processes

MDPI

Article

A Study of Text Vectorization Method Combining Topic Model and Transfer Learning

Xi Yang [1,2,*], Kaiwen Yang [1,*], Tianxu Cui [1], Min Chen [1] and Liyan He [1]

[1] School of Information, Beijing Wuzi University, Beijing 101149, China; cuitianxubwu@163.com (T.C.); chenmincmark@163.com (M.C.); hly199808@163.com (L.H.)

[2] School of Computer & Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China

* Correspondence: yangxi@bwu.edu.cn (X.Y.); ykw1234@163.com (K.Y.)

Abstract: With the development of Internet cloud technology, the scale of data is expanding. Traditional processing methods find it difficult to deal with the problem of information extraction of big data. Therefore, it is necessary to use machine-learning-assisted intelligent processing to extract information from data in order to solve the optimization problem in complex systems. There are many forms of data storage. Among them, text data is an important data type that directly reflects semantic information. Text vectorization is an important concept in natural language processing tasks. Because text data can not be directly used for model parameter training, it is necessary to vectorize the original text data and make it numerical, and then the feature extraction operation can be carried out. The traditional text digitization method is often realized by constructing a bag of words, but the vector generated by this method can not reflect the semantic relationship between words, and it also easily causes the problems of data sparsity and dimension explosion. Therefore, this paper proposes a text vectorization method combining a topic model and transfer learning. Firstly, the topic model is selected to model the text data and extract its keywords, to grasp the main information of the text data. Then, with the help of the bidirectional encoder representations from transformers (BERT) model, which belongs to the pretrained model, model transfer learning is carried out to generate vectors, which are applied to the calculation of similarity between texts. By setting up a comparative experiment, this method is compared with the traditional vectorization method. The experimental results show that the vector generated by the topic-modeling- and transfer-learning-based text vectorization (TTTV) proposed in this paper can obtain better results when calculating the similarity between texts with the same topic, which means that it can more accurately judge whether the contents of the given two texts belong to the same topic.

Keywords: text vectorization; topic model; pretrained model; transfer learning

check for updates

Citation: Yang, X.; Yang, K.; Cui, T.; Chen, M.; He, L. A Study of Text Vectorization Method Combining Topic Model and Transfer Learning. *Processes* **2022**, *10*, 350. https://doi.org/10.3390/pr10020350

Academic Editor: Mengchu Zhou

Received: 9 January 2022
Accepted: 6 February 2022
Published: 11 February 2022

1. Introduction

In the Internet era, the interaction of online information is becoming more and more frequent, and the development of big data and the cloud computing technology promotes the dissemination of online information, which is a great driving force for the development of industries such as news media with information dissemination as the business core. Compared with the traditional paper media, online news is more timely because of the network as the communication media [1]. At the same time, because it is not limited by the printing layout, its communication quantity is often huge. For users who receive and browse news, a large number of fast updated online news cause them to enjoy the great convenience brought around by the fact that the Internet can retrieve information in seconds without the need to leave home [1]. However, in front of a large amount of information, how can they easily find other news with similar topics to expand their reading under the condition of browsing news in their field of interest, that is, the relevant recommendations? This is our consideration in modeling in this paper.

For the above scenarios, the traditional data processing methods are difficult to deal with. Therefore, it is necessary to adopt the machine-learning-assisted intelligent processing method to structurally operate the unstructured text data, and optimize the parameters of the model by analyzing the actual supply–demand relationship in the system. Therefore, we can obtain that for the currently browsed text data, finding other text data with high similarity is an effective method to solve the above problems. Because the text data cannot be directly used for modeling, it needs to be numerically processed [2,3], that is, the text vectorization operation [4,5], and then the generated vector, is used as the training data sample of the machine learning model. Text vectorization can be realized in two ways: one is the word vector, generated by taking the word as the basic unit [6], and the other is the paragraph vector, considering paragraph factors on the basis of word vector [7–9]. Word vector is a distributed representation of words, which aims to capture the syntactic and semantic information of words [10,11]. Before the text data is input into the network layer, it must be vectorized according to some method to become structured data. After the text is converted into word vector, the discrete symbols will be encoded according to the index to reduce the amount of parameters and improve the generalization ability.

In this paper, the text vectorization method combining topic model [12,13] and transfer learning [14,15] is studied. Firstly, the topic model is used to extract keywords from the text, to solve the problem of latent semantic mining which is difficult to solve by the traditional term frequency inverse document frequency method; then, the paragraph vector model is used to vectorize the text to solve the problem of mining the semantic relationship between paragraph and context that is difficult to solve by the word vector model. At the same time, the pretrained model is used for transfer learning to facilitate deeper feature learning of text data [16–18], so that the generated text vector can more accurately reflect semantic information.

The main contributions of this paper are as follows:

- The topic model keyword extraction method is introduced into the text vectorization operation.
- The text vector that can deeply reflect the semantic relationship is generated by transferring the pretrained model.
- The above text vectorization method is applied to the calculation task of text similarity.

The rest of this paper is organized as follows: Section 2 is the literature review; Section 3 introduces the materials and methods; in Section 4, a comparative experiment is designed. The text vectorization method used in this paper is applied to the calculation of text similarity, and compared with the text similarity calculated by the vector generated by traditional text vectorization, and the experimental results are analyzed; Section 5 is the conclusion.

2. Literature Review

We review the related literature from the following three aspects: text vectorization, topic model, and pretrained model for transfer learning.

The generation of word vector can be understood as encoding text data according to a certain model. Its development has roughly experienced several stages, such as one-hot representation [19], bag of words (BOW) [20,21], language model (LM) [22], word2vec [6], and so on. One-hot representation generates word vectors by binary coding, and each dimension only indicates whether the corresponding word in the dictionary is taken at the location. This method will not only bring dimensional disaster and lead to data sparsity, but also cause lack of feature extraction of text semantics; BOW replaces binary data with word frequency data on the basis of one-hot, but it still fails to solve the problems of dimension disaster and semantic loss; the LM model uses conditional probability to express the association between words in text sequence. The semantic representation method of the LM model is relatively primitive, so it has been developed to word2vec model. The word2vec model is composed of continuous bag of words model (CBOW) and skip-gram model. The word2vec model uses a structure similar to neural network to establish the relationship between words, but the model only focuses on the local semantic information and fails to combine the global information of the text. The vector of the same word is still the same in different contexts, that is, it cannot solve the problem of polysemy. As can be seen from the above, word vectors mostly focus on the semantic connection between words themselves or local words [10,11], so the information such as paragraphs and word order of the text can not be accurately extracted and fully utilized. Therefore, a paragraph vector model is proposed, that is, para2vec [7], sometimes referred to as doc2vec. Para2vec is an extension of word2vec [8,9]. When generating vectors, not only the context words but also the corresponding paragraph information are considered. Para2vec model consists of a distributed memory model (DM) and a distributed bag of words model (DBOW).

