Document Similarity Benchmark

Jan-Gabriel Mylius

Lehrstuhl für Datenbanksysteme,
Institut für Informatik,
Ruprecht-Karls-Universität Heidelberg
mylius@posteo.de

10. February 2020, Heidelberg

Motivation

- NLP is often interested in meaning of a sentence
- Comparing the semantic similarity is a common task

Sentence 1	Sentence 2
The man is operating a stenograph.	The man is typing on a machine used for stenography.
”Fairies don’t exist” – fine.	”Satyrs don’t exist” – fine.

The Architecture

The benchmark suite consists of two essential elements:

- the algorithms
- the datasets

We want a framework:

- That compares algorithms on different datasets
- Is easily expandable
- Provides commonly used algorithms
- Provides common datasets

Algorithms

There is a template class for implementing custom algorithms. All algorithms do the following 3 things

- train - Training sentences pairs and scores available
- encode sentences - convert sentence to a vector
- compare sentences - two sentences given, number should be returned

Preimplemented Algorithms

For the benchmark suite I implemented the following:

- Bag of Words
- Bag of Words with lemmatization
- Word2Vec[Mik+15]
- BERT[Dev+18]
- Doc2Vec[LM14]

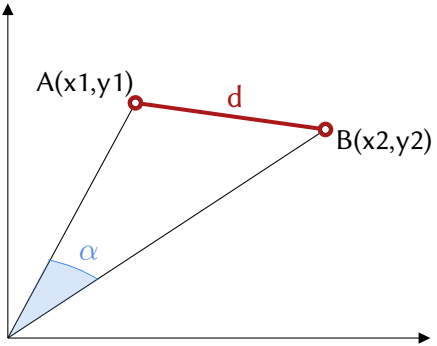
Functional Principle

How do these measures function:

1. Convert words to vector
2. Compare vectors using a similarity metric

For some algorithms several similarity metrics were implemented:

- Cosine similarity
- Euclidean similarity
- Jaccard similarity
- Word Mover's Distance[Kus+15]



Euclidean vs. Cosine Distance

The dataset class

The dataset class has the following essential functions:

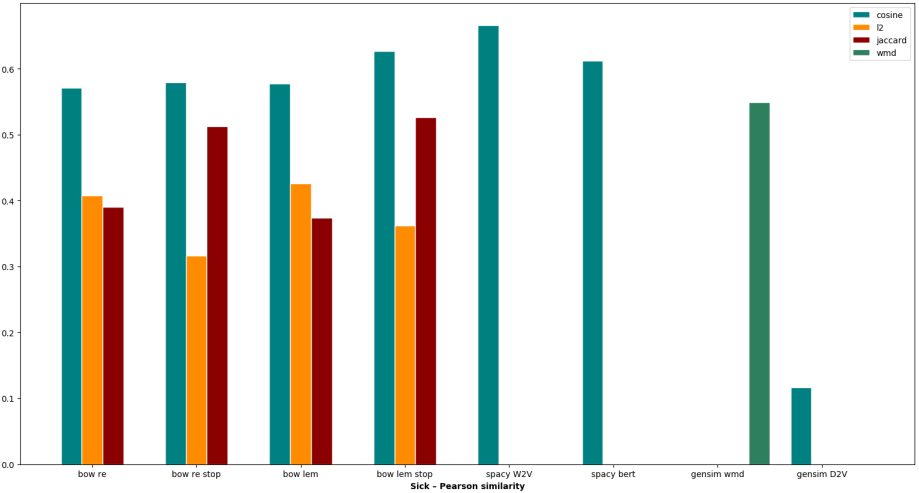
- load a dataset
  - STS
  - SICK
  - custom
- runs an algorithm
- norms results to match the datasets scale
- stores results for all algorithms that have run

The benchmark

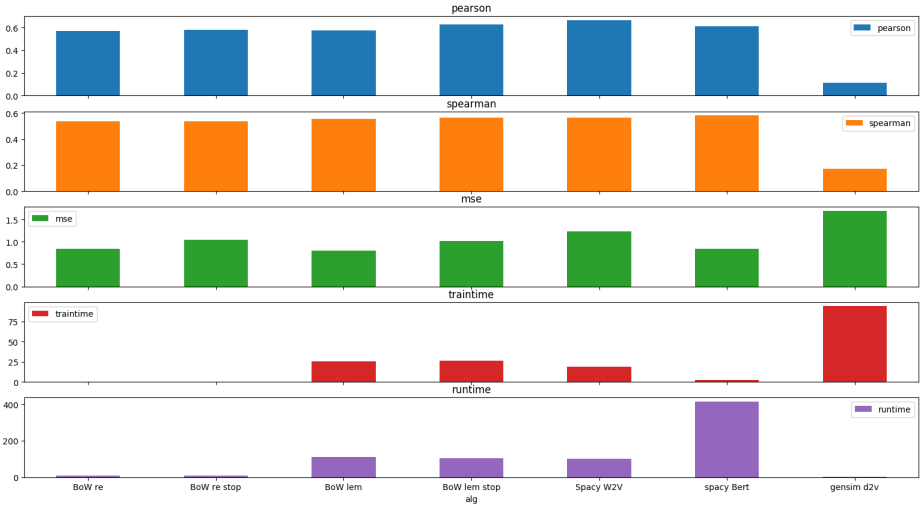
What the benchmark does:

