

Semantic Textual Similarity

Bruno Sánchez Gómez and María del Carmen Ramírez Trujillo

December 12, 2024

Table of Contents

- 1 Introduction
- 2 Methodology
- 3 Results
- 4 Conclusion

1 Introduction

2 Methodology

3 Results

4 Conclusion

Introduction



- 1 Introduction
- 2 Methodology
- 3 Results
- 4 Conclusion

Methodology overview

- Feature extraction:
 - Syntactic features (8)
 - Lexical features (32)
 - String features (18)
- Preprocessing: remove punctuation and convert to lower case.
 - Extra preprocessing (for lexical and string feature extraction):
 - Stop words
 - Lemmatization
 - Word sense disambiguation
- Training models:
 - Multi-Layer Perceptron (MLP)
 - Support Vector Regressor (SVR)
 - Random Forest Regressor (RFR)

Syntactic Features

Process of analyzing a sentence's structure according to the grammatical rules and how words are related withing the sentence.

- N-grams overlap removing function words (prepositions, conjunctions, articles)

Sentence: "a horse eats carrots" \Rightarrow "horse eats carrots" (content words)

$$ngo(S_1, S_2) = 2 \cdot \left(\frac{|S_1|}{|S_1 \cap S_2|} + \frac{|S_2|}{|S_1 \cap S_2|} \right)^{-1}$$

Syntactic Features

- Syntactic roles similarity

python library Stanza (Stanford NLP Group, 2006)

Sentence: "a horse eats carrots and the man cleans the farm for the owner"

```
'p':  [{ 'carrot', 'eat' }, { 'owner', 'farm', 'for', 'clean' } ],
's':  [{ 'horse' }, { 'man' } ],
'o':  [{ 'carrot' }, { 'farm' }, { 'for', 'owner' } ]
```

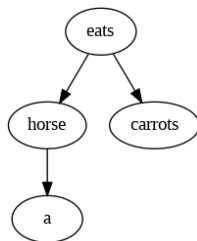
$$chunksim(C1, C2) = \sum_{l_1 \in C1} \sum_{l_2 \in C2} ssim(l_1, l_2)$$

Syntactic Features

- Syntactic dependencies overlap

python library Stanza (Stanford NLP Group, 2006)

Sentence: "a horse eats carrots"



$$\text{wdrc}(S_1, S_2) = \frac{\sum_{r \in S_1 \cap S_2} \max(\text{ic}(g(r)), \text{ic}(d(r)))}{\sum_{r \in S_2} \max(\text{ic}(g(r)), \text{ic}(d(r)))}$$

Lexical features

The process of analyzing a sentence's structure through the identification and classification of individual words.

- Jaccard distance
- Containment similarity
- Lin similarity
- Resnik similarity
- WordNet augmented word overlap
- Weighted word overlap
- Greedy lemma aligning overlap

Lexical features

The process of analyzing a sentence's structure through the identification and classification of individual words.

- Jaccard distance
- **Containment similarity**
- Lin similarity
- **Resnik similarity**
- **Weighted word overlap**
- **WordNet augmented word overlap**
- **Greedy lemma aligning overlap**

Lexical features

The sets S_1 and S_2 will be list of words or n-tuples of n-grams of each sentence.

- Containment similarity measure (Broder, 1997)

$$csimm(S_1, S_2) = \frac{|S_1 \cap S_2|}{\min\{|S_1|, |S_2|\}}$$

- Resnik similarity

$$resniksim(S_1, S_2) = ic(LCS(S_1, S_2))$$

- Weighted word overlap

$$wwc(S_1, S_2) = \frac{\sum_{w \in S_1 \cap S_2} ic(w)}{\sum_{w' \in S_2} ic(w')}$$

Lexical features

- WordNet augmented word overlap

$$P_{WN}(S_1, S_2) = \frac{1}{|S_2|} \sum_{w_1 \in S_1} \text{score}(w_1, S_2)$$

$$\text{score}(w, S) = \begin{cases} 1 & \text{if } w \in S \\ \max_{w' \in S} \{\text{pathsim}(w, w')\} & \text{otherwise} \end{cases}$$

- Greedy lemma aligning overlap

$$glao(S_1, S_2) = \frac{\sum_{(l_1, l_2) \in P} \max\{\text{ic}(l_1), \text{ic}(l_2)\} \cdot \text{linsim}(l_1, l_2)}{\max\{|S_1|, |S_2|\}}$$

P is the set of lemma pairs obtained by greedy alignment.

