Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 189 (2021) 128–135

Procedia Computer Science

www.elsevier.com/locate/procedia

5th International Conference on AI in Computational Linguistics

Detecting Semantic Similarity Of Documents Using Natural Language Processing

Saurabh Agarwala[a,*], Aniketh Anagawadi[a], Ram Mohana Reddy Guddeti[a]

[a]*Department of Information Technology, National Institute of Technology Karnataka, Surathkal 575025, India*

Abstract

The similarity of documents in natural languages can be judged based on how similar the embeddings corresponding to their textual content are. Embeddings capture the lexical and semantic information of texts, and they can be obtained through bag-of-words approaches using the embeddings of constituent words or through pre-trained encoders. This paper examines various existing approaches to obtain embeddings from texts, which is then used to detect similarity between them. A novel model which builds upon the Universal Sentence Encoder is also developed to do the same. The explored models are tested on the SICK-dataset, and the correlation between the ground truth values given in the dataset and the predicted similarity is computed using the Pearson, Spearman and Kendall's Tau correlation metrics. Experimental results demonstrate that the novel model outperforms the existing approaches. Finally, an application is developed using the novel model to detect semantic similarity between a set of documents.

Keywords: Embeddings; Natural Language Processing; Semantic Similarity; Deep Learning; Computational Linguistic

1. Introduction

Word embeddings represent semantically meaningful representations of words from local co-occurrences in sentences. They have become nearly ubiquitous because of their utility to compute the semantic similarity between two words with ease and can be used to find words similar to a given word. Embedding generation methods represent words as continuous vectors in a low dimensional space that captures the words' lexical and semantic properties. A word embedding identifies the semantics and syntax of a word to build a vector representation of this information. However, there is no universal consensus on how sentence embeddings must be computed.

Sentence embedding techniques represent entire sentences and their semantic information as vectors. The context, semantics and other subtle features of the sentence can be represented in these embeddings. Many Natural Language Processing (NLP) applications need to compute the similarity in meaning between two texts. For example, search

* Corresponding author. Tel.: +91 84158 59101
E-mail address: saur.agarwala@gmail.com, anikethanaga@gmail.com, profgrmreddy@nitk.edu.in

1877-0509
10.1016/j.procs.2021.05.076

engines need to model the relevance of a document to a query going beyond the mere overlap in words between the two. The semantic relationship between the two texts is often quite subtle. For example, the texts ”a man is playing a harp” and ”a man is playing a keyboard” are judged as dissimilar, although they have the same syntactic structure and their constituent words have very similar embeddings.

In this paper, the various existing approaches used to obtain the text embeddings are explored, and a novel model to generate embedding and then predict the similarity between two input texts is described. Further, the approaches are evaluated on the SICK dataset, and the experimental results demonstrate that the novel model outperforms the other models based on the correlation between the ground truth values given in the dataset and the similarity values predicted by the model. The semantic similarity score can be utilised in various applications in academia, medical, literature, etc. One such application to detect semantic similarity between documents is developed as a proof-of-concept of the proposed model.

The rest of the paper is organised as follows: Section 2 describes the background and related work; Section 3 deals with the Methodology; Section 4 gives the Results and Analysis; Conclusions are detailed in Section 5.

2. Background And Related Work

Distributed representations for words contribute to better performance across many NLP tasks as demonstrated by Ma and Hovy [1]. Word embeddings can be obtained using various models such as Word2vec [2], GloVe [3] or ELMo [4].

Semantically similar words also have embedding vectors that are similar in the corresponding embedding space. Also, word embeddings can be used to obtain texts embedding if that text can be considered a bag-of-words.

TF-IDF (Term Frequency–Inverse Document Frequency) can be used to assign weights to words based on how important the 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-IDF can be used as a weighting strategy to obtain the embedding corresponding to a text using the embeddings of its constituent words.

