International Journal of Scientific Research in Computer Science, Engineering and Information Technology

ISSN : 2456-3307

OPEN ACCESS

Available Online at : www.ijsrcseit.com

doi : https://doi.org/10.32628/CSEIT2390628

Contextual Sentence Similarity from News Articles

Nikhil Chaturvedi, Jigyasu Dubey

Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore, Indore, Madhya Pradesh, India

ARTICLEINFO

Article History:

Accepted: 20 Feb 2024

Published: 01 March 2024

Publication Issue

Volume 10, Issue 2

March-April-2024

Page Number

24-37

ABSTRACT

An important topic in the field of natural language processing is the measurement of sentence similarity. It's important to precisely gauge how similar two sentences are. Existing methods for determining sentence similarity challenge two problems Because sentence level semantics are not explicitly modelled at training, labelled datasets are typically small, making them insufficient for training supervised neural models; and there is a training-test gap for unsupervised language modelling (LM) based models to compute semantic scores between sentences. As a result, this task is performed at a lower level. In this paper, we suggest a novel paradigm to handle these two concerns by robotics method framework. The suggested robotics framework is built on the essential premise that a sentence's meaning is determined by its context and that sentence similarity may be determined by comparing the probabilities of forming two phrases given the same context. In an unsupervised way, the proposed approach can create high-quality, large-scale datasets with semantic similarity scores between two sentences, bridging the train-test gap to a great extent. Extensive testing shows that the proposed framework does better than existing baselines on a wide range of datasets.

Keywords: - sentence similarity, BERT, deep learning, Cosine Similarity

I. INTRODUCTION

Sentence similarity analysis is a well-established problem in natural language processing (NLP) (Luhn, 1957; Robertson et al., 1995; Blei et al., 2003; Peng et al., 2020). The job tries to use statistics to measure how similar two sentences are in terms of meaning. It has many uses in text search, Plagiarism detection, question answering, machine translation, and understanding natural language (Farouk et al., 2018; MacCartney and Manning, 2009).

The absence of large-scale labelled datasets containing phrase pairings with labelled semantic similarity scores is one of the biggest obstacles facing existing algorithms for sentence similarity. Such datasets need a lot of time and money to acquire. The STS benchmark (Cer et al., 2017) and SICK Relatedness dataset (Marelli et al., 2014) are examples of datasets with the right size for training deep neural networks. They have 8.6K and 9.8K labelled phrase pairs, respectively.

To solve this problem, a variety of learning techniques are offered, using word embeddings (Le and Mikolov, 2014) or BERT embeddings (Devlin et al., 2018) to unsupervised map sentences to fix-length vectors. Then, using the cosine or 2. product of various sentence representations, sentence similarity is calculated. Our research continues this trend by computing sentence similarity based on fixed-length sentence representations rather than by directly comparing sentences. The main problem with current methods is the significant time lag between model training and model testing (i.e., calculating semantic similarity between two sentences). For instance, BERT-style models are trained at the token level by predicting words in given situations; neither explicit sentence semantic modelling nor the production of phrase embeddings occurs during the training phase. However, to gain semantic similarity at test time, sentence semantics must be explicitly modelled. Due to the lack of consistency, the goals at the two stages are very different, and people do worse on tasks that require textual and semantic similarity.

The context of a sentence tells us what it means, just like the words around it tell us what a word means (Harris, 1954). Given the same situation, it is likely that two similar sentences will be made. If there is a small chance that two sentences will be made from the same context, there is a gap in the semantic space between these two sentences. Based on this idea, we propose a framework that measures semantic similarity by looking at how likely it is for two sentences to be generated in the same context. This is done without any human help. As for how it will be used, the framework has the following steps: (1) We train a contextual model by guessing how likely it is that a sentence will fit in the left and right contexts. (2) We find similarity between two sentences by comparing the scores that the contextual model gives in a lot of different contexts. To make it easier to draw conclusions, we train a surrogate model to act as step 2 based on the results of step 1. In an unsupervised setup, the surrogate model can be used right away to predict how similar two sentences are. In a supervised setup, it can be used as a starting point and then fine-tuned on later datasets. Note that the result of step 1 or the surrogate model is a sentence-specific vector with a fixed length.

