Hindawi
Scientific Programming
Volume 2021, Article ID 5916748, 11 pages
https://doi.org/10.1155/2021/5916748

Research Article

Parallel Cleaning Algorithm for Similar Duplicate Chinese Data Based on BERT

Biqiu Li, Jiabin Wang, and Xueli Liu

College of Engineering, Huaqiao University, Quanzhou 362000, Fujian, China

Correspondence should be addressed to Jiabin Wang; fatwang@hqu.edu.cn

Received 13 May 2021; Revised 24 October 2021; Accepted 9 November 2021; Published 23 December 2021

Academic Editor: Antonio J. Peña

Data is an important source of knowledge discovery, but the existence of similar duplicate data not only increases the redundancy of the database but also affects the subsequent data mining work. Cleaning similar duplicate data is helpful to improve work efficiency. Based on the complexity of the Chinese language and the bottleneck of the single machine system to large-scale data computing performance, this paper proposes a Chinese data cleaning method that combines the BERT model and a k-means clustering algorithm and gives a parallel implementation scheme of the algorithm. In the process of text to vector, the position vector is introduced to obtain the context features of words, and the vector is dynamically adjusted according to the semantics so that the polysemous words can obtain different vector representations in different contexts. At the same time, the parallel implementation of the process is designed based on Hadoop. After that, k-means clustering algorithm is used to cluster similar duplicate data to achieve the purpose of cleaning. Experimental results on a variety of data sets show that the parallel cleaning algorithm proposed in this paper not only has good speedup and scalability but also improves the precision and recall of similar duplicate data cleaning, which will be of great significance for subsequent data mining.

1. Introduction

Network resources are a huge and constantly updated ocean of information and an important channel for people to obtain information and knowledge. A large number of similar and repeated pages cause people to waste a lot of time when screening effective information. This phenomenon is more common in Chinese expression. Chinese expression has a rich diversity, so the similar repetition of the Chinese text is particularly obvious in synonyms and polysemy. With the development of information technology, the data growth mode has changed; the amount of data is gradually huge; and centralized computing is difficult to deal with massive data [1].

The existing data cleaning algorithms cannot meet the actual needs in cleaning efficiency and accuracy. The cleaning of Chinese similar duplicate data includes the cleaning algorithm based on literal similarity and the cleaning algorithm based on semantic similarity. Literal similarity cannot distinguish data with the same semantics but with the different font, so it is difficult to be applied to the processing of Chinese data. The existing algorithms based on semantic similarity cannot effectively clean out all similar duplicate records because of the loss of important information in the vectorization process. Moreover, many scholars use the idea of ensemble learning [2, 3] to combine multiple classifiers so that even if one weak classifier gets the wrong prediction, other weak classifiers can correct the error. This scheme has achieved good results in medical text classification, but its complexity is relatively high.

This paper obtains the text vector through the parallel design of the BERT language model, then calculates the distance between the texts, and cleans out the similar duplicate data through k-means clustering. The main tasks are as follows:

(i) Aiming at the phenomenon of synonyms and polysemy in Chinese, the BERT model is used to reduce the loss of original semantic information in the process of text to vector. At the same time, the

distributed computing of the process is designed based on a big data platform.

(ii) The idea of clustering is used to realize parallel cleaning of similar duplicate data. In this process, cosine similarity algorithm, canopy algorithm, and k-means clustering algorithm provided by Mahout algorithm library are used to detect semantically similar duplicate data and cluster them.

(iii) This paper analyzes the cleaning effect of this algorithm and the traditional algorithm from three aspects of precision, recall, and F1-score and illustrates the advantages of parallel computing from the aspects of speedup and scalability.

The organizational structure of this paper is as follows: Section 2 describes the relevant literature survey. The theoretical basis involved in this paper is introduced in Section 3. Section 4 introduces the parallel cleaning process of similar duplicate Chinese data in detail. The experimental results are analyzed in Section 5, and Section 6 gives the conclusion of this paper.

2. Related Work

In terms of data cleaning, the United States is the first country to start systematic research and the country with the fastest update of research results. Although the research time is long, most of the research objects are English data. For example, the sorted-neighborhood method (SNM) [4] selects one or several attributes or eigenvalues as keywords according to the characteristics of the data set, sorts the data set, places the possibly similar data in the adjacent position, and then sets a window size to compare the data similarity in the window. This method has a strong dependence on keyword selection and window size setting, and the cleaning result is unstable. Multipass sorted neighborhood (MPN) [5] improves the above algorithm by selecting different keywords for multipass sorting each time to avoid mistakes caused by improper selection at one time, but it also increases the cleaning time and reduces the cleaning efficiency. Priority queue method (PQM) [6] selects a record to represent a similar group of records and compares it with the records to be cleaned, which significantly reduces the cleaning time [7].

In terms of Chinese data cleaning, due to the great differences between Chinese and English languages and cultures in many aspects, the similarity detection method for English cannot be directly used to clean Chinese data. In addition to judging whether the font is consistent, we should also judge whether the meaning is the same. Accordingly, there are two similar duplicate detection methods for Chinese data.

In the research of literal similarity, there are many researches based on the editing distance method. The most basic algorithm is to take word granularity as the basic operation unit to obtain the cost of editing operation. Then synonyms are introduced to optimize the algorithm.

In the research of semantic similarity, the corresponding semantic similarity calculation method can be designed according to the characteristics of the research object. However, the research on data cleaning has made a breakthrough by adding knowledge and semantics to the framework of data cleaning. Semantic similarity and structural similarity can also be involved in the calculation at the same time so as to improve the accuracy of matching.

To sum up, the current research on similar duplicate data cleaning mainly includes: similarity detection with different distance measurement methods, elimination of similar duplicate data with different sorting methods, formulation of corresponding cleaning rules according to the characteristics of the data to be cleaned, and so on. Few people pay attention to the differences between Chinese and English and the preprocessing process of data cleaning, that is, text vectorization. The quality of text vectorization will directly affect the performance of subsequent cleaning, so this paper takes vectorization as the breakthrough point to deeply explore the impact of word position on word semantics so as to optimize the data cleaning process.