Text data is composed of a large number of sentences. In order to avoid the problem of dimension explosion, we cannot directly construct vectors with sentences as feature dimensions. Sentences can be regarded as a sequence of words [23], so we can extract the representative words, that is, keywords, and then generate vectors based on keywords by constructing certain models and algorithms. There are two kinds of text keyword extraction methods, one is the term frequency inverse document frequency (TFIDF) method based on frequency, and the other is the topic model method based on spatial mapping [12,13]. The TFIDF method measures the importance of a specific word in each sentence by two factors: term frequency, that is, the frequency of a word in the sentence, and inverse document frequency, that is, the reciprocal of the frequency of the sentence containing the above words. Although this method constructs the correlation between words and sentences, it only analyzes them from the perspective of frequency, and fails to construct the mapping relationship based on the understanding of text semantics. In order to extract keywords from text based on understanding semantic structure, the topic model method is proposed. The topic model is an unsupervised learning model for latent semantic mining and analysis of text data. Its core is to construct the mapping between sentence space and topic space and between topic space and word space by setting topic variables as intermediate variables. Using spatial mapping to analyze text semantics is not only conducive to clustering words to reduce redundancy, but also realizes dimension reduction operation, to better reflect the distribution features of words, topics, and sentences. The topic model is mainly composed of latent semantic analysis model (LSA) and latent Dirichlet allocation model (LDA). The core of LSA [24] is the operation of singular value decomposition of text matrix to generate the mapping relationship among words, topics, and sentences, that is, word space and sentence space are mapped to relatively lower dimensional topic space, respectively, to explore the latent semantic relationship [25–27]. Compared with LSA model, the LDA [28,29] model focuses on constructing the mapping relationship between the above three variable spaces from the perspective of probability distribution, to avoid the huge amount of matrix operation caused by singular value decomposition operation. By introducing the relevant theories of Dirichlet distribution and multinomial distribution [30,31], the LDA model can carry out latent semantic mining from the perspective of probability distribution [32–34].

By transferring the parameter information obtained from some large and general source tasks to the target task, transfer learning avoids the problem of starting from scratch every training [14,15]. In general, we often lack enough targeted datasets for training, so we can use the pretrained model trained by the general large dataset for transferring. In the field of natural language processing, the development of pretrained model has mainly experienced several stages, such as embeddings from language models (ELMo), generative pretraining (GPT), and bidirectional encoder representations from transformers (BERT). ELMo [35] first learns the word embedding of each word through the language model, and then dynamically adjusts the word embedding according to the context [36–38], which can solve the problem of polysemy and realize the function of semantic relationship judgment at the same time [39]. In terms of feature extractor, ELMo adopts the LSTM, but a new feature extractor transformer [40] was proposed later, and the feature extraction ability proved to be better than LSTM. Therefore, based on transformer as the feature extractor [41,42], GPT is proposed [43]. The model is first pretrained through the language model, and then fine-tuned to access the downstream tasks; however, the model is a one-way language model [44,45], that is, when predicting words in sentences, it only pays attention to the above without considering the following, so it is not comprehensive in semantic understanding. In order to consider the semantic information in two word order directions at the same time, BERT [16] is proposed. It is a two-way language model based on transformer. Through the training of large datasets, a general pre-trained model with strong transferring ability is obtained [46,47].

3. Materials and Methods

3.1. Word2vec

Word2vec is a word vector generation method based on word embedding. Its core idea is the distributed expression of words, which maps words to vectors with definable dimensions. Because the context of the current word is considered in the training process, a low-dimensional dense vector with semantic information can be generated.

Specifically, each statement in a text paragraph is essentially a word sequence, expressed as $d = \left[d^{(1)}, d^{(2)}, \ldots, d^{(j)}, \ldots, d^{(n)}\right]$, where $d^{(j)}$ represents the word at the current jth position in the sequence, and n is the sequence length, that is, the number of words contained. In order to obtain the word vector, we need to construct the mapping relationship between the current word and its context, that is, $d^{(i)} = f(d^{(j)}), i \neq j$, the mapping relationship is trained through the neural network structure, and the hidden layer weight obtained in the training process is the word vector. Word2vec consists of CBOW and skip-gram.

3.1.1. Continuous Bag of Words (CBOW)

CBOW constructs the mapping relationship between the two adjacent words before and after the current word as the context, that is, the current word is predicted by the context, that is, $d^{(j)} = f(d^{(j-2)}, d^{(j-1)}, d^{(j+1)}, d^{(j+2)})$. The specific structure of the model is shown in Figure 1a. The average value of the word vector of the context is input as the weight, and then converted into the conditional probability $p(d^{(j)} \mid d^{(j-2)}, d^{(j-1)}, d^{(j+1)}, d^{(j+2)})$ of the current word under the condition of the context through the softmax activation function, to realize the semantic association between the current word and the context.

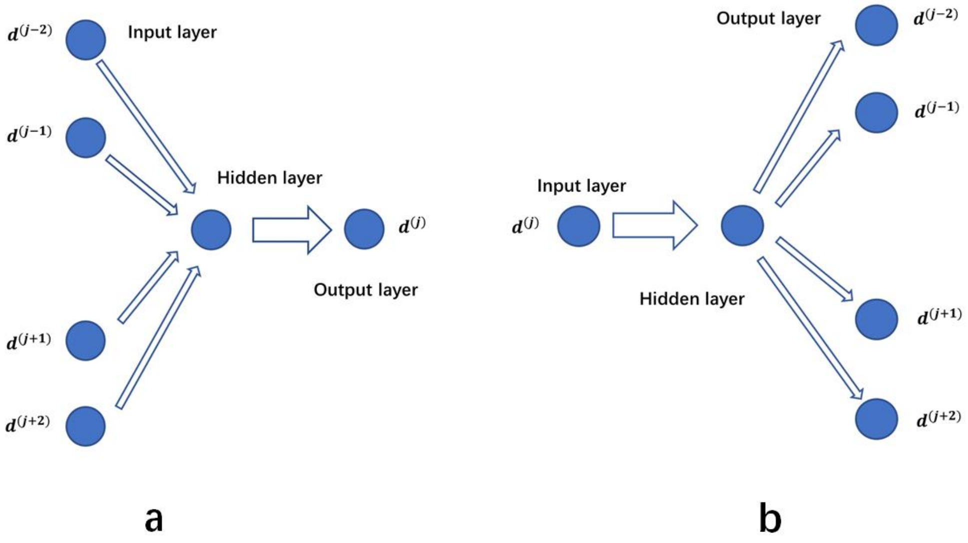

Figure 1. Word2vec ((**a**): CBOW (**b**): Skip-Gram).

3.1.2. Skip-Gram

Different from CBOW, skip-gram uses the current word to predict the context, and the scope of the context is definable. Its size of sliding window is called skip-window, as shown in Figure 1b. Letting the window size above and below be 2, respectively, and taking this as an example, the mapping relationship $d^{(j+c)} = f(d^{(j)}), c = \pm 1, \pm 2$ can be obtained. Then, similar to the training process of CBOW, feature extraction is carried out through the hidden layer, and finally probabilistic conversion is carried out through the softmax activation function to obtain the condition probability $p(d^{(j+c)} \mid d^{(j)}), c = \pm 1, \pm 2$.

3.2. Para2vec

Because the word2vec model does not consider paragraph factors in semantic learning, that is, the relationship between ordered clauses of a sentence, the para2vec model is proposed on this basis. The model mainly introduces a paragraph component with the same structure as the context component to predict the current word, namely the DM model, or the context is predicted by paragraph component, that is, the DBOW model.

3.2.1. Distributed Memory (DM)

DM is extended from CBOW. It also predicts the current word through context. On this basis, it adds paragraph ID and generates paragraph vector with the same structure as word vector. The paragraph component and context component are accumulated and connected together and sent to the network for training. Finally, it also uses softmax activation function for probabilistic conversion to obtain the conditional probability of the current word. In general, the DM model introduces word vector and paragraph vector into the process of semantic learning to predict the probability distribution of the current target word. The specific structure of the model is shown in Figure 2a.

