2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)

2020 IEEE 22nd International Conference on High Performance Computing and Communications | 978-1-7281-7649-9/20/$31.00 ©2020 IEEE | DOI: 10.1109/HPCC-SmartCity-DSS50907.2020.00159

A Novel SMOTE Algorithm based Portrait Model for Programmers

1st Pengyu Yu
College of Computer and Information
Hohai University
Nanjing, China
jadesperwalker@qq.com

2nd Yirui Wu*
College of Computer and Information
Hohai University
Nanjing, China
wuyirui@hhu.edu.cn

3nd Shun Zhao
College of Computer and Information
Hohai University
Nanjing, China
1912603538@qq.com

***Abstract*—With the rapid development of information technology, the scale of software products is getting larger. In order to ensure code quality, it's necessary to collect log information in the continuous integration development process, analyze the behavior of developers, build developer user profile models, and provide personalized suggestions to developers based on profile information. All these processes can be defined as goals of user portrait model. Essentially, user portrait model is designed to model users' coding behaviours based on a large amount of data. However, current methods often suffer from the problem of imbalanced data, which is the core challenge for a successful portrait model. In fact, most parts of log information refer to regular programmers, while only few samples correspond to programmers who are supposed to be improved by suggestions from the portrait model. To solve this problem, we propose to adopt SMOTE Algorithm to deal with the imbalanced log data, which is the core innovation of the proposed method. Experiments show the proposed SMOTE Algorithm based model could accurately classify programmers' personality types and offer suggestions.**

***Index Terms*—Bert-Capsule Network; Developer Portrait Model; SMOTE Algorithm; Imbalanced Data**

I. Introduction

In order to shorten the development cycle while ensuring code quality [1], more and more development teams have begun to turn to the continuous integration development model. In the continuous integration development model [2], the existing development team mainly through the loading of automated tools to compile, package, test, deploy and other integrated operations [3] to ensure the code quality, but rarely write from the code Think about how to ensure code quality from a perspective. At the same time, when the project integration fails, developers often deal with defects directly, and rarely analyze the causes of defects, causing developers to still have the same defect problems [4].

In the continuous progress of the project, developers will leave a large amount of continuous integration log information during the continuous integration development process, mainly including code function log information and defect problem feedback log information submitted by the developer. Compared with the log information in the traditional development process, the continuous integration log information can quickly locate the developer with the problem defect. It is no longer necessary to manually assign the problem to the corresponding developer, avoiding the wrong assignment of the defect problem. In addition, because developers frequently submit code to the main line of the project, they can collect more log information containing developer behavior data, and then analyze the behavior of developers. In the traditional development process, the code cannot be integrated frequently, and the collected log information is far less than the log information in the continuous integration development process. When analyzing the behavior of developers, there will be insufficient data. For this reason, it is necessary to collect continuous integration log information to analyze the behavior of developers.

In order to conduct a comprehensive analysis of the developer's behavior, after obtaining the continuous integration log information, the user profile technology can be used to extract the characteristics of the continuous integration log information from the three dimensions of personality, work activity and work attitude to construct a developer profile The model then obtains the recent work of the developer, and according to the work of the developer, proposes suggestions that can improve the code quality, and helps the developer to reduce code defects. At the same time, the project leader can also understand the working status of the developers in real time according to the portraits of the developers, and help developers with problems in time. Among them, work activity refers to the number of times that developers submit code. The more the number of submissions, the more active the work is. Work attitude refers to the number of times that developers have various defects. The more the number of times, the worse the work attitude. For the personality characteristics of developers, a five-factor personality model can be used to analyze.

When constructing the developer portrait model, if only analyze the behavior of the developer from a single dimension, the portrait will appear to be too one-sided and the actual meaning is not strong. Therefore, when constructing a portrait, it is necessary to analyze the behavior of

978-1-7281-7649-9/20/$31.00 ©2020 IEEE
DOI 10.1109/HPCC-SmartCity-DSS50907.2020.00159

developers from as many dimensions as possible to make the portrait more useful. At the same time, when analyzing the personality of the developer, if the behavior of the developer is analyzed directly based on the keywords in the feedback information and a specific sentence template, the sentence structure for complex emotions may no longer be applicable. When the deep learning algorithm extracts the semantic features in the feedback information [5], it often directly mines the global semantic features [6], ignoring the meaning of the local emotional words in the text [7], resulting in an insufficiently comprehensive character portrait model. Therefore, it is necessary to design a model that can integrate local emotional features with global features to make the constructed character portrait more comprehensive and improve the performance of character classification.