3. Theoretical Basis

This section introduces the k-means clustering algorithm for cleaning duplicate records, the BERT language model for transforming text data into vectors, MapReduce programming model, and Mahout algorithm library.

3.1. k-Means Clustering Algorithm. Clustering algorithm refers to the method of automatically dividing a pile of unlabeled data into several categories. This method should ensure that the same type of data has similar characteristics, so the idea of clustering can be used to clean similar duplicate text data. k-means algorithm [8] is one of the most widely used clustering methods in machine learning. Its working principle is to determine k clustering centers for a given instance, and divide each member into k clusters according to the distance between the members in the instance and the clustering center so that the points in the cluster are connected as closely as possible, and the distance between clusters is as large as possible. In order to determine the centroid coordinates and membership relationship of each instance, k-means first randomly initializes the centroid and then repeatedly performs the following two operations: calculates the distance between each member and the centroid and assigns it to the nearest centroid; the coordinates of the member instances of each centroid are used to recalculate the centroid coordinates of each cluster until the sum of squares of errors is the minimum or the specified number of iterations is reached.

For text clustering algorithm, in addition to k-means algorithm, krill herd algorithm and its improved algorithm are commonly used [9, 10]. The algorithm has the advantages of strong convergence, simple programming, and easy implementation. It also has the disadvantages of poor convergence accuracy, low computational efficiency, and less application fields. Although the algorithm has a good

effect on text clustering, considering the efficiency of massive data computing, the algorithm needs a more mature distributed design to better meet the needs of the big data market. From this point of view, the k-means algorithm is simpler than the algorithm idea; its parallel design is easy to implement and is integrated into the Mahout algorithm library. For enterprises processing data based on a distributed platform, k-means is more convenient and reliable.

3.2. BERT. BERT (bidirectional encoder representations from transformers) [11] is trained based on massive data. It has a strong generalization ability. Enterprises and individuals can use the model to make slight adjustments according to their own needs, making the construction model more simple and efficient. This model is mainly based on a bidirectional transformer encoder [12]. The bidirectional model of BERT is different from the traditional bidirectional model, which only considers the context information of the left and right sides of the sentence and also integrates the context information of the left and right sides that are commonly dependent in all layers. The structure is shown in Figure 1.In Figure 1, $w_1, w_2, \ldots, w_i$ represent the text input; $W_1, W_2, \ldots, W_i$ represent the vectorized representation of the text processed by the transformer. The process of obtaining vectors from text through BERT calculation on data cleaning is shown in Figure 2.

In Figure 2, input a text, [CLS] represents the beginning of the sentence, [SEP] represents the end of the sentence, and the words masked are used for model training.

BERT mainly uses the encoder of the transformer instead of its decoder. The transformer model is very different from the recurrent neural network. It does not need to perform multiple iterations. The calculation of the current word does not need to wait for the end of the previous word. It calculates all the words in the sentence at the same time. This parallel calculation greatly improves the calculation efficiency. In addition, the transformer considers the relationship between the position of the words in the language and the front and back words, which is identified by the position information. The position information is represented by the position encoding, and the linear transformation of sine and cosine functions is used to express the position relationship of each word in the natural language sequence. Words close to each other are closely related, while words far away are alienated. The location code is calculated as follows:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right),$$
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right), \quad (1)$$

where *pos* is the position of the word in the sentence, the value range is (0, *max*, *squence*, *length*), *i* is the dimension of the word vector, the value range is (0, *embe dd ing*, *di mension*), and $d_{mo\ de\ l}$ is the dimension of the total word

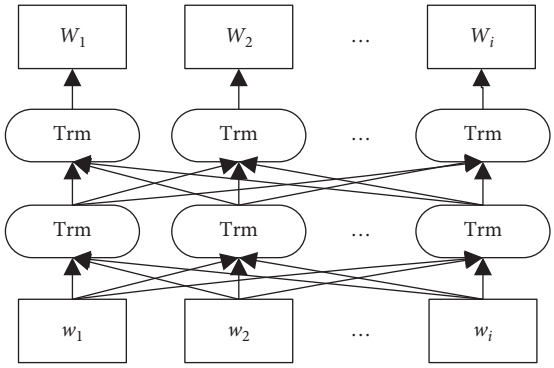

Figure 1: BERT model structure.

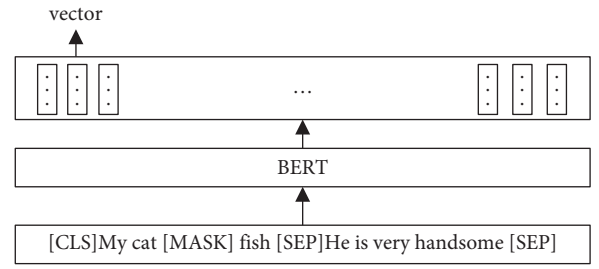

Figure 2: Text to vector by BERT.

vector. Each position will get a combination of cosine and sine functions with different periods so as to produce unique position information.

After the position information of the word is obtained, the corresponding vector expression is calculated.

3.2.1. Computing Word Vector and Position Coding. For sentence X, query the expression of each word in the word vector table, get the position embedding by the position-coding formula, and add the two, that is,

$$X = \text{EmbeddingLookup}(X) + \text{Positional Encoding},$$
$$X \in R^{\text{batchsize}*\text{seq.len}*\text{embed.dim}}. \quad (2)$$

The vector dimensions are as follows:

$X_{\text{embedding}}$ = [batch size, sequence length, embedding dimension], where *batchsize* is the number of texts, *sequence length* is the length of texts, and *embeddingdimension* is the dimension of word vector.

3.2.2. Self-Attention Mechanism Calculation. Multihead attention mechanism is used to extract multiple semantics so that each word in the sentence is related to all other words in the sentence. Each word vector contains the information of all word vectors in the current sentence, and the semantic information of each word in a different context is obtained. The structure is shown in Figure 3.

