# Patent Phrase to Phrase Matching Based on Bert

Zhan Chen[1, *]

[1]Department of statistics, Capital University of Economics and Business, Beijing, China

*Corresponding author: b20160901233@stu.ccsu.edu.cn

**Abstract.** Due to a large US patent archive, it is necessary to introduce a similarity matching system to judge if an invention has been granted a patent so that people just focus on high similarity patent items and ignore low similarity ones. First, the large-scale corpus is pre-trained using the Bert language model to acquire the semantic characteristics of general language. The pre-training Bert language model is used to tune the text data set of patent phrases to acquire the semantic features of the certain text and the specific meaning of the keywords to match similarity, given certain parameters according to the task, such as MSE as loss function and certain number as learning rate and so on. The validation results are good whether it is according to MSE loss or the Pearson correlation coefficient. Finally, applying this model to the test dataset and the results show that the Pearson correlation of all the variables is significant, and the model fits well.

**Keywords:** Similarity; Bert; Pearson correlation.

## 1. Introduction

Text scoring refers to the use of computers for manual problems. There are machine learning algorithms for text scoring by statistically analyzing the number of words, the number of spelling errors, the average number of words, the number of sentences, the term frequency-inverse document frequency (TF-IDF) and other characteristics of the text including project essay grader (PEG) [1]. TF-IDF adopts unsupervised learning, taking into account the two attributes of word frequency and freshness. It can filter certain common words and keep important words that can give more information. In practical applications such as search engines, this technology is the main means of information retrieval. However, it is not comprehensive to use word frequency to measure the importance of a word and this calculation cannot reflect the positional relationship; At the same time, it relies heavily on the level of word segmentation. Its effect is good, but it cannot extract the semantic features of the text. Latent semantic analysis (LSA) can not only obtain the semantic features of the texts, but also solve the synonym problem [2]. Its starting point is that there is some connection between words in the corpus, that is, there is some potential semantic relationship, which is hidden in the context pattern of words in the text. Therefore, it needs to analyze a large number of corpuses to find this potential semantic relationship. LSA does not need to determine the semantic code, only depends on the relationship of things in the text context, and uses semantic relations to represent the text, simplifying the purpose of the text vector. The main idea of this method is to decompose the matrix of document-term into the two independent matrices. The two matrices are the document-topic matrix and the topic-term matrix. However, it consumes a large number of computational resources and cannot obtain the order information of the text.

Recently, deep neural networks (DNN) have been widely used in text analysis that can extract deep semantic information and context-related information [3]. DNN is mainly divided into convolutional neural networks (CNN) and circulating neural networks (RNN). RNN have a strong ability to capture short context dependencies, but the longer the distance between contexts, the weaker the ability of RNN. Long short-term memory (LSTM) is better in this aspect [4]. In view of the shortcomings of the traditional natural language pre-training technology, the neural network natural language pre-training technology has taken improvement measures, mainly considering the context relationship between word orders into the actual corpus. Qiu et al. expounded the pre training and model related technologies from four aspects: the context-related word order, language model structure, task type and technology application scope [5].

Bidirectional encoder representations from transformers (Bert) are a deep-learning-based language representation model proposed by Google. It works best in 11 different natural language processing test assignments, including categorization tasks [6-8]. The tasks based on the Bert mainly include two processes: pre-training and fine-tuning. The former uses large-scale unlabeled text corpus for self-supervised training to effectively learn text features and text vector representation, thus forming the pre-training Bert model; in the pre-tuning, the network parameters are used as the initial model. When it comes to specific assignments, a labelled data set is input to complete fitting of the model. And a deep learning model available is obtained to complete specific NLP tasks. Compared with the traditional RNN and short-term memory network used for NLP tasks, the transformer has more powerful text coding ability and can also make more efficient use of high-performance devices such as graphics processing unit (GPU).

Bert, in this paper, aims to get a score for a phrase-based on extracted features from the raw data. For these features, Jawahar studied the types of language features captured by the 12 coding layers constituting Bert. The bottom layers of Bert mainly learn the shallow language features of the input text, the middle layers of Bert mainly learn the syntactic information features, and the high layers of Bert mainly learn the semantic information features of the text [9]. The above research shows that Bert can not only capture the shallow features in the text but also effectively learn the deep semantic information. With respect to data, the Kaggle competition has a publicly available dataset. Next, the second step is to split the dataset into two sub-datasets for training and testing. The training data set is a labeled data set used to create a model based on the selected characteristics of the data set. The trained model is then applied to the test data set to assign them the sentence score.

