Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2021"

Moscow, June 16–19, 2021

Russian News Similarity Detection with SBERT: pre-training and fine-tuning

Vatolin A. S.
SberBank / Moscow, Russia
vatolinalex@gmail.com

Smirnova E. Y.
SberBank / Moscow, Russia
kate15511@gmail.com

Shkarin S. S.
SberBank / Moscow, Russia
kouki.sergey@gmail.com

Abstract

Computation of text similarity is one of the most challenging tasks in NLP as it implies understanding of semantics beyond the meaning of individual words (tokens). Due to the lack of labelled data this task is often accomplished by means of unsupervised methods such as clustering. Within the DE2021: "Russian News Clustering and Headline Selection" we propose a method of building robust text embeddings based on Sentence Transformers architecture, pretrained on a large dataset of in-domain data and then fine-tuned on a small dataset of paraphrases leveraging GlobalMultiheadPooling.

Keywords: text similarity; clustering; sentence embedding; BERT; Sentence Transformers; paraphrase detection
DOI: 10.28995/2075-7182-2021-20-XX-XX

Поиск похожих новостей на русском языке с помощью SBERT: предварительное обучение и тонкая настройка

Ватолин А. С.
СберБанк / Москва, Россия
vatolinalex@gmail.com

Смирнова Е. Ю.
СберБанк / Москва, Россия
kate15511@gmail.com

Шкарин С. С.
СберБанк / Москва, Россия
kouki.sergey@gmail.com

Аннотация

Вычисление схожести текстов - одна из самых сложных задач в сфере автоматической обработки естественного языка, поскольку подразумевает понимание семантики всего текста, что выходит за рамки значений отдельных слов (токенов). Из-за недостатка размеченных данных эта задача часто решается методами обучения "без учителя", такими как кластеризация. В рамках соревнования DE2021: "Russian News Clustering and Headline Selection" мы предлагаем метод построения векторных представлений (эмбеддингов) текста на основе архитектуры Sentence Transformers, предварительно обученной на большом наборе данных заданного домена (новостные тексты), а затем тонко настроенных на небольшом наборе данных парафразов с использованием GlobalMultiheadPooling.

Ключевые слова: семантическая близость текстов; кластеризация; векторное представление текстов; BERT; Sentence Transformers; нахождение парафразов

1 Introduction

Computation of text similarity is a common, yet challenging task that plays an important role in a variety of Natural Language Processing (NLP) applications, such as search engines, plagiarism detectors, question answering systems, etc.

The main difficulty stems from the variability of human language, as the same meaning can be conveyed with different language units. As a result, we cannot rely on superficial resemblance of texts, such as sharing common words or expressions. Instead, we need to go beyond individual words, capture the semantic meaning of the texts and then evaluate this semantic similarity with some measurable score.

One method to create such a semantic similarity score between texts is based on weighted relations between words in linguistic resources, such as WordNet and Semantic nets (see [10], [12]). This approach has its constraints in the limited size and rigidity of manually constructed thesauri and ontologies.

Another approach is to create special vector representations, or embeddings, that can capture the semantics of a text. One of the successful solutions for creating such vectors for individual words was word2vec, introduced in 2013 [14]. The idea behind it is based on the famous assumption "you shall know a word by the company it keeps". Thus, the initial values of the embeddings are replaced during training in such a way that the words used in the same contexts (and hence similar) are located closer in the vector space.

As an extension of this idea, the doc2vec algorithm was developed in 2014 [9], which added a vector with the document (e.g., text) ID to the word2vec model. This made it possible to obtain embeddings of sentences and texts, which inherit the main feature of word2vec: similar texts are located closer in the vector space. Thus, the similarity between texts can be calculated as the distance between their embeddings (for example, cosine, Euclidean, Manhattan distance, etc.).

