NATURAL LANGUAGE PROCESSING

LESSON 8 : LEXICAL SIMILARITY

OUTLINE

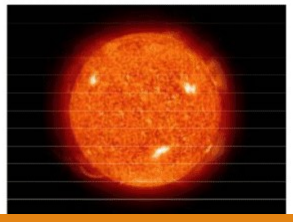
- **Lexical Similarity**
- **Similarity Measures**
  - Levenshtein Distance,
  - Jaccard Index,
  - Cosine Similarity
- **Lexical Vector Models**
  - Binary Weighting
  - Term Frequency (TF)
  - Term Frequency - Inverse Document Frequency (TF-IDF)

LEXICAL SIMILARITY

Lexical similarity is a perspective that focuses on the formal similarity of linguistic items. There is no goal of using any semantic details. The human brain responds most to visual stimuli. Therefore, Lexical similarity still has an important place in text comparison studies as it can provide a quick preliminary information.

peach 

peach 

sun 

LEXICAL SIMILARITY

Lexical similarity at the word level deals with the character position of the word, while semantic similarity considers the conceptual meaning depending on the approach.

	Lexical	Semantic
Sample-Simple	0.9	0
Sample-Instance	0.1	1

SIMILARITY MEASURES

Although there are many methods in the literature, the following three methods are still popular for lexical similarity. These are:

- Levenshtein Distance,
- Jaccard Index,
- Cosine Similarity.

* Levenshtein measures the dissimilarity of the two strings, while Jaccard and Cosine measure the similarity of them.

LEVENSHTEIN DISTANCE

Levenshtein is a distance metric for measuring the difference between two sequences. Informally, the Levenshtein distance between two words shows the minimum number of single-character edits required to change one word into the other. In measurement, three operations can be used:

1. Addition a character
2. Alternation a character
3. Deletion a character

LEVENSHTEIN DISTANCE

«HONDA» to «HYUNDAI»

(1) Add ‘Y’ => HYONDA
(2) Change ‘O’ to ‘U’ => HYUNDA
(3) Add ‘I’ => HYUNDAI

Levenshtein distance is 3

		H	Y	U	N	D	A	I
	0	1	2	3	4	5	6	7
H	1	0	1	2	3	4	5	6
O	2	1	1	2	3	4	5	6
N	3	2	2	2	2	3	4	5
D	4	3	3	3	3	2	3	4
A	5	4	4	4	4	3	2	3

H		O	N	D	A	
H	Y	U	N	D	A	I

H	O		N	D	A	
H	Y	U	N	D	A	I

JACCARD INDEX

Jaccard index is a statistic used for comparing the similarity of the diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets.

$$J(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} \times 100$$

JACCARD INDEX

Its algorithm contains these steps:

1. Count the number of elements (characters or words) in intersection of both sets (words or texts)
2. Count the number of elements in union of both sets
3. Divide the number of intersection by the number of union
4. Multiply the number found by 100

COSINE SIMILARITY

Cosine similarity is a function that measures the cosine value of the angle between two vectors.

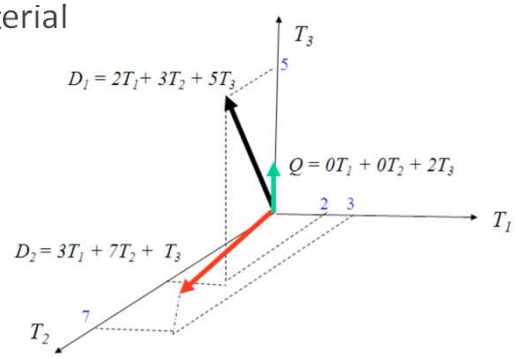
V1 and V2 : vectors

$$\mathrm{Cos}(\mathrm{V1}, \mathrm{V2}) = \frac{\overrightarrow{\mathrm{V1}}\ \overrightarrow{\mathrm{V2}}}{\|\mathrm{V1}\| * \|\mathrm{V2}\|}$$

However, in order to measure the cosine similarity of the words, it is first necessary to convert them to vectors.