When the developer profile model is built, it can provide the developer with personalized programming suggestions from the two aspects of work activity and work attitude according to the different personality of the developer, which is helpful for the developer to accept the suggestions more easily, and Make timely adjustments during programming to improve code quality. For example, when developer A is not active, has a poor working attitude, and has a negative and pleasant personality, he can ask developer A "You can set an alarm clock to remind yourself to submit code. You can search the Internet when writing code. Some new solutions to solve current problems.

In summary, the user profile technology can be applied to the analysis of developer programming behavior, and the behavior data of developers can be mined from the three dimensions of work activity, work attitude and personality, and the reasons for the defects caused by the developers can be analyzed, and the developers can be constructed. Portrait model. When analyzing the personality of a developer, a deep learning algorithm that combines global features and local emotional features can be designed to mine the emotional semantics in the feedback information.

II. Related Work

In the daily social process, users will leave a lot of behavioral information on the Internet [8], including comments posted by users, topics to participate in discussions, chat records, etc. These behavioral information can not only describe the important behavioral characteristics of users, but also indirectly reflect the psychological activities and personality characteristics of users at that time. By constructing character portraits, it is possible to make predictions about the user's behavior and preferences. At the same time, it can also provide users with personalized recommendations according to their different personalities. When predicting the user's personality, the user's behavior information can be analyzed and predicted based on the five-factor personality model, and then the user's personality characteristics can be obtained.

User portraits [9] are mainly based on the attributes and behaviors of users in real life, abstract tags from multiple dimensions, and restore the real appearance of users as much as possible. After the user is tagged, the company can accurately locate the specific user based on the tag, thereby customizing different strategies for different users and reducing the cost of pushing. At the same time, companies can obtain the user's personality characteristics based on the user's behavior patterns, design products or models that meet this type of user, and then improve the user experience. We believe this technology could be improved by quantity of novel methods, such as edge computing [10], data-driven intelligence [11] or big data processing [12].

The construction of user portraits first needs to collect user attribute data and behavior data [13]. The attribute data of the user is mainly the basic information of the user, and the behavior data of the user is mainly the operation information of the user in daily life. First, the user's attribute data and behavior data are counted to obtain user characteristics. Then, use relevant methods to analyze user characteristics and generate user tags. Finally, a portrait model that can describe the characteristics of users is constructed. After the user portrait is constructed, it is necessary to continue to collect user behavior data, and continue to analyze, continuously improve the user's characteristic dimension, and provide users with more personalized and characteristic services.

According to different usage methods, user portraits can be divided into rule portraits and machine learning algorithm portraits [14]. Rule-based portraits mainly use statistical methods to accurately and comprehensively describe user characteristics. The machine learning algorithm profile mainly uses traditional machine learning and deep learning algorithms to abstract user data, thereby abstracting more condensed labels.

The construction method of user profile can generally be divided into four steps: target analysis, data collection, data modeling and profile construction, as follows:

(1) Goal analysis: establish the goal, significance and expected result of the user portrait construction. User portraits are based on users' real data. By calculating users' personal information and specific behavior attributes, users can be classified and tagged. At the same time, the research focus of user portraits will vary greatly from user to scene, and user behavior may also change with time. Therefore, it is necessary to make a preliminary judgment on the goal, meaning and expected result of the user portrait before establishing the portrait;

(2) Data collection: Conduct comprehensive data collection work. The user's basic attribute information can be collected through the information filled in when the user registers and logs in. The user's behavior information can be obtained according to the user's browsing time, content, and the number of clicks to switch;

(3) Data modeling: Combine the collected data with the usage scenarios of the portrait, and perform multi-dimensional analysis on the data through machine learning algorithms or statistical methods, and calculate the probability of the category, so as to construct a model that meets the actual situation;

(4) Portrait construction: Re-extract the obtained modeling data, abstract multi-level tags that can describe the user, and build a three-dimensional, modeled, and standardized user portrait.

III. The Proposed Method

In this section, we firstly offer introduction to SMOTE algorithm. Afterwards, we describe the process to enhance continuous integration log information. Finally, we propose a novel BERT-Capsule model based on Incremental Learning idea.

