Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105
Volume **8**, Number **1**, February **2023**

A Classification Method of Imbalanced Big Data Based on Improved SMOTE and Stacked LSTM

Wentao Xu

College of politics
National Defense University
Shanghai, 200082, China
chaoticstich@163.com

Received July 16, 2022, revised September 10, 2022, accepted November 8, 2022.

Abstract. *In recent years, classification algorithms have made great progress, but with the continuous expansion of data sources, most of the data have different degrees of imbalance. Aiming at the problems of low classification accuracy and easy to fall into local optimum when most classification algorithms deal with unbalanced big data, an imbalanced big data classification algorithm based on Improved SMOTE (Synthetic Minority Oversampling Technique) and deep learning is proposed. The proposed method consists of data preprocessing and classification stages. Firstly, in order to limit the range of sample generation, two improved SMOTE algorithms are proposed to preprocess the dataset so as to solve the class imbalance problem. Secondly, the preprocessed dataset is classified using a stacked LSTM network, which solves the problem of weak adaptability of single-layer LSTM in extracting features. The experimental results show that the propose framework can effectively solve the classification problem of imbalanced datasets, and the classification accuracy is significantly better than other advanced machine learning and deep learning methods.*
Keywords: Big Data; Imbalanced Data Classification; SMOTE; Deep Learning; LSTM

1. **Introduction.** In recent years, with the rapid development of network technology, massive amounts of data have been accumulated in various fields such as medical care, finance, biology, etc. Big data plays an important role in information analysis and behavior prediction [1,2,3,4]. However, analyzing and extracting keypoints from such large data is a very difficult task due to the imbalanced data distribution [5]. In practical applications, there are a lot of imbalanced data, such as the number of bank bad debts [6], the number of cancer patients in the medical field [7] and so on. Making decisions based on these data is a typical classification task. In imbalanced data, the class with more instances is called the majority class (also known as negative samples), and the class with relatively few instances is called the minority class (also known as positive samples). Unbalanced big data will cause the learning performance of the classification algorithms to decline, and there are also problems of small samples, overlapping or small discontinuities, making the classification of unbalanced data a difficult problem [8].

In order to train a classification model more suitable for imbalanced data, it is very important to process imbalanced data in advance. The preprocessing methods for unbalanced data mainly involve changing the distribution of unbalanced datasets through different mechanisms to obtain balanced datasets [9]. Among them, feature selection refers to picking out the subset of features that make the classifier perform best from the sample space. The selection of feature subsets can be regarded as a search optimization problem.

Each possible feature combination is regarded as a node, then each node is connected to form a graph, and finally each node in the entire graph is traversed and the feature combinations of the nodes are used to train the classifier and calculate the performance of the classifier, therefore selecting the feature combination that enables the classifier to achieve optimal performance [10]. However, for high feature dimensions, there are many possibilities for each feature to be combined, and the established graph structure will be very large, resulting in a long time to traverse the entire graph. It is necessary to use specific algorithms to improve efficiency, such as Artificial bee Colony (ABC) and Particle Swarm Optimization (PSO).

For imbalanced datasets, the sampling method alleviates the degree of imbalance by re-balancing the sample space. The downsampling techniques mainly re-balance the dataset by removing the number of samples of the majority class in the training set, but when removing samples of the majority class, important sample information may be lost [11]. The upsampling techniques balance the minority and majority classes by generating minority class samples, and random upsampling is the simplest and most commonly used method. Such method repeatedly extracts random samples from the minority classes and puts them into the original sample space to form a new sample space. However, it is prone to model overfitting problem in the training of the classifier, and it will increase the impact of noise data on the model [12]. To this end, Chawla et al. proposed the SMOTE algorithm [13] based on the idea of random upsampling. By artificially constructing positive samples, the numbers of negative samples and positive samples in the dataset tend to be balanced, and by synthesizing new data rather than simply replicating the minority class samples, the problem of overfitting can be avoided to a certain extent. However, SMOTE generates the same amount of synthetic data for each minority class sample without considering the distribution characteristics of the neighboring samples, which increases the possibility of sample overlapping. Han et al. [14] proposes Borderline-SMOTE, which looks for samples close to the boundary of the sample spaces of the minority classes to form a new sample space, and then data is synthesized in the new sample space, making the newly added artificial data more effective. However, this method only synthesizes the minority class samples on all the boundaries, and does not distinguish the boundary samples, so it will add more noise data and seriously affect the performance of the classifier. In addition, Torres et al. [15] proposes SMOTE-D, in which the mean points of the k-nearest neighbor samples of the minority class are used to synthesize new samples, and Mathew et al. [16] proposes a Kernel-based K-SMOTE algorithm, but both methods are also very sensitive to noisy samples.