Kusner et al., 2015 [5] introduced a distance function called Word Mover's Distance (WMD) that leverages word embeddings to find the similarity between two texts. It is assumed that the text documents are represented as normalised bag-of-words vectors, $d \epsilon R^n$ for a finite size vocabulary of n words, and that greater frequency of a word implies that it is more important. The semantic similarity between individual word pairs can be incorporated as the travel cost associated with travelling from one word to another in a semantic embedding space using Euclidean distance. The net distance between two texts is the minimum weighted cumulative cost required to move all words from the first text to the second. WMD does not consider word ordering since it employs a bag-of-words approach and it has high time complexity. The complexity of the vanilla version is $O(p^3 \log p)$ and even the enhanced version has $O(p^2)$ complexity.

Arora et al., 2017 [6] introduced a method to generate text embeddings called smooth inverse frequency (SIF) using word embeddings of the constituent words. Given an input text, weights are assigned to the constituent words, and then the average is taken to get a single vector. The weight assigned to the word embedding is $\frac{a}{a+p(w)}$, where a is a parameter usually set to 0.001, and the estimated frequency of the word in the reference corpus is denoted by $p(w)$. The principal components of the resultant embeddings for a set of components are computed. The projection of the sentence embeddings on their first principal component is subtracted from the original embeddings to remove the variation related to frequency and syntax that is less relevant semantically. SIF downgrades frequently occurring words and is also reasonably robust to the weighting scheme, i.e. using the word frequencies estimated from different corpora does not harm the performance. A wide range of the parameters 'a' can achieve close-to-best results, and an even more comprehensive range can achieve significant improvement over the unweighted average. One limitation is that the algorithm is slow. Also, being a bag of words approach, it does not take into account word order in the sentence when constructing the sentence embedding.

Pre-trained encoders are neural networks trained in a supervised manner to learn to construct the embeddings of sentences while paying particular attention to word order. They aim to play the same role as Word2vec and GloVe but for sentences. Pre-trained encoders are built to capture as much semantic information as possible by training on various supervised and unsupervised tasks.

Conneau et al., 2017 [7] proposed Infersent, a pre-trained encoder model which is a bi-directional LSTM (BiLSTM) with max-pooling layers. It is trained on the SNLI (Stanford Natural Language Inference) dataset, which consists of 570k English sentence pairs labelled with one out of the three categories: entailment, contradiction, and neutral. It uses GloVe vectors for getting the word embeddings.

Cer et al., 2018 [8] proposed the Universal Sentence Encoder, which generates a 512-dimensional embedding given an input text using the concept of deep averaging network (DAN) encoder where input embeddings for words and bigrams are averaged together and passed through a feed-forward deep neural network.

Singh et al., 2021 [9] proposed a method to compare the similarity between the text in online news articles in two languages (Hindi and English) that refer to the same event. The textual data is pre-processed, and its features are represented using TF-IDF based vector space and Bag-of-Words models. The similarity is compared using the Jaccard, cosine and Euclidean distance similarity measures. The text is not vectorised using advanced models like the Universal Sentence Encoder.

Gomaa et al., 2013 [10] surveyed three types of text similarity approaches: String-based, Corpus-based and Knowledge-based. Cosine similarity is a measure of the cosine of the angle between two vectors of an inner product space. The similarity between the two texts can be judged based on the cosine similarity between their respective embedding vectors.

Cross-language plagiarism occurs when a text is translated from a fragment written in a different language without appropriate citation. Barrón-Cedeño et al., (2010) [11] compared two cross-language plagiarism detection methods: CL-CNG (Cross-Language Character n-Grams) based on character n-grams and statistical translation based CL-ASA (Cross-Language Alignment-based Similarity Analysis). They also proposed a novel approach based on monolingual similarity analysis and machine translation. It was also shown that the novel technique performed the best in two considered language pairs (Basque and Spanish, Basque and English).

An approach introduced by Buscaldi et al., 2012 [12] combined a module that calculates the N-gram based similarity between sentences and another module that calculates the similarity between concepts in the sentences using WordNet and a concept similarity measure. A good correlation is achieved between automatic and manual similarity results.

Bär et al., 2012 [13] introduced a system named UKP (Ubiquitous Knowledge Processing) which uses a simple log-linear regression model based on training data that combines multiple text similarity measures such as string similarity, semantic similarity and measures related to structure and style and text expansion mechanisms. Out of the possible 300 implemented features, the final UKP final model consists of a log-linear combination of about 20 features and achieved fairly good correlation results.

