Sentences Semantic Similarity Detection with Ensemble Methods integrating BERT-based models.

Le Hoang Minh Ha
Hanoi, Vietnam
21020621@vnu.edu.vn

Nguyen Dieu Nhat
Hanoi, Vietnam
dieunhat@gmail.com

Abstract

Semantic Similarity Detection in Natural Language Processing is pivotal for measuring the level of text resemblance and has been applied in various applications. In this experiment, we adopt an Ensemble Methods from three BERT-based models to take advantages of their complementary strengths and weaknesses. We conduct our methods on the *Microsoft Research Paraphrase* (MSRP) (Dolan and Brockett, 2005) dataset, achieving the best accuracy of 0.9125 and F1-score of 0.9003.

Our code, models, and output files can be found here: https://github.com/minhha-lehoang/NLP-Project-Similarity-Detection.

1 Introduction

Sentences Semantic Similarity Detection is a prominent task in *Natural Language Processing* (NLP) whose concept is to evaluate the level of resemblance between pairs of provided texts based on a specific similarity metric. This has found applications in various aspects such as question–answering systems, information retrieval, sentiment classification, and so on.

A traditional approach for this task is knowledge-based method which put high emphasis on the relationships and meanings of documents, using structured knowledge sources such as WordNet (Fellbaum, 1998) or ConceptNet (Speer et al., 2016) to calculate semantic similarity between terms. Another method can be mentioned is corpus-based approach, whose concept is to measure semantic resemblance between texts using information retrieved from large corpora based on the idea that similar words have tendency to occur together. However, it is notable that the versatility of natural language poses a challenge in establishing rule-based methods for the determination of semantic similarity measures, making those approaches may fail to resolve the problem of ambiguity and polysemy in semantics. With that in mind, we believe an approach using large language models can be more efficient for our task. We chose to use *BERT* (Bidirectional Encoder Representations from Transformers)-based (Devlin et al., 2019) models, leveraging the advantages in contextual understanding and pre-trained representations.

BERT is a state-of-the-art Transformer (Vaswani et al., 2017) network in NLP based on self-attention mechanism. BERT uses *Masked Language Models* (MLM), which randomly mask some of the tokens from the input, and the objective is to predict the masked word from the corpus based only on its context, to pre-train deep bidirectional representations. This allows BERT to learn from different types of linguistic contexts and achieves state-of-the-art performance on various *Natural Language Understanding* (NLU) tasks without heavily-engineered task-specific architectures. Fine-tuning BERT-based models on domain-specific dataset has been a popular approach for Sentences Semantic Similarity Detection.

In our experiment, we implemented an Ensemble Method (Dietterich, 2000) of three pre-trained BERT-based models utilizing Cross-Encoder architecture in the baselines, on MSRP dataset.

2 Methods

2.1 Cross-Encoder

In this section, we will review two different architectures for semantic similarity detection - *Bi-Encoder* and *Cross-Encoder*, to verify our choice of using Cross-Encoder in the baseline.

Bi-Encoders (Figure 1) integrate two encoders at a time, producing each input sentence a corresponding sentence embedding. Each sentence is sequentially passed through a BERT and pooling block, resulting in two separate sentence embedding vectors. The cosine similarity of these two

vectors will therefore be computed based on the formula below, indicating the level of resemblance between two input sentences ranging from 0 to 1.

$$CosineSimilarity(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|}$$

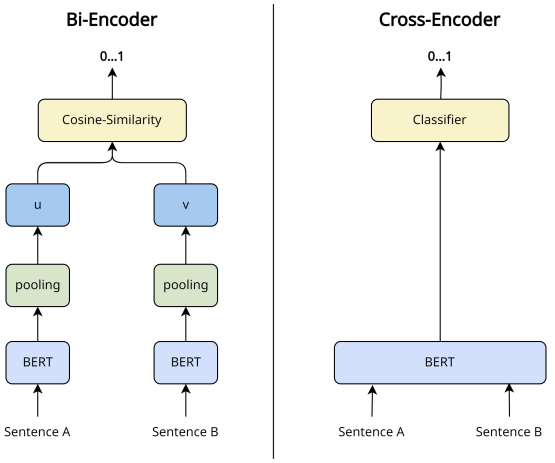

Figure 1: Illustration of Bi-Encoders *(left)* and Cross-Encoders *(right)* architectures.

On the other hand, **Cross-Encoders** (Figure 1), does not produce a sentence embedding for each input sentence but process them jointly in a single encoder. The combined input sentences will go through a BERT network and a classifier sequentially, resulting in an output between 0 and 1.

