ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

A FastText Classification Model Based on Simbert Data Augmentation

Haitao Wang[1, a, *], Jiaxing Fan[1, b]

[1] College of Computer Science and Technology, Henan Polytechnic University, Jiaozuo, 454000, China

[a]jz_wht@126.com, [b]212009010029@home.hpu.edu.cn

Abstract

FastText is a text classification and word training tool launched by Facebook. Its biggest feature is that it greatly reduces the classification time while ensuring accuracy. However, when the training set is too small, the FastText classification model is prone to overfitting during the classification process, resulting in a decline in the classification accuracy. Data augmentation technology can effectively expand the size of the training set using existing data without introducing external data, thereby improve the performance of the text classification model and solve the problem of overfitting of the classification model caused by too small samples or uneven sample distribution. This paper proposed a data augmentation algorithm based on Simbert to improve the performance of the FastText classification model. Firstly, use the Simbert generative model to augment the sample. Then calculate the similarity between the original sample and the generated sample, and select the top K most similar samples. Finally, the original sample and the generated sample are merged into a new training set as the input of the classification model for classification. Experimental results show that FastText classification model performance improved on four publicly available datasets.

Keywords

Data augmentation; Text classification; FastText; Simbert.

1. INTRODUCTION

Text is one of the important ways for users to communicate and exchange information using the Internet. Every day, tens of billions of text data are generated on different Internet platforms. In the face of a large number of generated texts, only relying on manual processing usually faces huge challenges[1].As one of the basic tasks in the field of NLP (natural language processing), text classification is an activity that people divide text into groups according to certain standards for certain needs. In recent years, with the rapid development of the deep learning field, a number of new deep learning classification models represented by the FastText [2] model have emerged. Compared with CNN(Convolutional Neural Network)[3]and RNN(Recurrent Neural Network)[4], FastText has the advantages of faster training and testing while maintaining accuracy, and does not require pre-trained word vectors. However, the classification effect of FastText is inseparable from the size and quality of the corpus. When the training set is too small, the model selects too few features of the text, which may easily lead to overfitting of the model. The problem of overfitting is one of the important research areas in deep learning. In many studies, dropout[5], batch normalization[6] are considered as good methods to overcome overfitting. When the number of training set data samples is too small, the above methods cannot play a corresponding role.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

Data augmentation technology generates new data by regularly or irregularly transforming the original data, and merges it into the original data set to achieve the purpose of enriching the data set. This method has been successfully applied to the field of computer vision [7].In this paper, we introduce and improve the Simbert model in the classification process to improve the number and quality of samples, thereby improving the generalization ability of FastText classification model. The content of our work is as follows:

1. We used the samples in the original training set to generate similar sentences through the Simbert model, increasing the number of samples in the training set.

2. In order to ensure the quality of the generated text, we used the text similarity algorithm to calculate the similarity between the original text and the generated text, and selected the top K most similar samples.

3. We combined the original text and the augmented samples into a new training set as the input of the FastText classification model to perform the classification task, and counted the classification results.

2. RELATED WORK

Data augmentation technology was originally applied in the field of computer vision, by flipping, cropping, rotating, scaling, shifting, style conversion [8], RGB channel adjustment [9], etc., to enrich and balance image datasets and improve model performance. In recent years, under the influence of data augmentation methodologies in the field of computer vision, new text augmentation methods have been proposed.