Aggarwal et al., 2012 [14] presented an approach that combines knowledge-based semantic similarity scores calculated for the words falling under the same syntactic roles in both sentences and corpus-based semantic relatedness measure over the entire sentence. The scores were considered as input features to various models like bagging models and linear regression. A single score representing the degree of similarity between sentences was obtained. This method used both knowledge-based similarity measure and corpus-based relatedness measure to get better results in calculating semantic similarity between sentences.

The various models used to compute sentence similarity can be grouped into two categories based on their approach: Bag-Of-Words models that leverage word embeddings of the constituent words and Pre-Trained Encoders.

3. Proposed Work

The main intention of this paper is to compare the performance of different natural language processing methods used to obtain the semantic similarity score between texts of various forms, select the best performing method out of it and then construct an application that detects semantic similarity of documents to serve as a proof of concept.

Fig. 1 describes the steps followed to find the best performing method and then construct the application using it. The Pearson, Spearman and Kendall's Tau correlation metrics measure the strength and nature of the relationship between two sets of data. A value between -1 and 1 is returned where -1 indicates a strong negative relationship, 0 indicates no relationship at all and 1 indicates a strong positive relationship.

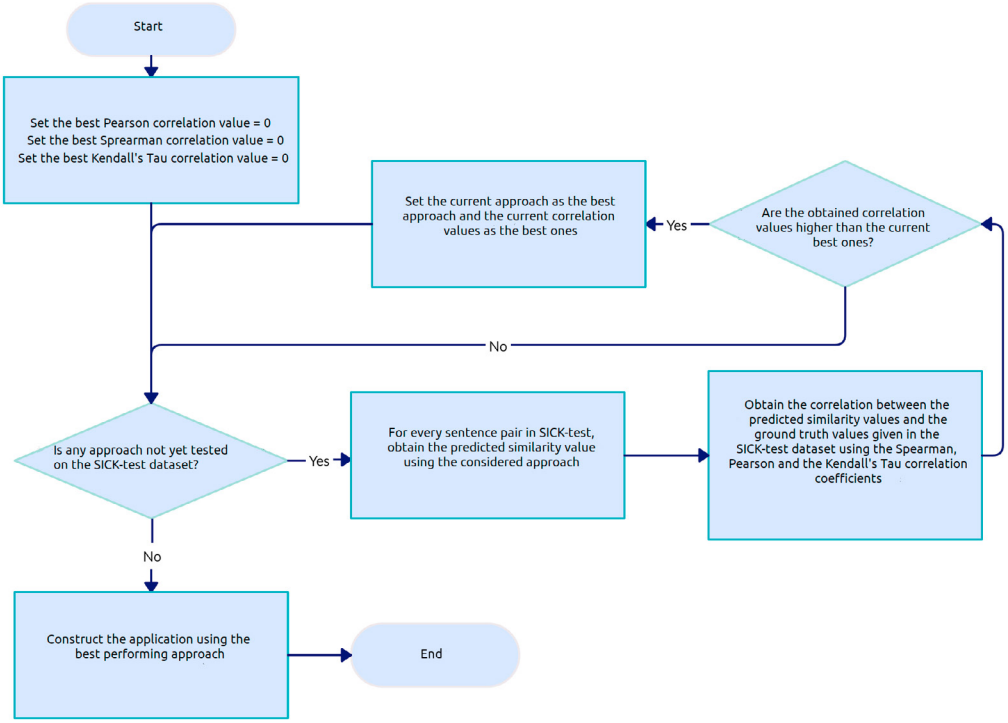

Fig. 1. Work flow of the proposed method

3.1. Dataset

The SICK dataset[1] which contains human similarity judgements for pairs of sentences, is used for experimentation. It contains 10,000 English sentence pairs labelled with their semantic relatedness. Each sentence pair is annotated on a scale from 0 to 5 for entailment through crowdsourcing techniques.

3.2. Methodology

The simplest way of estimating the similarity between a pair of texts is to take the average of the word embeddings of the constituent words and use a similarity function like cosine similarity between the resultant embeddings. This approach can be further improved by first removing the stopwords. This is considered as the baseline approach, against which the performance of all other models is compared.