In the classification of unbalanced big data, the main idea of traditional clustering algorithms is to divide the dataset into multiple clusters based on the internal characteristics of the data, so that the data within a cluster is as similar as possible, and the data similarities between the different clusters are as small as possible. However, the sparsity of the new samples generated by clustering is very low, that is, the new samples cannot well reflect the characteristics of the original samples. Therefore, cluster-based mining algorithms are inefficient in dealing with this imbalanced data classification [17]. The Boosting algorithms mainly combine a series of weak classifiers obtained through repeated learning into a strong classifier in which a weighted majority voting mechanism is used to increase the weights of weak classifiers with low prediction error rates and reduce the weights of weak classifiers with high prediction error rates. Li et. al [18] propose a PSO based adaptive boosting algorithm PSO-AdaBoost, which can re-initialize parameters to avoid falling into local optimum and prevent redundant or useless weak classifiers from consuming excessive system resources, thereby improving the performance of processing data with a high degree of imbalance. Boosting-based mining algorithms have

many advantages such as high accuracy and no prior knowledge required. However, it does not support parallel learning, making the classifiers slow to learn and susceptible to noise. SVM is a commonly used kernel-based algorithm. Based on the training set, a hyperplane with the largest margin is found in the sample space to divide samples within different categories. However, in the case of linear inseparability, there is no way to find a hyperplane that can divide the samples, so the kernel method is used by the SVM to perform high-dimensional mapping of the sample space to achieve the purpose of dividing the samples. Schölkopf et al. [19] propose an algorithm based on parameter tuning of SVM to deal with imbalanced data sets, and the performance was improved compared to the original SVM algorithm. Nurlaily et al. [20] propose an imbalanced big data classification method combining ACO , genetic algorithm (GA) and SVM, in which SMOTE is used to deal with data imbalance problem. Lu et al. [21] propose a data classification method based on improved weighted extreme learning machine (ELM), in which a voting mechanism is introduced into the weighted ELM to solve the data imbalance problem. Although SVM and ELM have the advantages of good self-learning and self-adaptive ability, strong nonlinear mapping and parallel processing ability, but such methods rely too much on parameter tuning, so the application scope is severely limited.

Nowadays, with the rapid development of deep learning, compared with shallow neural networks, deep neural networks can mine more information under the same circumstances. The most advanced processing way is to balance the dataset with traditional sampling methods and replace traditional machine learning models with deep learning network models for training and classification. Zhang et al. [22] propose a network traffic detection model based on convolutional neural network (CNN). The SMOTE combined with nearest neighbor algorithm (SMOTE-ENN) is used to balance the dataset before CNN training, thereby improving the classification performance. Yan et al. [23] propose an improved local adaptive gated recurrent unit (LA-GRU) model, in which the local adaptive SMOTE technology is used to deal with imbalanced data, and then GRU model is used to extract features from the data and perform classification. However, it does not solve the noise sensitivity problem of SMOTE. In response to the above problems, an imbalanced big data classification method combining improved SMOTE and stacked LSTM is proposed. The main contributions of this paper are listed as follows:

1) The SMOTE algorithm is improved to solve the marginalization problem of the distribution of positive samples in the dataset.

2) The LSTM network is used for data classification, and three LSTM models are continuously used to design a three-layer stacked structure to solve the problem of weak adaptability of the single-layer LSTM network when performing feature extraction, and further improve the classification performance.