The linear mapping of $X_{embedding}$ forms three matrices Q, K, V among which

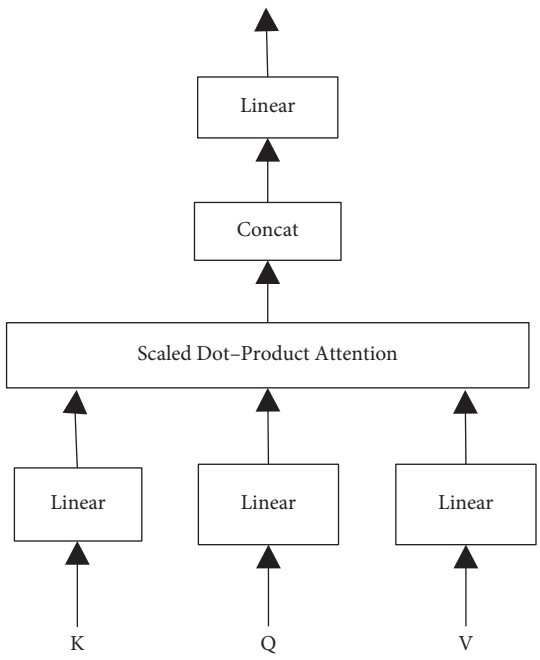

Figure 3: Multihead attention mechanism.

$$\begin{aligned} Q &= Linear\left(X_{embedding}\right) = X_{embedding}W_Q, \\ K &= Linear\left(X_{embedding}\right) = X_{embedding}W_K, \\ V &= Linear\left(X_{embedding}\right) = X_{embedding}W_V, \end{aligned} \tag{3}$$

where W_Q, W_K, and W_V are the corresponding weights of matrix Q, K, and V, respectively. Define a superparameter h (divisible by *embe dd ing di mension*), that is, the number of heads, and segment Q, K, V. After segmentation, the dimensions of matrix Q, K, and V are the same, and the lengths in each direction are batch size, h, sequence length, and embedding dimension/h, respectively For the convenience of subsequent calculation, transpose the three matrices after segmentation, and the length of Q^T, K^T, and V^T in each direction becomes batch size, h, sequence length, and embedding dimension/h accordingly. Next, we calculate the self-attention mechanism as follows:

$$\begin{aligned} X_{attention} &= Self\,Attention(Q, K, V), \\ &= soft\max\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \end{aligned} \tag{4}$$

$$soft\max\left(z_1, z_2, ..., z_N\right) = \frac{1}{\sum_1^N e^{z_i}}\left(e^{z_1}, e^{z_2}, ..., e^{z_N}\right). \tag{5}$$

In formula (4), QK^T is the attention matrix, and the dot product of the matrix, and its transposition can express the degree of association between each word in the sentence and all other words in the sentence. The dot product results in a d_k fold increase in variance. Therefore, we need to scale the attention matrix to the original gradient to become a standard normal distribution. Then, normalization is used to make the sum of attention weights of each word and all other words 1. In formula (5), N is the sentence length, and $(z_1, z_2, ...z_N)$ is the N-dimensional row vector. The results after normalization are more stable so that the balanced gradient can be obtained during backpropagation. After that, we use the result to set a weight for V and integrate the information of all other words in the sentence into the current word.

3.2.3. Residual Connection and Standardization. Transpose the result obtained in the previous step to make it the same as the dimension of X. Add the previous value and the value after self-attention operation, and then do the same operation for each module to avoid the gradient disappearing in the subsequent cross-layer connection:

$$X_{attention} = X + X_{attention}. \tag{6}$$

In order to reduce the execution time, the added results are normalized to make the output of the self-attention layer obey the standard normal distribution as follows:

Calculate the mean value by the behavior unit of the matrix:

$$\mu_i = \frac{1}{m}\sum_{i=1}^{m} x_{ij}. \tag{7}$$

Calculate the variance with the behavior unit of the matrix:

$$\sigma_j^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x_{ij} - \mu_j\right)^2. \tag{8}$$

Then, the average value of (7) is subtracted from each element of each row of the matrix one by one and then divided by the standard deviation of this row to get the normalized result. ε is used to prevent the denominator from being 0, and parameters α , β are used to make up for the lost information in the process:

$$LayerNorm(x) = \alpha \bullet \frac{x_{ij} - \mu_i}{\sqrt{\sigma^2 + \varepsilon}} + \beta. \tag{9}$$

Then (9) is normalized as follows:

$$X_{attention} = LayerNorm\left(X_{attention}\right). \tag{10}$$

The normalized vector list is transferred to a fully connected feedforward neural network.

3.2.4. Calculation of Feedforward Neural Network. The feedforward neural network layer is activated by activation function through two-layer linear mapping:

$$X_{output} = Activate\left(Linear\left(Linear\left(X_{attention}\right)\right)\right). \tag{11}$$

Then, after summation and standardization, the word vector list is output.

In addition, two new pretraining tasks MLM (masked language model) and NSP (next sentence prediction) are proposed in BERT. MLM refers to processing 15% of the words in a sentence with special markers. The selected words included in the 15% may be covered, replaced with other characters, or do nothing. The probabilities of these three

kinds of possibilities are 80%, 10%, and 10%, respectively. Then, the model infers what the processed words should be by learning and understanding the context of the processed words.

The model relies on context to predict the covered or replaced words, so this task can make the model have the ability of error correction [13] and can deal with the text with nonstandard language expression calmly, without making the deviation of vectorization result too large.

In the NSP task, we take out some data from the corpus. Half of the data are likely to be matched before and after the sentence, and half of the data are likely to be confused. We can learn the relationship between sentences through training. This operation can improve the ability of the model to judge the relationship between sentences in the question answering system.

3.3. MapReduce Programming Model and Mahout Library. The combination of the machine learning algorithm and the parallel computing mode can greatly improve the efficiency of the algorithm [14]. MapReduce programming model is a typical parallel computing model [15]. The MapReduce processing framework is shown in Figure 4.

Mahout, as a machine learning algorithm library, integrates a series of classification, clustering, dimension reduction, and recommendation algorithms [16], such as k-means, Canopy [17], and so on. Some algorithms on Mahout support the implementation of the MapReduce programming model and can run on the Hadoop platform [18].