Each element in the vector shows how well the input sentence fits the context of that element, and the vector can be thought of as the meaning of the input sentence in the context space. Then, to figure out how similar two sentence vectors are, we use the cosine distance between them.

The proposed framework has the potential to solve both problems: (1) the context regularisation provides a reliable way to generate a large-scale high-quality dataset with semantic similarity scores from an unlabeled corpus; and (2) the train-test gap can be naturally bridged by training the model on the large-scale similarity dataset, which leads to significant performance gains compared to using pretrained models directly.

We test different datasets in both supervised and unsupervised settings, and the results show that the proposed framework does a much better job than other sentence similarity models.

II. Related Work

Sentence embeddings are ways to represent sentences with a lot of details. They should have a lot of information about what sentences mean so that metrics like cosine similarity can be used to figure out how similar two sentences are to each other. Le and Mikolov (2014) came up with the paragraph vector, which learns on its own by guessing the words in a paragraph based on the paragraph vector. In a followup, methods like FastText, Skip-Thought vectors (Kiros et al., 2015), Smooth Inverse Frequency (SIF) (Arora et al., 2016), Sequential Denoising Autoencoder (SDAEs) (Hill et al., 2016), InferSent (Conneau et al., 2017), QuickThought vectors (Logeswaran and Lee, 2018), and Universal Large-scale pretraining models have had a lot of success (Devlin et al., 2018; Liu et al., 2019). This has recently sparked a line of work on producing sentence embeddings based on the pretraining finetuning paradigm (Reimers and Gurevych, 2019; Zhang et al., 2020; Ke et al., 2020; Wu et al., 2020; Li et al., 2020).

His work is about learning how words are represented based on their contexts (Mikolov et al., 2013; Le and Mikolov, 2014). This is based on the idea that the context of a word determines what it means. Our work is based on large, unlabeled corpora and aims to learn useful representations of sentences to measure how similar two sentences are.

For that Bag-of-words (BoW) (Li et al., 2006), term frequency inverse document frequency (TF-IDF) (Luhn, 1957; Jones, 1972), BM25 (Robertson et al., 1995), latent semantic indexing (LSI) (Deerwester et al., 1990), and latent dirichlet allocation (LDA) are all statistical methods for measuring sentence similarity (Blei et al., 2003). Deep learning methods for figuring out how similar two sentences are are based on distributed representations (Mikolov et al., 2013; Le and Mikolov, 2014) and can be roughly put into three groups: matrix-based, word-distance-based, and sentence embedding-based methods.

Table 1 show all categories of methods with its benefits and findings

S.No.	Paper Title	Author	Year	Method	Category	Benefit	Findings
	Deep Convolutional Extreme Learning Machine and Its Application in Handwritten Digit Classification	Pang el. al.	2016	Two-layer CNN	Matrix based	Evaluate systematic ally similarity measure	Deep CNN used for better performance
	Relay Backpropagation for Effective Learning of Deep Convolutional Neural Networks	He and Lin et. al.	2016	Deep CNN	Matrix based	Better performan ce	Used only for some specific dataset
	Inter-Weighted Alignment Network for Sentence Pair Modeling	Shen et. al.	2017	Sequentia l LSTM	Matrix based	emphasize d on each word in a sentence	They use additional lexical features
	Multiway Attention Networks for	Tan et. al.	2018	multiway attention networks	Matrix based	sentence pairs by encoding each	two sentences do not interact during the encoding part

	Modeling Sentence Pairs					sentence separately	
	Semantic Sentence Matching with Densely-Connected Recurrent and Co-Attentive Information	Kim el. al.	2019	densely-connected co-attentive RNN	Matrix based	They used attentive features as well as hidden features	alleviate the problem of an ever-increasing size of feature vectors due to dense concatenation operations
	Word Mover's Embedding: From Word2Vec to Document Embedding	Wu et. al.	2018	WMD (word mover distance)	Word distance based	Its outperformed word2vec model	Used only for word embedding
	Hierarchical Optimal Transport for Document Representation	Yurochkin et. al.	2019	hierarchical optimal topic transport document distances	Word distance based	Improved version of WMD	Used only for specific purpose
	Word Rotator's Distance	Yokoi et. al.	2020	optimal transport cost with alignment method	Word distance based	Its use two aspects of feature cover distance and angle	Strictly defind rule and norms of word vector
	Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks	Reimers et. al.	2019	SBERT	Sentence embedding based	This reduces the effort for finding the most similar pair	They implemented a smart batching strategy
	SBERT-WK: A Sentence Embedding Method by	Wang et. al.	2020	SBERT-WK	Sentence embedding based	They capture different properties	Still needed to fine the results