The rest of this paper is organized as follows. Section 2 explains the proposed unbalanced data preprocessing algorithms based on SMOTE. Section 3 describes the proposed stacked LSTM-based classification network in detail. Section 4 presents the experiment results and analysis. Finally, the Section 5 summarizes the full text.

2. **Dataset preprocessing based on improved SMOTE.**

2.1. **SMOTE algorithm.**

The basic idea of the SMOTE algorithm is to deal with the attribute domain instead of the instance domain by creating synthetic instances of the minority classes. Synthetic instances are created along the k nearest neighbors of the minority class, and each instance in the minority class will be oversampled. However, the SMOTE algorithm cannot solve the problem of marginal distribution of positive samples in the dataset. In the original SMOTE algorithm, the positive samples are firstly

grouped according to the Euclidean distance. Given a sample Z ,$Z = \{z_1, z_2, ..., z_n\}$, and $z_1, z_2, ..., z_n$ are the n dimensional values of the sample Z. The same is true for sample $V = \{v_1, v_2, ..., v_n\}$. Then the Euclidean distance E between sample Z and sample F is:

$$E = \sqrt{\sum_{k=1}^{n} (z_k - v_k)^2} \tag{1}$$

The 6 samples with the closest Euclidean distances are grouped together. According to the idea of clustering, the samples with a closer spatial distance to the positive samples are also positive. In each group of 6 samples, new positive samples are randomly generated synthetically on the connection line between each two samples:

$$Z_{new} = Z + rand(0, 1) \times (V_i - Z) \tag{2}$$

Where $i = 1, 2, ..., 6$. Z represents a sparse positive sample. V_i is the $i - th$ nearest neighbor of Z. rand(0, 1) represents a random number between 0 and 1. Z_{new} represents a newly generated sample. Multiple iterations are performed according to Eq. (2) to balance the dataset. The original SMOTE algorithm process is shown in Algorithm 1.

Algorithm 1 SMOTE algorithm
Input: positive sample set $X_{positive}$, negative sample set $X_{negative}$.
Step 1 Group the samples in the positive sample set $X_{positive}$, and the 6 samples with the nearest Euclidean distances are grouped into a group.
Step 2 The positive samples are randomly generated according to Eq. (2) on the connection line between the two samples in each group, and the newly generated samples are added to the data set.
Step 3 **While** $\frac{X_{positive}}{X_{negative}} \neq 1$
Return to Step 2
END

2.2. **C-SMOTE.** The SMOTE algorithm balances the tendency of classification results to a certain extent, and can improve the classification performance of the model for positive samples. However, the SMOTE algorithm cannot solve the marginalization problem of the distribution of positive samples in the dataset. To solve this problem, this paper proposes a centroid-based SMOTE (C-SMOTE) algorithm. The C-SMOTE algorithm focuses on the area where new samples are generated, avoiding further marginalization of the dataset distribution caused by adding new samples.

The proposed C-SMOTE algorithm first divides the sample set into groups, each of which is a group of 6 samples. Then 3 samples such as Z_1, Z_2, Z_3 are randomly selected from each group,$Z_i = \{Z_{i1}, Z_{i2}, ..., Z_{in}\}$. The centroid Z_T of the 3 samples is calculated as:

$$Z_T = (\tfrac{1}{3}\sum_{i=1}^{3} Z_{i1}, \tfrac{1}{3}\sum_{i=1}^{3} Z_{i2}, \tfrac{1}{3}\sum_{i=1}^{3} Z_{i3}) \tag{3}$$

These 3 samples form a triangle, and each sample is a vertex of the triangle. A positive sample is randomly generated on the connection line between each vertex and the centroid, and three new positive samples are generated for a triangle. The generation area of each new group of samples is limited to a certain extent, and the generated new samples are closer to the centroid, which better improves the further marginalization of the new sample distribution with the original SMOTE algorithm.