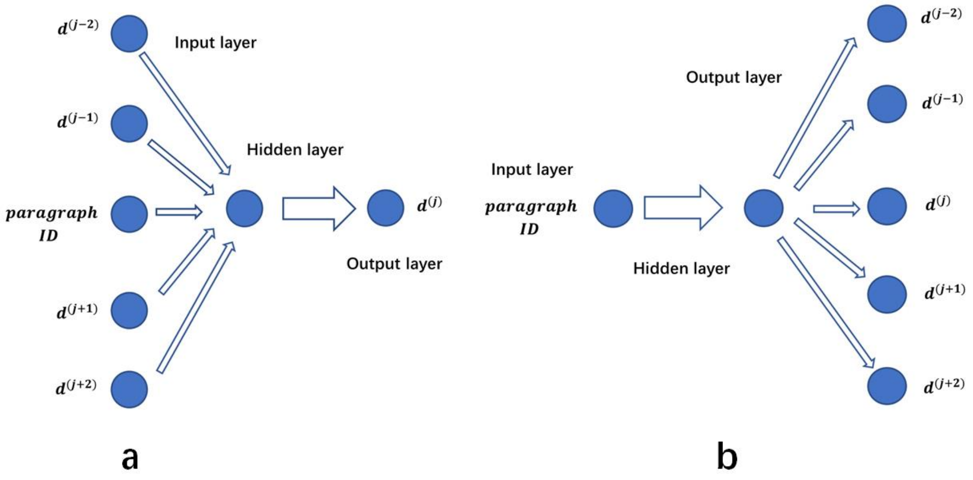

Figure 2. Para2vec ((**a**): DM (**b**): DBOW).

3.2.2. Distributed Bag of Words (DBOW)

DBOW is extended through skip-gram. It takes the context as the prediction target, and the range of the context to be predicted can also be determined by defining the size of the sliding window; the specific structure of the model is shown in Figure 2b. However, unlike skip-gram, DBOW does not directly use the current word, but takes the paragraph vector based on the paragraph ID as the input, and the predicted context word also comes from the inside of each input paragraph. Due to the two-way and orderly semantic relationship between input and output, such a para2vec learns not only semantic information, but also word order information.

3.3. Topic Model

Topic model is a kind of unsupervised learning model that analyzes text semantic information by constructing relationship mapping between sentence space, topic space, and word space.

Let the sentence space be D, in which each element d represents a sentence. The sentence can be regarded as a word sequence, so the sentence variable can be expressed as

$$d_i = \left[d_i^{(1)}, d_i^{(2)}, \ldots, d_i^{(j)}, \ldots, d_i^{(n_i)}\right] \quad i = 1, 2, \ldots, N, j = 1, 2, \ldots, n_i. \tag{1}$$

where $d_i^{(j)}$ represents the jth word in the ith sentence, N is the total number of sentence elements, and n_i is the total number of words in the ith sentence.

Let the topic space be T, where each element $t_k, k = 1, 2, \ldots, K$ represents the topic, and K is the number of types of topics.

Let the word space be W, where each element $w_v, v = 1, 2, \ldots, V$ represents a word and V is the number of types of words.

The distribution of each word in the sentence sequence is not random, but connected through the intermediate variable of topic. Because it is unobservable, it is called hidden variable, and the relative sentence variable and word variable are called observable variable. The goal of the topic model is to mine text semantic information by constructing the mapping relationship between the spaces of the above three variables. The common topic model consists of latent semantic analysis (LSA) and latent Dirichlet allocation (LDA).

3.3.1. Latent Semantic Analysis (LSA)

LSA is a topic model based on allocation matrix. Let the allocation matrix be $A_{N\times n}$, in which each element $a_i^{(j)}$ represents the word allocation probability at the jth position on the ith sentence. The allocation matrix can be expressed by the product of the allocation matrix $B_{N\times K}$ between sentence space and topic space and the $C_{K\times n}$ between topic space and word space, that is, $A_{N\times n} = B_{N\times K} \cdot C_{K\times n}$. Since the above distributions are hidden and cannot be solved directly, the singular value decomposition method is used to obtain the approximate estimates of $B_{N\times K}$ and $C_{K\times n}$.

3.3.2. Latent Dirichlet Allocation (LDA)

Because the singular value decomposition method used in LSA is mechanical matrix disassembly, it not only has a large amount of computation, but also does not consider the problem of probability distribution. LDA avoids the above defects. The model models each element in the sentence space, and then expands to the whole sentence space based on the principle of independent and identically distributed.

The specific modeling process is as follows:

The topic distribution of the sentence can be expressed as a parameter vector

$$\theta_i = [\theta_{i1}, \theta_{i2}, \ldots, \theta_{ik}, \ldots, \theta_{iK}], i = 1, 2, \ldots, N. \tag{2}$$

where each component θ_{ik} represents the distribution probability of the kth topic to which the ith sentence belongs. The parameter vector obeys the Dirichlet distribution, and its probability density function is

$$p(\theta_i \mid a) = \frac{\Gamma\left(\sum_k^K a_k\right)}{\prod_k^K a_k} \prod_k^K \theta_{ik}^{a_k - 1}. \tag{3}$$

where $a = [a_1, a_2, \ldots, a_k, \ldots, a_K], a_k > 0$ is distribution parameter.

Then, the topic distribution of the ith sentence is constructed by the multinomial distribution whose parameter is the parameter vector, and its probability density function is

$$p(t \mid \theta_i) = \frac{\left(\sum_k^K t_k\right)!}{\prod_k^K t_k!} \prod_k^K \theta_{ik}^{t_k}. \tag{4}$$

If each sample satisfies the principle of independent and identical distribution, the topic distribution $p(t \mid d_i)$ of the sentence can be randomly selected from the above distribution.

The word distribution of the topic can be expressed as a parameter vector

$$\eta_k = [\eta_{k1}, \eta_{k2}, \ldots, \eta_{kv}, \ldots, \eta_{kV}], k = 1, 2, \ldots, K. \tag{5}$$

where each component η_{kv} represents the distribution probability of the vth word corresponding to the kth topic. The parameter vector obeys the Dirichlet distribution, and its probability density function is

$$p(\eta_k \mid b) = \frac{\Gamma\left(\sum_v^V b_v\right)}{\prod_v^V b_v} \prod_v^V \eta_{kv}^{b_v - 1}. \tag{6}$$

where $b = [b_1, b_2, \ldots, b_v, \ldots, b_V], b_v \geq 0$ is distribution parameter.

Then, the word distribution corresponding to the kth topic is constructed by using the multinomial distribution with the parameter vector as the distribution parameter, and its probability density function is

$$p(w \mid \eta_k) = \frac{\left(\sum_v^V w_v\right)!}{\prod_v^V w_v!} \prod_v^V \eta_{kv}^{w_v} \tag{7}$$

If each sample satisfies the principle of independent and identical distribution, the word distribution $p(w \mid t_k)$ of the topic can be randomly selected from the above distribution.

The symbolic representation is shown in Table 1 below, the probability distribution relationship between variables is shown in Figure 3, and the modeling process is shown in Algorithm 1.