	Dissecting BERT-Based Word Models					using space span by word representation	
	Whitening Sentence Representations for Better Semantics and Faster Retrieval	Liu et. al.	2021	BERT with Whitening technique	Sentence embedding based	They outperformed the flow-based model	In this model require dimensional reduction operation
	Universal Sentence Encoder	Cer et. al.	2018	Word embedding	Sentence embedding based	They create universal sentence encoder for all purpose of NLP task	They use only word embedding technique
	CLEAR: Contrastive Learning for Sentence Representation	Wu et. al.	2020	Transformer encoder	Sentence embedding based	They focus on sentence level training	Better for sentence representation task
	Self-Guided Contrastive Learning for BERT Sentence Representations	Kim et. al.	2021	BERT	Sentence embedding based	Utilizing self-guidance for better sentence representation	Not present max of mean pooling in lowest layer
	A novel hybrid methodology of measuring sentence similarity	Yoo et. al.	2021	Hybrid (lexical and deep learning)	Hybrid	Advantages of both approach	They used only one similarity matrix
	Evaluating keyphrase extraction algorithms for finding similar	Sarwar et. al.	2022	Word embedding	Sentence embedding based	Both lexical and semantic feature used	Topic wise selection missing

	news articles using lexical similarity calculation and semantic relatedness measurement by word embedding						
	Text Similarity Measures in News Articles by Vector Space Model Using NLP	Singh et. al.	2020	Vector Space Model	Sentence embedding based		

III.Proposed Model

The main idea behind the proposed paradigm is to figure out how similar in meaning two sentences are by looking at how likely they are to be made in different situations.

We can reach this goal by doing the following: We need to first train a contextual model to figure out how likely it is that a sentence fits in the left or right context. This can be done either with a discriminative model, i.e., figuring out how likely it is that the concatenation of a sentence plus its context makes a text that makes sense, or a generative model, which predicts how likely it is that a sentence will be made given its context; Next, we can measure a pair of sentences by comparing their scores to see how similar they are. by contextual models given different contexts, after that test for any pair of sentences, we need to look at a lot of different situations to figure out scores given by models based on context, which is time-consuming.

So, we want to train a "surrogate" model that takes two sentences as input and predicts how similar the contextual model thinks they are. The surrogate model can be used directly to get sentence similarity scores in an unsupervised way, or it can be used as a starting point for a model that will be fine-tuned on datasets that come after it. Below, we'll talk about the specifics of each module in order.

3.1 Contextual Models: We need a model of context to figure out how likely it is that a sentence will fit in the left or right context. We do this by attempting to put together a generative model and a discriminative model, allowing us to take advantage of both models of text coherence (Li et al., 2017).

Notations we used: Let s_i denote the i^{th} sentence, which consists of a sequence of words $s_i = \{s_{i,1} \ldots, s_{i,ni}\}$, where n_i denotes the number of words in s_i . Let $s_{i:j}$ denote the i^{th} to j^{th} sentences. s_i respectively denote the preceding and subsequent context of s_i.

3.1.1 Discriminative Models

The discriminative model takes a sequence of consecutive sentences ($s_{<i}$, s_i, $s_{>i}$) as the input, and maps the input to a probability indicating whether the input is natural and coherent. We treat sentence sequences taken from the original articles written by humans as positive examples and sequences with replacements for the centre sentence ci as negative ones. Half of replacements s_i come from the original document, and half of the replacements come from random sentences from the corpus. For implementation, we use a single-layer bi-directional LSTM as the backbone with the size of hidden states is set to 300. The concatenation of LSTM representations at the last step (right-to-left and left-to-right) is used to represent the sentence. sentence representations for consecutives. To obtain the final probability, sentences are concatenated and fed into the sigmoid function:

$$P\ (y == 1 \mid s_{<i}, s_i, s_{>i}) = \text{sigmoid}\ (h^T [\ h_{<i}, h_i, h_{>i}]) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

where "h" stands for parameters that can be taught. We made the discriminative model simple on purpose for two reasons. First, using the discriminative approach to predict coherence is an easy task. Second, and more importantly, the discriminative approach will be used in the next selection stage for screening, where speed is more important.