The detailed steps of the C-SMOTE algorithm are shown in Algorithm 2.
Algorithm 2 C-SMOTE algorithm

Input: positive sample set $X_{positive}$, negative sample set $X_{negative}$.
Step 1. Group the samples in the positive sample set $X_{positive}$, and the 6 samples with the nearest Euclidean distances are grouped into a group.
Step 2. Randomly select 3 samples from each group as triangle vertices, and calculate the centroid based on Eq. (3);
Step 3. New samples are randomly generated on the lines connecting the triangle vertices and the centroid.
Step 4. **While** $\frac{X_{positive}}{X_{negative}} \neq 1$
Return to Step 3
END

2.3. **F-SMOTE.** The C-SMOTE algorithm better limits the area where new samples are generated, and improves the sample distribution of the dataset, but the computational complexity is still high. In practical applications, the sample points are generally multi-dimensional. For hundreds, thousands or even tens of thousands of dimensions, the time consumption of the SOMTE algorithm and the C-SMOTE algorithm will increase significantly. Aiming at this problem, this paper further proposes a farthest point-based SMOTE (F-SMOTE).

The F-SMOTE focuses only on two sample points, the centroid point of all positive samples, and the positive sample point farthest from the centroid. New sample points are randomly generated along the line connecting the farthest point and the centroid point:

$$X_{new} = X_c + rand(0,1) \times (X_F - X_c) \tag{4}$$

where X_{new} are the newly generated points, X_c represents the centroid of all positive samples, and X_F is the farthest sample point from the centroid.

The F-SMOTE algorithm abandons the idea of grouping positive sample points in the traditional SMOTE algorithm, instead only focusing on the centroid of all positive samples and the farthest positive sample point from this centroid, which greatly reduces the complexity of the algorithm.. In addition, the F-SMOTE algorithm only needs to iterate once, and a batch of new samples is generated according to Eq. (4) to directly balance the entire dataset. The algorithm is simple and easy to implement. The detailed steps of the F-SMOTE algorithm are as follows:
Algorithm 3 F-SMOTE algorithm
Input: positive sample set $X_{positive}$, negative sample set $X_{negative}$.
Step 1 Calculate the centroid of all positive samples, and traverse all positive samples to find the sample point farthest from the centroid.
Step 2 Based on Eq.(4), a large number of positive samples are generated along the line connecting the centroid and the farthest point to balance the dataset.

3. Stacked LSTM-based Classification.

3.1. **Feature Attributes Construction.** Given a dataset $X = \{x_1, x_2, ..., x-n\}$, each sample x_i has a w-dimensional statistical attribute $V = \{v_1, v_2, ..., v_2\}$. Firstly, the feature attributes are used to construct an attribute matrix $\boldsymbol{K} \in \Omega^{n \times w}$. An adjacency matrix $\boldsymbol{F} \in \Omega^{n \times n}$ is established based on sample similarity. For an element in the adjacency matrix, if $F(i,j) = 1$, it means that x_i and x_j are similar, otherwise they are not similar. In order to obtain the main mode of the attribute matrix, a model of extracting representative rows and columns from the attribute matrix is established to form a new attribute matrix $\boldsymbol{K'} \in \Omega^{n \times w}$. This matrix represents the main pattern of the attribute matrix $\boldsymbol{K}$, and the data is classified based on the difference between $\boldsymbol{K}$ and $\boldsymbol{K'}$. Finally,

the feature extraction and training are completed by the proposed deep learning method, and the results are input to a softmax layer to complete the data classification.

3.2. **LSTM Network.** Recurrent Neural Networks (RNN) is a neural network with a feedback structure, the input is not only related to the current input and network weights, but also related to the previous network inputs. The RNN network will memorize the information of the historical moment, and apply the information left by the memory to the input calculation of the current neuron. However, when the RNN learns the long-term dependency problem, there will be problems such as gradient disappearance or gradient explosion, which may cause the model to fail to train [24]. Therefore, this paper uses the long short-term memory module in the LSTM model to solve the above problems.

LSTM is essentially a gated RNN[25,26]. Three gates and an update parameter of the cell state are added to the hidden layer of the RNN model, so that the detection network have variable and self-circulating weights. The internal structure of LSTM neurons is shown in Figure 1.