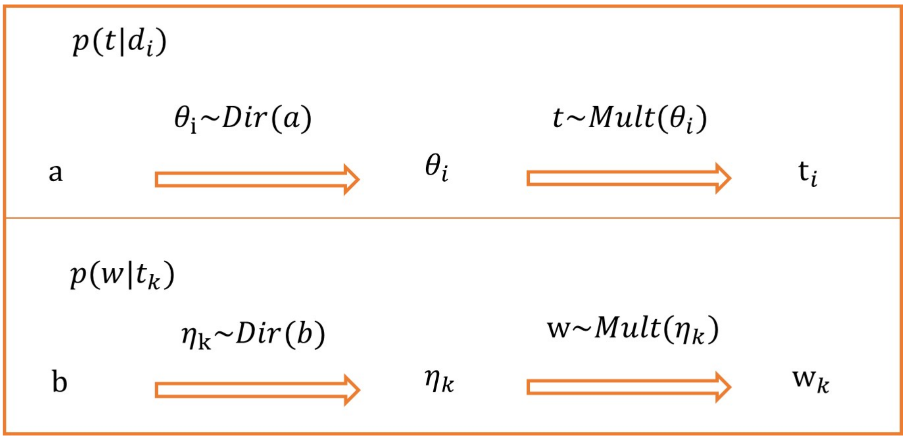

Figure 3. Probability distribution relationship between variables.

Table 1. List of mathematical symbols.

Symbol	Meaning
$d_i^{(j)}$	The component of the text sequence, which represents the jth word of the ith sentence
η_{kv}	Allocation probability when the kth topic corresponds to the vth word
b	Distribution parameters of Dirichlet distribution obeyed by η
$t_i^{(j)}$	The component of the topic sequence, which represents the topic corresponding to the jth word of the ith sentence
θ_{ik}	Distribution probability of the kth topic to which the ith sentence belongs
a	Distribution parameters of Dirichlet distribution obeyed by θ

Algorithm 1 Modeling based on LDA model.

```
Input: text set $D$; topic set $T$; word set $W$; parameters $a, b$ of Dirichlet distribution; length $n_i$ of each sentence $d_i \in D$.
Output: allocation $\theta_{ik}$ based on text-topic; allocation $\eta_{kv}$ based on topic-word; probability density $p(t \mid \theta)$ of topic; probability density $p(w \mid \eta)$ of word; topic sequence $t$; sentence sequence $d$.
Initialize $a$ and $b$ to 1 respectively.
for each $d_i \in$ range $(D)$ do
    for each $t_k \in$ range $(T)$ do
        generate $\theta_{ik} \sim Dir(a)$ as $p(t \mid d_i)$.
    end for
end for
for each $i \in$ range $(N)$ do
    for each $j \in$ range $(n_i)$ do
        generate $t_i^{(j)} \sim Mult(\theta_i)$.
        generate $d_i^{(j)} \sim Mult(\eta_{t_i^{(j)}})$.
    end for
end for
return $\theta_{ik}, \eta_{kv}, p(t \mid \theta), p(w \mid \eta)$, t $= \left[t_i^{(j)}\right]$, d $= \left[d_i^{(j)}\right]$.
```

3.4. *Pretrained Model and Transfer Learning*

Pretraining means that when the model starts training, it does not need to initialize the parameters randomly from scratch, but initialize and then train on the basis of a set of model parameters that have learned a large number of general features in advance. BERT is a pretrained model. Through massive corpus training, word vectors with wide applicability are obtained. At the same time, it can be optimized in specific tasks, which greatly improves the experimental effect. The BERT model is a pretrained model based on multitask learning on the basis of bidirectional deep transformer. A major structural feature of BERT model is that it adopts a two-way self attention mechanism, which can take into account the context information in this encoder and left and right adjacent encoders at the same time.

The input layer of BERT consists of three layers, and the values of the three layers are added as the output of transformer. The token layer is used to embed word vectors, the segment layer is used to retrieve the sentences to which the word belongs, and the position layer is used to locate the position of the word in the word vector sequence. Taking Figure 4 as an example, when a piece of text data is input into the model, all words and the identification between internal segments are embedded through the token layer, so the composition of each word can be clearly distinguished in the token sequence. Then, the words and paragraph identifiers belonging to the same paragraph in the text are marked as the same category in the segment layer, that is, two sub-sentences in the sentence pair or multiple sub-sentences in the long sentence can be distinguished. Finally, in the position layer, each word and paragraph are identified and numbered consecutively as position embedding, which also provides a basis for the model to learn word order information.

With its strong transferring ability, BERT can adapt to the transfer learning of different downstream tasks. Therefore, it can be fine-tuned on the basis of the pretrained model, build word vectors in the way of transfer learning, improve the model performance and generalization ability, and reduce the demand for large-scale labeled data.