4. Parallel Cleaning Process of Similar Duplicate Chinese Data

The data cleaning method based on the combination of the BERT model and k-means clustering algorithm of the Hadoop platform includes three processes: text preprocessing, text to vector, and text clustering. The specific steps are as follows:

Step 1: get the source data and preprocess the data

Step 2: use the BERT model to vectorize the preprocessed data

Step 3: calculate the cosine similarity of the vectorized data

Step 4: cluster similar texts and delete them

4.1. Text Preprocessing. Usually, there are some interference factors in the source data, which cannot be cleaned directly. The following treatment is required:

(1) Special character processing: By removing the blank and other special symbols, the subsequent vectorization calculation is more focused on the word itself, and the interference of noise data on the accuracy of vectorization is reduced.

(2) Font conversion: In order to prevent the initial input from being affected by the lack of word representation in the word vector table, the traditional characters are converted into simplified characters, which can ensure semantic integrity and obtain the correct initial vector.

(3) Character format conversion: The data format of the original text is iso-8859-1, which is presented in garbled format on Linux system, so it needs to be converted to utf8 format.

(4) Text cutting: Each line of the original text file is cut to form a new file.

(5) Conversion of text to sequential format: The process uses SequenceFile as its basic data exchange format.

(6) Text capture: The maximum length of the text sequence accepted by the BERT base is 512 characters, and the text with more than 512 characters needs to be processed. The results of different processing methods are shown in Table 1. According to the experimental data of the existing paper [19], it can be seen that the error rate of the first 128 plus the last 382 is the lowest, so this paper does the same processing.

4.2. Text to Vector. The computer cannot process the text data directly, so it is necessary to transform the text data into a mathematical expression that can be processed by the computer. At first, the text is represented directly by the simplest word set model, which has no semantic information of the text. In this paper, we use the technical idea of the BERT language model to extract the features of the text and transform the text into the corresponding mathematical expression so as to reduce the loss of semantic information as much as possible.

In this paper, the processing of Chinese text directly takes a single word as the basic unit of the text, so the text is divided into words, and then each word in the text is converted into a one-dimensional vector as the word vector by querying the word vector table. In the process of model training, the text vector integrated with the semantic information of a single word is automatically learned, and the words in different positions are given different position vectors. The sum of the three vectors is used as the model input [20]. The output of the model is the vector representation of the input words, which is used to describe the global semantic information of the text. The process of text vectorization is shown in Figure 5.In Figure 5, w_i represents the text input of the word, and W_i represents the word vector list processed by the bidirectional transformer encoder, an important part of BERT.

Because Mahout does not implement the BERT algorithm, this experiment designs the parallel process of BERT based on the Hadoop platform. The process is shown in Figure 6.

The vectorization result format is {word ID in text: eigenvalue}, and the text content is in a bracket, as shown in Figure 7.

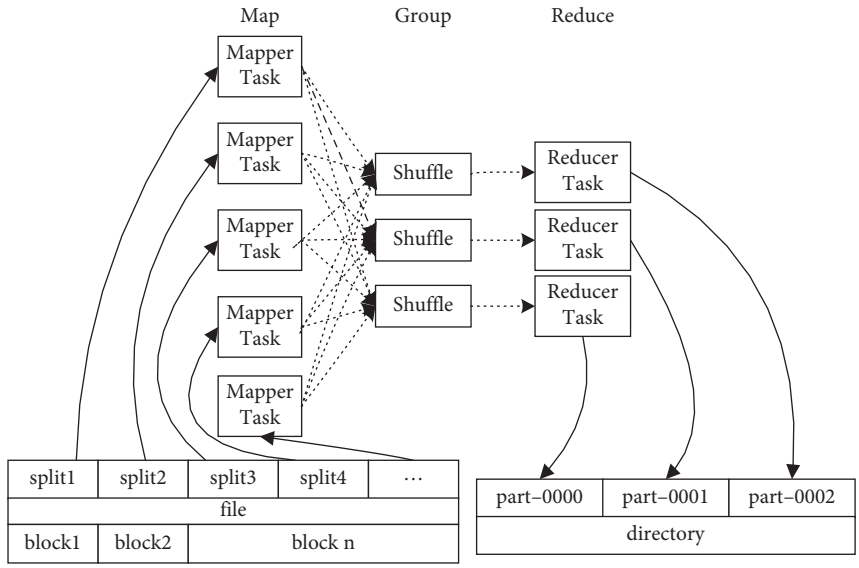

Figure 4: MapReduce processing framework.

Table 1: Error rate on IMDB and Sogou.

Method	IMDB	Sogou
Head only	5.63	2.58
Tail only	5.44	3.17
Head + tail	5.42	2.43
hier.mean	5.89	2.83
hier.max	5.71	2.47
hier.self-attention	5.49	2.65

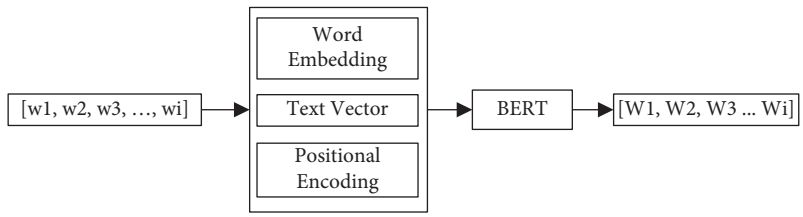

Figure 5: Text to vector process.

4.3. Similar Text Cleaning. Mahout has built-in text vectorization methods, such as TF-IDF [21], but this method only considers the importance of word frequency to document semantic expression, which cannot fully reflect the information contained in the text. Therefore, this experiment uses the above text vectorization method and, after parallel design, the output path of the result is "/BERT-vectors" in HDFS, which is subsequently used as the input for executing the canopy algorithm and k-means algorithm.

Execution Mahout canopy:

```
mahout canopy -i /user/root/clean3-ck/log-sparse/BERT
-vectors -o /user/root/clean3-ck/canopy-output -dm or
g.apache.mahout.common.distance.CosineDistanceMeas
```

Because for the random given data set, its characteristics are not clear, and the subjective setting of k value will cause the uncertainty of clustering results, so we need canopy to get the clustering center k [22].

