robustness of decision making.

   It is apparent that we made decisions, mainly, based on a set of words that are common for all of the 4 documents: **29%** of $Doc1 \cap Doc2$, **27%** of $Doc3 \cap Doc4$.

   $Doc1 \cap Doc2 \cap Doc3 \cap Doc4$ { 'to', 'are', 'also', 'this', 'other', 'a', 'of', 'that', 'for', 'be', 'with', 'and', 'used', 'development', 'not', 'the', 'in', 'is', 'as' }

   In the context of NLP, such words that appear on every document are called "**stop-words**" or rather "function words".

   It is assumed that function words are used in languages because of grammatical necessity rather than serving in part of knowledge.

   On the other hand,
   The words that are not considered as function words are key words.

   Key words are those content bearing words that serve in part of knowledge.

Analysis of the Results: Observations

2. The `str.split()` as a tokenizer did not perform well for the current job: there are several mistakes that should be fixed for better measurement of similarity:

   `'science,'`, `'language.'`, `'(NLP)'`, `'linguistics,'`, `'(e.g.,'`, `'Web)'`, `'network).'`, `'(World'`, …

3. Text normalization (i.e. case-folding for the example) is necessary.
4. Stemming or Lemmatization may be applied to merge different surface forms of the same words having the same meaning, e.g.

   `'developing'`, `'development'`, `'developers'`, …

Model Improvement

1. Fine-tune tokenizer (w.r.t. Punctuations)
2. Apply normalization (case folding, contractions, abbreviations, etc.)
3. Stop-word elimination (not always help be cautious).
4. Apply either Stemming (Porter) or Lemmatization but not both. (Both method should not (cannot) be applied to the same text.)
5. Calculate similarity scores at each of the above steps progressively, i.e., after applying 1st improvement, after applying 1st and 2nd improvement, after 1st, 2nd and 3rd, so on.
6. Analyze the results as we did above at every progressive similarity score calculation cycle in 4.
7. Based on your observations, make suggestion of new improvements and if any, apply them and “repeat” starting from 4 until there left no room for improvement!?

Stemming & Lemmatization

Stemming

Stemming is a process that reduces words to their root form, **removing suffixes or prefixes**.

This is done to normalize words and group related words together, which can be helpful for tasks like text classification, information retrieval, and search.

Examples of stemming:

Running -> stemmed to "**run**"
Jumping -> stemmed to "**jump**"
Happily -> stemmed to "**happy**"
Countries -> stemmed to "**country**"
Unhappiness -> stemmed to "**unhappy**"

Grouping related words to gether

"*jumping*", "*jumped*", and "*jumps*"
all stemmed to
"**jump**"

MSKU NLP | CENG-3526 Natural Language Processing

36

Lemmatization

Lemmatization is a process of reducing words to their **dictionary form or lemma**.

Unlike stemming, lemmatization takes into account the grammatical context of a word to determine its root form.

This can result in more accurate results, especially for irregular verbs and nouns.

Examples of lemmatization:

Playing becomes "**play**"
Played becomes "**play**"
Plays becomes "**play**"

Happiness becomes "**happy**"
Unhappy becomes "**happy**"

Am becomes "**be**"
Is becomes "**be**"
Are becomes "**be**"

MSKU NLP | CENG-3526 Natural Language Processing

37

Document/Text Representation

Common Document Representation Models

Bag-of-Words (**BoW**):
Represents documents as a collection of words without considering order.

TF-IDF (Term Frequency-Inverse Document Frequency):
Combines word frequency with its importance across the corpus.

Word Embeddings:
Represents words as dense vectors in a continuous space, capturing semantic relationships.

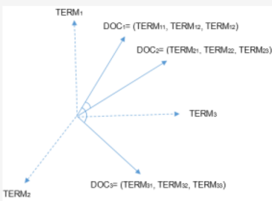
Document Embeddings:
Represents entire documents as dense vectors, capturing the overall semantic meaning.

Topic Modeling:
Identifies latent topics within a collection of documents.

Neural Network-Based Models:
Uses neural networks to learn complex representations of text data.

Bag-of-words (BOW) Model: Vector Space Model

Vector Space Model (Bag of Words Model)



3D Term-Space (TERM1, TERM2, TERM3)

Each doc is a point in term-space (i.e. 3D Vector)

Bag-of-Words (BoW)

Concept: Represents a document as a bag of words, where each word is assigned a numerical value based on its frequency in the document.

