Plagiarism Detection Using Word Embedding

Erfaneh Gharavi, Kayvan Bijari, Kiarash Zahirnia, Hadi Veisi

University of Tehran

Outline of Presentation

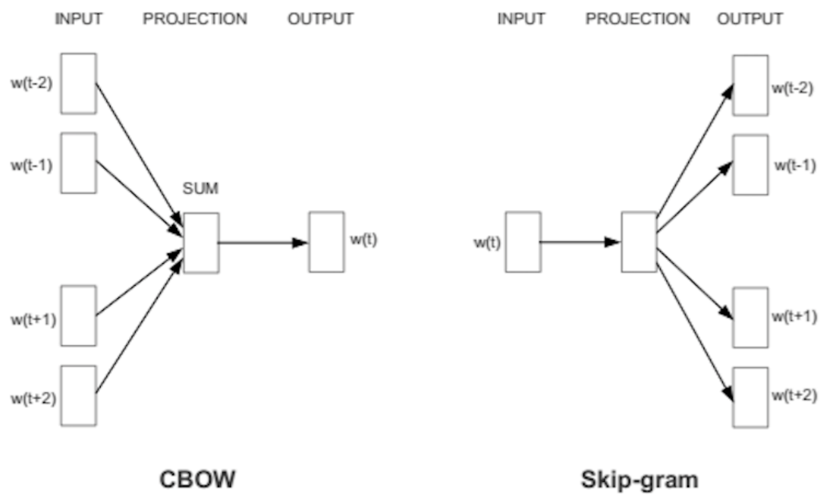
- Proposed Method
- Results
- Future work

1

Proposed Method

Persian word embedding

- Word2Vec
- Corpus: ISNA corpus
  - 500,000,000 words
- Retrieval:
  - HashMap

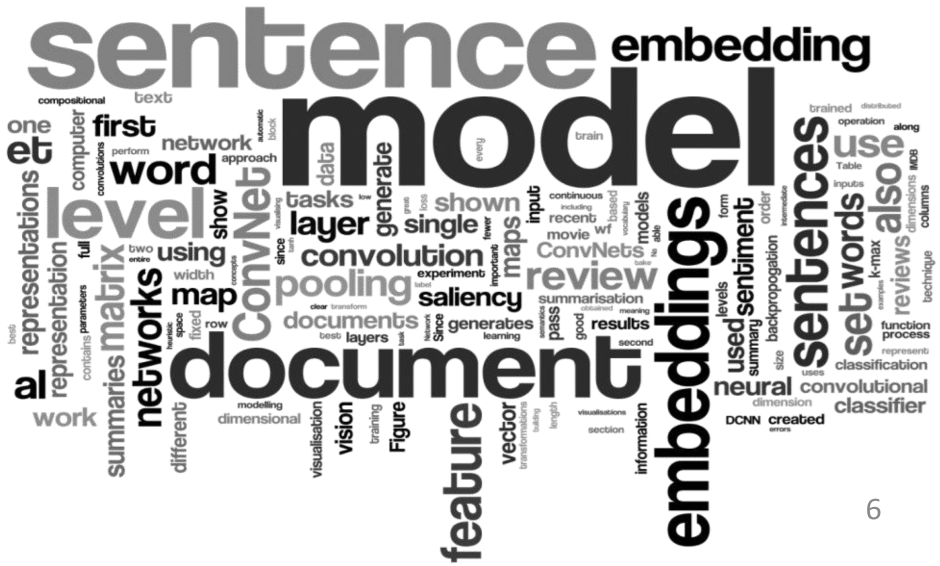


Split Documents

- Split
  - regex
- Preprocessing
  - Remove stop words
  - Omit punctuation

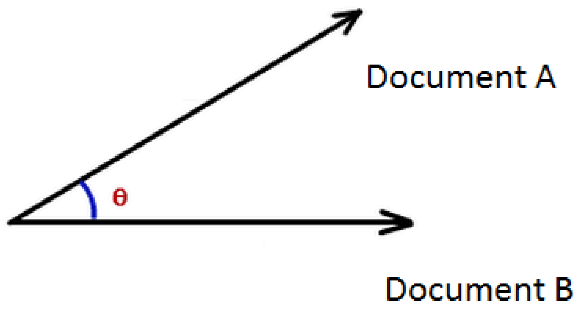
Sentences Representation

- Composition Function
  - Paragraph Vector
  - RAE (Recursive Auto-encoder)
  - Matrix Vector Recurrent Neural Networks
  - Mathematical
    - Averaging