The differences between Bi-Encoders and Cross-Encoders lie on the mechanism of how they process and embed the input sentences pair, provoking a different use in different contexts. Bi-Encoders, with the ability to generate embedding for each sentence, will be suitable for tasks in which document retrieval and ranking is crucial, especially in large datasets. But in semantic similarity detection task where the understanding of the context and the relationship between input sentences are put in the center, the use of Bi-Encoders will not be beneficial as its mechanism indicates that the more overlapping between two inputs, the more similar they are. This suggests a light emphasis on the contextual understanding and semantic similarity of the sentences pair. In that setting, Cross-Encoders stand in and showcase its out-performance as for the ability to maintain semantic relationship between input texts.

2.2 Models

For this task, we utilize two BERT-based models: RoBERTa and DeBERTa.

RoBERTa (**R**obustly **O**ptimized **BERT** Pretraining **A**pproach) (Liu et al., 2019) is based on the original BERT architecture but was trained on a much larger dataset and used a more effective training procedure. Specifically, RoBERTa was trained with dynamic masking on longer sequences without *Next Sentence Prediction* (NSP), with much larger batches. RoBERTa achieves state-of-the-art performances and out-performs BERT on the *GLUE* (General Language Understanding Evaluation) (Ye et al., 2020) benchmark.

DeBERTa (**D**ecoding-**e**nhanced **BERT** with Disentangled **A**ttention) (He et al., 2020) was built on RoBERTa with *Disentangled Attention* and *EMD* (Enhanced Mask Decoder) training with half of the data used in RoBERTa. Unlike BERT where each word in the input layer is represented using a vector which is the sum of its word (content) embedding and position embedding, each word in DeBERTa is represented using two vectors that encode its content and position, respectively, and the attention weights among words are computed using disentangled matrices based on their contents and relative positions, respectively. Using EMD, instead of incorporating absolute positions in the input layer like BERT, DeBERTa incorporates absolute word position embeddings right before the softmax layer where the model decodes the masked words based on the aggregated contextual embeddings of word contents and positions (Figure 2). With those two improvements, DeBERTa outperforms RoBERTa on a majority of NLU tasks with 80GB training data. DeBERTaV3 (He et al., 2021) is an improved version of DeBERTa using ELECTRA-Style (Clark et al., 2020) pre-training with a new *GDES* (Gradient Disentangled Embedding Sharing) method. DeBERTaV3 demonstrates a remarkable improvement in performance compared to DeBERTa on downstream tasks.

In our experiment, we exploit the advantage of transfer learning by using those models pretrained on the *Semantic Textual Similarity* benchmark (STS-B) (Cer et al., 2017). STS-B comprises a selection of the English datasets used in the sentence semantic similarity detection tasks organized in the context of SemEval between 2012 and 2017. The selection of datasets include text from image captions, news headlines and user forums. Each

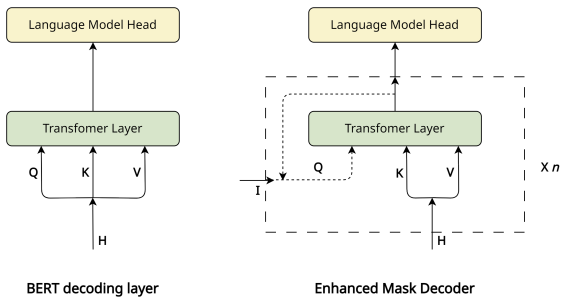

Figure 2: Illustration of BERT output softmax layer (*left*) and DeBERTa enhanced mask decoder (*right*)

sentence pair of the dataset received a score between 0 and 5, with 5 means that the human raters believe the sentences mean exactly the same, and 0 means that the sentences are completely unrelated to each other. According to Herbold's experiment results on semantic similarity detection across different datasets, the model fine-tuned for the STS-B predicts the similarity of sentences better than other approaches.

We hypothesize that using RoBERTa and DeBERTa models fine-tuned on the STS-B may achieve good performance at classifying whether a sentence pair is paraphrased (i.e., has the same meaning) or not.

2.3 Ensemble Methods

To leverage the strengths of different models, we applied a straightforward ensemble method to combine predictions from the three aforementioned models.

Ensemble method (Dietterich, 2000) is a learning algorithm whose main idea is to aggregate predictions from various models rather than relying solely on one with the aim of achieving more robust performance. Many different ensemble choices can be chosen, ranging from simple methods such as averaging, weighted averaging or voting, to more advanced techniques like bagging, boosting and stacking. The decision to choose ensemble methods should be made carefully, taking into consideration factors such as computational resources, complexity, and appropriateness with the task.

As our task is a binary classification problem and the models used already incorporate complexity, particularly with the attention-based mechanism, we opted for a straightforward ensemble method – majority voting (Figure 3). As suggested from the name, the prediction from this technique is determined by the class that receives the majority of votes from each model. It is worth noticing that to ensure a clear majority and avoid ties in votes, the number of models combined should be an odd.