Data augmentation based on deep learning is a new method emerging in recent year. Creating new sentences using the trained language model for data augmentation. Kobayashi [10] uses a pre-trained language model to predict the top k words replaced in the original text, and generate k samples. Fadaee et al [11] proposed to generate new sentences containing rare words in a newly created context, and applied them to low-resource neural machine translation with good results .Wu et al [12] proposed conditional BERT context augmentation. By introducing a new conditional mask language model to transform BERT into c-BERT (conditional BERT), it turns out that well-trained conditional BERT can be used for context augmentation. Kafle et al [13] used task-specific heuristic rules and LSTM(Long Short Term Memory)-RNN LMs to create new sentences. Anaby-Tavor et al [14] proposed a text augmentation technique based on the GPT architecture, which requires pre-training the data augmentation model in a large amount of text, so that the model can capture the structure of the language and generate coherent sentences. Hu et al [15] proposed the VAEHD model. The model fuses a variable auto-encoder with an ensemble encoder, so that it can learn latent explanatory features from text and autonomously generate more discriminative data based on a given sentiment and tense. Fedus et al [16]proposed a GANs network suitable for text generation, which can train the model to generate high-quality samples. Zhang et al [17] proposed a data augmentation method DAGAN based on a generative adversarial model, which improves the performance of the classifier in GAN to learn classification boundaries, so that new data can be generated by category.

The advantages of data augmentation methods based on deep learning lie in lower cost, faster speed, more authenticity, and can effectively eliminate the imbalance of data in the original data set. The downside is that training a good language model usually takes a lot of time.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

3. MODEL STRUCTURE

In this paper, we proposed a FastText classification model based on Simbert data augmentation. The model structure is shown in Figure 1. It consists of three parts: Simbert pre-trained model, text similarity calculation, FastText classification model.

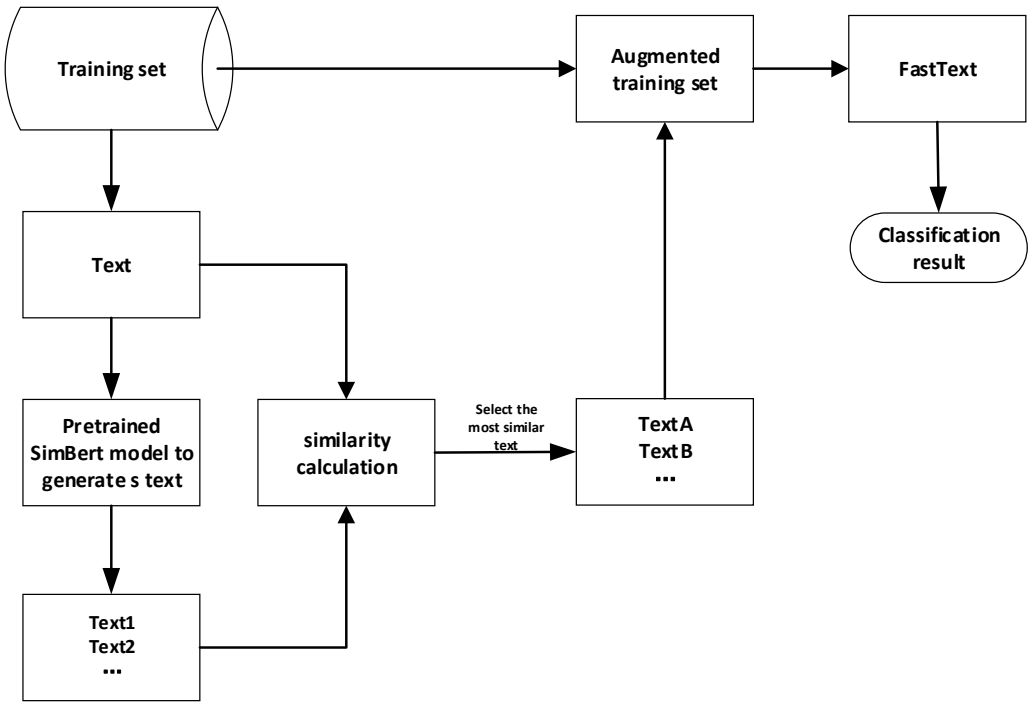

Figure 1. FastText classification model process based on Simbert

3.1. Simbert pre-trained model

Simbert[18] is a generative language model that integrates generation and retrieval based on the Bert model and adopts the UniLM[19] model proposed by Microsoft. The Simbert training process is shown in Figure 2. Simbert belongs to supervised training. The training corpus is similar sentence pairs collected by itself, and the Seq2Seq part is constructed by predicting the similar sentence generation task of another sentence.