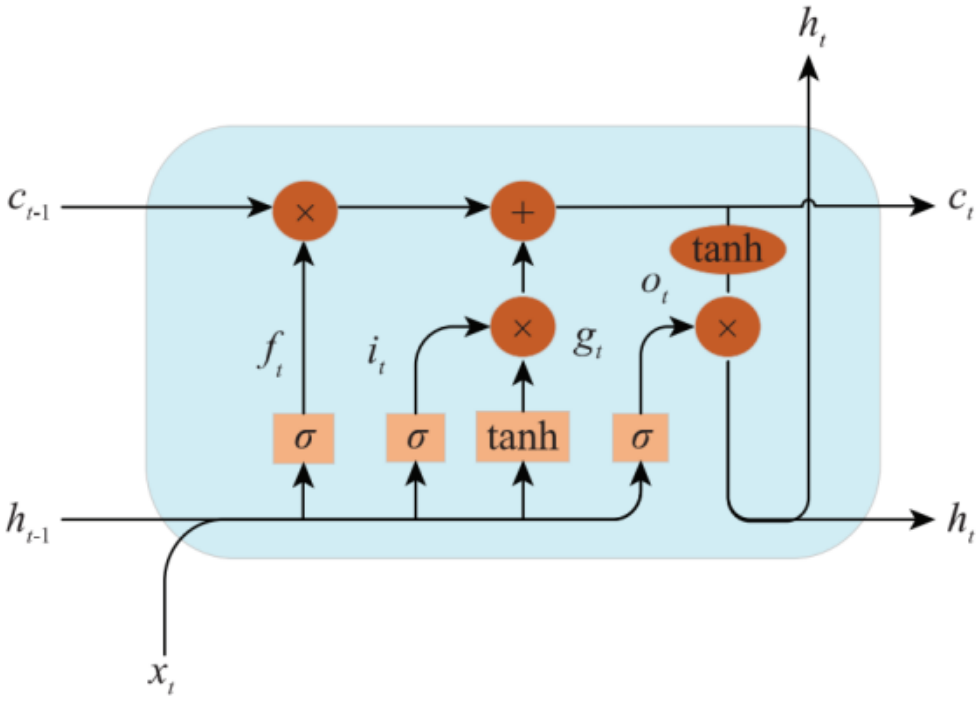

FIGURE 1. LSTM structure.

As shown in Figure 1, the LSTM network is formed by a combination of 4 independent structures, including one state (ie, the cell state), and a main structure consisting of 3 gates (ie, forget gate, input gate, and output gate). The forget gate is responsible for clearing the information from the previous cell $c_{t-1}$to the current cell c_t. The input gate computes the decision update value it and the candidate cell c_t by computing the decision. The updated c_t is obtained through the calculation of the forget gate, the input gate and the g_t, and the c_t is continuously propagated, thereby reaching the next state c_{t+1}. h_t is the output value of the LSTM network, which is calculated and decided by the output gate. The specific calculation process is as follows:

Firstly, the forget gate ft controls which long-term memories of the LSTM layer can be forgotten:

$$f_t = \sigma(\boldsymbol{W}_f \boldsymbol{h}_{t-1} + \boldsymbol{U}_f \boldsymbol{X}^t + \boldsymbol{b}_f) \tag{5}$$

Next, the input gate calculates what information can be obtained from the input and identifies which parts of the information should be stored in the cell state:

$$\boldsymbol{g}_t = \tanh(\boldsymbol{W}_g \boldsymbol{h}_{t-1} + \boldsymbol{U}_g \boldsymbol{X}^t + \boldsymbol{b}_g) \tag{6}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \boldsymbol{h}_{t-1} + \boldsymbol{U}_i \boldsymbol{X}^t + \boldsymbol{b}_i) \tag{7}$$

Then, the long-term memory in the cell state is updated as follows:

$$\boldsymbol{c}_t = \boldsymbol{c}_{t-1} \otimes \boldsymbol{f}_t + \boldsymbol{g}_i \otimes \boldsymbol{i}_t \tag{8}$$

Finally, the state of the current hidden layer is updated by the output gate based on the input, the cell state and the previous hidden state:

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \boldsymbol{h}_{t-1} + \boldsymbol{U}_o \boldsymbol{X}^t + \boldsymbol{b}_o) \tag{9}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \otimes \tanh(\boldsymbol{c}_t) \tag{10}$$