LEXICAL VECTOR MODELS

In our previous lessons, we talked about the *benefits of converting texts to vectors*. Here, however, we will focus on methods that, unlike the previous ones, require less material in computation.



LEXICAL VECTOR MODELS

Considering the chronological evolution of lexical vector models, the three approaches, in order from primitive to complex, are as follows:

1. Binary Weighting,
2. Term Frequency (TF),
3. Term Frequency - Inverse Document Frequency (TF-IDF).

BINARY WEIGHTING

If a document includes a term, the weight of that term for the document is 1, otherwise 0.

Document1 : Bir kalem ve bir defter aldım.
Document2: Bir kitap aldım.
BoW: [aldım bir defter kalem kitap ve] (union of all words)

D1: [1 **1** 1 1 0 1]
D2: [1 1 0 0 1 0]

TERM FREQUENCY (TF)

If a document includes a term, the weight of that term for the document is TF (# of term), otherwise 0.

Document1 : Bir kalem ve bir defter aldım.
Document2: Bir kitap aldım.
BoW: [aldım bir defter kalem kitap ve] (union of all words)

D1: [1 **2** 1 1 0 1]
D2: [1 1 0 0 1 0]

TF-IDF

TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

The TF-IDF value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word.

$TF\text{-}IDF = TF_{norm} * IDF$ $TF_{norm} = TF / TF_{max}$

$IDF = 1+\ln(\#docs/\#docs_w)$

TF-IDF

Document1 : Bir kalem ve bir defter aldım.
Document2: Bir kitap aldım.

BoW : [aldım bir defter kalem kitap ve]

TF_{max}

TF(D1): [1 **2** 1 1 0 1] TF_{norm}(D1):[0.5 1.0 0.5 0.5 0.0 0.5]
TF(D2): [1 1 0 0 1 0] TF_{norm}(D2):[0.5 0.5 0.0 0.0 0.5 0.0]

TF-IDF

Document1 : Bir kalem ve bir defter aldım.
Document2: Bir kitap aldım.

BoW : [aldım bir defter kalem kitap ve]

$IDF_{aldım}$= IDF_{bir}=1+ln(2/2)=1
IDF_{kalem}=IDF_{defter}=IDF_{kitap}=IDF_{ve}=1+ln(2/1)=1.693

IDF:[1 1 1.693 1.693 1.693 1.693]

TF-IDF

TF_{norm}(D1):[0.5 1.0 0.5 0.5 0.0 0.5]
TF_{norm}(D2):[0.5 0.5 0.0 0.0 0.5 0.0]

IDF:[1 1 1.693 1.693 1.693 1.693]

TF*IDF (D1):[0.5 1.0 0.85 0.85 0.00 0.85]
TF*IDF (D2):[0.5 0.5 0.00 0.00 0.85 0.00]

EXAMPLE

Q: «Metin benzerliğini bulmak için TF-IDF ve kosinüs benzerliğini kullanacağız.»
D1: «DDİ bilgisayar biliminin bir konusudur.»
D2: «Metin benzerliğini bazı DDİ yöntemleri sayesinde buluruz.»

Stop Words: «bir», «için», «ve», «sayesinde», «bazı».

EXAMPLE - *JACCARD* INDEX

Q: «Metin benzerliğini bulmak için TF-IDF ve kosinüs benzerliğini kullanacağız.»
D1: «DDİ bilgisayar biliminin bir konusudur.»
D2: «Metin benzerliğini bazı DDİ yöntemleri sayesinde buluruz.»

Stop Words: «bir», «için», «ve», «sayesinde», «bazı».

# of shared words(Q, D1) = 0
# of shared words(Q, D2) = 2
# of total words = 11

J(Q,D1) = 0/11 = 0
J(Q,D2) = 2/11 = 0.18
According to Jaccard Index, Q is more similar to D2.

EXAMPLE - COSINE SIMILARITY

Q: «Metin benzerliğini bulmak için TF-IDF ve kosinüs benzerliğini kullanacağız.»
D1: «DDİ bilgisayar biliminin bir konusudur.»
D2: «Metin benzerliğini bazı DDİ yöntemleri sayesinde buluruz.»