- Accepts command line parameters:
  - No parameters: Run all algorithms
  - Else: Run given algorithms
- Measures:
  - Training Time
  - Run Time
  - Pearson correlation
  - Spearman's correlation
  - Mean squared error
- Outputs results as json file

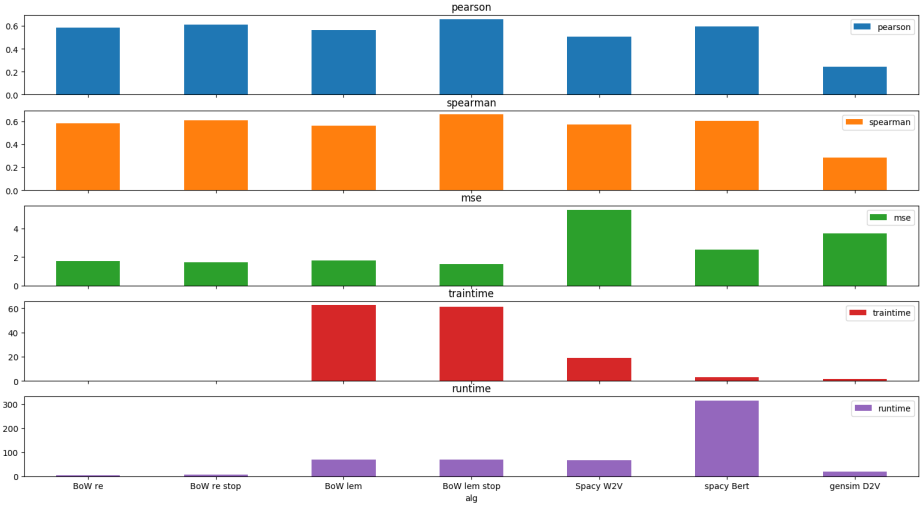
Results – Overview



Results – SICK cosine only



Results – STS cosine only



Discussion of Results

- l2, jaccard and wmd similarity not useful
- Filtering stop words improves results
- Lemmatization improves results
- Doc2Vec needs more training data
  - Maybe allow additional training data in future version
  - There is no currently compatible pretrained Doc2Vec model
    → Some one should train one for current gensim
  - More and larger datasets would be interesting to have
- BERT and Word2Vec not reliably better than BoW with lemmatization

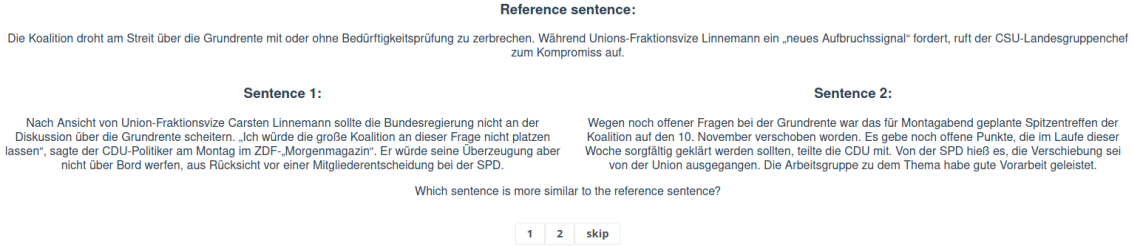
The Datasets – Lessons Learned

Set	Sentence 1	Sentence 2	Score
SICK	The man is operating a stenograph.	The man is typing on a machine used for stenography.	4.2(1-5)
STS	”Fairies don’t exist” – fine.	”Satyrs don’t exist” – fine.	1.2(0-5)
SICK	A man is jumping into an empty pool	There is no biker jumping in the air	1.6(1-5)

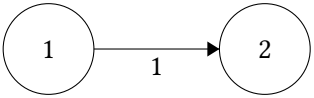
- Similarity and relatedness are not identical
- Semantic similarity/relatedness are hard to quantify
  - STS/SICK ratings don’t match each other
    → inter annotator consistency?
  - Finding consens on scale difficult
    → annotators training