Semantic Similarity

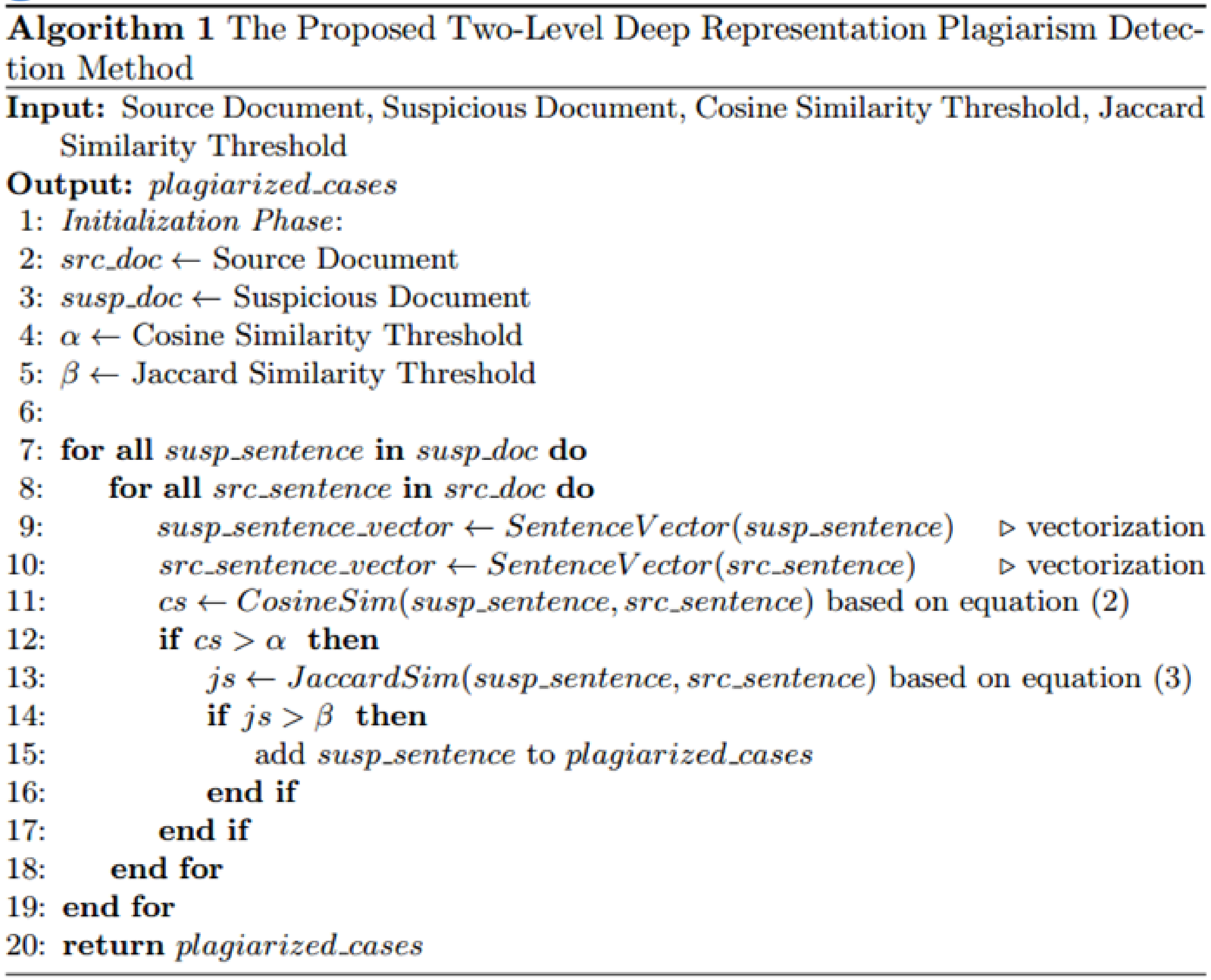
- Cosine Similarity of two sentence representation
- What is the minimum of cosine similarity measure to consider two sentences similar semantically?
- Threshold:
  - α=0.3



Lexical Similarity

- Jaccard Similarity of two sentences
  - including stop words
- What is the minimum of Jaccard similarity measure to consider two sentences similar lexically?
- Threshold:
  - β=0.2

Algorithm:



Algorithm 1 The Proposed Two-Level Deep Representation Plagiarism Detection Method

Input: Source Document, Suspicious Document, Cosine Similarity Threshold, Jaccard Similarity Threshold