	DDİ	Bilgisayar	Bilim	Konu	Metin	Benzerlik	Bulmak	TF-IDF	Kosinüs	Kullanmak	Yöntem
TF	0	0	0	0	1	2	1	1	1	1	0
IDF	**1.41**	**2.1**	**2.1**	**2.1**	**1.41**	**1.41**	**1.41**	**2.1**	**2.1**	**2.1**	**1.1**
TF_{norm}	0	0	0	0	0.5	1	0.5	0.5	0.5	0.5	0
TF*IDF	0	0	0	0	0.7	1.41	0.2	1.05	1.05	1.05	0

For Query (Q) document

EXAMPLE - COSINE SIMILARITY

Q: «Metin benzerliğini bulmak için TF-IDF ve kosinüs benzerliğini kullanacağız.»
D1: «DDİ bilgisayar biliminin bir konusudur.»
D2: «Metin benzerliğini bazı DDİ yöntemleri sayesinde buluruz.»

	DDİ	Bilgisayar	Bilim	Konu	Metin	Benzerlik	Bulmak	TF-IDF	Kosinüs	Kullanmak	Yöntem
TF	1	1	1	1	0	0	0	0	0	0	0
IDF	**1.41**	**2.1**	**2.1**	**2.1**	**1.41**	**1.41**	**1.41**	**2.1**	**2.1**	**2.1**	**1.1**
TF_{norm}	0.5	0.5	0.5	0.5	0	0	0	0	0	0	0
TF*IDF	0.7	1	1	1	0	0	0	0	0	0	0

For Document 1 (D1)

EXAMPLE - COSINE SIMILARITY

Q: «Metin benzerliğini bulmak için TF-IDF ve kosinüs benzerliğini kullanacağız.»
D1: «DDİ bilgisayar biliminin bir konusudur.»
D2: «Metin benzerliğini bazı DDİ yöntemleri sayesinde buluruz.»

	DDİ	Bilgisayar	Bilim	Konu	Metin	Benzerlik	Bulmak	TF-IDF	Kosinüs	Kullanmak	Yöntem
TF	1	0	0	0	1	1	1	0	0	0	1
IDF	**1.41**	**2.1**	**2.1**	**2.1**	**1.41**	**1.41**	**1.41**	**2.1**	**2.1**	**2.1**	**1.1**
TF_{norm}	0.5	0	0	0	0.5	0.5	0.5	0	0	0	0.5
TF*IDF	0.5	0	0	0	0.7	0.7	0.7	0	0	0	0.5

For Document 2 (D2)

EXAMPLE - COSINE SIMILARITY

Q: «Metin benzerliğini bulmak için TF-IDF ve kosinüs benzerliğini kullanacağız.»
D1: «DDİ bilgisayar biliminin bir konusudur.»
D2: «Metin benzerliğini bazı DDİ yöntemleri sayesinde buluruz.»

Cos(Q,D1) = 0

Cos(Q,D2) = 0.55

According to Cosine Similarity, Query (Q) is more similar to Document2 (D2).

DISADVANTAGE OF LEXICAL VECTOR MODEL

Lexical vector models have some disadvantages as well as advantages such as fast computation, easy adaptation to small corpora and clarity. Perhaps *the most significant drawback is the large vector sizes* that cannot be adjusted. For example, a corpus was created with 62K reviews from a website containing reviews of IMDB movie reviews. Approximately 160K words were identified in the bag-of-words of this corpus. The TF-IDF vectors to be calculated in such a corpus will have 160K dimensions. Considering that each cell of the vector will be 4 bytes, approximately 37 GB (160K x 62K x 4B) of free memory will be needed for all comments.

GET RID OF DISADVANTAGE

- If so, describe any other problems you notice.
  - Problems special to Turkish
- Next, let's discuss how we can get rid of these disadvantages.
- Please indicate your opinions on this subject by relating them to other subjects you have learned within the scope of NLP.

DOCUMENT SIMILARITY WITH LEVENSHTEIN

Can we measure document similarities by using Levenshtein distance?

If your answer is yes, how can we do it?

If you say no, why?