3.2.1. Generating Text Embeddings

Text embeddings can be obtained from bag-of-words based approaches by using word embedding models like Word2vec, GloVe to obtain the embeddings for the constituent words of the text and then combining them using a strategy such as Word Mover's Distance, Smooth Inverse Frequency and TF-IDF. These text embeddings can also be obtained using pre-trained encoders like Infersent, etc. These generated text embeddings can be used to calculate the semantic similarity score between pair of texts.

[1] The SICK dataset can be found at: https://github.com/alvations/stasis/tree/master/SICK-data

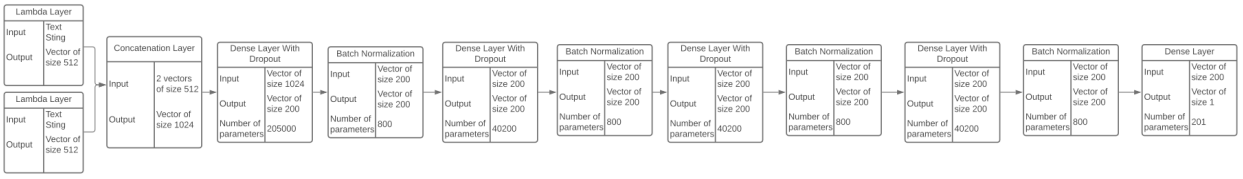

Fig. 2. Diagrammatic representation of the architecture of the novel model. The Universal Sentence Encoder is used in the Lambda layer.

3.2.2. Novel Model

A novel deep learning model is developed to find the semantic similarity between two texts. The novel model leverages the ability of the Universal Sentence Encoder to vectorize the input. The universal sentence encoder generates a fixed-length embedding of size 512 from the input sentences. These generated embeddings are concatenated and fed to the subsequent layers. The description of the layers used in the novel model (Fig. 2) are as follows:

- Input Layer: Treats text1 and text2 as input to the model
- Lambda Layer: Using the lambda layer of Keras, the embedding of the input is obtained before passing it into the rest of the model.
- Concatenate Layer: Concatenates the embeddings of text1 and text2
- Batch Normalization Layer: The distribution of each layer's inputs changes during training because the parameters of the previous layers change. Therefore the input at each layer is normalized.
- Dense Layer: Fully connected layer which consists of several activation units.
- Dropout Layer: To avoid overfitting on the test data, dropout regularisation is added to the hidden layers.

The development and training sets of the SICK dataset are combined and used to train the model. The ground truth similarity scores in the dataset are scaled to fit the interval [0,1], and the dataset is split into train and test set in the ratio of 80:20. Adam optimizer [15] is used for faster convergence and the ReLU (Rectified Linear Unit) [16] activation function is used in the hidden layers. The model is trained for 1000 epochs using a batch size of 512. The various approaches of generating the embeddings from input texts and computing the semantic similarity score between them are summarized in Table 1.

4. Experimental Results and Analysis

The experiments were carried out using Python3 on the Google Colab platform. Various Python3 libraries like NumPy, scikit-learn and Keras were used to create the novel model and test the approaches on the dataset.

Every approach listed in Table-1 is tested by feeding the pairs of sentences given in the SICK-test dataset to the model, obtaining the text embeddings and then calculating the semantic similarity value between them. The correlation between the predicted similarity values and the human determined (ground truth) similarity values for the same pairs of sentences given in the dataset is calculated using the Pearson, Spearman and Kendall's Tau correlation coefficients.

Fig. 3, Fig. 4 and Fig. 5 are graphical representations of the resultant Pearson, Spearman and Kendall's Tau correlation values between the predicted similarity obtained using the approaches mentioned in Table-1 and ground-truth values of the SICK-test dataset, respectively.

Based on the results obtained, it can be inferred that considering only the unweighted average of the word embeddings of the constituent words in the sentence (AVG-W2V and AVG-GLOVE) performs well for a simple baseline. Bag-of-Words based approaches that use Word2vec embeddings outperform approaches that use GloVe embeddings. It is also observed that the WMD based approaches perform poorly against the baseline approaches and that the InferSent performs slightly better than SIF-based approaches.