This paper takes advantage of the Bert model to establish a similarity system to judge the similarity between two patent phrases. In section 2, the method adopted is described from the transformer to Bert. In section 3, as to the patent data from Kaggle, some parts of the result are shown on the accuracy of the model applied to the certain task, including training in the training dataset, picking the best model and then inferring. In section 4, the result is simply analyzed and the relevant discussion is given.

## 2. Method

### 2.1 Transformer

Bert makes use of the transformer architecture, so the transformer is necessary to be described roughly [10].

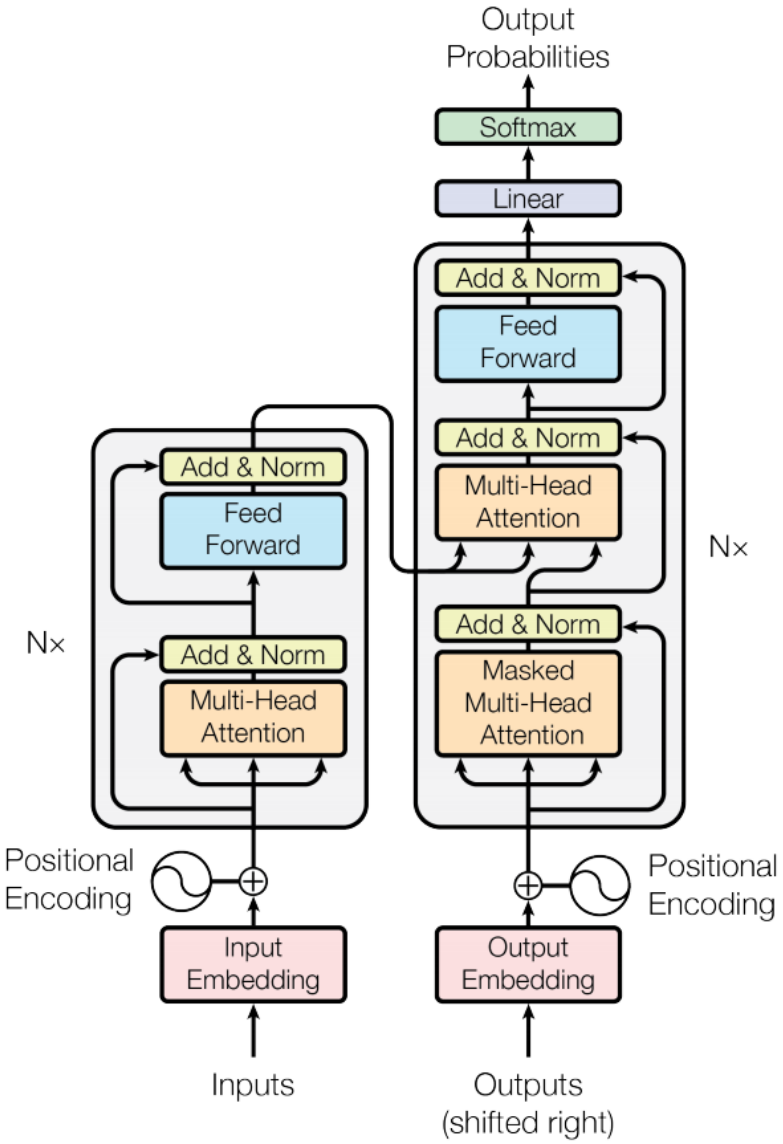


**Fig. 1** The basic architecture of the transformer

As Fig. 1 shows, the transformer consists of encoders shown on the left of Fig. 1 and decoders shown on the right of Fig. 1. Bert mainly adopts the encoders. Encoders are many encoder blocks, each being a standard structure. The encoder is a type of neural network that helps people deal with a range of issues such as classification and regression. The main difference between encoders and other neural network is that encoders have an attention mechanism. For each encoder, it is reflected in Multi-Head Attention which comprises multiple self-attention.