A. Introduction to SMOTE Algorithm

The SMOTE (Synthetic Minority Oversampling Technique) algorithm is an oversampling algorithm proposed by Chawla et al., which is mainly used to synthesize new minority samples to achieve class balance. In binary classification experiments, there is often a problem of imbalance between positive and negative sample categories, that is, the number of samples in one type is much larger than the number of samples in the other type. When the model is trained with unbalanced sample categories, it is easy to make the model more inclined to large samples, resulting in lower accuracy and recall of small sample data. The traditional random oversampling algorithm mainly uses simple copying to increase the samples of the minority class, which leads to the similarity between the generated minority class sample and the original data sample, and the classifier is prone to overfitting. The SMOTE algorithm mainly analyzes minority samples, and synthesizes new samples based on the minority samples to add to the data set, thereby eliminating the imbalance of positive and negative sample categories in the original data set.

In the SMOTE algorithm, the k nearest neighbors of the sample x_i are first found from all samples T of the minority class, and a sample $x_i(nm)$ is randomly selected from the k nearest neighbors, and then a random number between 0 and 1 is generated Count, get a new sample x_{i1} through equation (1), and finally use equation (1) to get N new data samples. Among them, x_{i1} can be obtained by formula (1).

$$x_{i1} = x_i + * \left(x_i(nm) - x_i\right) \tag{1}$$

It can be found that the SMOTE algorithm only uses the sample information of the minority class when generating the minority class.Taking into account the samples of the majority class, it is likely that the generated minority class samples will overlap with the original data.

To this end, Han et al. proposed the Borderline-SMOTE algorithm based on the SMOTE algorithm. By identifying the data points at the boundary, over-sampling the minority samples to generate new data to achieve a balanced sample category. In the Borderline-SMOTE algorithm, the m nearest neighbors of each sample p in the minority sample set S in the total training set are first calculated, and then the m nearest neighbors are classified. If the m nearest neighbors are all samples of the majority class, they are defined as noise samples and no operation is performed. If the m nearest neighbors are all minority classes, then the sample is far from the classification boundary, and no operation is performed. If there are both majority and minority samples in the m nearest neighbors, they are considered as boundary samples and put them into set B. Finally, the SMOTE algorithm is used to generate new sample points.

For the boundary sample set $B = b_1, b_2, ..., b_n$. First calculate the k nearest neighbors $bi(nm)$ of each sample bi in the minority sample S in the boundary set B. and then randomly select a(1¡a¡n) nearest neighbors, calculate the difference between a nearest neighbor and the sample b_i, then multiply it by a random number, and generate an artificial minority sample h_i through equation (2). Finally, the formula (2) is repeatedly used to generate artificial minority samples until the positive and negative sample categories are balanced.

$$h_i = b_i + * \left(b_{i(nm)} - b_i\right) \tag{2}$$

Liang et al. [15] proposed the LR-SMOTE algorithm based on the traditional over-sampling SMOTE algorithm, so that the generated samples can be closer to the center of the sample, and avoid generating abnormal samples or changing the distribution of the data set. First use the K-means algorithm to find the center point of the minority samples c_i. Then calculate the Euclidean distance d and the average distance d_{mean} for each sample point m_i to c_i, and then calculate the ratio of d_{mean} to d, and finally use the SMOTE algorithm to generate a minority of balanced sample data until the sample. Minority samples can be generated by formula (3). The specific process of minority generation is shown in Fig.1.

$$\text{new} = c_i + \text{rand}\left(0, \frac{d_{\text{mean}}}{d}\right) * (m_i - ci) \tag{3}$$

B. Enhancement of Continuous Integration Log Information

Due to the large difference between the number of samples of neurotic personality and other personality samples, in order to prevent data imbalance from being unable to meet the classification requirements, this paper uses an improved SMOTE algorithm to oversample the neurotic personality data. At the same time, in order to make the sample size of all personalities reach a ratio of

1:1, this paper also oversampling the sample data of other personalities through the improved SMOTE algorithm.

Since the SMOTE algorithm mainly considers the imbalance of samples between categories, that is, the imbalance between the minority class and the majority class, it does not take into account the imbalance of the distribution of samples of the same type, and the problem of high similarity of synthetic samples, so in order to reduce To synthesize the similarity between samples and improve the representativeness of the samples, this paper combines the K-means algorithm with the SMOTE algorithm, and divides the oversampling operation of the SMOTE algorithm into two processes, clustering and oversampling.