3.1.2 Generative Models

Using SEQ2SEQ structures (Sutskever et al., 2014) as a backbone, the generative model predicts the likelihood of generating each token in sentence ci sequentially given contexts si and s>i.

$$p(s_i \mid s_{<i}, s_{>i}) = \prod p(s_{i,j} \mid s_{<i}, s_{>i}, s_{i,<j}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

The forward probability of generating the two sentences given the same context (p(si|si, s>i)) and the backward probability of generating contexts given sentences (p(si|si, s>i)) can be used to calculate the semantic similarity of two sentences. Predicting previous contexts given subsequent contexts (p (si|si, s>i)) and predicting subsequent contexts given previous contexts (p(s>i|si, si)) can be used to model the context-given-sentence probability.

We use Transformer-large as the backbone to implement the above three models: p(si|si, s>i), p(si|s>i, si), and p(s>i|si, si). These models are based on the SEQ2SEQ structure. Word embeddings are made better by adding sentence position embeddings and token position embeddings. We use Adam (Kingma and Ba, 2014), which has a learning rate of 1e-4, β_1 = 0.9, and β_2 = 0.999. The model is trained with 100B tokens from Common Crawl, which are taken from a corpus.

3.2 Scoring Sentence Pairs

The score for g_i fitting into context given context [s_i, $s_{>i}$] is the linear combination of scores from discriminative and generative models:

$$S(g_i, s_{<i}, s_{>i}) = \lambda_1 \log p(y=1 \mid s_i, s_{<i}, s_{>i}) + \lambda_2 \frac{1}{|gi|} \log p(g_i \mid s_{<i}, s_{>i}) + \lambda_3 \frac{1}{|s<i|} \log p(s_{<i} \mid g_i, s_{>i}) + \lambda_4 \frac{1}{|s>i|} \log p(s_{>i} \mid g_i, s_{<i}) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

where λ_1, λ_2, λ_3, and λ_4 control how different modules work together. To make things easier, we use c to show that $s_{<i}$ or $s_{>i}$ is a context. So, $S(g_i, s)$ is the same as $S(g_i, s_{>i}, s_{<i})$.

Let's call this group of contexts C, where N_C is the number of contexts in C. The semantic representation of a sentence, v_s, is an N_C-dimensional vector, with each value being S(g, c), where c is less than C. Based on v_{s1} and v_{s2}, different metrics, such as cosine similarity, can be used to figure out how similar s_1 and s_2 are semantically.

Constructing C we need to pay close attention to how C is put together. The best thing to do is to use all contexts, where C stands for the whole corpus. Unfortunately, this is not possible because we would have to go through the whole corpus for each sentence.

We suggest the following workaround for a computation that is easy to do. Instead of using the whole corpus as C for a sentence s, we build its sentence-specific context set Cs so that s can fit into any context in Cs. Here's what makes sense: When it comes to sentence s_1, contexts can be put into two groups: those in which s_1 fits and those in which it doesn't. We'll use the first group to figure out if s_2 also fits, and the second group to figure out if it doesn't. We mostly care about the first, and we can ignore the second. The reason is that the latter can also be split into two groups: situations that don't fit either s_1 or s_2, and situations that don't fit s1 but do fit s_2. We can ignore contexts that don't fit either s_1 or s_2, since two sentences not being in the same context doesn't mean they don't have the same meaning. If a context doesn't fit s_1 but fits s_2, we can leave it until we figure out Cs_2.

In practice, for a given sentence, we first perform primary screening on the entire corpus with TF-IDF weighted bag-of-word bi-gram vectors to find related text chunks (20K for each sentence). Next, we use the discriminative model from Eq.1 to rank all the situations. For discriminative models, we store sentence representations in advance and calculate model scores in the last neural layer, which is much faster than the generative model. This two-step selection strategy is like the pipelined selection system (Chen et al., 2017; Karpukhin et al., 2020) in open-domain QA, which includes document retrieval using IR systems and fine-grained question answering using neural QA models.