In the above equations, $\boldsymbol{W}_i, \boldsymbol{W}_f, \boldsymbol{W}_g, \boldsymbol{W}_o$ are the weights of the hidden layer between the current hidden layer and the previous hidden layer, and $\boldsymbol{U}_i, \boldsymbol{U}_f, \boldsymbol{U}_g, \boldsymbol{U}_o$ are the weights between the current input layer and the current hidden layer; $\boldsymbol{b}_i, \boldsymbol{b}_f, \boldsymbol{b}_g, \boldsymbol{b}_o$ are the bias vectors; $\otimes$ is the element-wise multiplication operator; σ is the sigmoid function; and tanh is the activation function.

Afterwards, a loss function is defined to calculate the difference between the predicted value of the LSTM model and the actual value, in order to adjust the weights and biases using the optimizer to minimize the loss value after training and improve the classification performance of the model. Loss is calculated using cross entropy, the smaller the cross-entropy, the closer the predicted value is to the actual value:

$$loss = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log(y'_i) + (1 - y_i) \log(1 - y'_i)] \tag{11}$$

where y_i is the ith actual value, and y'_i is the i-th predicted value of the model, and n is the number of categories to be classified.

3.3. **Stacked LSTM.** The main ideas of the detection of the proposed model are as follows. Firstly, after the data set is preprocessed, the output X is obtained. Secondly, the output X is used as the input of the three-layer stacked LSTM network to perform feature optimization processing, so as to obtain the output X_0, and this output is used as the input to obtain Y_0 after Dropout. Finally, the final output Y is obtained through the Softmax layer. To this end, a three-layer stacked LSTM is constructed. Based on the LSTM network described above, three LSTM models are continuously used to design a three-layer stacked structure to solve the problem of weak adaptability of single-layer LSTM network for feature extraction. The input of the stacked LSTM is the dataset

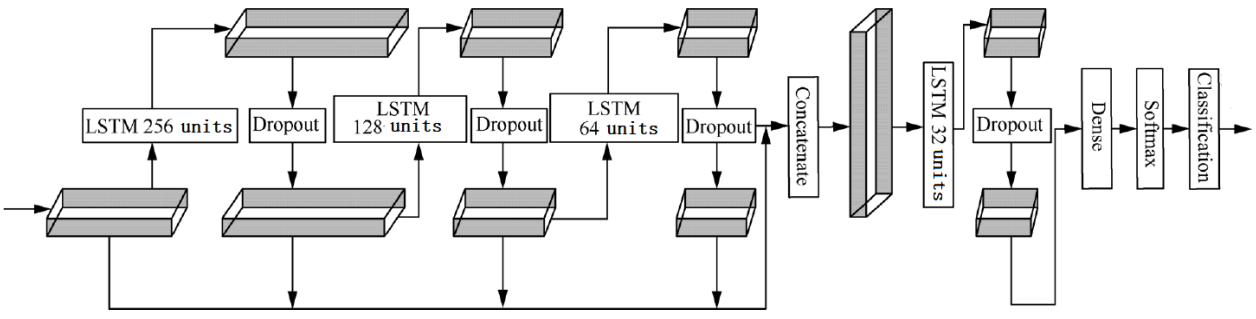

FIGURE 2. Stacked LSTM structure.

X that has been optimized and preprocessed in the pre-processing stage. Three layers of the stacked LSTM are used to extract features of different depths from the data, and then the processed sample features are fused and analyzed, and then passed to the next LSTM network. Dropout and Dense layers are used to appropriately mitigate the vanishing gradient problem, and the output is passed as input to the Softmax layer for classification.