UniLM is a Transformer model that combines NLU (Natural Language Understanding) and NLG (Natural Language Generation) capabilities. It jointly pre-trains multiple disparate language models with a common goal through a special Attention Mask method, and implements parameter sharing for different language models through a single Transformer model during the training process and mainly trained jointly on three language models, Bidirectional LM, Unidirectional LM and Seq2Seq LM.

In Simbert training, different sentences in similar sentence pairs are separated by [SEP] identifiers, and on this basis, a special Attention Mask method is used, so as to have the ability of NLG. In addition, Simbert also added random [MASK] to the input, so that the model can do the MLM task during the training process, and the MLM task can train the NLU capability of the model.

The Simbert training process is as follows:

1) If A and B are a group of similar sentences, then in the same batch, add [CLS] A[SEP] B[SEP] and [CLS] B[SEP] A[SEP] to training to do the task of generating similar sentences , which is Seq2Seq.

2) Use the [CLS] vector in the entire batch to get a sentence vector matrix $V \in Rb \times dV \in Rb \times d$ (b is batch_size, d is hidden_size).

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

3) After l_2 regularization in the hidden_size dimension, obtaining the regular matrix $\bar{V}$. Do the inner product in pairs to get the similarity matrix $\tilde{V}\tilde{V}^{\mathrm{T}}$, then multiply by a scale, and mask the diagonal part.

4) Finally, softmax is performed on each line, and the target label of each sample is the similar sentence of A and B.

Figure 2. Simbert model training process

3.2. Text similarity calculation

The cosine theorem, also known as the cosine similarity theorem, uses the cosine value of two angles in a vector space as a measure of the difference between two individuals. The closer the cosine value is to 1, the closer the included angle is to 0 degrees, the more similar the two vectors are. The value range of the cosine of the included angle is [-1,1]. Text vectorization is trained using the word2cev model.

Word similarity calculation. two-dimensional vector A(x_1, y_1) and B(x_2, y_2) the calculation formula is as follows:

$$\cos\theta = \frac{x_1 x_2 + y_1 y_2}{\sqrt{x_1^2 + y_1^2}\sqrt{x_2^2 + y_2^2}} \tag{1}$$

Text similarity calculation. N-dimensional vector, A $(x_{11}, x_{21}, x_{31}, \dots, x_{n1})$ and B$(x_{21}, x_{22}, x_{23}, \dots, x_{21})$, the calculation formula is as follows:

$$\cos(\theta) = \frac{\sum_{k=1}^{n} x_{1k} x_{2k}}{\sqrt{\sum_{k=1}^{n} x_{1k}^2}\sqrt{\sum_{k=1}^{n} x_{2k}^2}} \tag{2}$$

3.3. FastText classification model

3.3.1 Model structure

The FastText model has three layers: input layer, hidden layer, and output layer (Hierarchical Softmax), as shown in Figure 3. The input layer is the word represented by multiple vectors, the output layer is a specific category, and the hidden layer is the superposition average of multiple word vectors. The execution process is as follows:

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

After each sentence is preprocessed, each word is vectorized, and the average value of all word vectors is calculated in the hidden layer using formula (3).

$$\mathrm{h}_{text} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{3}$$

Calculate the probability distribution of each category based on the mean value, and select the category corresponding to the maximum value as the result.

Update the loss function by stochastic gradient descent, then update the model parameters W_o. The forward-propogation process can be described as formula (4)(5):

$$h = \frac{1}{n}\sum_{i=1}^{n} w_i \tag{4}$$

$$z = \mathrm{sigmoid}(W_o h) \tag{5}$$

4. z is the input vector of the final output layer, W_o represents the weight from the hidden layer to the output layer.