In the clustering process, this paper uses the K-means clustering algorithm to cluster the minority sample sets, and assigns different sampling weights $W(i)$ according to the number of samples in the sub-categories. The weight $W(i)$ can be obtained by formula (4).

$$W(i) = 1 - \frac{num(i)}{\sum_{i=1}^{c} num(i)} \tag{4}$$

Among them, c represents the total number of subcategories, and num(i) represents the number of samples in the i-th category. It can be found that when the number of samples is larger, the sampling weight is smaller, and the number of synthesized samples will be smaller, thereby achieving a balanced distribution among similar samples.

In the process of SMOTE algorithm oversampling, if the quality of the synthesized sample is higher, the performance of the model is better. Conversely, if the synthesized sample contains both the features of the minority class and the majority class, it is likely to affect the classification effect of the majority class, and the synthesized sample is not representative. For this reason, the concept of centroid can be introduced to improve the SMOTE algorithm. The improved formula is shown in equation (5). Among them, the centroid refers to the cluster center $C_{i,center}$ of each sub-category of the minority category.

$$x_{new} = C_{i,\text{center}} + \xi * (x_{i,j} - C_{i,\text{center}}) \tag{5}$$

Among them, $C_{i,center}$ represents the cluster center of the i-th subcategory, $x_{i,j}$ refers to the j-th sample in the i-th subcategory, ξ is a random number.

To sum up, when the improved SMOTE algorithm synthesizes new samples, it first uses a clustering algorithm to divide the minority sample set into multiple sub-clusters, and then according to the number of new samples to be synthesized and the number of samples in each cluster, each cluster is obtained. The weight of clusters and the number of samples that need to be synthesized are then synthesized by formula (5) to synthesize new samples, and finally the synthesized minority sample set is output.

The distribution of personality data before and after the expansion of the SMOTE algorithm is shown in Table I.

TABLE I
DISTRIBUTION OF PERSONALITY DATA BEFORE AND AFTER SMOTE ALGORITHM EXPANSION.

Character	Before SMOTE algorithm		After SMOTE algorithm	
	positive	negative	positive	negative
Extraversion	2179	1826	2179	2179
Openness	1662	1429	2179	2179
Agreeableness	2043	1674	2179	2179
Neuroticism	326	956	2179	2179
Conscientiousness	1851	1738	2179	2179

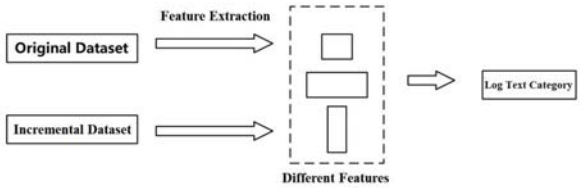


Fig. 1. The proposed BERT-Capsule model based on incremental learning.

That is, the number of positive samples and negative samples for the five personalities of extraversion, openness, agreeableness, conscientiousness and neuroticism are both 2,179, and the total number of samples for all personalities is 21790.

C. BERT-Capsule Model based on Incremental Learning

In order to improve the classification performance of the BERT-Capsule model for personality characteristics, this paper uses incremental learning to collect the log feedback information of developers, and then update the model to improve the classification performance of the model. Incremental learning means that a model can continuously learn new knowledge from new samples, and save most of the previously learned knowledge, without having to access the original data that has been used for training, which can greatly shorten the training time [16].

The incremental learning method of the BERT-Capsule model proposed in this paper is shown in Fig. 2. It is mainly divided into two stages, namely feature extraction stage and text classification stage. First input the text data set and label, then use the BERT-Capsule model to extract the features of the original data set text and the incremental data set text, and finally classify the text data through the classification method.

In the process of incremental learning of the BERT-Capsule model, the initial training data is used to train the model to obtain the initial classification model, and then it is judged whether it is a new category. If it is a new category, collect it, then use the sample data of the new category and the initial training data to incrementally

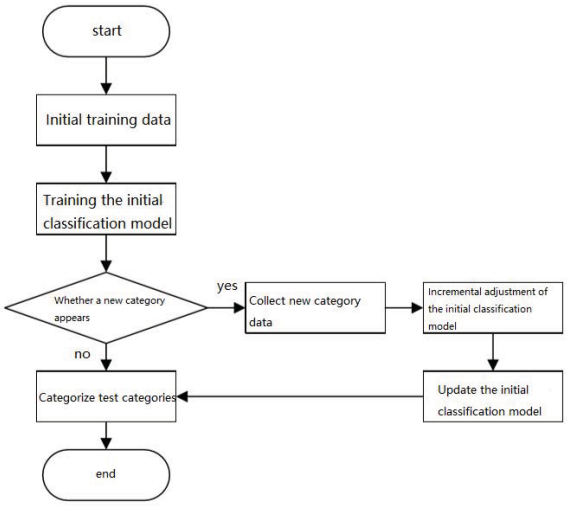


Fig. 2. Work flow chart of BERT-Capsule model based on incremental learning.

adjust the model, update the initial classification model, and finally use the test data to evaluate the performance of the model. If there is no need to classify the new category, the initial test set is used to evaluate the performance of the initial classification model. The flow of the entire method is shown in Fig.3.