Execute Mahout k-means:

```
mahout kmeans -i /user/root/clean3-ck/log-sparse/BER
T-vectors -c /user/root/clean3-ck/canopy-output/clusters
-0-final -o /user/root/clean3-ck/k-means-output -dm or
g.apache.mahout.common.distance.CosineDistanceMeas
ure -x 200 -ow --clustering
```

5. Analysis of Experimental Results

5.1. Experimental Configuration. The experimental environment consists of four personal computers. The cluster consists of one master node and three slave nodes. The operating system of each node is CentOS 7.5; Hadoop version is Hadoop 2.8; and JDK version is jdk-1.8.0_121, and other configuration parameters are shown in Table 2.

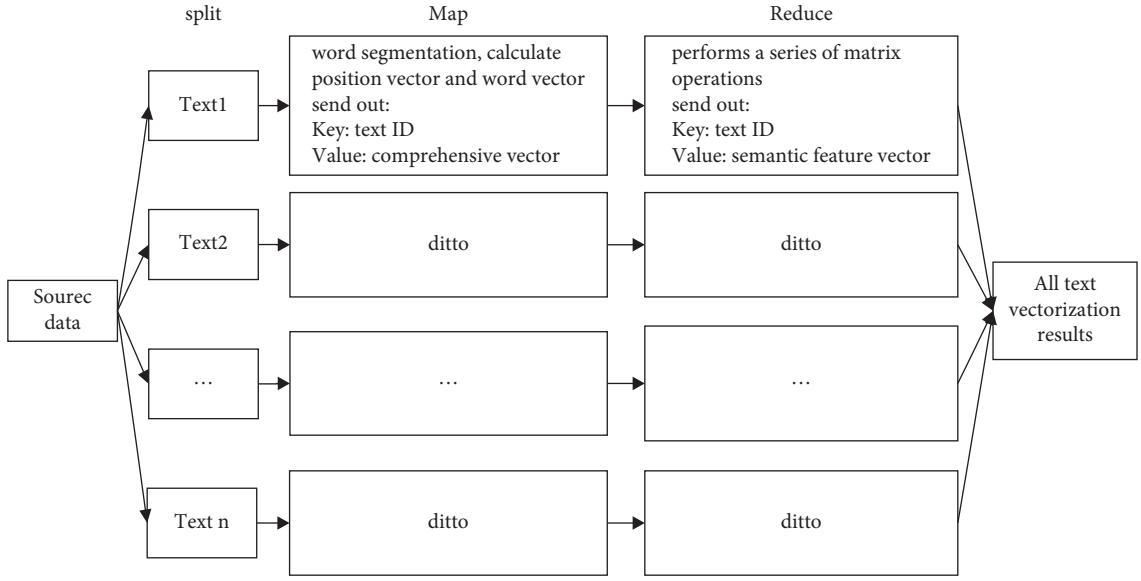

Figure 6: MapReduce process of BERT.

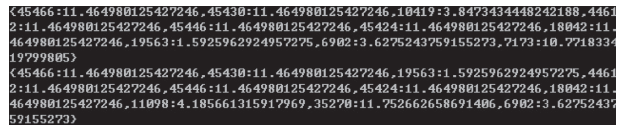

Figure 7: Text to vector results.

Table 2: Configuration parameters of cluster nodes.

Host name	IP	Chip model	Number of cores	Running memory (GB)	Hard disk size
Master	192.168.2.101	Intel® Core™ i7-6700 CPU @3.40 GHz	8	8	1,000 GB
slave1	192.168.2.103	Intel® Xeon® CPU E5-1603 v3@2.80 GHz	4	16	2,000 GB
slave2	192.168.2.102	Intel® Core™ i3-2120 CPU @3.30 GHz	4	8	500 GB
slave3	192.168.2.104	Intel® Core™ i5-4590 CPU @3.30 GHz	4	8	500 GB

5.2. Experimental Data. The experimental data are from ant financial NLP competition data set (data-1), 2018 Weizhong Bank Intelligent Customer Service Question Matching contest (data-2), the Third Magic Glass Cup Competition (data-3), and the business process log (data-4) generated during the operation of a platform in the laboratory.

The first three data sets are all customer service conversation data. There are many ways to describe and answer the same question. Through data cleaning, we can find out the questions with similar semantics. The last data set contains 20 running logs, with a total of 3.1711 million records. There are a large number of similar duplicate log contents between the log records generated. In the process of constructing the data set, the semantically similar duplicate data are filtered by a regular filter and then labeled manually.

5.3. Experiment I. Analysis of the Influence of Semantic Understanding on Cleaning Results. In order to verify the importance of semantic understanding in cleaning similar duplicate Chinese data, this paper compares the algorithm with the SNM, MPN, and PQM in terms of precision, recall, and F1-score. In order to avoid the randomness of the results, each group of comparative experiments was repeated five times, and the mean value of all results was taken as the final result.

5.3.1. Precision. The precision [23] is calculated based on the prediction results as follows:

$$\text{precision} = \frac{N_2}{N_1}, \tag{12}$$

where N_2 means that it is recognized as repeated, and it is also repeated actually. N_1 means that it is recognized as repeated. The value is between 0 and 1. The closer it is to 1, the better the understanding of polysemy and the better the cleaning effect. The results are shown in Figure 8.

From Figure 8, the algorithm in this paper has more advantages in precision than other traditional algorithms because the other three algorithms only judge whether the surface of characters is similar and do not consider the deep semantic information. In daily expression, the same word may represent different meanings in different contexts. For

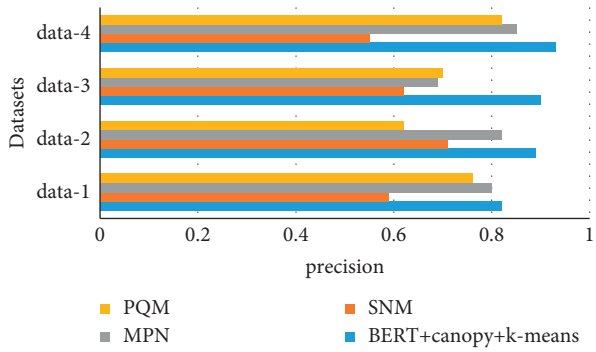

Figure 8: Comparison of precision results.