5. Output category. Since the model finally predicts the probability that the article belongs to a certain category, selecting the softmax layer. Equations (6) (7) (8) define the loss function:

$$\hat{y} = \mathrm{softmax}(z) \tag{6}$$

$$CE(y, \hat{y}) = -\sum_{j} y_j \log(\hat{y}_j) \tag{7}$$

$$\mathrm{loss} = \frac{1}{M}\sum_{i=1}^{m} CE(y_i, \hat{y}_i) \tag{8}$$

When there are many categories, the calculation of the softmax layer is time-consuming, so negative sampling is used, that is, several labels other than the current label are selected as negative samples each time, and the probability of occurrence of negative samples is calculated and added to the loss function, as shown in formula (9):

$$\mathrm{loss} = -\frac{1}{M}\sum_{i=1}^{m}\left(\log\sigma(u_o^T h_i) + \sum_{j\sim P(w)}\left[\log\sigma(-u_j^T h_i)\right]\right) \tag{9}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

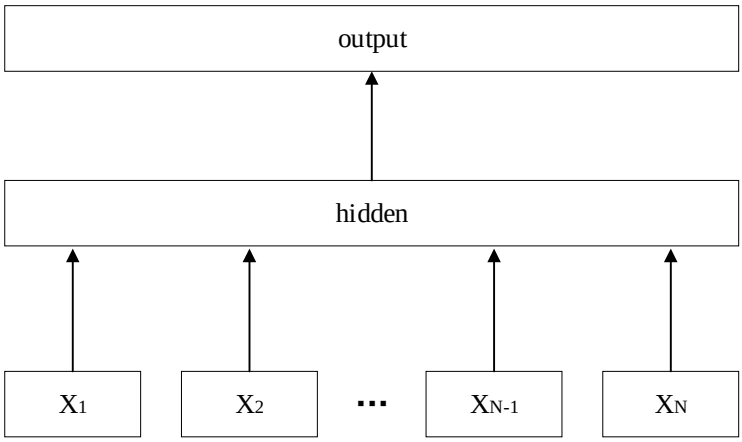

Figure 3. FastText model structure

3.3.2 Hierarchical softmax

To improve runtime, the FastText model uses hierarchical softmax. The purpose of hierarchical softmax is to reduce the computational complexity of the softmax layer. Instead of standard softmax, construct a Huffman tree according to the frequency of categories, the complexity can be reduced from N to $\log N$ by hierarchical softmax. An example of hierarchical softmax is shown in Figure 4.

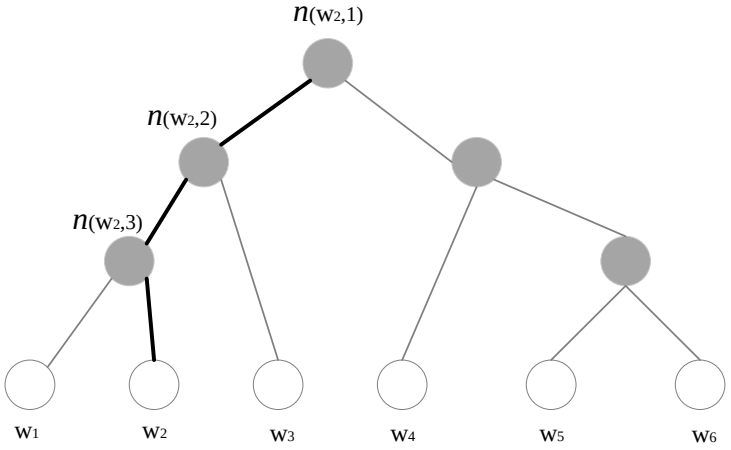

Figure 4. Hierarchical Softmax structure

In the hierarchical softmax model, leaf nodes represent different categories, while non-leaf nodes have corresponding outputs. The probability of the target word $\mathbf{W}$ is calculated by formula (10):