When judging whether it is a new labeled sample, this paper uses the SVDD algorithm to judge whether the new sample belongs to a new category. In the SVDD algorithm, it is mainly through the establishment of a hypersphere, the object to be described as a whole, so that all the described objects or as far as possible are in this sphere. In order to minimize the influence of abnormal points, the volume of the hypersphere needs to be minimized.

For the training set $T = \left\{X_i \in R^d\right\}_{i=1}^{l}$, the optimization goal of SVDD is to find the minimum radius R in the training set T. The optimization problem is shown in equation (6).

$$\min_{R,a} R^2 + C\sum_{i=1}^{n}\xi_i \text{ s.t. } \|x_i - a\|^2 \le R^2 + \xi_i, \quad \xi_i \ge 0, i = 1 \ , \cdots, n \tag{6}$$

Among them, R is the minimum radius of the ball, C is the penalty coefficient, ξ_i is the penalty item, and a is the center of the hypersphere.

When training the SVDD algorithm, first train the original sample set as a whole, and then use the new sample data to test whether it belongs to the original sample set. If it belongs to the original sample set, no operation is performed. If it does not belong to the original sample set, the sample is added to the incremental sample set. Among them, it can be judged by formula (7) whether the new sample data t belongs to the original data set. Finally, use the incremental sample data set to update the BERTCapsule model.

$$(\mathrm{t} - \mathrm{a})^T (t - a) \le R^2 \tag{7}$$

IV. Experiments

A. Evaluation index establishment

In order to evaluate the BERT-Capsule model, this article uses the precision, recall and F1 values in the two classifications as the experimental evaluation criteria.

In the two-class classification, the category that is concerned during the experiment is set as a positive category, and the category that is not concerned during the experiment is set as a negative category. According to the prediction result of the classifier and the true category, it can be divided into the following four situations:

(1) TP: The prediction result of the classifier is a positive class, and the real class is a positive class;

(2) FP: The prediction result of the classifier is a positive class, and the true class is a negative class;

(3) FN: The prediction result of the classifier is a negative class, and the true class is a positive class;

(4) TN: The prediction result of the classifier is a negative class, and the true class is a negative class.

The accuracy is:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

The recall is:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

F1 value is the harmonic average of precision rate and recall rate, namely:

$$F1 = 2 * \frac{\text{Precision* Recall}}{\text{Precision } + \text{Recall}} \tag{10}$$

B. Dataset settings

In order to evaluate the performance of the BERT-Capsule model, this article selects IMDB emotional data set and continuous integration log information data set (CILDDB) for experiments. Since the CILDDB data set has unbalanced categories before expansion, this article uses the expanded sample data of the SMOTE algorithm for the personality analysis experiments of developers. Among them, 80% of the data is selected as the training set to train the model, 10% of the data is selected as the test set to test the performance of the trained model, and 10% of the data is selected as the validation set to adjust the experimental parameters of the model. The scale of the experimental data set is shown in TableII.

The IMDB data set is a movie review data set, which contains 50,000 pieces of movie review information with obvious emotional tendencies, and is widely used for sentiment analysis in natural language. The IMDB data set is mainly divided into two categories: positive reviews

TABLE II
EXPERIMENTAL DATA SET SIZE.