The specific process is as follows: Firstly, input the preprocessed sample feature X into the first layer of LSTM with 256 units to obtain first-order features. Before the features are passed to the second layer of LSTM of the stacked network, the neurons are optimized by the Dropout layer, and the optimized data is passed to the second layer of LSTM with 128 units to obtain second-order features; Before the features are passed into the third layer LSTM of the stacked network, the neurons are optimized by the Dropout layer, and the optimized second-order feature data is passed into a third-layer LSTM with 64 units to obtain third-order features, and then the neurons are optimized through the Dropout layer. These features with different depths are concatenated with the data to obtain new data with features of different depths. Finally, the data with features of different depths is input into the LSTM with 32 units, and the data of the hidden layer state at the last moment is obtained. The data passing through the last hidden layer state of the LSTM is passed to a fully connected layer to obtain 32-dimensional features, and finally a Softmax layer is used for classification.

4. **Experiment and Analysis.** In order to evaluate the performance of the proposed method, experiments are carried out on a computer with i5-9400 CPU @ 2.8GHz, 16 GB RAM, running Windows 10 operation system, all experiments have been done under the Keras (Version 2.3.1) deep learning framework with Tensorflow (Version 1.15.0) as the backend. The LSTM network learning rate is set to 0.01, the batch size is 16, the training epoch is 50, the dropout rate is set to 0.5, and the optimizer used is Adam [27]. And under the same conditions, the proposed method is compared with PSO-Adaboost [18], ACO-SVM [20] and LA-GRU [23].

4.1. **Datasets.** From the UCI and KEEL repositories [27], the pima and emotions datasets with relatively low imbalance rates and the pageblocks, votel, and pop_failures datasets with high imbalance rates are selected for experiments. The statistics of the experimental datasets are shown in Table 1. In order to improve the training effect, this paper divides the training set and the test set into different combinations for training. After experimental verification, it is known that when the training set is 75% and the test set is 25%, the proposed method achieves the best classification accuracy indicating that the output of the training model is more optimized.

Table 1. statistics of the experimental datasets

Datasets	Samples	Majority samples	Minority samples	Attributes	Imbalance Ratio
pima	767	500	267	9	1.87
emotions	592	419	173	72	2.42
pageblocks	547	491	56	11	8.77
vowel	987	898	89	14	10.09
pop_failures	539	494	45	21	10.98

4.2. **Evaluation Metrics.** In the experiment, three indicators, F-value, G-mean and AUC, which are commonly used in the classification performance evaluation of unbalanced datasets, are used to evaluate the performance of different classification methods. These three indicators are all based on the confusion matrix. Taking the binary classification model as an example, the definition of the confusion matrix is shown in Table 2. In Table

Table 2. Confusion matrix.

		Predicted class	
		Positive	Negative
Actual class	Postive	TP	FN
	Negative	FP	TN

2, TP means the predicted class and the actual class are all positive class. TN means the predicted class and the actual class are all negative class. FP means the predicted class is positive and the actual class is negative. FN means the predicted class is negative and the actual class is positive. In the field of imbalanced data classification, the positive class refers to the majority class and the negative class refers to the minority class.
G-mean: This indicator comprehensively considers the classification performance of the minority class and majority class samples. If the classification of the classifier is biased towards one class, it will affect the classification accuracy of the other class:

$$\text{G-mean} = \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}} \tag{12}$$

F-score: This indicator can effectively reflect the sensitivity of the classifier to the classification performance of minority class samples:

$$\text{F-score} = \frac{2 \times \frac{TP}{TP+FN} \times \frac{TP}{TP+FP}}{\frac{TP}{TP+FN} + \frac{TP}{TP+FP}} \tag{13}$$

AUC: Area Under Curve, this indicator calculates the area under the Receiver Operating Characteristic (ROC) curve, where TPR is the true positive rate and FPR for the false positive rate:

$$AUC = \frac{1 + TPR \text{ - } FPR}{2} = \frac{1 + \frac{TP}{TP+FN} - \frac{FP}{TN+FP}}{2} \tag{14}$$