Cs is made up of the top contexts chosen by Eq. 3. We build by adding one context at a time, which is called the incremental construction method. To make sure there are a lot of different Cs, each text chunk can only contribute one context, and the Jaccard similarity between the i-th sentence in the context to choose and the sentences already chosen should be less than 0.5. Cs is set to a size of 500. To figure out how similar s_1 and s_2 are semantically, we put Cs_1 and Cs_2 together and use the result as the context set C. The score for how similar s_1 and s_2 are in terms of meaning is:

$$v_{s1} = [S(s_1, c) \text{ for } c \in Cs_1 + Cs_2]$$

$$v_{s2} = [S(s_2, c) \text{ for } c \in Cs_1 + Cs_2]$$

$$\text{sim}(s_1, s_2) = \text{cosine}(v_{s1}, v_{s2}) \quad \ldots\ldots(4)$$

1.3 Training Surrogate Model

The method explained in Section 3.2 is a straightforward way to figure out scores for semantic relatedness. But it is very slow at the time of inference given any two sentences, the model still must go through the whole corpus, collect the context set Cs, and go through each instance in Cs to calculate the context score based on Eq. (3). Each of these steps takes a long time. We plan to solve this problem by training a surrogate model to speed up inference.

We first get the similarity scores for each pair of sentences by following the steps in Section 3.3. We get scores for a total of 100M pairs, which are then split into train, development, and testing by 98/1/1. Next, we train a neural model that takes a pair of sentences as input and predicts their similarity score by using the collected similarity scores as gold labels.

We use the RoBERTa model (Liu et al., 2019) as the framework, and we use the Siamese structure (Reimers and Gurevych, 2019), in which RoBERTa is used to map two sentences to vector representations. We get the sentence representation by taking the average of the pools in the last RoBERTa layer. The predicted semantic similarity is the cosine similarity between the two sentence representations, and we try to minimise the L2 distance between the predicted and golden similarities.

The Siamese structure makes it possible to get and store fixed-size vectors for input sentences, which speeds up semantic similarity searches. We will talk more about this in the section on the ablation study.

When trained from scratch, the trained surrogate model gets an average L2 distance of 7.4 X10^{-4} on the dev set. When initialised with the RoBERTa-large model, it gets 6.1 X 10^{-4}. (Liu et al., 2019).

The pros and cons of the surrogate model should be considered. First, it can make inference much faster because it doesn't have to go through the time-consuming process of iterating over the whole corpus to build C. Second, the surrogate has the same structure as widely used models like BERT and RoBERTa, so it can be easily tuned with human-labeled datasets in supervised learning. On the other hand, the origin model in Section 3.2 can't be easily combined with other human-labeled datasets. As for the cons, the surrogate model is always less accurate because its upper limit is the origin model from Section 3.2.

IV. Experiments and Results

First, we use both unsupervised and supervised settings to test the proposed method on the Semantic Textual Similarity (STS) tasks. For the unsupervised setting, we use the STS tasks 2012–2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016), the STS benchmark (Cer et al., 2017), and the SICK-Relatedness dataset (Marelli et al., 2014) to evaluate. All datasets have pairs of sentences that are numbered from 0 to 5 to show how similar they are to each other. We find the Spearman's rank correlation between the cosine similarity of each pair of sentences and the gold labels.

We use the STS benchmark (STSb) to measure how well supervised STS systems work in the supervised setting. This dataset has 8,628 sentence pairs from three categories: captions, news, and forums. The training, development, and test sets each have 5,749, 1,500, and 1,379 sentence pairs. With the L2 regression objective function, we use the training set to fine-tune all of the models. For the test, we also figure out the Spearman's rank correlation between the cosine similarity of the sentence pairs and the gold labels.