Data category	Training set	Test set	Validation set	Complete Works
IMDB	40000	5000	5000	50000
CILDDB	17430	2180	2180	21790

TABLE III
EXPERIMENTAL RESULTS OF THE DEEP LEARNING MODEL UNDER THE IMDB DATASET.

model	Precision(%)	Recall(%)	F1(%)
Capsule-A [17]	86.54	86.10	86.32
Capsule-B [17]	87.45	86.67	87.06
CNN-multichannel [18]	86.38	86.14	86.26
LR-Bi-LSTM [19]	86.74	86.38	86.56
CSVM [20]	85.26	86.68	84.97
LSTM-CNN [21]	86.91	85.88	86.39
MLCNN	**89.25**	**88.57**	**88.91**
BERT-Capsule	87.46	87.90	87.68

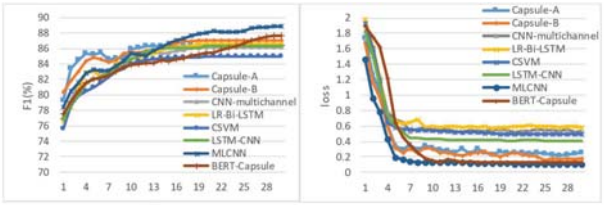
and negative reviews. The number of positive reviews is 25,000 and the number of negative reviews is 25,000. You can select 20,000 positive reviews and 20,000 negative reviews from the IMDB data set as the training set, 2500 positive reviews and 2500 negative reviews as the test set, and 2500 positive reviews and 2500 negative reviews as the validation set.

In the CILDDB data set, it can be divided into five categories of personality data: extraversion, openness, agreeableness, neuroticism, and conscientiousness. Each category contains two situations: positive affective tendencies and negative affective tendencies. Among them, from the two emotional tendencies of each personality, 10*1743 are selected as the training set, 10*218 are selected as the test set, and 10*218 are selected as the verification set.

C. Comparison of classification performance with existing deep learning algorithms

Experimental results of the deep learning model under the IMDB dataset. In order to further verify the effectiveness of the BERT-Capsule model proposed in this article, this article selects the IMDB data set, and compares the BERT-Capsule model with Capsule-A, Capsule-B, CNN-multichannel, LR-Bi-LSTM, CSVM, LSTMCNN, MLCNN. Seven deep learning models were compared. The classification performance of each model under the IMDB data set is shown in TableIII, the F1 value comparison is shown in Fig.4(a), and the loss comparison is shown in Fig.4(b).

It can be found that the convergence speed of the BERT-Capsule model proposed in this article on the IMDB data set is relatively slow, and the accuracy, recall, F1 value and loss value are lower than the experimental effect of



(a) F1 value of deep learning model under IMDB dataset (b) Deep learning model loss under IMDB dataset

Fig. 3. Experimental results of deep learning model under IMDB dataset.

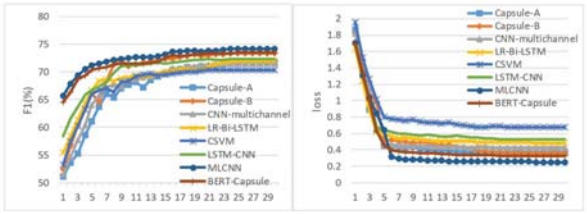
TABLE IV
EXPERIMENTAL RESULTS OF THE DEEP LEARNING MODEL UNDER THE CILDDB DATA SET.

model	Precision(%)	Recall(%)	F1(%)
Capsule-A [17]	71.46	71.95	71.70
Capsule-B [17]	73.18	73.66	73.42
CNN-multichannel [18]	71.63	71.49	71.56
LR-Bi-LSTM [19]	72.25	71.83	72.04
CSVM [20]	70.57	70.14	70.35
LSTM-CNN [21]	72.33	72.29	72.31
MLCNN	**74.65**	**73.76**	**74.20**
BERT-Capsule	73.25	73.85	73.55

MLCNN, but it is better than deep learning other than MLCNN Experimental results of the model. In terms of F1 value comparison, the BERT-Capsule model is 1.36% higher than the Capsule-A model, 0.62% higher than the Capsule-B model, 1.42% higher than the CNNmultichannel model, and 1.12% higher than the LR-Bi-LSTM model. The CSVM model is improved by 2.71%, 1.29% higher than the LSTM-CNN model, and 1.23% lower than the MLCNN model. This shows that the BERT-Capsule model has a good classification when it positively and negatively classifies the emotional semantics in the text. ability. At the same time, the experimental effect of the BERT-Capsule model is lower than that of the MLCNN model. The reason may be that MLCNN adopts the mechanism of multiple CNN feature fusion when extracting local features, which can more effectively extract the emotional semantics in the text, thereby making the MLCNN model more effective. The experimental effect is better than that of the BERT-Capsule model in this paper.

Experimental results of the deep learning model under the CILDDB data set. In order to further verify the effectiveness of the model in this paper, based on the five-factor personality model, this paper selects the CILDDB data set for further verification. The experimental results are shown in TableIV. The comparison of F1 values is shown in Fig.5(a). The loss comparison is shown in Fig.5(b).