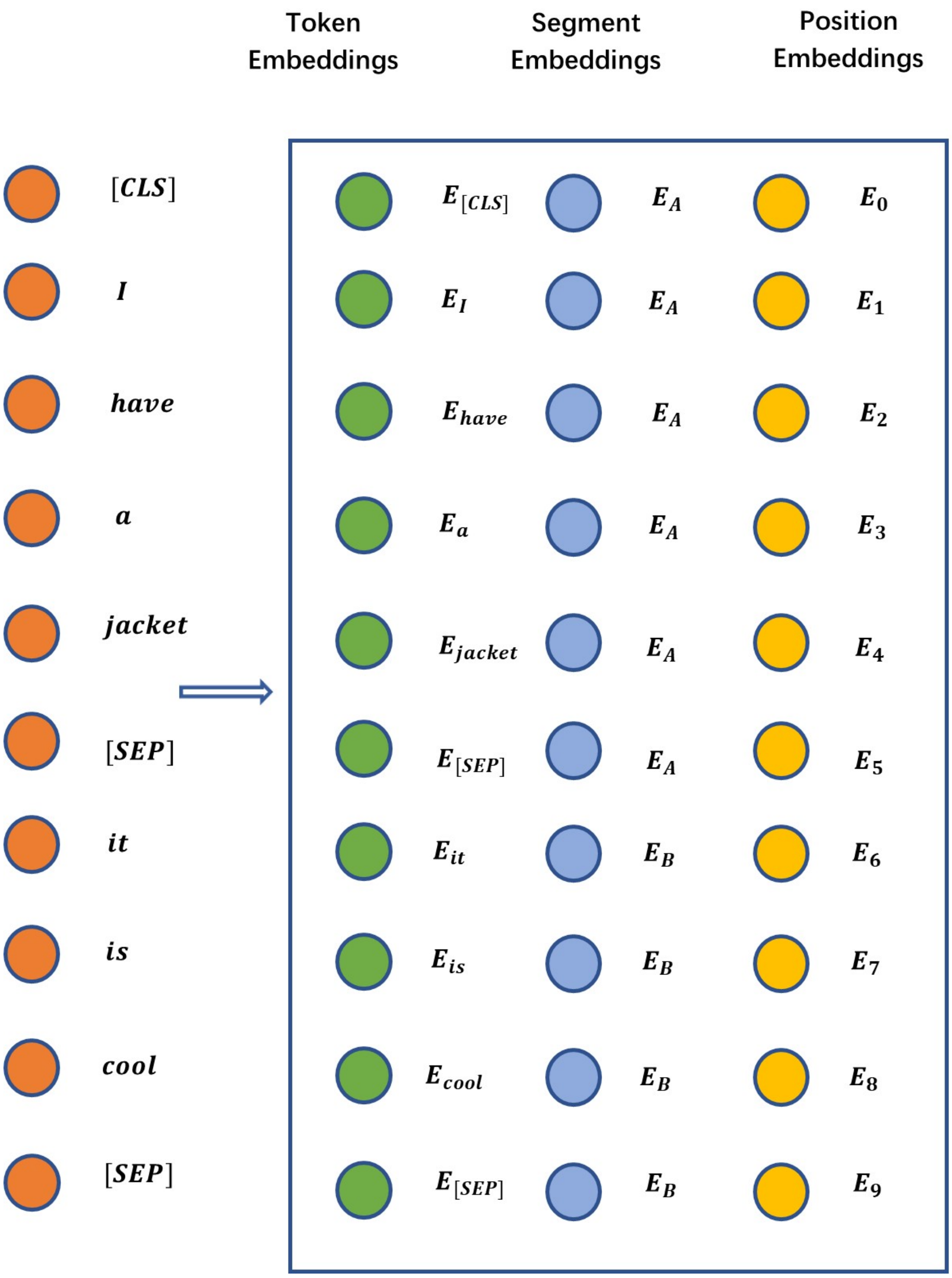

Figure 4. Input layer of BERT.

3.5. Text Similarity Calculation Based on Topic Modeling and Transfer-Learning-Based Text Vectorization (TTTV)

Deep learning involves a large number of operations. At the same time, the training of a neural network model also needs the supply of big data [48]. If the training task is initialized from scratch and solved iteratively every time, it will not only be inefficient, but also cause a waste of computing resources [49]. Therefore, this paper adopts the transfer learning method based on a pretrained model for training. The pretrained model was trained by large datasets, and we can directly use its corresponding structure and weight. In order to adapt to specific tasks, we can freeze some of its network layers, and then train the unfrozen parts on a small scale based on specific task datasets, and dynamically adjust the parameters to adapt to the characteristics of downstream tasks.

TTTV can be used to measure the similarity of texts with different contents belonging to the same topic. The first is keyword extraction. The text data is modeled by LDA belonging to the topic model, the mapping relationship among sentence space, topic space, and word space is constructed, and the number of keywords to be extracted is set. The P (precision), R (recall), and F1 (F1 measure) are used as measurement indicators, i.e.,

$$\mathrm{P} = \frac{|T_1| \cap |T_2|}{|T_1|}. \tag{8}$$

$$\mathrm{R} = \frac{|T_1| \cap |T_2|}{|T_2|}. \tag{9}$$

$$\mathrm{F1} = 2\mathrm{PR}/(\mathrm{P} + \mathrm{R}). \tag{10}$$

where T_1 represents the extracted keyword set, and T_2 represents the keyword set marked by the original text. We comprehensively consider the above indicators and select the topic model parameters with the highest extraction performance for subsequent operations. After that, text vectorization is carried out, and the text vector is obtained through the transfer learning of the BERT model. We select the text vectors x_1 and x_2 generated by pairs of text data d_1 and d_2 belonging to the same topic but different contents, calculate their cosine distance, and define it as the similarity of the text to which they belong, i.e.,

$$\mathrm{similarity}(d_1, d_2) \overset{\mathrm{def}}{=} \mathrm{cosine}(x_1, x_2) = \frac{x_1 \cdot x_2}{||x_1|| \cdot ||x_2||} \tag{11}$$

See Algorithm 2 for the specific process.

Algorithm 2 Text similarity based on TTTV.

```
Input: text set $D$; number $K$ of keywords.
Output: similarity $(d_1, d_2); \forall d_1, d_2 \in D$.
for each $d_i \in$ range $(D)$ do
    for each $k \in$ range $(K)$ do
        extract $keyword_k$ of $d_i$ through LDA.
    end for
    generate vector $x_i$ of text $d_i$.
    based on set $\{keyword_k\}$ through BERT.
end for
cosine$(x_1, x_2) \leftarrow \frac{x_1 \cdot x_2}{||x_1|| \cdot ||x_2||}$.
similarity$(d_1, d_2) \leftarrow$ cosine$(x_1, x_2)$.
return similarity$(d_1, d_2)$.
```

4. Experiment

4.1. Experimental Environment and Data

The experimental environment is an Intel i7 processor with 16 GB memory, the programming language is Python, the Jieba is used for text word segmentation, and the Gensim is used for text vectorization training. The data used in the experiment is Sogou laboratory news dataset, including 18 subjects in international, sports, society, entertainment, and other fields, with a total of 1.02 GB news text data.

4.2. Experimental Setup and Analysis of Experimental Results

In this paper, cosine distance is used as the evaluation index to calculate the similarity between texts, to evaluate whether the vectors generated by different text vectorization methods can accurately judge whether the contents of any two texts belong to the same topic, and a comparative experiment is set up. In the keyword extraction stage, the traditional frequency-based TFIDF method and the topic model method used in this paper are used to extract the keywords of the text, to compare the impact of the extraction results on the performance of the final generated text vector. The topic model method can be compared in two cases: latent semantic analysis model (LSA) and latent Dirichlet allocation model (LDA). In the text vector generation stage, the word2vec method and the para2vec method are used for text vectorization, respectively. The word2vec method can be compared in the continuous bag of words (CBOW) model and skip-gram model, and the para2vec method can be compared in the distributed memory model (DM) and distributed bag of words model (DBOW). Finally, it is compared with the transfer learning method using the BERT model.

The experimental results are shown in Tables 2 and 3 and Figures 5–8. The first is text keyword extraction. We set the keyword extraction number to 3–18, respectively, and calculate the P (precision), R (recall), and F1 (F1 measure) to evaluate the extraction effect.

Table 2. Keyword extraction.

Method	**K = 3**			**K = 4**			**K = 5**			**K = 6**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
TFIDF	0.67	0.11	0.19	0.75	0.17	0.27	0.80	0.22	0.35	0.83	0.28	0.42
LSA	0.67	0.11	0.19	0.50	0.11	0.18	0.60	0.17	0.26	0.67	0.22	0.33
LDA	0.66	0.11	0.19	0.75	0.17	0.27	0.60	0.17	0.26	0.67	0.22	0.33
Method	**K = 7**			**K = 8**			**K = 9**			**K = 10**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
TFIDF	0.71	0.28	0.40	0.63	0.28	0.38	0.56	0.28	0.37	0.50	0.28	0.36
LSA	0.71	0.28	0.40	0.75	0.33	0.46	0.67	0.33	0.44	0.60	0.33	0.43
LDA	0.86	0.33	0.48	0.75	0.33	0.46	0.78	0.39	0.52	0.70	0.38	0.50
Method	**K = 11**			**K = 12**			**K = 13**			**K = 14**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
TFIDF	0.45	0.28	0.34	0.42	0.28	0.33	0.46	0.33	0.39	0.50	0.39	0.44
LSA	0.55	0.33	0.41	0.50	0.33	0.40	0.46	0.33	0.39	0.43	0.33	0.38
LDA	0.82	0.50	0.62	0.75	0.50	0.60	0.77	0.56	0.65	0.79	0.61	0.69
Method	**K = 15**			**K = 16**			**K = 17**			**K = 18**		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
TFIDF	0.53	0.44	0.48	0.56	0.50	0.53	0.59	0.56	0.57	0.61	0.61	0.61
LSA	0.40	0.33	0.36	0.38	0.33	0.35	0.35	0.33	0.34	0.50	0.50	0.50
LDA	0.73	0.61	0.67	0.75	0.67	0.71	0.71	0.67	0.69	0.72	0.72	0.72

Table 3. Average text similarity.