We compare our proposed model to the following cutting-edge methods:

- **Avg. Glove embeddings** is the average of the word embeddings made by looking at how often words appear together in the corpus (Pennington et al., 2014).
- **Avg. BERT embeddings** is the average number of words that BERT embeds (Devlin et al., 2018).
- **BERT CLS-vector** is the vector representation of the special token [CLS] in BERT.
- **Universal Sentence Encoder** is a way to turn sentences into their corresponding embeddings. It is designed to help people learn how to do other NLP tasks (Cer et al., 2018).
- **SBERT** is a BERT-based method for getting sentence embeddings from the Siamese structure that can be compared using cosine similarity (Reimers and Gurevych, 2019).

In the unsupervised setup, the proposed models are used right away for inference. In the supervised setup, they are tweaked using the labelled datasets. We also fine-tune the model using both the SNLI (Bowman et al., 2015) and the Multi-Genre NLI (Williams et al., 2018) datasets. The SNLI has 570K sentence pairs and the multi-Genre NLI has 433K sentence pairs from different types of sources. Both sets of sentences are marked with one of the labels contradiction, entailment, or neutral. When the model is fine-tuned on NLI datasets, there is no labelled similarity dataset used, so the results are like those of unsupervised models. If the model is then fine-tuned on similarity datasets like STS, the results are like those of supervised models. Let's call the model that was also trained on NLI datasets *- NLI. For our proposed framework, we use Origin to represent the original model, where C for each sentence is made by searching the entire corpus, as described in Section 3.3, and similarity scores are calculated using Eq (4). We also tell you how Surrogate models of different sizes did (i.e., base, and large).

Table 1 shows what happened when no one watched. For the fully unsupervised setup, we see that the proposed models do better than baselines in a big way. Notably, the proposed models that are trained in an unsupervised setting (both origin and surrogate) can get results that are comparable to models that are trained on more annotated NLI datasets. Another thing that can be seen is that, as expected, the surrogate models do worse than the origin model. This is because the origin model acts as a ceiling for the surrogate model, but it does so at the cost of inference speed.

Model (unsupervised)	**STS12**	**STS13**	**STS14**	**STS15**	**STS16**	**STSb**	**Avg**
Avg. Glove embeddings	55.14	70.66	59.73	68.25	63.66	58.02	61.32
Avg. BERT embeddings	38.78	57.98	57.98	63.15	63.15	46.35	54.81
BERT CLS-vector	20.16	30.01	20.09	36.88	38.08	16.50	29.19
Universal Sentence Encoder	64.49	67.80	64.61	76.83	73.18	74.92	71.22
Origin	**72.41**	**74.30**	**75.45**	**78.45**	**79.93**	**78.47**	**76.93**
Surrogatebase	70.62	72.14	72.72	76.34	75.24	74.19	74.06
Surrogatelarge	71.93	73.74	73.95	77.01	76.64	75.32	75.20
Model (Supervised)	**STS12**	**STS13**	**STS14**	**STS15**	**STS16**	**STSb**	**Avg**
SBERT -NLI$_{base}$	70.97	76.53	73.19	79.09	74.30	77.03	74.89
SRoBERTa-NLI$_{base}$	71.54	72.49	70.80	78.74	73.69	77.77	74.46
Surrogate-NLI$_{base}$	74.15	76.50	72.23	81.24	78.75	79.32	77.25
SBERT-NLI$_{large}$	72.27	78.46	74.90	80.99	76.25	79.23	76.55
SRoBERTa-NLI$_{large}$	74.53	77.00	73.18	81.85	76.82	79.10	76.68
Surrogate-NLI$_{large}$	**76.98**	**79.83**	**75.15**	**79.32**	**80.82**	**79.64**	**79.33**

Table 1 Spearman rank correlation ρ between the cosine similarity of sentence representations and the gold labels for various Textual Similarity (STS) tasks under the unsupervised setting

Table 2 shows the results that were checked by an adult. We can see that the proposed Surrogate model outperforms baseline models by a large amount for both model sizes (base and large) and setups (with and without NLI training), giving an average gain of over 2 points on the STSb dataset.