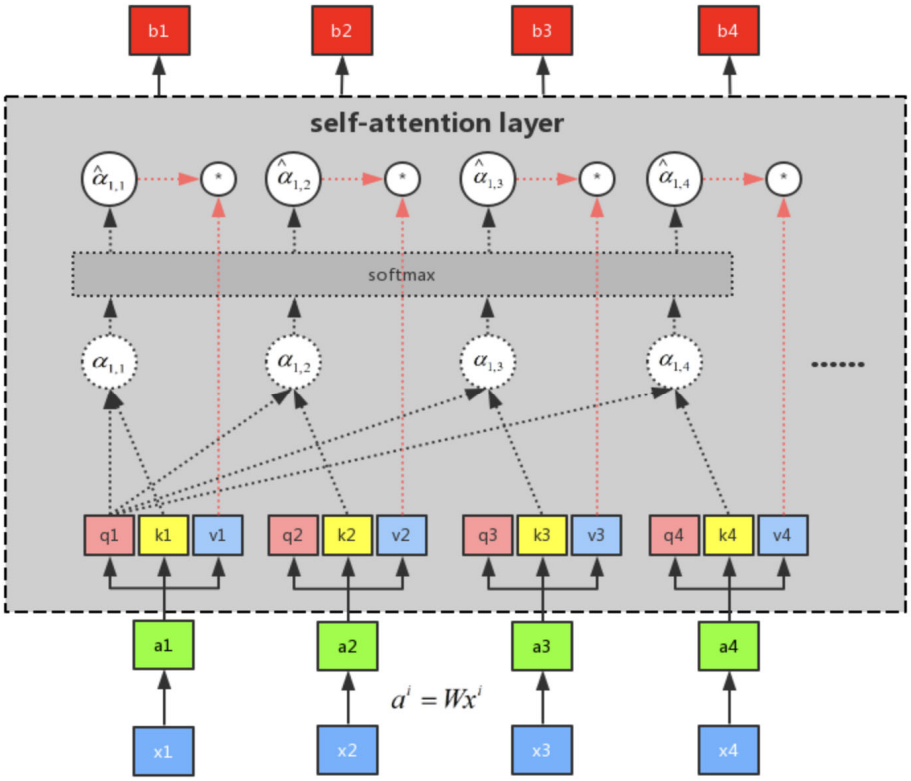


**Fig. 2** The construction of one self-attention layer

As Fig. 2 shows, for each encoder, the input is a sequence of vectors $\{x_i\}$, and the output is a sequence of vectors for each input vectors. In this structure, the parameters are $q_i, k_i, v_i$ that people train.

This construction considers the relationship between one word and the context (every word). Take $x_1$ for example, if people enquiry the coefficient between $x_1$ and every word, firstly we compute $q_1 = W^Q a_1, k_i = W^K a_i, v_i = W^V a_i$, then

$$a_{1,i} = \frac{q_1 k_i}{\sqrt{d_{qk}}} \tag{1}$$

$$\hat{a}_{1,i} = soft\max(a_{1,i}) = \frac{e^{a_{1,i}}}{\sum_i e^{a_{1,i}}} \tag{2}$$

then the output $b_1$ is computed as below

$$b_1 = \sum_i \hat{a}_{1,i} v_i \tag{3}$$

According to (3), one could get the output $b_2, b_3 ..., b_n$

In fact, this is one self-attention structure and the Multi-Head Attention contains many self-attention structures. The actual model will use multiple attention, which is called multi-head attention. The performance of the attention layer is enhanced through multi-head attention in two aspects. The first is that it extends the model to focus on different locations. In the above example, although each code is more or less embodied in $b_1$, it may be dominated by the actual word itself. If people translate a sentence, such as “animal did not cross the street because it was too tired”, they will want to know which it refers to. At this time, the multi-head attention mechanism of the model may play a role. The second is that it provides multiple “representation subspaces”. For the multi-head attention mechanism, people have multiple sets of weight matrices (eight attention heads are used in the

transformer. Therefore, there are eight matrix sets for each encoder and decoder). These matrices are calculated independently and finally integrated, which can prevent overfitting.

### 2.2 Bert

Bert is a Natural Language Processing (NLP) neural network, using Transformer as a framework. The model includes pre-training and fine-tuning.

#### 2.2.1 Pre-training

Pre-training in NLP is a breakthrough work, although it has been very mature and widely used in Computer Vision (CV). Bert builds two assignments, namely masked language model (MLM) and next sentence prediction (NSP).