It is also evident that the novel model is clearly superior in performance to all the other approaches when judged using all three metrics. This demonstrates that the Novel model often gets the ordering of the sentences right and can generate accurate embeddings from input text which effectively captures the semantic information.

Table 1. List of the various approaches used to obtain text embeddings and the semantic similarity

Description Of The Approach	Notation
Obtaining the word embeddings of constituent words of the input text using Word2vec and averaging them	AVG-W2V
Obtaining the word embeddings of constituent words of the input text using Word2vec and averaging them after removing stopwords	AVG-W2V-STOP
Obtaining the word embeddings of constituent words of the input text using Word2vec and averaging them after applying TF-IDF weighting scheme	AVG-W2V-TFIDF
Obtaining the word embeddings of constituent words of the input text using Word2vec and averaging them after removing stopwords and then applying TF-IDF (Term Frequency - Inverse Document Frequency) weighting scheme	AVG-W2V-TFIDF-STOP
Obtaining the word embeddings of constituent words of the input text using GloVe and averaging them	AVG-GLOVE
Obtaining the word embeddings of constituent words of the input text using GloVe and averaging them after removing stopwords	AVG-GLOVE-STOP
Obtaining the word embeddings of constituent words of the input text using GloVe and averaging them after applying TF-IDF weighting scheme	AVG-GLOVE-TFIDF
Obtaining the word embeddings of constituent words of the input text using GloVe and averaging them after removing stopwords and applying TF-IDF (Term Frequency - Inverse Document Frequency) weighting scheme	AVG-GLOVE-TFIDF-STOP
Word Mover's distance in combination with Word2vec	WMD-W2V
Word Mover's distance in combination with Word2vec and stopword removal	WMD-W2V-STOP
Word Mover's distance in combination with GloVe	WMD-GLOVE
Word Mover's distance in combination with GloVe and stopword removal	WMD-GLOVE-STOP
Smooth Inverse Frequency in combination with Word2vec	SIF-W2V
Smooth Inverse Frequency in combination with GloVe	SIF-GLOVE
Infersent version 1	INF
The Novel Model	NOV

5. Conclusions

The semantic similarity score between a given pair of texts has a wide range of applications. This semantic similarity score is obtained by measuring the similarity between the embeddings of the pair of texts. In this paper, to calculate the semantic similarity, various custom-made bag-of-words based approaches are built using word embeddings techniques such as Word2vec and GloVe. Concepts like TF-IDF, Word Mover's distance and Smooth Inverse Frequency are also incorporated in the bag-of-words based approaches to generate better embeddings from the text. Pre-trained encoder models like Infersent are also used to generate the embeddings. Even a novel model to generate embeddings from input text is built using a Keras ensemble of Universal Sentence Encoder as a base and various layers evolving into a deep learning model trained on the SICK-train and SICK-dev datasets to calculate the semantic similarity score between two texts.

The various approaches are evaluated on the SICK-test dataset using the cosine similarity function between the generated embeddings for sentence pairs. The predicted similarity values are judged against the ground truth values using the Pearson, Spearman and Kendall's Tau correlation metrics.