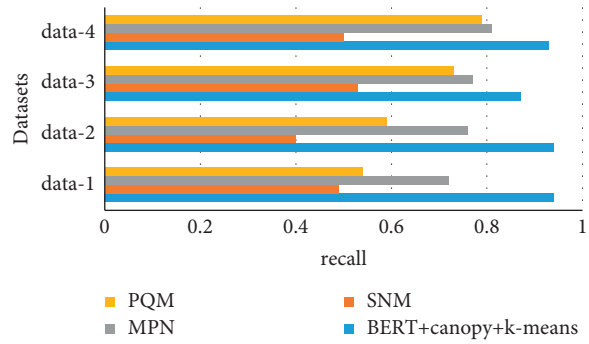

Figure 9: Comparison of recall results.

example, "I bought a bag of apples in the store" and "I bought an Apple phone in the store," the meaning of the two sentences is obviously different. The "apple" in the first sentence represents fruit in its context, and the "apple" in the second sentence represents the mobile phone brand in its context, which is identified as similar duplicate data according to the traditional data cleaning algorithm. In fact, the meaning is completely different. The position information is introduced into BERT to analyze the correlation between the words on the left and right sides of the word and the word itself, and this relationship is integrated into the vector coding. Therefore, the vector expression of the word "apple" when the mobile phone appears in the context is completely different from that when the fruit appears in the context, so it is more accurate in calculating the similarity between sentences, higher precision, and more reliable in clustering results.

5.3.2. Recall. The recall [24] is calculated based on the original sample as follows:

$$\text{recall} = \frac{N_2}{N}. \tag{13}$$

where N_2 means that it is recognized as repeated, and it is also repeated actually. N means all duplicate data in the sample. The value is between 0 and 1; the closer it is to 1, the more comprehensive the cluster coverage is. The results are shown in Figure 9.

As can be seen from Figure 9, the algorithm still has a good recall because, for two texts expressing the same meaning with synonyms or synonymous poems, the co-occurrence of words is very small, so the traditional algorithm cannot recognize the sentences with different glyphs and the same meaning and cannot recognize the original similar sentences, and the word is added in the process of text to vector through the BERT model. The BERT model adds word position information in the process of text to vector so that the vector expression of synonyms is similar, and the subsequent recognition of similar duplicate records is better.

5.3.3. F1-Score. Precision and recall are mutually exclusive quantities, and generally, the optimal value cannot be obtained at the same time. F1 is the harmonic average of precision rate and recall rate, and the calculation formula is as follows:

$$F1 - score = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{14}$$

F1-score is between 0 and 1; the closer it is to 1, the better the clustering effect. The results are shown in Figure 10.

Based on the above experimental results and analysis, it can be seen that this algorithm can recognize the situation of "polysemy" and "synonymy" in Chinese, the cleaning effect on different data sets is better than the other three algorithms, and the stability is more than 80%. Therefore, this algorithm can be used for Chinese data cleaning.

5.4. Experiment II. Influence of Text Vectorization on Cleaning Results. The vectorization result of text will directly affect the subsequent cleaning effect. In this section, experiments are designed to verify the importance of text vectorization to data cleaning. BOW [25], TF-IDF, and CBOW [26] vectorization methods are selected to convert the preprocessed text into a mathematical expression, and then canopy and k-means clustering algorithms are used to clean similar duplicate data as comparative experiments, which are compared with the cleaning algorithm based on BERT. The comparison results of precision, recall, and F1-score are shown in Figures 11–13:

It can be seen from the above figures that the precision, recall, and F1-score of the text vectorized by BERT are generally higher than those of other algorithms in subsequent similarity detection, and the cleaning effect of the word bag model is the worst. The model analyzes each word in the text as an individual, completely ignoring the relationship between words, the position, and grammar between words and sentences. Secondly, the vector space of the word bag model will increase with the increase of word lists, which requires a lot of storage space. Moreover, the word vector is sparse; most areas are 0; and the vector space structure is not good enough. Therefore, the dimension disaster and semantic gap lead to poor performance in similarity detection. CBOW effect is sometimes good or bad, with great contingency and instability. The model produces word vectors without context, which cannot describe the context information of words. Therefore, for sentences such as "I am higher than you" and "you are higher than me," there is still

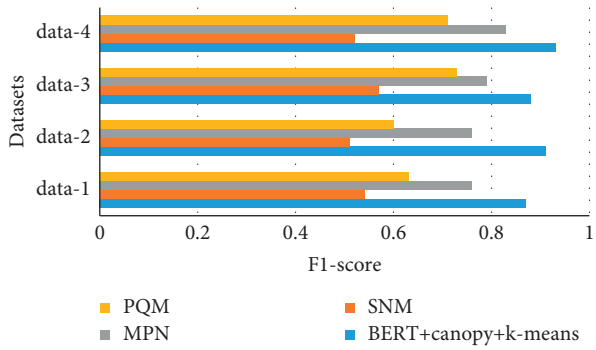

Figure 10: F1-score comparison.

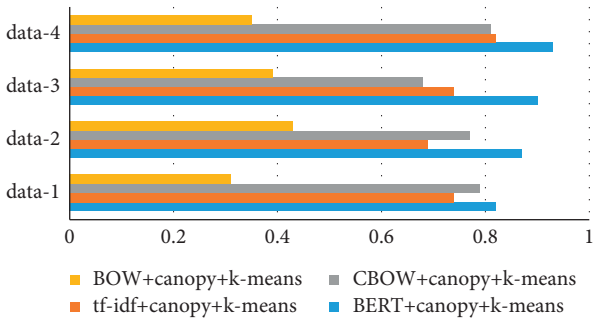

Figure 11: Comparison of precision results.

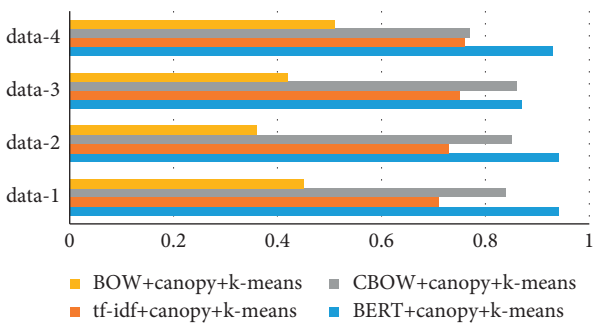

Figure 12: Comparison of recall results.