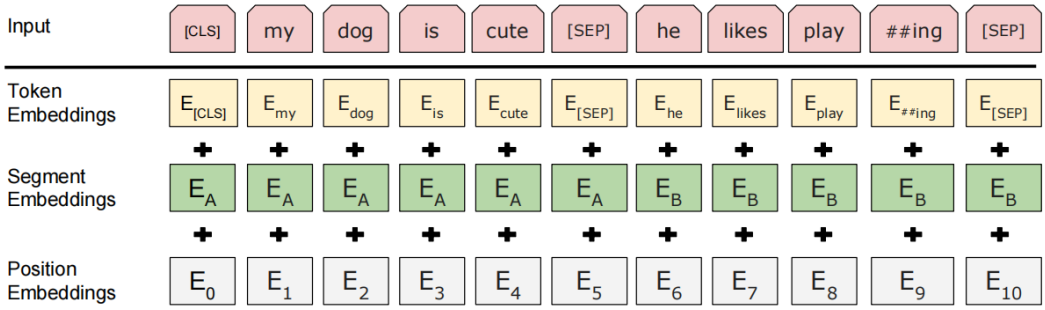


**Fig. 3** the input representation of Bert

As is shown in Fig. 3, “CLS” indicates the classification and “SEP” represents a clause symbol, which is used to disconnect two sentences. The input is the sum of token embeddings, segment embeddings and position embeddings. They represent token representation, sentence representation and position representation respectively.

MLM is used to pre-train transformers to generate a deep bidirectional language representation. In Bert, 15 percent of word-piece tokens would be randomly masked. The task of NSP is to judge if sentence A is the next sentence B, this which is saved in the “cls” symbol in Fig. 1.

#### 2.2.2 Fine-tuning

The principle of fine-tuning is to utilize the known network structure and network parameters, alter the output layer to a certain layer, and fine-tune the parameters of several layers before the last layer. In this way, the powerful ability to generalize the deep neural network is used efficiently, and complex model design and time-consuming training are avoided. Thus, fine tuning is a relatively appropriate choice when there is not enough data. For different downstream tasks, the structure of Bert may have different slight changes. Normally, the only need is to add an output layer based on Bert to finish the fine-tuning.

In my task, it can be considered a regression task, so a full connect layer is added and the output is one dimension.

## 3. Results and Discussion

### 3.1 Data Description

The data is about patents. When judging if a new invention has been described before, determining the semantic similarity between phrases is important, because it can allow people to narrow down the search range. People can judge if the invention has been granted a patent just on the high similarity patent items.

The number of the training data is 36473. The data comes from Kaggle. The training data contains features: the training set, containing phrases, contexts, and their similarity scores. The number of test

data is 36 used to test the performance of the model to be trained. The test data contains features: the training set, containing phrases and contexts without the score.

Part of the training data is shown in Table 1. Anchor is the first phrase and target is the second phrase. The score is the similarity level between two phrases in the context. The higher the score (ranging from 0 to 1), the higher the similarity level.

**Table 1**. Description of part of training data

| ID | Anchor | Target | Context | Score |
|---|---|---|---|---|
| $x_1$ | abatement | abatement of pollution | PERFORMING OPERATIONS; TRANSPORTING. PHYSICAL OR CHEMICAL PROCESSES OR APPARATUS IN GENERAL | 0.5 |
| $x_2$ | abnormal position | position shown | TEXTILES; PAPER. NATURAL OR MAN-MADE THREADS OR FIBRES; SPINNING | 0.25 |

### 3.2 Data Description Data Processing and Results

In fact, data processing is done a little. The data is almost clear. What this paper just does is to remove some data containing missing values or strange symbols. Then the data is fed into the model and the model helps segment the sentence through WordpieceTokenizer. The tokenizer does not segment the sentence just according to space. For some unusual words, the tokenizer is to divide these words because some words occur rarely. If one store these rare words into a dictionary, the dictionary is big and redundant. In order to reduce the space of the dictionary, WordpieceTokenizer is to divide a word occurring a little into several sub-words in order from left to right, and each sub-word is as long as possible. According to the source code, this method is called the greedy longest match first algorithm.

As for context features in data, this paper connects different sentences and separates them with "SEP" such as "ack SEP indications of cystoscopy SEP physics horology".

### 3.3 Model and results

In this model, some important parameters should be illustrated. Firstly, the max input length is set to 140, which means the input comprises 140 words. If the number of the words of the input is over 140, the machine will cut the sentence and let the length of the input be 140; and if the number of words of the input is less than 140, the machine will supplement the corresponding number of zero. These "zeros" will not make a difference in the machine.

Secondly, the batch size is eight, which means only eight samples of data are used for one training. This paper uses GPU to run the model. However, due to the limitation of the memory of GPU in my company, this paper just sets the batch size to 8. And the learning rate is 2e-5 which is a flexible choice according to experience from others.

Thirdly, this paper adopts 3-fold cross-validation to choose the best model. That means splitting the training data into 3 equal parts, so there are 3 models. For each model, it trains 2 parts of the data and the rest of the data functions as evaluation data to test the accuracy of the model.