Keyword Extraction	**word2vec**		**para2vec**		**Pre-Trained**
	CBOW	**Skip-Gram**	**DM**	**DBOW**	**BERT**
TFIDF	0.68	0.67	0.81	0.79	0.82
LSA	0.71	0.72	0.82	0.82	0.83
LDA	0.78	0.79	0.83	0.84	0.86

It can be seen from Figures 5–7 that in terms of extraction accuracy, when the keyword extraction number is less than or equal to six, the extraction precision of TFIDF method and LSA method shows an increasing trend, while the extraction precision of LDA method fluctuates but is relatively stable. When the keyword extraction number is located in [6,12] and [8,17], respectively, the extraction precision of TFIDF method and LSA method begins to decline, and the extraction precision of LDA method tends to be stable between 0.7–0.8. It can be seen that the method using non-LDA model is not suitable for the extraction of more keywords.

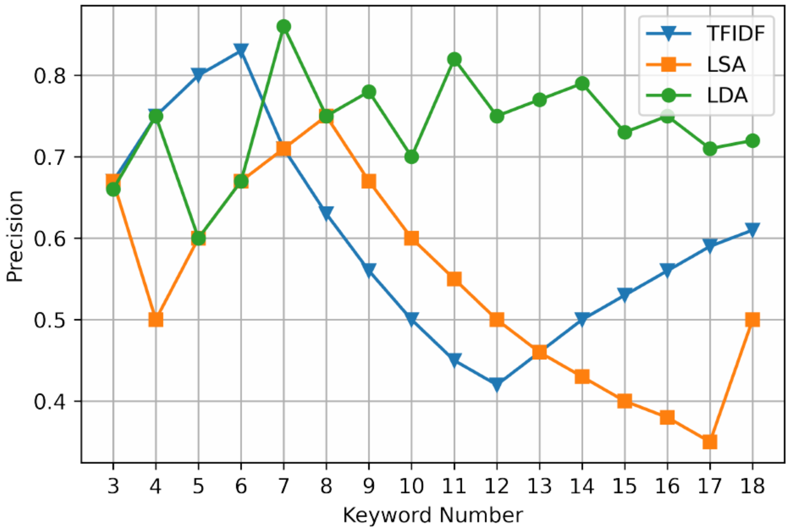

Figure 5. Experimental results of P.

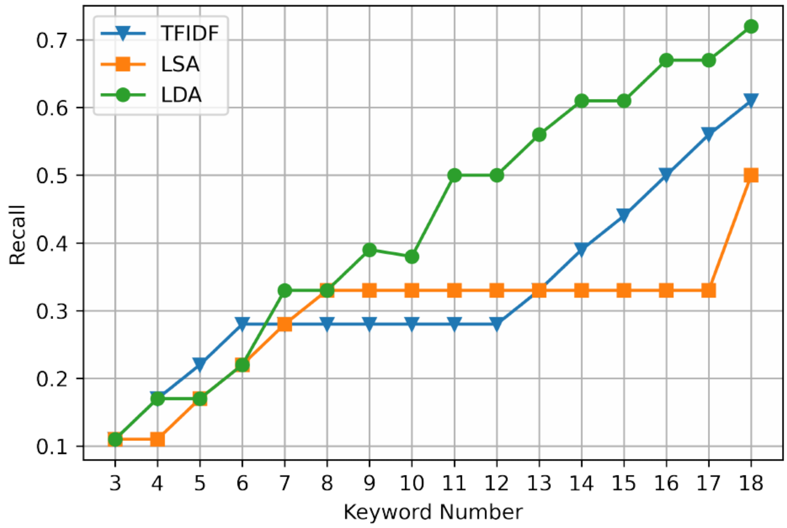

Figure 6. Experimental results of R.

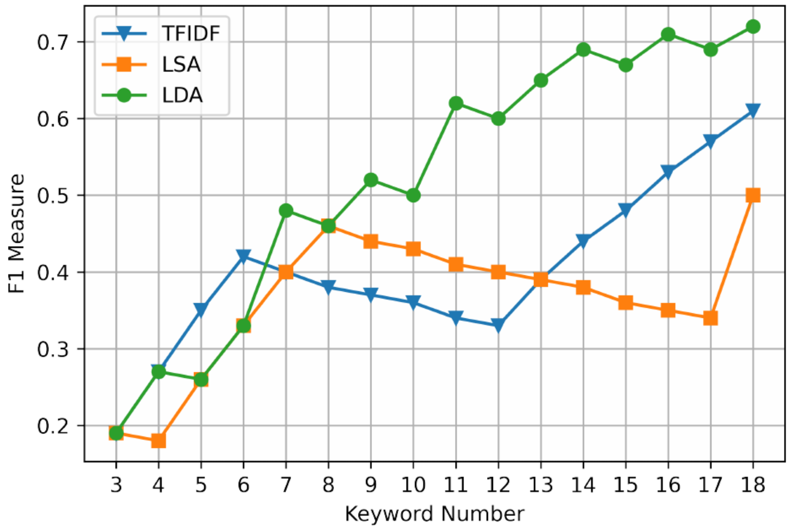

Figure 7. Experimental results of F1.

In terms of extraction recall, when the keyword extraction number is less than or equal to six, the extraction recall of each extraction method shows an increasing trend. When the keyword extraction number is located in the [6,12] interval and [8,17] interval, respectively, the extraction recall of the TFIDF method and the LSA method does not increase or decrease, indicating that the learning ability of non-LDA model for sample features is limited by local optimization. The recall of LDA extraction still shows a gradual upward trend.

Since it is not comprehensive to evaluate the extraction performance by precision or recall alone, the extraction performance can be comprehensively evaluated by its harmonic mean F1 measure. The F1 measure of the TFIDF method and the LSA method continues to decrease in [6,12] and [4,17], respectively, and the overall F1 measure is lower than that of the LDA method. Therefore, text keyword extraction is carried out through the LDA model, the extraction accuracy is high, and the performance is not easily affected by the change of the set number of keywords.

To sum up, we set the keyword extraction number to 18, then carry out the next vectorization operation, measure the similarity of paired text data belonging to the same topic but different contents in the dataset, and sum and average the similarity calculation results to obtain the average text similarity, as shown in Table 3 and Figure 8.