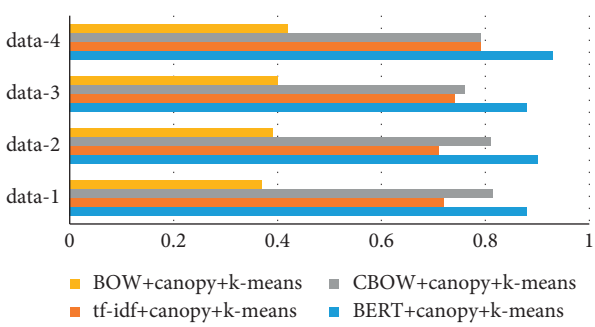

Figure 13: F1-score comparison.

no difference in vectorization results without considering the position information of words in sentences. TF-IDF method selects keywords with strong discrimination to represent text, but it is not friendly to long text and can only distinguish the same subject rather than the same meaning, so it may not surpass CBOW in some cases. The BERT vectorization method introduced in this paper considers the location information, produces the word vector focusing on the context, and processes the polysemy more appropriately so as to clean out the synonymous sentences. According to the comparison diagram of precision, recall, and F1-score and the above analysis, the cleaning precision rate of text data after using the vectorization method in this paper is more than 80%, and the F1-score is stable. Compared with other algorithms, similar repeated clustering after generating vectors by this method can get a better cleaning effect.

5.5. Experiment III. Performance Analysis of Parallel Algorithm. In this paper, the parallel cleaning method based on clustering is improved. Taking the data set data-4 as an example, the advantages of parallel computing for big data processing are illustrated from the aspects of speedup and scalability. Choose a different amount of data (unit: 10,000) and different cluster sizes to experiment.

5.5.1. Speedup. Speedup is used to measure the performance and effect of parallel systems or program parallelization [27]. It is the ratio of the time consumed by the same task in a single processor system and a parallel processor system.

$$S = \frac{Ts}{Tp}, \tag{15}$$

where Ts is the running time of a single node and Tp is the running time of P nodes ($P = 1, 2, 3, 4$). The experimental results of acceleration are shown in Figure 14.

From Figure 14, the overall trend of speedup is linear growth; especially when the data volume increases, the speedup is particularly excellent, which indicates that parallel computing has a strong advantage in processing large data. The data volume is a little bit small, and the speedup is slow. Because with the increase of the number of nodes, the communication time of each node also increases. Meanwhile, parallel processing has no obvious advantage on the small data volume, which results in the communication time of each node having a great impact on the time performance of the parallel calculation of small data. In short, the larger the data and computing tasks, the higher the efficiency of execution in a distributed environment.

5.5.2. Scalability. The scalability can reflect the utilization of cluster, and the formula is as follows:

$$E = \frac{S}{P}, \tag{16}$$

where S is acceleration ratio and P is the number of nodes. The experimental results are shown in Figure 15.

Ideally, the overall computing capacity of distributed clusters should increase linearly with the increase in the number of machines. In fact, there is a certain communication consumption between nodes, so scalability cannot reach the ideal value. From the above figure, when the data

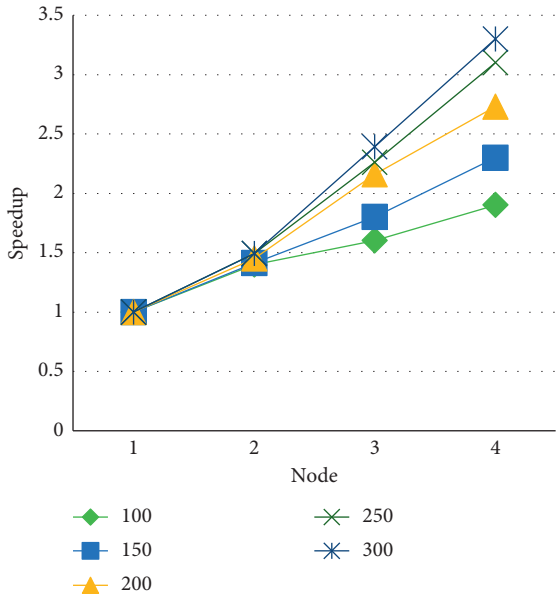

Figure 14: Comparison of speedup results.

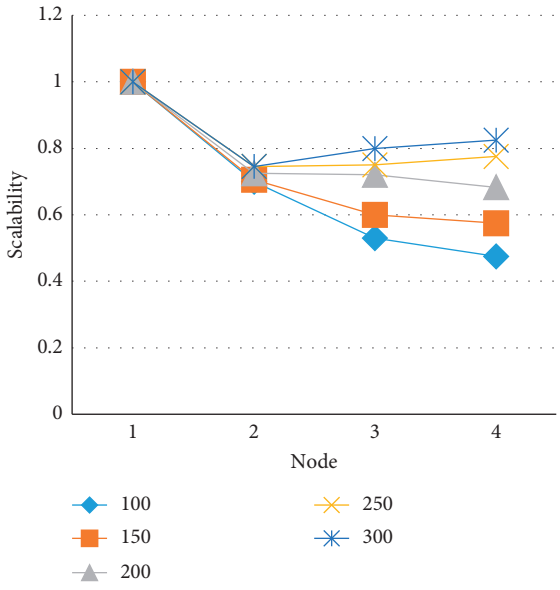

Figure 15: Comparison of scalability results.

volume is more than 2.5 million, the cluster scalability is higher than 80%, which indicates that the computing ability of distributed system has a good effect on processing big data. When the data scale is constant, the more machines in the cluster, the stronger the overall computing ability. However, with the increase of the number of machines, the expansion rate of processing the same data volume is decreasing because the communication between nodes consumes more resources; when the number of nodes is fixed, the larger the data scale is, the higher the utilization rate of the cluster. In general, increasing the cluster nodes and expanding the cluster scale can meet the high-performance requirements of big data processing.