In addition, the evaluation method to judge whether the result is good or not is the Pearson correlation coefficient. The formula is

$$r = \frac{\sum (G_i - \overline{G})(K_i - \overline{K})}{\sqrt{\sum (G_i - \overline{G})^2} \sqrt{\sum (K_i - \overline{K})^2}} \tag{4}$$

where $G, K$ are two sequences. Thus, it describes the degree of linear correlation between two sequences.

**Table 2**. The result of cross validation

| Fold | Training loss | Validation loss | Pearson |
|---|---|---|---|
| 1 | 0.023 | 0.022 | 0.832 |
| 2 | 0.024 | 0.026 | 0.819 |
| 3 | 0.023 | 0.020 | 0.839 |

As Table 2 shows, the result of the 3rd fold is the best whether it is according to MSE loss or Pearson correlation coefficient. For example, as to id $x_{77}$, the real score is 0.75, which means the similarity of the given two phrases is relatively high, and the estimated score is 0.76, which means at least for the piece of data, the fitting result is good. Applying the 3rd fold model to the test data, part of the result got is shown in Table 3 below.

**Table 3**. Part of the result of test data

| ID | Score | ID | Score |
|---|---|---|---|
| $t_{15}$ | 0.762 | $t_{18}$ | 0.005 |
| $t_{17}$ | 0.261 | $t_{19}$ | 0.517 |
| $t_{28}$ | 0.001 | $t_{20}$ | 0.399 |

Finally, the Pearson correlation coefficient in the test data is 0.829, so the scores of the real test data set and estimated scores are significantly high relevant, which means if two patent phrases are highly similar, the model will give a high score.

## 4. Conclusion

In this paper, the deep learning method is utilized in the phrase similarity matching task. And a similarity scoring model based on Bert is constructed. Then the model is optimized by fine-tuning strategy. The results of the experiment show that in this similarity scoring task, especially in the case of limited dataset size, the Pearson correlation coefficient on the test set reaches 0.829. At the same time, the best validation result of the model is 0.839. That means the model is robust, because the Pearson correlation coefficient in the test data set and the Pearson correlation coefficient in the validation data set are very close. This model can be applied to other fields for text scoring.

This paper describes the research on the text similarity scoring algorithm based on Bert. For the text data of specific content such as patents, on the basis of collecting relevant data sets, the semantic representation of sentences is obtained through the Bert language model, and then the data is scored by the regression layer. The results display that this proposed method's effect is good on the patent phrase similarity task. On the other hand, the problem of the algorithm in this paper is that it requires high-quality data sets, needs to collect enough data sets in specific fields, and needs to add correct labels to the data sets, thus increasing the use cost. Patent similarity matching is only one aspect of extracting the value of a large amount of data. In future work, we can also conduct value judgment, entity recognition, automatic generation and other research according to the text content. At a time when information technology is an integral part of people's lives, the value of data is worth exploring and studying.

## References

[1] Page E.B. The imminence of grading essays by computer. Phi Delta Kappan, 1966, 48: 238-243.

[2] Deerwester S, Ddumais S T, Furnas G W, et al. Indexing by latent semantic analysis. Journal of the American Society for Information Science, 1990, 41(6): 391-407.

[3] Tang D.Y. Sentiment-Specific Representation Learning for Document-Level Sentiment Analysis. ACM, 2015:447-452.

[4] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735-1780.

[5] Qiu X P, Sun T X, Xu Y G, et al. Pre-trained models for natural language processing: a survey. Science China: Technological Sciences, 2020(10): 1872-1897.

[6] Liu Huan, Zhang Zhixiog, Wang Yufei. A review on main optimization methods of BERT. Data Analysis and Knowledge Discovery, 2021, 5(1): 3-15.

[7] Fang Xiaodong, Liu Changhui, Wang Liya, et al. Chinese text classification based on BERT's composite network model. Journal of Wuhan Institute of Technology, 2020, 42(6): 688-692.

[8] Duan Dandan, Tang Jiashan, Wen Yong, et al. Chinese short text classification algorithm based on BERT model. Computer Engineering, 2021, 47(1): 79-86.

[9] Jawahar G, Sagot B, Seddah D. What does BERT learn about the structure of language? //Pro-ceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence: [s.n.], 2019.

[10] Vaswani A, Shazeer N, Parmar N, et al. Attention Is All You Need[C]// arXiv, 2017.