→ different annotation method might be better

Interface



- Annotations are dictionaries of tuples
- Allows creation of different task format
- Annotations can be viewed as graph for each reference
  - Nodes: sentences
  - Edges: more similar relationship
  - Edge weights: number of annotators agreeing



Idea: Page-Rank-Algorithm to create scores

Conclusion

Lessons learned:

- Lemmatization and stop word filtering work
- Cosine similarity metric of choice
- Difference between similarity and relatedness matters
- BoW probably preferable to pretrained Word2Vec for unclean data
- Word2Vec probably generally better

Future work:

- Create option to generate additional training data
- Implement cutting edge algorithms
- Implement Page-Rank for annotator
- Create dataset to test similarity vs relatedness

Bibliography I

[Dev+18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[Kus+15] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. "From word embeddings to document distances". In: *International conference on machine learning*. 2015, pp. 957–966.

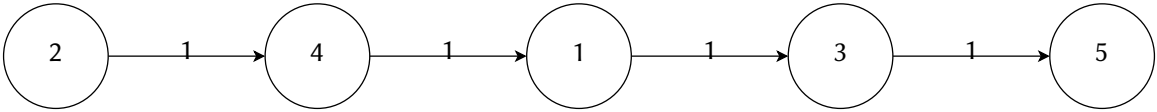
[LM14] Quoc Le and Tomas Mikolov. "Distributed representations of sentences and documents". In: *International conference on machine learning*. 2014, pp. 1188–1196.

[Mik+15] Tomas Mikolov, Kai Chen, Gregory S Corrado, and Jeffrey A Dean. *Computing numeric representations of words in a high-dimensional space.* US Patent 9,037,464. 2015.

[Pag+99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. *The pagerank citation ranking: Bringing order to the web.* Tech. rep. Stanford InfoLab, 1999.

Outline

The linear case

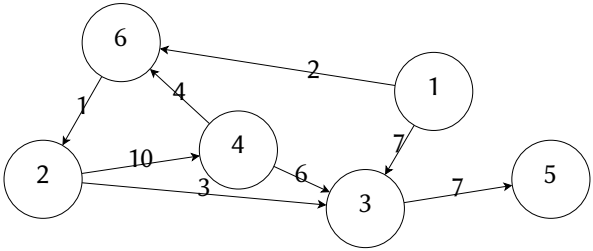


Expected result from a rational single user:

- Rank sentences, then divide by number of sentences $\approx$ score
- Problems:
  - Users aren't rational
  - Users might disagree

What does a realistic result look like?

The realistic case



Problems:

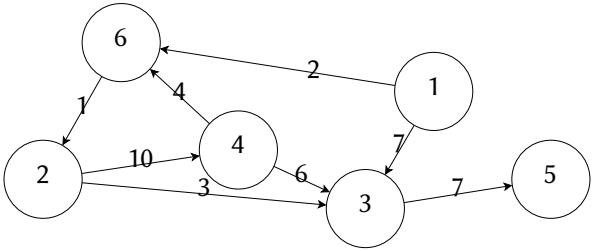
- Circular relationships exist
- Ranking is non obvious

Naive approach:

- Add up the edge weights for each node:
  - Node 3: 16, Node 5: 7
  - Does this match our expectations?

Better Solution?

The realistic case



Problems:

- Circular relationships exist
- Ranking is non obvious

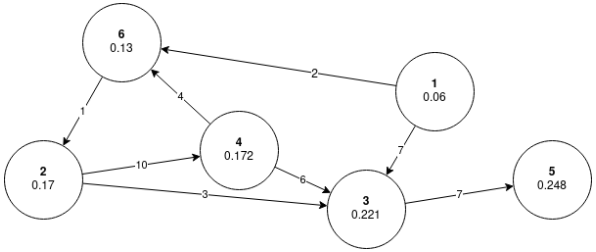
Naive approach:

- Add up the edge weights for each node:
  - Node 3: 16, Node 5: 7
  - Does this match our expectations?

Better Solution?

Page-Rank Algorithm[Pag+99]

The realistic case



Problems:

- Circular relationships exist
- Ranking is non obvious

Naive approach:

- Add up the edge weights for each node:
  - Node 3: 16, Node 5: 7
  - Does this match our expectations?

Better Solution?

Page-Rank Algorithm[Pag+99]

Suggested Page-Rank Implementation

- make sure reference sentence is rated to itself
- run page-rank over annotations
- norm the results over all sentences

Reference Sentence compared to itself $\longrightarrow$ upper bound.
Therefore this should give us decent scores.
Advantages:

- Easier to annotate than 1-5 scale.
- Quicker to annotate.
- Direct comparison data exists for sentence pairs.