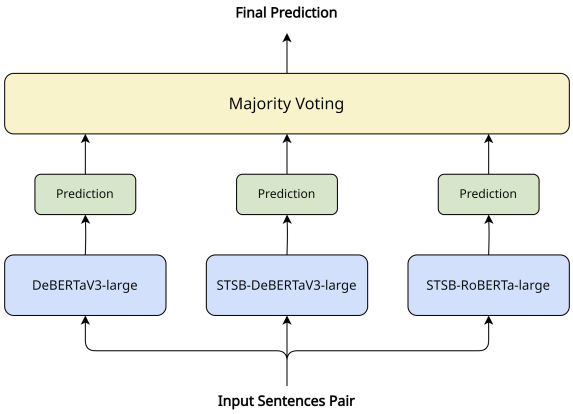

Figure 3: Illustration of ensemble methods using majority voting for our models.

3 Experiments

3.1 Datasets

The MSRP dataset is a collection of 5801 pairs of sentences from news articles, which have been annotated by humans to indicate whether they are paraphrases or not. The ratio of paraphrase to non-paraphrase pairs is 2:1. The dataset was split into training, development, and test sets, with 3576, 500, and 1725 number of samples each set, respectively.

3.2 Training Details

Initially, in the data pre-processing stage, We used NLTK (Bird and Loper, 2004) word tokenizer[1] to tokenize each sentence into tokens, remove punctuation tokens, before passing the input sentences pairs into the model for fine-tuning.

For the baseline, we employed the `CrossEncoder`[2] class, pre-defined in Sentence-Transformers (Reimers and Gurevych, 2019), a prominent Python framework for sentence embeddings.

All the pre-trained models we have used for this task are retrieved from *Huggingface*:

- RoBERTa-large fine-tuned on the STS benchmark using `CrossEncoder` class

[1] https://www.nltk.org/api/nltk.tokenize.html
[2] https://www.sbert.net/examples/applications/cross-encoder/README.html

by Reimers and Gurevych (2019) (STSB-RoBERTa-large)[3].

- DeBERTaV3-large[4].
- DeBERTaV3-large fine-tuned on the STS benchmark using `CrossEncoder` class (STSB-DeBERTa-V3-large)[5].

We initialized each individual pre-trained models as a regression Cross-Encoder that outputs a continuous score from 0 to 1, measuring the similarity between sentences in a pair. We fine-tuned each `CrossEncoder` model on 5 epochs with 10% warmup steps and 1500 evaluation steps. To achieve the expected outputs (binary labels) from models' continuous outputs on test set, a threshold of 0.5 was used to classify the scores. For the final predictions, we applied hard voting ensemble method from 3 previously chosen models.

3.3 Evaluation Metric

In our experiment, we evaluate the performance of the model based on accuracy and F1-score, widely used for binary classification tasks. The formulas of those metrics are shown below.

$$Accuracy = \frac{NumberofCorrectPredictions}{TotalNumberofPredictions}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

3.4 Results

Our method using ensembles achieves an accuracy of 0.9125 and macro F1-score of 0.9003 on the test set. Table 1 compares the results of each model individually with final results employing majority voting ensemble methods. This also highlights the superior-performance in every metrics by using ensemble methods instead of each separate model.

[3]https://huggingface.co/cross-encoder/stsb-roberta-large

[4]https://huggingface.co/microsoft/deberta-v3-large

[5]https://huggingface.co/yunyu/cross-encoder-stsb-deberta-v3-large

4 Conclusion

The outcomes of our experiments demonstrate that large pre-trained language models, such as RoBERTa and DeBERTa, exhibit strong performance on the task of semantic similarity detection within the MSRP benchmark, utilizing the Cross-Encoder architecture. Models that have undergone further pre-training on the STS benchmark are slightly more robust at differentiating between paraphrased sentence pairs and non-paraphrased pairs compared to the base models. The application of a hard voting ensemble method on the predictions from these models yields superior results compared to any of our single model.

Nevertheless, it is worth keeping in mind that our approach using ensembles aims at accomplishing the best performance as regards two aforementioned evaluating metrics. In the settings of limited computational resources or concentrating on fast inferences, opting for STSB-DeBERTa-V3-large can be considered a more appropriate choice as for its lightweightness yet achieving comparative results.

References

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15, Berlin, Heidelberg. Springer Berlin Heidelberg.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Model	Accuracy	Macro F1	Positive F1	Negative F1
STSB-RoBERTa-large	0.8951	0.8825	0.9209	0.8441
DeBERTaV3-large	0.8986	0.8836	0.9253	0.8419
STSB-DeBERTa-V3-large	0.9014	0.8881	0.9276	0.8496
Ensemble models (hard voting)	**0.9125**	**0.9003**	**0.9351**	**0.8655**

Table 1: Comparison of classification results from each model separately and ensemble models using hard voting.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. Language, Speech, and Communication. MIT Press, Cambridge, MA.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Steffen Herbold. 2023. Semantic similarity prediction is better than other semantic similarity measures.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2016. Conceptnet 5.5: An open multilingual graph of general knowledge. *CoRR*, abs/1612.03975.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2020. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models.