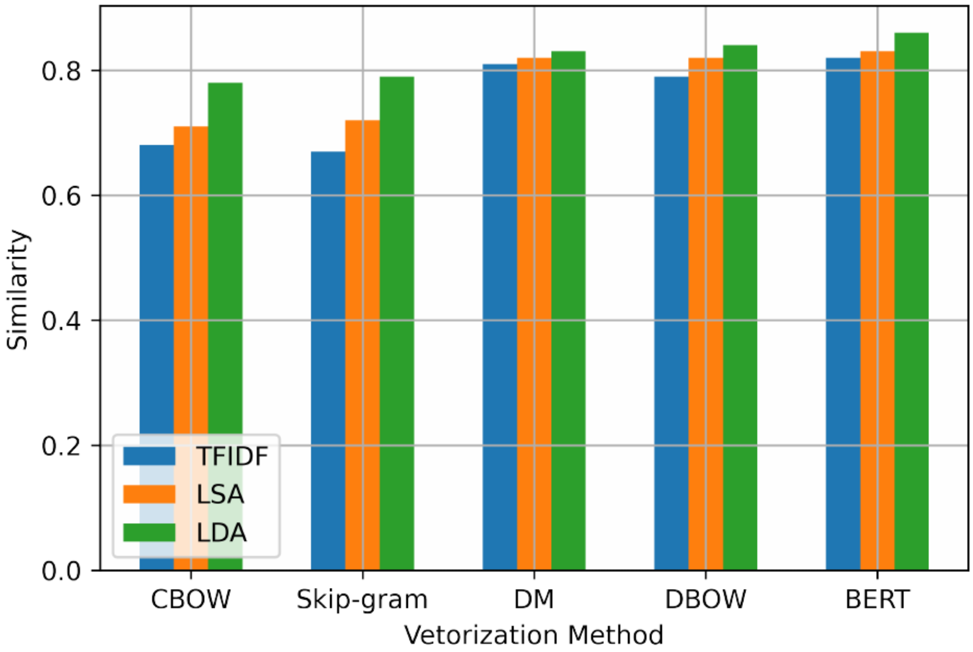

Figure 8. Comparison of text similarity calculation results.

Figure 8 lists five text vectorization methods: CBOW, skip-gram, DM, DBOW, and BERT. On the premise of three keyword extraction methods, TFIDF, LSA, and LDA, we measure the similarity of pairs of text data belonging to the same topic but different contents, and calculate the average text similarity.

It can be seen from the figure that for each vectorization model, the average text similarity calculated by the text vector generated on the premise of selecting LDA for keyword extraction is higher than that calculated based on TFIDF and LSA, respectively. This means that we can more accurately judge the text pairs with similar, or the same, topics. At the same time, we can see that the similarity judgment performance of the text vector generated by DM or DBOW is better than that calculated by CBOW or skip-gram method, which means that the para2vec method based on paragraph vector combined with word vector can grasp the text topic information more accurately than the word2vec method considering only word vector.

Compared with the word2vec method, both the para2vec method and the BERT method learn the semantic information in the text more effectively, so the performance is significantly improved when judging the text similarity. When comparing para2vec with BERT, it is found that although BERT improves the similarity results, the improvement range is not very large. This is because the BERT model is pretrained through a large amount of corpus, so it focuses more on the learning of general features, while para2vec is more suitable for the local feature learning of data with appropriate scale, but it is not good at learning general features. In other words, these two models have their own advantages and disadvantages, and the scalability brought by BERT with transfer learning will be an important aspect to make up for its shortcomings. In addition to the differences in model structure, another reason for this insignificant difference is that the dataset itself is relatively clustered and stored through similar topics, so it does not create a more general data distribution, that is, whether the texts of similar topics are clustered in the same round of tests is completely random. Therefore, the performance superiority of the model proposed in this paper is not obvious, although it does improve the experimental results.

Through comparison, the average text similarity results calculated by the text vector generated by the LDA topic model combined with the BERT pretrained model are higher than the above comparative experiments, that is, this method can judge the text pairs with similar, or the same, topics relatively more accurately.

5. Conclusions

In order to measure the similarity of text pairs with similar, or the same, topic in news texts, this paper proposes a text vectorization method combining topic model and transfer learning. The text data is transformed into vectors, and the cosine distance is used as the measurement index to calculate the similarity. By setting up comparative experiments, it is proved that this method can obtain higher results when calculating the similarity of text pairs with similar, or the same, topic, which means that it can more accurately judge whether the text pair belongs to the same subject.

There is still much room for improvement for such methods, such as considering the mapping relationship between vector dimension and semantic information, or adding additional feature weights to text paragraphs. This is also a collection of methods to be gradually improved in the next step, to facilitate more accurate semantic analysis and research of the text. In addition to the BERT model, there are many available pretrained models, such as RoBERTa [50], ALBERT [51], and SpanBERT [52]. Adding these models to the comparative experiment to study the performance differences in the process of transfer learning will become an entry point in our subsequent research.

Author Contributions: This paper was completed by five authors. Their contributions are as follows: Conceptualization, X.Y. and K.Y.; methodology, K.Y. and T.C.; software, X.Y. and K.Y.; validation, X.Y., K.Y. and T.C.; formal analysis, K.Y.; investigation, T.C.; resources, X.Y.; data curation, M.C. and L.H.; writing—original draft preparation, X.Y., K.Y., T.C. and L.H.; writing—review and editing, K.Y. and T.C.; visualization, K.Y. and L.H.; supervision, X.Y. and T.C.; project administration, M.C.; funding acquisition, M.C. and X.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was funded by National Natural Science Foundation of China (No. 61803035), Beijing Intelligent Logistics System Collaborative Innovation Center (No. BILSCIC-2019KF-08).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jeffrey, C. South Online resources for news about toxicology and other environmental topics. *Toxicology* **2001**, *157*, 153–164.
2. Macskassy, S.A.; Hirsh, H.; Banerjee, A.; Dayanik, A.A. Converting numerical classification into text classification. *Artif. Intell.* **2003**, *143*, 51–77. [CrossRef]
3. Qi, D.; Liu, X.; Yao, Y.; Zhao, F. Numerical characteristics of word frequencies and their application to dissimilarity measure for sequence comparison. *J. Theor. Biol.* **2011**, *276*, 174–180.
4. Kang, X.; Ren, F.; Wu, Y. Exploring latent semantic information for textual emotion recognition in blog articles. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 204–216. [CrossRef]