$$p(w) = \prod_{j=1}^{L(w)-1} \sigma\left(\text{sign}(w,j) \cdot \theta_{n(w,j)}^{T} h\right) \tag{10}$$

where $\boldsymbol{\theta}_{\boldsymbol{n}(\boldsymbol{w},\boldsymbol{j})}$ is the vector representation of the non-leaf node $\boldsymbol{n}(\boldsymbol{w},\boldsymbol{j})$; $\boldsymbol{h}$ is the output value of the hidden layer, calculated from the vector of the input word; The $\mathbf{sign}(\boldsymbol{x},\boldsymbol{j})$ function is defined as formula (11):

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

$$\text{sign}(w,j) = \begin{cases} 1, & \text{if } n(w,j+1) \text{ is the left child of } n(w,j) \\ -1, & \text{if } n(w,j+1) \text{ is the right child of } n(w,j) \end{cases} \quad (11)$$

In addition, the sum of the probabilities of all words is 1, as shown in Equation (12):

$$\sum_{j=1}^{n} p(w) = 1 \quad (12)$$

The formula after parameter update is shown in formula (13):

$$\theta_j^{(\text{new})} = \theta_j^{(\text{old})} - \eta\left(\sigma\left(\theta_j^T h\right) - t_j\right)h \quad (13)$$

3.3.3 N-Gram Features

To prevent word order loss, FastText uses character-level n-grams to represent a word. n-gram is an algorithm based on language model. The basic idea is to perform a window sliding operation of size N according to the subsection order of the text content, and finally form a sequence of byte fragments with a window size of N. The specific steps are as follows:

1) First, the N-gram is regarded as a word and represented by an embedding vector.
2) Second, when calculating the hidden layer, add the embedding vector of the N-gram to sum and take the average.

4. EXPERIMENTAL SETUP

4.1. Experimental dataset

Our experiments selected four publicly available datasets as our experimental benchmark datasets. The ratio of training set, validation set, and test set is 6:2:2.

Table 1. Experimental dataset details

Dataset	Class	Average length	Dataset size	Training size
SouGou	6	18	10000	6000
TouTiao	5	41	3000	1800
FuDan	6	16	6000	3600
WeiBo	2	32	8000	4800

4.2. Model parameter settings and experimental environment

The parameter settings in the FastText classification model, the dropout value is set to 0.5, the epoch value is 10, the mini-batch is 128, the pad_size is 32, the learning rate is 1e-3, the size of the hidden layer is set to 256, and the dimension of the word vector is 300. In the Simbert model, num is set to 10, and num is the number of generated ones. Max_len is set to 50, and Max_len is the maximum processing text content length.

The experimental environment settings are shown in Table 2.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

Table 2. Experimental environment settings

Experimental environment	Specific configuration
operating system	Windows 10
CPU	Intel(R) Core(TM) i7-8750H
GPU	NVIDIA Tesla V100
RAM	16 GB
Python	3.7
TensorFlow	1.14
keras	2.3.1
bert4keras	0.7.7
Pytorch	1.9.0

5. RESULTS AND ANALYSIS

In order to comprehensively compare the classification effect of the FastText model before and after data enhancement, the evaluation indicators used in the experimental part are Accuracy and F1 value to measure the classification effect.