Model	**Spearman ρ**
BERT$_{base}$	84.30
SBERT$_{base}$	84.67
SRoBERTa$_{base}$	84.92
Surrogate$_{base}$	**87.91**
BERT-NLI$_{base}$	88.33
SBERT-NLI$_{base}$	85.35
SRoBERTa-NLI$_{base}$	84.79
Surrogate-NLI$_{base}$	**89.95**
BERT$_{large}$	85.64
SBERT$_{large}$	84.45
SRoBERTa$_{large}$	85.02

Surrogate$_{large}$	**88.52**
BERT-NLI$_{large}$	88.77
SBERT-NLI$_{large}$	86.10
SRoBERTa-NLI$_{large}$	86.15
Surrogate-NLI$_{large}$	**90.69**

Table 2 : Spearman correlation ρ for the STSb dataset under the supervised setting.

Note that the Origin model can't be easily adapted to the partially supervised or supervised setting because it's hard to fine-tune the Origin model when the context set C needs to be built first. So, we tweak the surrogate model to make up for the loss of accuracy caused by switching from origin to surrogate. As we can see from Tables 1 and 2, the performance loss can be fixed by fine-tuning Surrogate on NLI datasets and STSb.

V. MODEL STRUCTURE

At first, we used the Siamese network structure to train the surrogate model. In this structure, two separate sentences are fed into the same model. It would be interesting to see what happens if you feed the model two sentences at once, like [CLS], s_1, [SEP], s_2, and then use the special token [CLS] to figure out how similar they are. This is how BERT classifies sentence pairs. To compare it to the Siamese model, we call it the BERT-style model.

By training the BERT-style model with the L_2 regression loss using the same harvested sentence pairs as the Siamese model, we get a Spearman's rank correlation of 77.43, which is slightly better than the Siamese model's result of 77.32. This is because the BERT structure does a better job of modelling how words and phrases in two sentences interact with each other. This is because interactions between words and phrases in two sentences start at the input layer with self-attentions. The two sentences don't interact in the Siamese structure until the output cosine layer.

The BERT structure's benefit of having enough interactions comes with a cost: for every new sentence pair, we must run the whole model again. This isn't the case with the Siamese structure, which speeds up searches for semantic similarity by storing representations of sentences ahead of time. In practise, we prefer the Siamese structure because it speeds up semantic similarity searches more than the BERT structure's small performance boost.

We initially used a Siamese network to train our surrogate model, where two distinct sentences are processed by the same model. However, we also explored feeding the model with two sentences at once, using the special tokens [CLS], s1, [SEP], s2, and using the token [CLS] to determine the similarity between the sentences. This approach, like BERT's method for classifying sentence pairs, is referred to as the BERT-style model.

Comparing the results, the BERT-style model using L2 regression loss and the same sentence pairs as the Siamese model resulted in a slightly higher Spearman's rank correlation of 77.43 compared to 77.32 for the Siamese model. This is due to the BERT structure's ability to better model the interactions between words and phrases in two sentences, starting at the input layer with self-attentions, while the Siamese structure only considers interactions at the output cosine layer.

However, the BERT structure's advantage comes at the cost of having to re-run the entire model for each new sentence pair, while the Siamese structure speeds up semantic similarity searches by storing sentence representations in advance. In practice, we prefer the Siamese structure for its faster search speed despite the slightly lower performance.

VI.CONCLUSION

We present a novel method for determining the similarity of two sentences. The approach is based on the principle that the likelihood of generating two similar sentences within the same context should be equal. Our method involves a pipeline process in which a vast number of sentence pairs and their similarity scores are initially gathered. We then use this data to train a surrogate model for faster inference. Results from numerous tests indicate that this framework outperforms existing sentence embedding-based techniques.

VII. REFERENCES

[1] Agirre, E., Banea, C., Cardie, C., Cer, D., Diab, M., Gonzalez-Agirre, A., ... & Wiebe, J. (2015, June). Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015) (pp. 252-263).

[2] Agirre, E., Banea, C., Cardie, C., Cer, D. M., Diab, M. T., Gonzalez-Agirre, A., ... & Wiebe, J. (2014, August). SemEval-2014 Task 10: Multilingual Semantic Textual Similarity. In SemEval@ COLING (pp. 81-91).

[3] Agirre, E., Banea, C., Cer, D., Diab, M., Gonzalez Agirre, A., Mihalcea, R., ... & Wiebe, J. (2016). Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511.. ACL (Association for Computational Linguistics).