5. Tan Z.; Chen, J.; Kang, Q.; Zhou, M.C.; Sedraoui, K. Dynamic embedding projection-gated convolutional neural networks for text classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *99*, 1–10. [CrossRef]
6. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
7. Le, Q.; Mikolov, T. Distributed Representations of Sentences and Documents. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014.
8. Kim, D.; Seo, D.; Cho, S.; Kang, P. Multi-co-training for document classification using various document representations: Tf–idf, lda, and doc2vec. *Inform. Sci.* **2019**, *477*, 15–29. [CrossRef]
9. Gómez-Adorno, H.; Posadas-Durán, J.P.; Sidorov, G.; Pinto, D. Document embeddings learned on various types of n-grams for cross-topic authorship attribution. *Computing* **2018**, *100*, 741–756. [CrossRef]
10. Zhang, Y.; Xu, B.; Zhao, T. Convolutional multi-head self-attention on memory for aspect sentiment classification. *IEEE/CAA J. Autom. Sin.* **2020**, *7*, 1038–1044. [CrossRef]
11. Liu, H.; Chatterjee, I.; Zhou, M.C.; Lu, X.S.; Abusorrah, A. Aspect-based sentiment analysis: A survey of deep learning methods. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 1358–1375. [CrossRef]
12. Lan, D.; Buntine, W.; Jin, H. A segmented topic model based on the two-parameter poisson-dirichlet process. *Mach. Learn.* **2010**, *81*, 5–19.
13. Yang, Y.; Liu, Y.; Lu, X.; Xu, J.; Wang, F. A named entity topic model for news popularity prediction. *Knowl.-Based Syst.* **2020**, *208*, 106430. [CrossRef]
14. Buiu, C.; Dnil, V.R.; Rdu, C.N. Mobilenetv2 ensemble for cervical precancerous lesions classification. *Processes* **2020**, *8*, 595. [CrossRef]
15. Shin, S.J.; Kim, Y.M.; Meilanitasari, P. A holonic-based self-learning mechanism for energy-predictive planning in machining processes. *Processes* **2019**, *7*, 739. [CrossRef]
16. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2019**, arXiv:1810.04805.
17. Lai, P.T.; Lu, Z. Bert-gt: Cross-sentence n-ary relation extraction with bert and graph transformer. *arXiv* **2021**, arXiv:2101.04158.
18. Abdulnabi, I.; Yaseen, Q. Spam email detection using deep learning techniques. *Procedia Comput. Sci.* **2021**, *184*, 853–858. [CrossRef]
19. Boncalo, O.; Amaricai, A.; Savin, V.; Declercq, D.; Ghaffari, F. Check node unit for ldpc decoders based on one-hot data representation of messages. *Electron. Lett.* **2018**, *51*, 907–908. [CrossRef]
20. Wu, L.; Hoi, S.; Yu, N.; Declercq, D.; Ghaffari, F. Semantics-preserving bag-of-words models and applications. *IEEE Trans. Image Process.* **2010**, *19*, 1908–1920.
21. Lei, W.; Hoi, S. Enhancing bag-of-words models with semantics-preserving metric learning. *IEEE Multimed.* **2011**, *18*, 24–37.
22. Bengio, Y.; Ducharme, R.; Vincent, P.; Jauvin, C. A neural probabilistic language model. *J. Mach. Learn. Res.* **2003**, *3*, 1137–1155.
23. Ahn, G.; Lee, H.; Park, J.; Sun, H. Development of indicator of data sufficiency for feature-based early time series classification with applications of bearing fault diagnosis. *Processes* **2020**, *8*, 790. [CrossRef]
24. Deerwester, S.; Dumais, S.T.; Furnas, G.W.; Landauer, T.K.; Harshman, R. Indexing by latent semantic analysis. *J. Assoc. Inf. Sci. Technol.* **2010**, *41*, 391–407. [CrossRef]
25. Hofmann, T. Unsupervised learning by probabilistic latent semantic analysis. *Mach. Learn.* **2001**, *42*, 177–196. [CrossRef]
26. Ozsoy, M.G.; Alpaslan, F.N.; Cicekli, I. Text summarization using latent semantic analysis. *J. Inf. Sci.* **2011**, *37*, 405–417. [CrossRef]
27. Yong, W.; Hu, S. Probabilistic latent semantic analysis for dynamic textures recognition and localization. *J. Electron. Imaging* **2014**, *23*, 063006.
28. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent dirichlet allocation. Advances in Neural Information Processing Systems 14. In Proceedings of the Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, Vancouver, BC, Canada, 3–8 December 2001.
29. Kang, H.J.; Kim, C.; Kang, K. Analysis of the trends in biochemical research using latent dirichlet allocation (lda). *Processes* **2019**, *7*, 379. [CrossRef]
30. Chao, C.; Zare, A.; Cobb, J.T. Partial membership latent dirichlet allocation. *IEEE Trans. Image Process.* **2015**, *99*, 1.
31. Biggers, L.R.; Bocovich, C.; Ca Pshaw, R.; Eddy, B.P.; Etzkorn, L.H.; Kraft, N.A. Configuring latent dirichlet allocation based feature location. *Empir. Softw. Eng.* **2014**, *19*, 465–500. [CrossRef]
32. Jia, Z. A topic modeling toolbox using belief propagation. *J. Mach. Learn. Res.* **2012**, *13*, 2223–2226.
33. Zhu, X.; Jin, X.; Jia, D.; Sun, N.; Wang, P. Application of data mining in an intelligent early warning system for rock bursts. *Processes* **2019**, *7*, l55. [CrossRef]
34. Yao, L.; Huang, H.; Chen, S.H. Product quality detection through manufacturing process based on sequential patterns considering deep semantic learning and process rules. *Processes* **2020**, *8*, 751. [CrossRef]
35. Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Zettlemoyer, L. Deep Contextualized Word Representations. *arXiv* **2018**, arXiv:1802.05365.
36. Catelli, R.; Casola, V.; Pietro, G.D.; Fujita, H.; Esposito, M. Combining contextualized word representation and sub-document level analysis through bi-lstm + crf architecture for clinical de-identification. *Knowl.-Based Syst.* **2021**, *213*, 106649. [CrossRef]

37. Subramanyam, K.K.; Sangeetha, S. Deep contextualized medical concept normalization in social media text. *Procedia Comput. Sci.* **2020**, *171*, 1353–1362. [CrossRef]
38. Cen, X.; Yuan, J.; Pan, C.; Tang, Q.; Ma, Q. Contextual embedding bootstrapped neural network for medical information extraction of coronary artery disease records. *Med Biol. Eng. Comput.* **2021**, *59*, 1111–1121. [CrossRef]
39. Feng, C.; Rao, Y.; Nazir, A.; Wu, L.; He, L. Pre-trained language embedding-based contextual summary and multi-scale transmission network for aspect extraction—Sciencedirect. *Procedia Comput. Sci.* **2020**, *174*, 40–49. [CrossRef]
40. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Llion Jones, L.; Aidan, N.; Gomez, A.N.; Kaiser, L. Attention Is All You Need. In Proceedings of the 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
41. Shan, Y.A.; Hl, B.; Sk, B.; Lx, A.; Jx, A.; Dan, S.B.; Lei, X.; Dong, Y. On the localness modeling for the self-attention based end-to-end speech synthesis. *Neural Netw.* **2020**, *125*, 121–130.
42. Mo, Y.; Wu, Q.; Li, X.; Huang, B. Remaining useful life estimation via transformer encoder enhanced by a gated convolutional unit. *J. Intell. Manuf.* **2021**, *2*, 1997–2006. [CrossRef]
43. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. *Improving Language Understanding by Generative Pre-Training*; OpenAI: San Francisco, CA, USA, 2018.
44. Yao, C.; Cai, D.; Bu, J.; Chen, G. Pre-training the deep generative models with adaptive hyperparameter optimization. *Neurocomputing* **2017**, *247*, 144–155. [CrossRef]
45. Chan, Z.; Ngan, H.W.; Rad, A.B. Improving bayesian regularization of ann via pre-training with early-stopping. *Neural Process. Lett.* **2003**, *18*, 29–34. [CrossRef]
46. Sun, S.; Liu, H.; Meng, J.; Chen, C.; Yu, Y. Substructural regularization with data-sensitive granularity for sequence transfer learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 2545–2557. [CrossRef] [PubMed]
47. Ohata, E.F.; Bezerra, G.M.; Chagas, J.; Neto, A.; Albuquerque, A.B.; Albuquerque, V.; Filho, P.P.R. Automatic detection of COVID-19 infection using chest x-ray images through transfer learning. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 239–248. [CrossRef]
48. Luo, X.; Li, J.; Chen, M.; Yang, X.; Li, X. Ophthalmic diseases detection via deep learning with a novel mixture loss function. *IEEE J. Biomed. Health Inform.* **2021**, *25*, 3332–3339. [CrossRef] [PubMed]
49. Luo, X.; Sun, J.; Wang, L.; Wang, W.; Zhao, W.; Wu, J.; Wang, J.H.; Zhang, Z. Short-term wind speed forecasting via stacked extreme learning machine with generalized correntropy. *IEEE Trans. Ind. Inf.* **2018**, *14*, 4963–4971. [CrossRef]
50. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
51. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A Lite Bert for Self-Supervised Learning of Language Representations. *arXiv* **2019**, arXiv:1909.11942.
52. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Levy, O. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]