$$Accuracy = \frac{TP + TN}{P + N} \tag{14}$$

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{15}$$

5.1. FastText model classification results comparison

We extracted 20%, 40%, 60%, 80%, and 100% of the texts in the four training sets, respectively, using Simbert for one to five-fold augmentation, and performing multiple classification task experiments. Table 3 presents the F1 scores of FastText classification models before and after using data augmentation techniques. From the Table 3 we can see that experimenting with a smaller training set, the classification results of the FastText model are severely distorted before using Simbert data augmentation techniques. The reason is that the training set is too small, which leads to the overfitting problem of the FastText model. With the continuous expansion of the proportion of extracted training sets, the classification results of FastText have gradually improved. After using Simbert data augmentation technology, the overfitting problem of FastText in small data is solved, and the value of F1 is increased by up to 65.22%. The reason is that the Simbert data augmentation technology improves the quantity and quality of the text in the training set, and the FastText classification model can obtain more text features, which in turn greatly improves the generalization ability of the FastText classification model. At the same time, with the continuous improvement of the extraction ratio, the classification results of FastText also continue to rise. The above fully demonstrates the effectiveness of Simbert data augmentation technology on the FastText model.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

Table 3. Classification results of different datasets in FastText model

Train set	FastText	Simbert+FastText
FuDan*20%	24.08%	89.30%
FuDan*40%	82.54%	92.00%
FuDan*60%	88.50%	92.43%
FuDan*80%	90.17%	92.72%
FuDan*100%	90.42%	93.65%
SouGou*20%	60.45%	77.95%
SouGou*40%	73.05%	80.30%
SouGou*60%	78.25%	82.65%
SouGou*80%	78.55%	83.25%
SouGou*100%	81.15%	83.30%
TouTiao*20%	24.37%	72.46%
TouTiao*40%	24.74%	78.52%
TouTiao*60%	24.70%	81.05%
TouTiao*80%	78.27%	83.59%
TouTiao*100%	82.86%	86.42%
WeiBo*20%	51.38%	60.71%
WeiBo*40%	58.19%	61.56%
WeiBo*60%	60.62%	62.69%
WeiBo*80%	61.06%	63.50%
WeiBo*100%	61.50%	65.04%

5.2. The effect of K value on FastText classification results

In order to explore the impact of selecting the top K most similar samples on the performance of the FastText classification model during the Simbert data augmentation process, we conducted experiments with different K values. The classification results are shown in Figure 5. On the whole, as the value of K increases, the performance of the FastText classification model gradually improves, and when k is 1, the increase in the four training sets is the largest. In terms of the growth rate, with the continuous increase of the k value, the value of F1 has slowed down, and even in the SouGou training set and WeiBo training set, there is a trend of flatness and decline. The reason is that there are many original samples in these two training sets, and the advantages of data augmentation technology in the larger training set are limited. Secondly, the more augmented samples are selected, the lower the similarity between the augmented samples and the original samples will be, the quality of the generated samples will be degraded, and the content of the generated samples will not match the original labels, which will affect the performance of the FastText classification model.

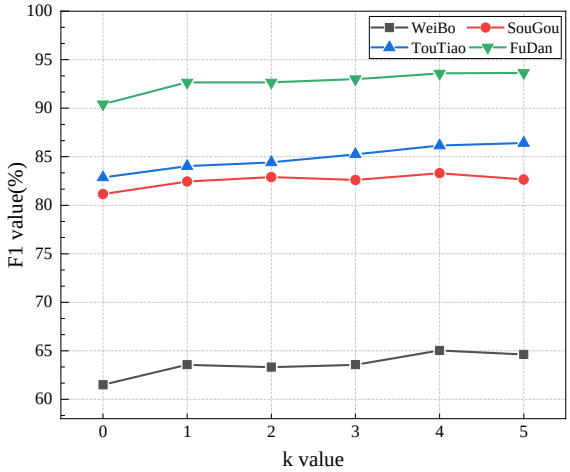

Figure 5. Classification results under different k values

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

5.3. Sample visualization distribution

In order to better demonstrate the quality of the samples generated by the Simbert data augmentation technology, we extracted 100 pieces of data from the four original data sets as test samples, and used the Simbert data augmentation technology to expand the data by 5 times, and visualize the data before and after the augmentation. First, the text is vectorized by doc2vec technology. Then use PCA (Principal Component Analysis) technology to reduce the dimension of the high-dimensional text vector. Finally, two-dimensional images are used to visualize the text. The distribution of test samples and augmented samples is shown in Figure 6. From the figure, we can see that the samples generated by data augmentation are closely connected with the test samples, and most of the samples are more concentrated. At the same time, we can also see that the distribution of some augmented samples is too scattered, because with the increase of the augmentation multiple, the similarity between the original samples and the partially augmented samples decreases. Overall, the quality of the text generated by the Simbert data augmentation technique can be guaranteed.