Recent advances in deep learning have allowed for the creation of text embeddings that are more powerful in representing the semantics. The Skip-Thought method [8] trains a model to predict surrounding sentences based on the encoder-decoder architecture. InferSent [5] trains a bi-directional Long-Short Term Memory (LSTM) model on the labelled Natural Language Inference (NLI) data, achieving better performance. ELMo [11] introduces contextualized word embeddings, created by training a bi-directional LSTM. Taking the weighted mean of ELMo word embeddings to get a text embedding results in great performance on sentence similarity benchmarks as shown in [11].

Finally, the current state-of-the-art method is to pre-train Transformer models on language modelling (LM) and then fine-tune it for downstream tasks. The best results are achieved by Universal Sentence Encoder models [3], fine-tuned on the NLI task, and Sentence Transformers [15], fine-tuned on the NLI and Semantic text similarity (STS) benchmarks.

There are several pre-trained Sentence Transformer models, as well as the RuBERT model by DeepPavlov, that can be exploited for calculating similarity of texts in Russian. However, to obtain the best results it is necessary to pre-train the transformer model on the in-domain data and then fine-tune the model for a specific task.

2 Shared task description

This paper is the result of the research done for the first track of the Dialogue-2021 shared task 'Russian News Clustering and Headline Generation' [6]. The main purpose of this track is to investigate different approaches to clustering similar news texts in Russian.

Data for this competition was sourced from the Telegram Data Clustering Contest and annotated via Yandex.Toloka. The main dataset represents couples of sentences marked according to whether they are related to the same story (OK) or not (BAD). The same story is considered to be a text in different media aka newspapers, web sources, etc. about a certain event that happened with certain people and at a certain time. Such dataset design essentially turns the clustering task into paraphrase detection. Though a common solution would be to build a classifier on top of BERT model (see Cross-Encoder in [7]), methods such as Bi-Encoders that produce text embeddings instead of classification label were considered more preferable (but not required) by the contest organizers as they are more consistent with the initial idea of the clustering task.

The participants' results were evaluated using f1-score, calculated for positive (OK) class.

3 System description

Transformer-based models, such as BERT, show start-of-the-art performance on most NLP tasks. In the original paper [7], the BERT model uses the cross-encoder approach: two texts are fed to the input and the target value is predicted from them. However, this approach is not optimal when predicting the target value for all text pairs from a large dataset. The complexity can be estimated as $O(n(n-1)/2)$.

Another approach is bi-encoder: the model projects the text into a dense vector space and then uses similarity metrics such as cosine similarity or Manhattan/Euclidean distance to calculate the semantic similarity between the two texts.

What is more, the latter approach is more consistent with the clustering task of the DE 2021 competition. Thus, the simplest way to undertaking the DE2021 shared task is to use multi-language pre-trained bi-encoder model, fine-tuned for paraphrase identification or semantic similarity. The advantage of this method is the speed of work, there is no need to train the model, all it takes to make inference. The disadvantage of this approach is low quality.

The model can be improved in 4 ways: data preparation, base transformer model, pooling method, and loss function.

3.1 MLM pretrain

The improvement over a base BERT model was a pre-training model with masked language modelling (MLM). This is a common way of improving these models for specific tasks. Generally, many researches found that MLM pre-training models are more stable, train faster, and more often reach higher scores than training from the original pretrained model. You can also use other transformer model architectures, such as RoBERTa or XLM-R, but not all models have Russian language support.

As a pooling method, the authors of the SBERT model [15] suggest using averaging over the token vectors of the last layer of the model. They also suggest taking the CLS token vector and max-over-time for output vectors.

3.2 Weighted Layer Pooling

In the article [13], the authors claim that the information useful for solving the problem is contained not only in the last layer, but also in the middle layers. To leverage this information the following operation can be performed: taking mean, max, CLS pooling for the outputs of each layer (output), and then averaging the resulting vectors with the trainable weights w.

$$u = softmax(w) * output$$

3.3 Global Multihead Pooling

In the article [4], to convert token vectors to a fixed-length vector, it is proposed to use the sum of the vectors weighted with attention weights. The authors use this method to aggregate the outputs of the Bi-LSTM model. In our article, we propose to adapt this method to transformers.

$$A = softmax(W_2 ReLU(W_1 output^T + b_1)^T + b_2),$$

where output is a matrix of token vectors, the output of the last layer of the BERT model, W_1, W_2 are trainable weight matrices, b_1, b_2 are bias vectors, and A is a vector of word weights. Because of softmax function, the weights are always non-negative and sum up to 1.