Pros: Simple to implement, computationally efficient.
Cons: Ignores word order and semantic relationships.

MSKU NLP | CENG-3526 Natural Language Processing

40

Bag-of-words (BOW) Model: Vector Space Model

Example Corpus

Doc 1: "The quick brown fox jumps over the lazy dog"
Doc 2: "The lazy dog sleeps in the sun"
Doc 3: "The quick brown fox jumps over the dog"
Doc 4: "The dog is lazy"

Vocabulary

Term-by-Document matrix

freq("the", doc3)=2

Word	Doc 1	Doc 2	Doc 3	Doc 4
the	2	2	2	1
quick	1	0	1	0
brown	1	0	1	0
fox	1	0	1	0
jumps	1	0	1	0
over	1	0	1	0
lazy	1	2	0	1
dog	1	1	1	1
sleeps	0	1	0	0
in	0	1	0	0
sun	0	1	0	0
is	0	0	0	1

Doc1 vector

Doc2 vector

MSKU NLP | CENG-3526 Natural Language Processing

41

TF-IDF Model: Vector Space Model

TF-IDF (Term Frequency-Inverse Document Frequency)

Concept: Combines term frequency (TF) with inverse document frequency (IDF) to assign weights to terms based on their importance within a document and across the corpus.

Pros: Addresses the shortcomings of BoW by considering term importance.
Cons: Can be sensitive to stop words and rare terms.

$$score(term_i, doc_j) = TF(term_i, doc_j) \times IDF(term_i)$$

where,

$$TF(term_i, doc_j) = freq\ of\ term_i\ in\ doc_j$$

$$IDF(term_i) = \log_{10}\left(\frac{N}{df(term_i)}\right)$$

$$N = \#\ of\ docs\ in\ corpus$$

$$df(term_i) = \#\ of\ docs\ that\ contain\ term_i$$

TF-IDF Model: Vector Space Model

Word	Doc 1	Doc 2	Doc 3	Doc 4
the	0	0	0	0
quick	0.693	0	0.693	0
brown	0.693	0	0.693	0
fox	0.693	0	0.693	0
jumps	0.693	0	0.693	0
over	0.693	0	0.693	0
lazy	0.173	0.693	0	0.347
dog	0.173	0.347	0.173	0.347
sleeps	0	0.462	0	0
in	0	0.462	0	0
sun	0	0.462	0	0
is	0	0	0	0.347

N = Total number of documents = 4

df(the) = 4 IDF(the) = log(4/4) = 0

df(quick) = 2 IDF(quick) = log(4/2) ≈ 0.693

$$score(the, doc_1) = TF(the, doc_1) \times IDF(the)$$
$$= 2 \times 0$$
$$score(quick, doc_1) = 1 \times 0.693$$

TF-IDF values are higher
for words that are common in a document but rare in the corpus.

TF-IDF Model: Vector Space Model

Word	Doc 1	Doc 2	Doc 3	Doc 4
the	0	0	0	0
quick	0.693	0	0.693	0
brown	0.693	0	0.693	0
fox	0.693	0	0.693	0
jumps	0.693	0	0.693	0
over	0.693	0	0.693	0
lazy	0.173	0.693	0	0.347
dog	0.173	0.347	0.173	0.347
sleeps	0	0.462	0	0
in	0	0.462	0	0
sun	0	0.462	0	0
is	0	0	0	0.347

N = Total number of documents = 4

df(the) = 4 IDF(the) = log(4/4) = 0

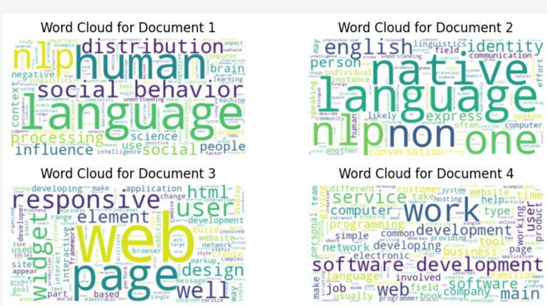
df(quick) = 2 IDF(quick) = log(4/2) ≈ 0.693

$$score(the, doc_1) = TF(the, doc_1) \times IDF(the)$$
$$= 2 \times 0$$
$$score(quick, doc_1) = 1 \times 0.693$$

TF-IDF values are higher
for words that are common in a document but rare in the corpus.

BoW Model vs TF-IDF Model

BoW Model



TF-IDF Model