[4] Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., & Guo, W. (2013, June). * SEM 2013 shared task: Semantic textual similarity. In Second joint conference on lexical and computational semantics (* SEM), volume 1: proceedings of the Main conference and the shared task: semantic textual similarity (pp. 32-43).

[5] Agirre, E., Cer, D., Diab, M., & Gonzalez-Agirre, A. (2012). Semeval-2012 task 6: A pilot on semantic textual similarity. In * SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012) (pp. 385-393).

[6] Arora, S., Liang, Y., & Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. In International conference on learning representations.

[7] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. Journal of machine Learning research, 3(Jan), 993-1022.

[8] Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large, annotated corpus for learning natural language inference. arXiv preprint arXiv:1508.05326.

[9] Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. arXiv preprint arXiv:1708.00055.

[10] Cer, D., Yang, Y., Kong, S. Y., Hua, N., Limtiaco, N., John, R. S., ... & Kurzweil, R. (2018). Universal sentence encoder. arXiv preprint arXiv:1803.11175.

[11] Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. arXiv preprint arXiv:1704.00051.

[12] Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised learning of

universal sentence representations from natural language inference data. arXiv preprint arXiv:1705.02364.

[13] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[14] Dor, L. E., Mass, Y., Halfon, A., Venezian, E., Shnayderman, I., Aharonov, R., & Slonim, N. (2018, July). Learning thematic similarity metric from article sections using triplet networks. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (pp. 49-54).

[15] Farouk, M., Ishizuka, M., & Bollegala, D. (2018, October). Graph matching based semantic search engine. In Research conference on metadata and semantics research (pp. 89-100). Springer, Cham.

[16] Hill, F., Cho, K., & Korhonen, A. (2016). Learning distributed representations of sentences from unlabelled data. arXiv preprint arXiv:1602.03483.

[17] Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., ... & Yih, W. T. (2020). Dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2004.04906.

[18] Ke, P., Ji, H., Liu, S., Zhu, X., & Huang, M. (2019). SentiLARE: Sentiment-aware language representation learning with linguistic knowledge. arXiv preprint arXiv:1911.02493.

[19] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[20] Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-thought vectors. Advances in neural information processing systems, 28.

[21] Le, Q., & Mikolov, T. (2014, June). Distributed representations of sentences and documents. In International conference on machine learning (pp. 1188-1196). PMLR.

[22] Li, B., Zhou, H., He, J., Wang, M., Yang, Y., & Li, L. (2020). On the sentence embeddings from pre-trained language models. arXiv preprint arXiv:2011.05864.

[23] Li, J., Monroe, W., Shi, T., Jean, S., Ritter, A., & Jurafsky, D. (2017). Adversarial learning for neural dialogue generation. arXiv preprint arXiv:1701.06547.

[24] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

[25] Logeswaran, L., & Lee, H. (2018). An efficient framework for learning sentence representations. arXiv preprint arXiv:1803.02893.

[26] Peng, S., Cui, H., Xie, N., Li, S., Zhang, J., & Li, X. (2020, April). Enhanced-RCNN: an efficient method for learning sentence similarity. In Proceedings of The Web Conference 2020 (pp. 2500-2506).

[27] Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.

[28] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[29] Williams, A., Nangia, N., & Bowman, S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference. arXiv preprint arXiv:1704.05426.

[30] Wu, Z., Wang, S., Gu, J., Khabsa, M., Sun, F., & Ma, H. (2020). Clear: Contrastive learning for sentence representation. arXiv preprint arXiv:2012.15466.

[31] Yang, M., Wang, R., Chen, K., Utiyama, M., Sumita, E., Zhang, M., & Zhao, T. (2019, July). Sentence-level agreement for neural machine translation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 3076-3082).

[32] Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X., & Zhou, X. (2020, April). Semantics-aware BERT for language understanding. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 34, No. 05, pp. 9628-9635).

Cite this article as :

Nikhil Chaturvedi, Jigyasu Dubey, "Contextual Sentence Similarity from News Articles", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 10, Issue 2, pp.24-37, March-April-2024. Available at doi : https://doi.org/10.32628/CSEIT2390628
Journal URL : https://ijsrcseit.com/CSEIT2390628