The text vector can be calculated using the following formula:

$$u = A \odot output$$

Usually, the attention weights focus on a specific part of the text, so we can extend the pooling method to a multi-head way:

$$u_i = A_i \odot output$$
$$u = [u_1, u_2, \dots, u_n],$$

where n is the number of heads. Since the text vectors from each head are concatenated, the dimension of the output vector increases significantly with a larger n. So, we used $n \leq 5$.

3.4 Contrastive loss

Contrastive loss takes the output of the network for a positive example and calculates its distance to an example of the same class and contrasts that with the distance to negative examples. To put it another way, the loss is low if positive samples are encoded to similar representations and negative examples are encoded to different representations.

$$loss = y\,d^2 + (1-y)(margin - d, 0)^2$$

3.5 Online contrastive loss

Contrastive loss modification, where loss computed only for hard positive and hard negative pairs.

4 Experimental setup

4.1 Data

The initial data for the competition were presented in the format of HTML files containing articles in various languages. Each article consists of a title, text, and additional metadata.

After preprocessing, including the removal of non-news and non-Russian texts, three datasets were generated - 20,000 labeled news pairs in one day for training, 40,000 unlabeled news pairs in 2 days for testing, and about 1,200,000 raw news texts for pre-training. The average text length is about 1500 characters, including spaces.

4.2 Experiments

Multilingual transfer learning

As a base model, we used distiluse-base-multilingual-cased-v2 from the sentence transformers library.

Transformer model

We used the Deeppavlov RuBERT base cased model as the optimal model by quality and training speed [16]. We also used the sbert_large_nlu_ru model [1], based on the Russian BERT Large model, but it showed worse quality compared to RuBERT. When training, the length of the texts was limited to 250 tokens, increasing the maximum length reduced the quality of the model. We think that this is due to a decrease in the size of the batch with a longer sequence length, and therefore a decrease in the stability of the gradients. To optimize the parameters, we used AdamW with learning rate 2e-5 and 10% warmup steps.

Pooling method

To aggregate token vectors into a text vector, we tried averaging word vectors, and also used Weighted Layer pooling and Global Multihead attention approaches. The Global Multihead attention layer is best in terms of quality, as it allows to increase the weight of important words and not take into account the words of the general vocabulary.

Loss function

For our better model, we used Online contrastive loss with cosine proximity functions and margin 0.5. On the final epochs, for most batches, the loss was 0. To avoid this problem, we tried to increase the margin to 0.8, but this did not lead to an improvement in quality.

5 Results

We present the results of model evaluation in Table 1. We use the f1 metric to measure quality. We split the train dataset in 70%, 15% and 15% for train, validation and test set accordingly.

Model	val f1	test f1	public f1	private f1
Distiluse v1*	0.8955	0.8846	0.8733	0.8816
Distiluse v2*	0.8974	0.8845	0.8840	0.8763
Deeppavlov RuBERT	0.9538	0.9455	0.9331	0.9343
Deeppavlov RuBERT, Global Multihead pooling	0.9619	0.9601	0.9467	0.9438
sbert_large_nlu_ru	0.9498	0.9415	0.9108	-
Deeppavlov RuBERT pretrained	0.9556	0.9511	0.9435	0.9387
Deeppavlov RuBERT pretrained, cross-encoder	0.9668	0.9656	0.9516	0.9545
Deeppavlov RuBERT pretrained, Global Multihead pooling	**0.9631**	**0.9617**	**0.9547**	**0.9548**

Table 1: models evaluation results

* scores without fine-tuning on news dataset.
All models are trained with mean pooling, unless GlobalMultihead pooling is specified in the name. Our target approach reaches the results of cross-encoder.