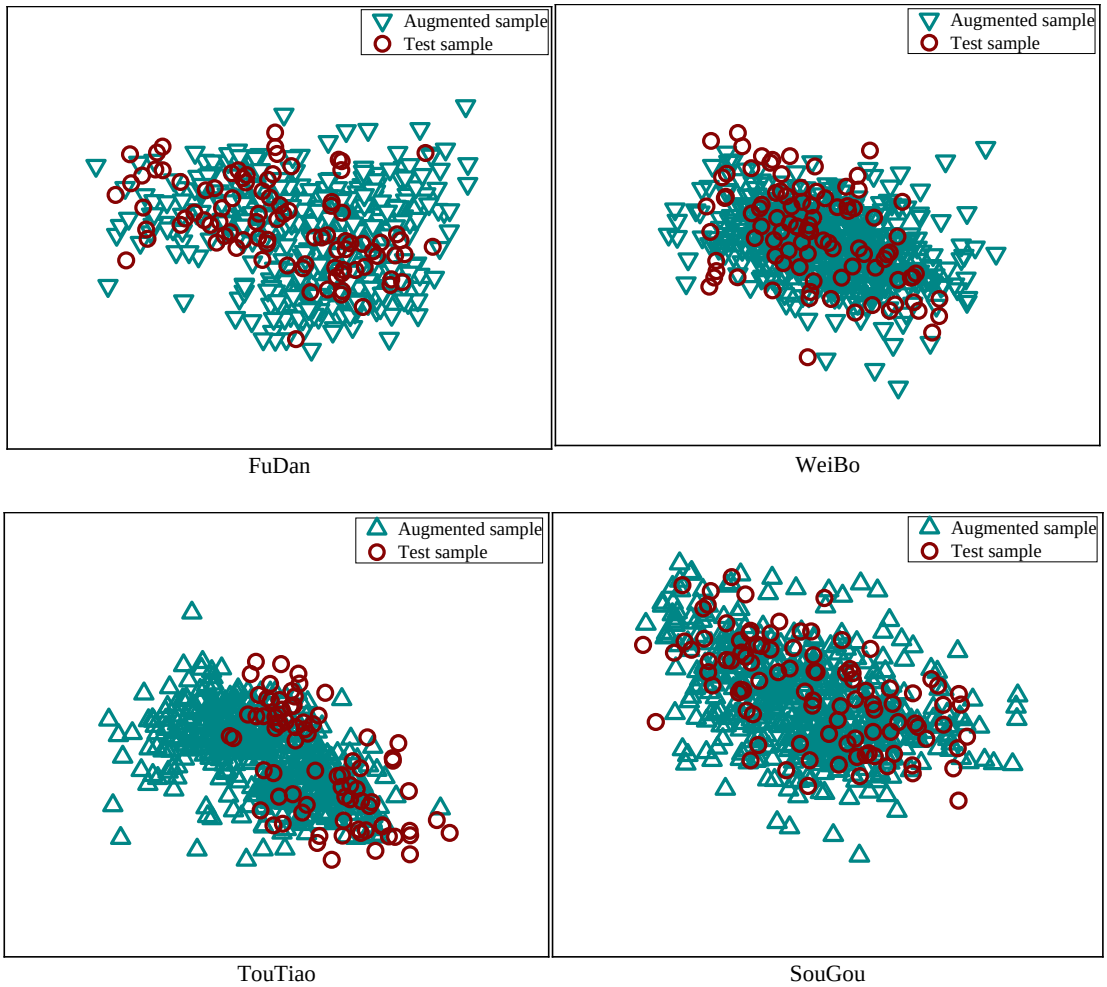

Figure 6. Test sample and augmented sample distribution

6. CONCLUSION AND FUTURE WORK

This paper combines Simbert data enhancement with the FastText classification model, and adds a similarity calculation algorithm to the Simbert enhancement process, improves the quality of the samples generated by the Simbert algorithm, and solves the problem of overfitting of the FastText model due to too few samples, improved the generalization ability of the

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

classification model. In future research, we also need to apply the Simbert data augmentation method to different text classification models to prove that Simbert data augmentation is universal in text classification. At the same time, whether the Simbert data augmentation technology is suitable for long text classification is also a question worth exploring.

ACKNOWLEDGMENTS

This work is support by the National Natural Science Foundation of China (No. 11601129, 61772159).

REFERENCES

[1] Ye, H. J., & Kankanhalli, A. (2017). Solvers' participation in crowdsourcing platforms: Examining the impacts of trust, and benefit and cost factors. The Journal of Strategic Information Systems, 26, 101–117.

[2] Yao T , Zhai Z , Gao B . Text Classification Model Based on fastText[C]// 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS). IEEE, 2020.

[3] Yoon Kim. 2014. Convolutional neural networks for sentence classification. CoRR, abs/1408.5882.

[4] Lai S , Xu L , Liu K , et al. Recurrent Convolutional Neural Networks for Text Classification. 2015.

[5] Loshchilov , I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. arXiv 2016, arXiv:1608.03983.

[6] Pan, S.J.; Yang, Q. A survey on transfer learning. IEEE T rans. Knowl. Data Eng. 2009, 22, 1345–1359. [CrossRef].

[7] He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In 2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence) (pp. 1322–1328). IEEE.