String Feature

- Character n-grams (Barrón-Cedeño, 2010)
 - `from sklearn.feature_extraction.text import TfidfVectorizer`

Sentence: "a horse eats carrots"

3-grams strings: ['a h', ' ho', 'hor', 'ors', 'rse', 'se ', 'e e', ' ea', 'eat', 'ats', 'ts ', 's c', ' ca', 'car', 'arr', 'rro', 'rot', 'ots']

$$\text{cossim}(A, B) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

- Greedy String Tiling

$$\text{gtsim}(S_1, S_2) = \frac{\sum_{t \in S_1 \cap S_2} \text{len}(t)}{\max\{\text{len}(S_1), \text{len}(S_2)\}}$$

Threshold for the length of the tile: 5, 10.

Model training and testing

- Multi-Layer Perceptron (MLP)
 - Support Vector Regressor (SVR)
 - Random Forest Regressor (RFR)
-
- Feature selection: 10 features with the highest absolute pearson correlation with gs.

- 1 Introduction
- 2 Methodology
- 3 Results**
- 4 Conclusion

Model & Feature Overview

- Models: MLP, SVR, RFR.
- Features:
 - Lexical, Syntactic, Strings (individually).
 - Unrestricted (Lexical + Syntactic + Strings).
 - FeatureSelection, based on:
 - Pearson correlation for MLP/SVR
 - Feature importance for RFR
- Performance measured using Pearson correlation with the Gold Standard

Results Summary

Features	MLP	SVR	RFR
Lexical	0.607	0.681	0.728
Syntactic	0.666	0.658	0.661
Strings	0.674	0.676	0.685
Unrestricted	0.652	0.744	0.757
FeatureSelection	0.744	0.742	0.745

- Best performance: RFR with Unrestricted (0.757)
- Syntactic features less informative than Lexical/Strings
- Feature combination improves SVR/RFR
- MLP suffers from overfitting

Top Features: Pearson Correlation

Top 5 features based on Pearson correlation with the Gold Standard:

Feature	Correlation
lemmas_wn_aug_overlap	0.7233
normal_char_2gram	0.7216
lemmas_char_2gram	0.6902
sw_char_2gram	0.6876
sw_gst_5	0.6666

Three key feature types:

- WordNet-Augmented Overlap (Lexical)
- Character n-grams (String-based)
- Greedy String Tiling (String-based)

Top Features: Feature Importance

Top 5 features based on Feature Importance scores from RFR:

Feature	Importance
lemmas_wn_aug_overlap	0.4325
normal_char_2gram	0.1630
chunk_sim_s	0.0413
lemmas_weighted_overlap	0.0275
normal_char_5gram	0.0199

The top 2 features:

- Common with Pearson correlation table.
- Have significantly higher importance, indicating their dominance in sentence similarity prediction

Feature types: 2 Lexical, 1 Syntactic, 2 Strings-related.

- 1 Introduction
- 2 Methodology
- 3 Results
- 4 Conclusion**

Conclusion

- Best performance: RFR with Unrestricted features (0.757 Pearson correlation).
- Key features: *WordNet-Augmented Overlap* and *Character n-grams*.
- Lexical and String-based features encode most of the relevant information for STS.
- Combining feature types (Lexical, Syntactic, Strings) significantly boosts performance.

References

- D. Bär, C. Biemann, I. Gurevych and T. Zesch. (2012). UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. 435-440.
- F. Sari, G. Glavaš, M. Karan, J. Snajder, B. Dalbelo Bašić. (2012). TakeLab: Systems for Measuring Semantic Text Similarity. Proceedings of SemEval-2012.