6 Conclusion and future work

Sentence similarity calculation is a common task, crucial for many NLP applications. Models based on one of the most cutting-edge architectures - Transformer - show state-of-the-art results in many down-stream tasks, including paraphrase detection. Pretrained multilingual Transformer models show decent quality without any additional training. However, the best scores are achieved by pre-training on in-domain data and follow-up fine-tuning for a specific task (paraphrase detection). Another improvement suggested in this article is the use of GlobalMultihead pooling.

As for future work, we should try SBERT-WK [2] model and in particular WK Pooling. The SBERT-WK model shows a higher quality compared to the SBERT model. The SBERT-WK model uses qr matrix decomposition, which in the Pytorch implementation is very slow on the GPU at the moment. Because of this, model training takes a significant amount of time.

Acknowledgements

We thank the organizers of Shared Task Ilya Gusev and Ivan Smurov for providing the data and holding the competition. We are also grateful to members of the DeepPavlov team for their pretrained BERT models. We thank the anonymous reviewers whose valuable comments helped to improve the paper.

References

[1] BERT large model (uncased) for Sentence Embeddings in Russian language, Access mode: https://huggingface.co/sberbank-ai/sbert_large_nlu_ru
[2] Bin Wang, Jay Kuo C.-C.: SBERT-WK: A Sentence Embedding Method by Dissecting BERT-based Word Models. – 2020. – Vol. arXiv:2002.06652. – Access mode: https://arxiv.org/abs/2002.06652
[3] Cer D. et al.: Universal Sentence Encoder. – 2018. – Vol. arXiv:1803.11175. – Access mode: https://arxiv.org/abs/1803.11175
[4] Chen Q., Ling Z.H., Zhu X.: Enhancing Sentence Embedding with Generalized Pooling. – 2018. – Vol. arXiv:1806.09828. – Access mode: https://arxiv.org/abs/1806.09828
[5] Conneau A. et al.: Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. – 2017. – Vol. arXiv:1705.02364. – Access mode: https://arxiv.org/abs/1705.02364.
[6] DE2021: Russian News Clustering and Headline Selection – Clustering, Access mode: https://competitions.codalab.org/competitions/28830
[7] Devlin J., Chang M.W., Lee K., Toutanova K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. – 2019. – Vol. arXiv:1810.04805. – Access mode: https://arxiv.org/abs/1810.04805.
[8] Kiros R et al.: Skip-Thought Vectors. – 2015. – Vol. arXiv:1506.06726. – Access mode: https://arxiv.org/abs/1506.06726
[9] Le Q.V., Mikolov T.: Distributed Representations of Sentences and Documents. – 2014. – Vol. arXiv:1405.4053. – Access mode: https://arxiv.org/abs/1405.4053.
[10] Li Y. et al. (2006), Sentence Similarity Based on Semantic Nets and Corpus Statistics. IEEE Transactions on Knowledge and Data Engineering 18(8), 1138–1150.
[11] Matthew E. Peters et al.: Deep contextualized word representations. – 2018. – Vol. arXiv:1802.05365. – Access mode: https://arxiv.org/abs/1802.05365
[12] Mihalcea R., Corley C., Strapparava C.: Corpus-based and Knowledge-based Measures of Text Semantic Similarity. – 2006. – Access mode: https://www.aaai.org/home.html
[13] Mikhailov V., Taktasheva E., Sigdel E., Artemova E.: RuSentEval: Linguistic Source, Encoder Force! – 2021. – Vol. arXiv:2103.00573. – Access mode: https://arxiv.org/abs/2103.00573.
[14] Mikolov T.: Efficient Estimation of Word Representations in Vector Space. – 2013. – Vol. arXiv:1301.3781. – Access mode: https://arxiv.org/abs/1301.3781.
[15] Reimers N, Gurevych I.: Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. – 2019. – Vol. arXiv:1908.10084. – Access mode: https://arxiv.org/abs/1908.10084.
[16] Shavrina T. et al.: RussianSuperGLUE: A Russian Language Understanding Evaluation Benchmark. – 2020. – Access mode: https://www.aclweb.org/anthology/2020.emnlp-main.381/