6. Conclusion

Based on the distributed computing platform, this paper studies the similar repeated Chinese data cleaning algorithm, analyzes the advantages and disadvantages of the traditional algorithm, and introduces the BERT text vectorization algorithm combined with the characteristics of the Chinese language so as to improve the accuracy of vectorization results at the semantic level and make the subsequent calculation of Chinese semantic similarity more credible. At the same time, considering the current situation of big data applications, the comparative experiment shows the advantages of parallel computing, which proves that the algorithm design idea of this paper is more in line with the current expectation of high efficiency of data processing.

Data Availability

The data sets used to support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61902133).

References

[1] J. Chen, K. Li, Z. Tang et al., "A parallel random forest algorithm for big data in A spark cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 919–933, 2016.

[2] O. Aytuğ, "An ensemble scheme based on language function analysis and feature engineering for text genre classification," *Journal of Information Science*, vol. 44, no. 1, pp. 28–47, 2018.

[3] O. Aytuğ, "Two-stage topic extraction model for bibliometric data analysis based on word embeddings and clustering," *IEEE Access*, vol. 7, pp. 145614–145633, 2019.

[4] P. G. Zhang and S. C. Huang, "A nearest neighbor sorting algorithm for Chinese data cleaning," *Computer Applications and Software*, vol. 35, pp. 286–288, 2018.

[5] M. A. Hernández and S. J. Stolfo, "Real-world data is dirty: data cleansing and the merge/purge problem," *Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 9–37, 1998.

[6] A. Monge and C. Elkan, "An efficient domain independent algorithm for detecting approximately duplicate database records," in *Proceedings of the SIGMOD Workshop on Data Mining and Knowl-edge Discovery*, pp. 23–29, Tucson, Arizona, January 1997.

[7] H. Z. Ye and D. W, "Review of similar duplicate records cleaning methods," *Modern library and information technology*, vol. 9, pp. 56–66, 2010.

[8] T. Lin and C. Zhao, "K-means clustering algorithm based on nearest neighbor optimization," *Computer science*, vol. 46, pp. 216–219, 2019.

[9] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "Hybrid clustering analysis using improved krill herd algorithm," *Applied Intelligence*, vol. 48, no. 11, pp. 4047–4071, 2018.

[10] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A combination of objective functions and hybrid krill herd algorithm for text document clustering analysis," *Engineering Applications of Artificial Intelligence*, vol. 73, pp. 111–125, 2018.

[11] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, ACL, Minneapolis, MN, USA, June, 2019.

[12] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the 31st Annual Conference on Neural Information Processing Systems(NIPS)*, pp. 5998–6008, Curran Associates, Long Beach, CA, USA, December, 2017.

[13] P. M. Yan and W. Q. Tang, "Chinese text classification based on improved Bert," *Industrial control computer*, vol. 33, no. 7, pp. 108–110, 2020.

[14] J. Chen, K. Li, K. Bilal, X. Zhou, K. Li, and P. S. Yu, "A Bi-layered parallel training architecture for large-scale convolutional neural networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 5, pp. 965–976, 2018.

[15] P. Sowkuntla, S. Dunna, and P. S. V. S. Sai Prasad, "MapReduce based parallel attribute reduction in Incomplete Decision Systems," *Knowledge-Based Systems*, vol. 213, p. 106677, 2021.

[16] P. D. Hung and D. L. Phan, "E-commerce recommendation system using mahout," in *Proceedings of the 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, pp. 86–90, IEEE, Singapore, February, 2019.

[17] J. Y. Liu and J. B. Wang, "Improvement of slope one Bi algorithm and its parallel application in big data platform," *Journal of Huaqiao University*, vol. 40, no. 06, pp. 786–792, 2019.

[18] U. Demirbaga and D. N. Jha, "Social media data analysis using MapReduce programming model and training a tweet classifier using Apache mahout," in *Proceedings of the 2018 IEEE 8th International Symposium on Cloud and Service Computing (SC2)*, pp. 116–121, Paris, France, November, 2018.

[19] C. Sun, X. P. Qiu, Y. Xu, and X. Huang, "How to fine-tune BERT for text classification," in *Proceedings of the Chinese Computational Linguistics - 18th China National Conference*, pp. 194–206, Kunming, China, October, 2019.

[20] D. D. Duan, J. S. Tang, Y. Wen, and K. H. Yuan, "Short Chinese text classification algorithm based on Bert model," *Computer Engineering*, vol. 47, no. 1, pp. 79–86, 2021.

[21] I. Arroyo-Fernández, C. F. Méndez-Cruz, G. Sierra, J.-M. Torres-Moreno, and G. Sidorov, "Unsupervised sentence representations as word information series: revisiting TF–IDF," *Computer Speech & Language*, vol. 56, pp. 107–129, 2019.

[22] H. Y. Wang, W. C. Cui, P. D. Xu, and L. I. Chuang, "Optimization of K selection in partition clustering algorithm by canopy," *Journal of Jilin University (Science Edition)*, vol. 58, no. 3, pp. 634–638, 2020.

[23] Y. S. Liu and X. Li, "Improved SNM algorithm based on length filtering and dynamic fault tolerance," *Computer application research*, vol. 34, no. 1, pp. 147–150, 2017.

[24] Z. P. Huang, L. Wang, and M. F. Zhang, "Design of intelligent cleaning system for massive big data based on cloud computing," *Modern electronic technology*, vol. 43, no. 3, pp. 116–120, 2020.

[25] Y. Shi and Q. Peng, "Definition of customer requirements in big data using word vectors and affinity propagation clustering," *Proceedings of the Institution of Mechanical Engineers - Part E: Journal of Process Mechanical Engineering*, vol. 235, no. 5, pp. 1279–1291, 2021.

[26] H. Wang, J. H. Pan, H. C. Wang, Q. Zhang, and Y. Zhang, "Research on text classification based on improved CBOW and BI-LSTM-ATT," *Computer and Digital Engineering*, vol. 49, no. 7, pp. 1372–1376, 2021.

[27] D. H. Yang, N. N. Li, H. Z. Wang, J. Z. Li, and H. Gao, "Optimization of parallel big data cleaning process based on task merging," *Acta computer Sinica*, vol. 39, no. 1, pp. 97–108, 2016.