It can be found that after fusing local feature semantic information and global feature semantic information, BERT The classification effect of the Capsule model is



(a) The F1 value of the deep learning model under the CILDDB dataset (b) The loss value of the deep learning model under the CILDDB data set

Fig. 4. Experimental results of the deep learning model under the CILDDB data set.

better than the classification effect of other deep learning models other than the MLCNN model. It has achieved good results in the three indicators of precision, recall and F1 value and loss value, which shows that the BERT-Capsule model can Better extraction of emotional semantic information in log information. At the same time, the accuracy rate, recall rate, F1 value and loss value of the MLCNN model are better than the BERT-Capsule model proposed in this paper. The main reason is that the MLCNN model can be better after fusing the features of the LSTM model and the multiple CNN convolution model. Analyze the different emotional tendencies of the same personality in the feedback information, and can better mine the main emotional semantic information in the feedback information. Therefore, the classification result of the MLCNN model is better than the classification result of the BERT-Capsule model proposed in this paper. In addition, the classification effect of the Capsule-B model is second only to the effect of the BERTCapsule model. The main reason is that although the Capsule-B model can obtain emotional semantic information in the text through multiple channels, it is mainly contextual semantic information, and does not pay attention to the semantic information of local emotions. It is not comprehensive enough in analyzing the personality characteristics of the developers, so the effect of the experiment Slightly worse than the experimental effect of the BERT-Capsule model.

D. Incremental learning classification performance comparison

As time changes, the personality characteristics of developers may also change, so the BERT-Capsule model needs to be updated in time. In this paper, the IMDB data set is used as the original data set, and the BERT-Capsule model parameters under the IMDB data set are used to train the IMDB data set, and then the CILDDB data set expanded by the SMOTE algorithm is used as the incremental data set to compare the BERT-Capsule model The incremental learning performance is evaluated.

In order to verify the effectiveness of the method in this paper, the original data set is used to train the network in the training phase to obtain the emotional tendency classification model BERT-Capsule, and then the test data set is used to test the model BERT-Capsule, and the test results are obtained. Then add the incremental data set to the original data set, and retrain the network to obtain the emotional tendency classification model BERT-Capsule-Inc, and use the original test set plus the incremental test set to test the model BERT-Capsule-Inc. Finally, compare the test results of BERT-Capsule and BERT-Capsule-Inc. The experimental results are shown in Table V.

TABLE V
PERFORMANCE COMPARISON BEFORE AND AFTER THE UPDATE OF THE BERT-CAPSULE MODEL BASED ON INCREMENTAL LEARNING.

model	Precision(%)	Recall(%)	F1(%)
BERT-Capsule	87.46	87.90	87.68
BERT-Capsule-Inc	**89.38**	**88.34**	**88.86**

It can be found that the accuracy rate of the BERT-Capsule-Inc model after using the incremental learning method has increased by 1.92%, the recall rate has increased by 0.44%, and the F1 value has increased by 1.18%, indicating that the CILDDB data set is used to perform the IMDB data set. After incremental learning, the BERT-Capsule-Inc model can better learn the emotional semantic features in the text. At the same time, it also indicates that the BERT-Capsule-Inc model can combine two different emotional semantics of the IMDB data set and the CILDDB data set when classifying the emotional tendency of the text, so that the BERT-Capsule-Inc model has better generalization performance , Which can better extract the emotional tendency in the text, which is helpful to make more accurate classification of the personality characteristics of the developers.

V. CONCLUSION

User portrait model is designed to model users' coding behaviours based on a large amount of data. Since current methods often suffer from the problem of imbalanced data, we propose to adopt SMOTE Algorithm to deal with the imbalanced log data, which is the core innovation of the proposed method. Experiments show the proposed SMOTE Algorithm based model could accurately classify programmers' personality types and offer suggestions.

ACKNOWLEDGE

This work was supported by National Key R&D Program of China under Grant 2018YFC0407901, the Fundamental Research Funds for the Central Universities under Grant B200202177, the Natural Science Foundation of China under Grant 61702160, the Natural Science Foundation of Jiangsu Province under Grant BK20170892.

REFERENCES

[1] Boby John and Rajeshwar S Kadadevaramath. Optimization of software development life cycle process to minimize the delivered defect density. *OPSEARCH*, 56(4):1199–1212, 2019.