Output: *plagiarized_cases*

```
 1: Initialization Phase:
 2: $src\_doc \leftarrow$ Source Document
 3: $susp\_doc \leftarrow$ Suspicious Document
 4: $\alpha \leftarrow$ Cosine Similarity Threshold
 5: $\beta \leftarrow$ Jaccard Similarity Threshold
 6:
 7: for all $susp\_sentence$ in $susp\_doc$ do
 8:     for all $src\_sentence$ in $src\_doc$ do
 9:         $susp\_sentence\_vector \leftarrow SentenceVector(susp\_sentence)$    ▷ vectorization
10:         $src\_sentence\_vector \leftarrow SentenceVector(src\_sentence)$      ▷ vectorization
11:         $cs \leftarrow CosineSim(susp\_sentence, src\_sentence)$ based on equation (2)
12:         if $cs > \alpha$ then
13:             $js \leftarrow JaccardSim(susp\_sentence, src\_sentence)$ based on equation (3)
14:             if $js > \beta$ then
15:                 add $susp\_sentence$ to $plagiarized\_cases$
16:             end if
17:         end if
18:     end for
19: end for
20: return $plagiarized\_cases$
```

2

Results

Results…

Rank / Team	Runtime (h:m:s)	Recall	Prec	Gran	F1	PlagDet
1 Mashhadirajab	02:22:48	**0.9191**	0.9268	1.0014	**0.9230**	**0.9220**
2 Gharavi	**00:01:03**	0.8582	0.9592	**1**	0.9059	0.9059
3 Momtaz	00:16:08	0.8504	0.8925	**1**	0.8710	0.8710
4 Minaei	00:01:33	0.7960	0.9203	1.0396	0.8536	0.8301
5 Esteki	00:44:03	0.7012	0.9333	**1**	0.8008	0.8008
6 Talebpour	02:24:19	0.8361	**0.9638**	1.2275	0.8954	0.7749

Results

Detection performance of the six approaches submitted, dependent on obfuscation type

Team	No obfuscation					Artificial Obfuscation					Simulated Obfuscation			
	Recall	Precision	Granularity	PlagDet		Recall	Precision	Granularity	PlagDet		Recall	Precision	Granularity	PlagDet
Mashhadirajab	**0.9939**	0.9403	**1**	0.9663		**0.9473**	0.9416	1.0006	**0.9440**		**0.8045**	0.9336	1.0047	**0.8613**
Gharavi	0.9825	0.9762	**1**	**0.9793**		0.8979	0.9647	**1**	0.9301		0.6895	**0.9682**	**1**	0.8054
Momtaz	0.9532	0.8965	**1**	0.9240		0.9019	0.8979	**1**	0.8999		0.6534	0.9119	**1**	0.7613
Minaei	0.9659	0.8663	1.0113	0.9060		0.8514	0.9324	1.0240	0.8750		0.5618	0.9110	1.1173	0.6422
Esteki	0.9781	0.9689	**1**	0.9735		0.7758	0.9473	**1**	0.8530		0.3683	0.8982	**1**	0.5224
Talebpour	0.9755	**0.9775**	**1**	0.9765		0.8971	**0.9674**	1.2074	0.8149		0.5961	0.9582	1.4111	0.5788

Corpora Evaluation

PlagDet performance of some submitted approaches on the submitted corpora

Team	Niknam	Samim	Mashhadi	ICTRC	Abnar
Gharavi	0.8657	**0.7386**	**0.5784**	**0.9253**	0.3927
Momtaz	0.8161	-	-	0.8924	-
Minaei	0.9042	0.6585	0.3877	0.8633	0.7218
Esteki	0.5758	-	-	-	0.3830
Ehsan	0.7196	0.5367	0.4014	0.7104	0.5890
Mansourizadeh	0.2984	-	0.1286	-	0.2687

More test …

Evaluated Word embedding and Jaccard similarity methods, separately on PAN@FIRE 2016 Persian PlagDet provided training data set.

	PlagDet	precision	recall
Two-Level Evaluation (word embedding+ Jaccard)	0.93	0.96	0.90
Word embedding	0.82	0.87	0.77
Jaccard	0.81	0.92	0.72

3

Conclusion & Future works

Advantages

- Fast
- Scalable
- No lexicon required (WordNet, FarsNet, ...)
- No handcrafted feature engineering
- No preprocessing
  - POS-tagging
  - Stemming

Challenges

- Simulated Plagiarism
- Threshold tuning
- Splitting
- Merging
  - coincident

Future works

- Other composition functions
- Word-by-word comparison

Thank
You
Gracias
Merci
Arigato
bolzïn
Grazie
Biyan
Juspaxar
Shukria
suksama
Mehrbani
Dankscheen
Tashakkur
Shukuria
gozaimashita
atu
Maake
Tingki
Yaqhanyelay
Efcharisto
Komapsumnida
Ekhmet