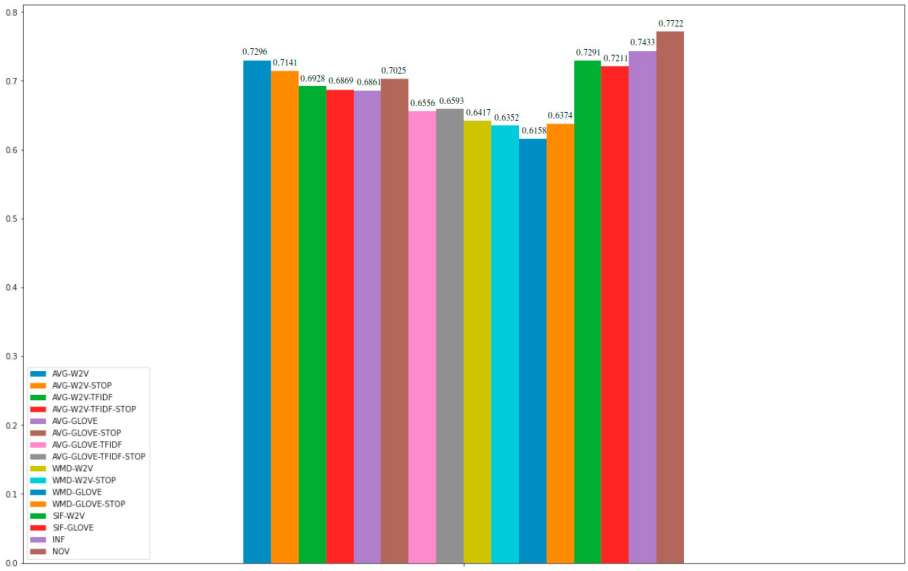

Fig. 3. Pearson Correlation Values

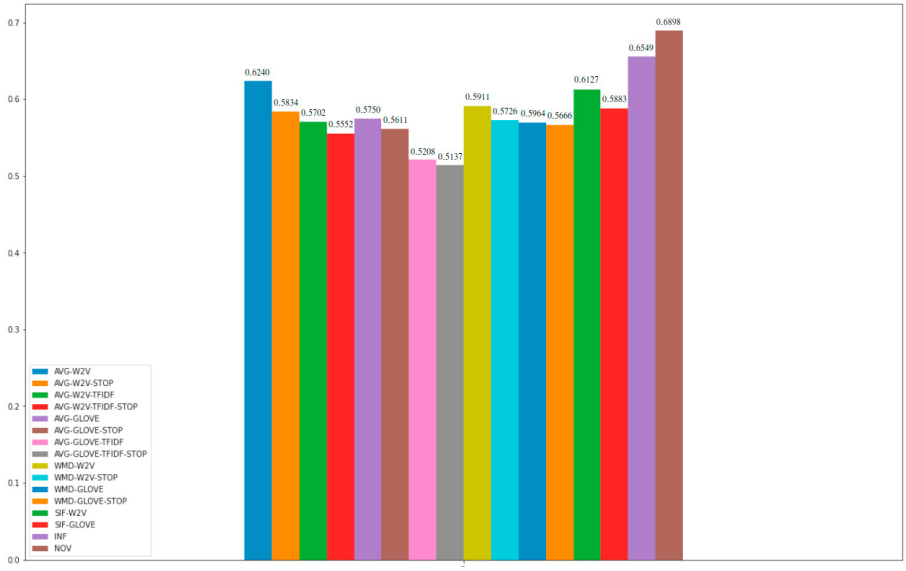

Fig. 4. Spearman Correlation Values

Experimental results demonstrate that the novel model outperforms all the other models and, therefore, can be used to capture the semantic information from the input text effectively.

Various applications are possible using semantic similarity. As a proof-of-concept, an application is built using the best performing approach (the novel model) to detect the semantic similarity given a set of documents. Further, the proposed work can be used in various fields such as medicine, literature and general academia. Lastly, it would also be interesting to incorporate techniques to compare the similarity between sets of images embedded in different documents in the future.