[2] Miklós Biró, Ricardo Colomo-Palacios, and Richard Messnarz. Addressing evolving requirements faced by the software industry. *Journal of Software: Evolution and Process*, 32(3):e2237, 2020.
[3] Muhammad Azeem Akbar, Jun Sang, Arif Ali Khan, and Shahid Hussain. Investigation of the requirements change management challenges in the domain of global software development. *Journal of Software: Evolution and Process*, 31(10):e2207, 2019.
[4] Reginald Putra Ghozali, Herry Saputra, M Apriadin Nuriawan, Ditdit Nugeraha Utama, Ariadi Nugroho, et al. Systematic literature review on decision-making of requirement engineering from agile software development. *Procedia Computer Science*, 157:274–281, 2019.
[5] Shunxiang Zhang, Zhongliang Wei, Yin Wang, and Tao Liao. Sentiment analysis of chinese micro-blog text based on extended sentiment dictionary. *Future Generation Computer Systems*, 81:395–403, 2018.
[6] Mehdi Emadi and Maseud Rahgozar. Twitter sentiment analysis using fuzzy integral classifier fusion. *Journal of Information Science*, 46(2):226–242, 2020.
[7] Qiannan Xu, Li Zhu, Tao Dai, and Chengbing Yan. Aspect-based sentiment classification with multi-attention network. *Neurocomputing*, 388:135–143, 2020.
[8] Xiaolong Xu, Xihua Liu, Zhanyang Xu, Fei Dai, Xuyun Zhang, and Lianyong Qi. Trust-oriented iot service placement for smart cities in edge computing. *IEEE Internet Things J.*, 7(5):4084–4091, 2020.
[9] Ricardo Mitollo Bertani, Reinaldo AC Bianchi, and Anna Helena Reali Costa. Combining novelty and popularity on personalised recommendations via user profile learning. *Expert Systems with Applications*, 146:113149, 2020.
[10] Xiaolong Xu, Bowen Shen, Xiaochun Yin, Mohammad R Khosravi, Huaming Wu, Lianyong Qi, and Shaohua Wan. Edge server quantification and placement for offloading social media services in industrial cognitive iov. *IEEE Transactions on Industrial Informatics*, 2020.
[11] Lei Ren, Zihao Meng, Xiaokang Wang, Lin Zhang, and Laurence T Yang. A data-driven approach of product quality prediction for complex production systems. *IEEE Transactions on Industrial Informatics*, 2020.
[12] Xiaokang Wang, Laurence Tianruo Yang, Yihao Wang, Lei Ren, and M Jamal Deen. Adtt: A highly-efficient distributed tensor-train decomposition method for iiot big data. *IEEE Transactions on Industrial Informatics*, 2020.
[13] Young-Gab Kim and Sang-Min Park. User profile system based on sentiment analysis for mobile edge computing. *Computers, Materials & Continua*, 62(2):569–590, 2020.
[14] Sara Ouaftouh, Ahmed Zellou, and Ali Idri. Social recommendation: A user profile clustering-based approach. *Concurrency and Computation: Practice and Experience*, 31(20):e5330, 2019.
[15] XW Liang, AP Jiang, T Li, YY Xue, and GT Wang. Lr-smote–an improved unbalanced data set oversampling based on k-means and svm. *Knowledge-Based Systems*, page 105845, 2020.
[16] Xiaokang Wang, Laurence T Yang, Liwen Song, Huihui Wang, Lei Ren, and M Jamal Deen. A tensor-based multi-attributes visual feature recognition method for industrial intelligence. *IEEE Transactions on Industrial Informatics*, 2020.
[17] Wei Zhao, Jianbo Ye, Min Yang, Zeyang Lei, Suofei Zhang, and Zhou Zhao. Investigating capsule networks with dynamic routing for text classification. *arXiv preprint arXiv:1804.00538*, 2018.
[18] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.
[19] Qiao Qian, Minlie Huang, Jinhao Lei, and Xiaoyan Zhu. Linguistically regularized lstms for sentiment classification. *arXiv preprint arXiv:1611.03949*, 2016.
[20] Huaiguang Wu, Daiyi Li, and Ming Cheng. Chinese text classification based on character-level cnn and svm. *International Journal of Intelligent Information and Database Systems*, 12(3):212–228, 2019.
[21] Yue Li, Xutao Wang, and Pengjian Xu. Chinese text classification model based on deep learning. *Future Internet*, 10(11):113, 2018.