[8] Gatys, L. A., Ecker, A. S., & Bethge, M. (2016). Image style transfer using convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2414–2423).

[9] Krizhevsky A ,Sutskever I ,Hinton G. ImageNet Classification with Deep Convolutional Neural Networks[J]. Advances in neural information processing systems,2012, 25(2).

[10] Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations.In NAACL-HLT.

[11] Fadaee, M.; Bisazza, A.; Monz, C. Data Augmentation for Low-Resource Neural Machine Translation.In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC,Canada, 30 July–4 August 2017; pp. 567–573.

[12] Wu, X.; Lv , S.; Zang, L.; Han, J.; Hu, S. Conditional BERT Contextual Augmentation. In International Conference on Computational Science; Springer: Seoul, Korea, 2019; pp. 84–95.

[13] Kafle, K.; Yousefhussien, M.; Kanan, C. Data augmentation for visual question answering. In Proceedings of the 10th International Conference on Natural Language Generation, Santiago de Compostela, Spain, 4–7 September 2017; pp. 198–202.

[14] Anaby-Tavor, Ateret, et al. "Not Enough Data? Deep Learning to the Rescue!." arXiv preprint arXiv:1911.03118 (2019).

[15] Hu Z, Yang Z, Liang X, et al. Toward controlled generation of text[C]//Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017: 1587-1596.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202301_9(1).0004

[16] William Fedus, Ian Goodfellow, and Andrew M. Dai.2018. MaskGAN: Better text generation via filling in the __. In ICLR.

[17] XiaoFeng Zhang, G. W. (2021). A Survey on the Development of Image Data Augmentation. Computer Science and Application, Vol. 11 No. 02.

[18] Su Jianlin. Have it both ways: Simbert model for fusion retrieval and generation [EB/OL]. [2020-05-18]. https://kexue.fm/archives/7427.

[19] Bao H , Dong L , Wei F , et al. UniLMv2: Pseudo-Masked Language Models for Unified Language Model Pre-Training[J]. 2020.