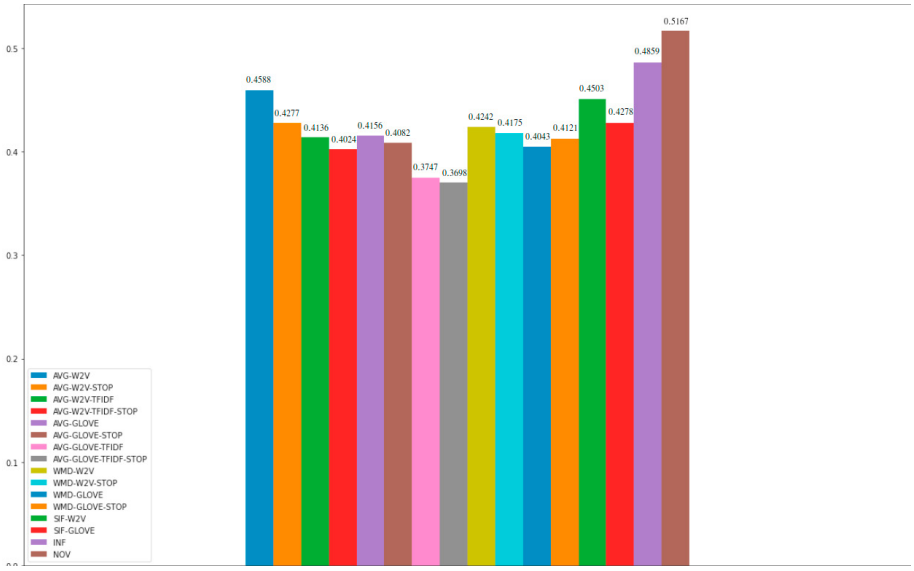

Fig. 5. Kendall's Tau Correlation Values

References

[1] DMa, Xuezhe and Hovy, Eduard (2016) "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF" *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* , Pages 1064-1074

[2] Tomas Mikolov and Kai Chen and G. S. Corrado and J. Dean (2013) "Efficient Estimation of Word Representations in Vector Space" *ICLR*

[3] Pennington, Jeffrey and Socher, Richard and Manning, Christopher (2014) "GloVe: Global Vectors for Word Representation" *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* , Pages 1532-1543

[4] Peters, Matthew and Neumann, Mark and Iyyer, Mohit and Gardner, Matt and Clark, Christopher and Lee, Kenton and Zettlemoyer, Luke (2018) "Deep Contextualized Word Representations" *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1* , Pages 2227-2237

[5] Kusner, Matt J. and Sun, Yu and Kolkin, Nicholas I. and Weinberger, Kilian Q. (2015) "From Word Embeddings to Document Distances" *Proceedings of the 32nd International Conference on International Conference on Machine Learning* , Pages 957–966

[6] Sanjeev Arora and Yingyu Liang and Tengyu Ma (2017) "A Simple but Tough-to-Beat Baseline for Sentence Embeddings" *ICLR*

[7] Conneau, Alexis and Kiela, Douwe and Schwenk, Holger and Barrault, Loïc and Bordes, Antoine (2017) "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data" *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* , Pages 670–680

[8] Daniel Matthew Cer and Yinfei Yang and Sheng-yi Kong and Nan Hua and Nicole Limtiaco and Rhomni St. John and Noah Constant and Mario Guajardo-Cespedes and Steve Yuan and C. Tar and Yun-Hsuan Sung and B. Strope and R. Kurzweil (2018) "Universal Sentence Encoder" *ArXiv* , abs/1803.11175

[9] Singh, R., Singh, S. (2021) "Text Similarity Measures in News Articles by Vector Space Model Using NLP" *J. Inst. Eng. India Ser. B* 102, 329–338

[10] Gomaa, Wael and Fahmy, Aly. (2013) "A Survey of Text Similarity Approaches" *International Journal of Computer Applications.* 68. 10.5120/11638-7118

[11] Barrón-Cedeño, Alberto and Rosso, Paolo and Agirre, Eneko and Labaka, Gorka (2010) "Plagiarism Detection across Distant Language Pairs" *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)* , Pages 37–45

[12] Buscaldi, Davide and Tournier, Ronan and Aussenac-Gilles, Nathalie and Mothe, Josiane (2012) "IRIT: Textual Similarity Combining Conceptual Similarity with an N-Gram Comparison Method" *First Joint Conference on Lexical and Computational Semantics (*SEM)* , Pages 552–556

[13] Bär, Daniel and Biemann, Chris and Gurevych, Iryna and Zesch, Torsten (2012) "UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures" *First Joint Conference on Lexical and Computational Semantics (*SEM)* , Pages 435-440

[14] Aggarwal, Nitish and Asooja, Kartik and Buitelaar, Paul (2012) "DERI & UPM: Pushing Corpus Based Relatedness to Similarity: Shared Task System Description" *First Joint Conference on Lexical and Computational Semantics (*SEM)* , Pages 643–647

[15] Diederik P. Kingma and Jimmy Ba (2015) "Adam: A Method for Stochastic Optimization" *CoRR* , abs/1412.6980

[16] Nair, Vinod and Hinton, Geoffrey E. (2010) "Rectified Linear Units Improve Restricted Boltzmann Machines" *Proceedings of the 27th International Conference on International Conference on Machine Learning* , Pages 807–814