4.3. **Results and discussions.** The classification performance for imbalanced data using the proposed stacked LSTM network on different datasets preprocessed with the original SMOTE, the proposed C-SMOTEs and the proposed F-SMOTE are compared. Table 3 lists the G-means results. It can be seen from the table that in the five datasets, the G-mean values of the SMOTE, C-SMOTE and F-SMOTE algorithms show an increasing trend. Among them, the G-mean values of the C-MOTE algorithm are greatly improved compared with the original SMOTE algorithm, and the G-mean values of the F-SMOTE algorithm are slightly improved compared with the C-SMOTE algorithm. In the pop_failures dataset, the G-mean value of the SMOTE algorithm is only 81.33%, the G-mean value of the C-SMOTE algorithm reaches 93.05%, and the G-mean index value of the F-SMOTE algorithm is 96.31%. This is because in the pop_failures dataset, positive samples are severely marginalized, resulting in poor performance of the original SMOTE algorithm. The proposed two improved SMOTE algorithms adopt the idea of limiting the area where new samples are generated, so that the positive samples in the dataset are not marginalized, but are centered around the centroid, which better improves the sample distribution of the dataset and improves the performance of the classifier. Overall,

the proposed F-SMOTE algorithm achieves the best performance. Next, the classification

Table 3. G-mean performance comparison with different preprocessing methods (%).

Dataset	SMOTE	C-SMOTE	F-SMOTE
pima	97.28	97.55	97.58
emotions	95.07	97.03	97.33
pageblocks	82.38	95.98	97.57
vowel	81.64	94.15	97.28
pop_failures	81.33	93.05	96.31

results of the proposed algorithm and other advanced algorithms on five datasets are compared, and the G-mean and F-score indicators are used for evaluation. Table 4 shows the classification results of different algorithms on 5 datasets, in which the proposed method uses F-SMOTE as the preprocessing method to achieve the best performance. It can be seen from Table 4 that for the Glass Identification and Iris datasets, because the imbalance rates in these two datasets are very low, each sample is more likely to become a support vector, the performance of the proposed algorithm in terms of F-score and G-mean are relatively close to that of the ACO-SVM and LA-GRU algorithms. For the pageblocks, votel, and pop_failures datasets with high imbalance rates, the PSO-Adaboost classifier is more likely to ignore the minority class, so the classification performance is very poor. The ACO-SVM and LA-GRU algorithms have good applicability for imbalanced datasets, but they are worse than the proposed method. This is because the proposed method solves the marginalization problem of positive sample distribution with the F-SMOTE algorithm, ensures that the number of majority and minority classes participating in the training of the classifier is balanced each time. In addition, in the proposed method, the stacked LSTM is used for better feature extraction, and at the same time, the sample information of the class boundary is fully considered, so the classification performance of the proposed method is significantly better for severely imbalanced datasets. As a variant of LSTM, GRU has fewer parameters and faster convergence than LSTM, and its performance is close to LSTM. Experimental results show that the proposed stacked LSTM performs slightly better than GRU. The AUC value is the the area enclosed by the ROC curve and

Table 4. Performance comparison of different methods(%).

Dataset	PSO-Adaboost		ACO-SVM		LA-GRU		Proposed Method	
	G-mean	F-score	G-mean	F-score	G-mean	F-score	G-mean	F-score
pima	94.28	93.97	97.24	96.85	97.61	97.28	97.58	97.01
emotions	93.14	92.85	95.17	94.72	94.52	94.26	97.33	96.41
pageblocks	83.38	81.92	90.48	88.97	93.47	91.99	97.57	96.92
vowel	80.28	79.92	90.57	89.81	90.62	89.84	97.28	96.35
pop_failures	74.92	73.75	89.90	89.38	90.57	89.82	96.31	96.29

the horizontal axis, and this metric can reflect the performance of the classifier. The closer the curve is to the (0, 1) point, and the larger the area enclosed by the curve and the horizontal axis, the better the classifier performance. Figure 3 presents the AUC results of each method on the pima dataset. It can be seen from Figure 3 that for the pima dataset with a low imbalance rate, the AUC value of the PSO-Adaboost algorithm is 0.878, and the other methods have achieved slightly better performance than the PSO-Adaboost

method, indicating that each method is effective for the classification of the balanced data. When the four different curves are not interleaved, the performance of the classifier corresponding to the ROC curve on the top is better than that of the classifier on the bottom. It can be seen from the curve distribution that the classification performance of the proposed method is better than other models, which verifies the feasibility of the proposed method. As shown in Figure 4, for the pop_failures dataset with a high imbalance

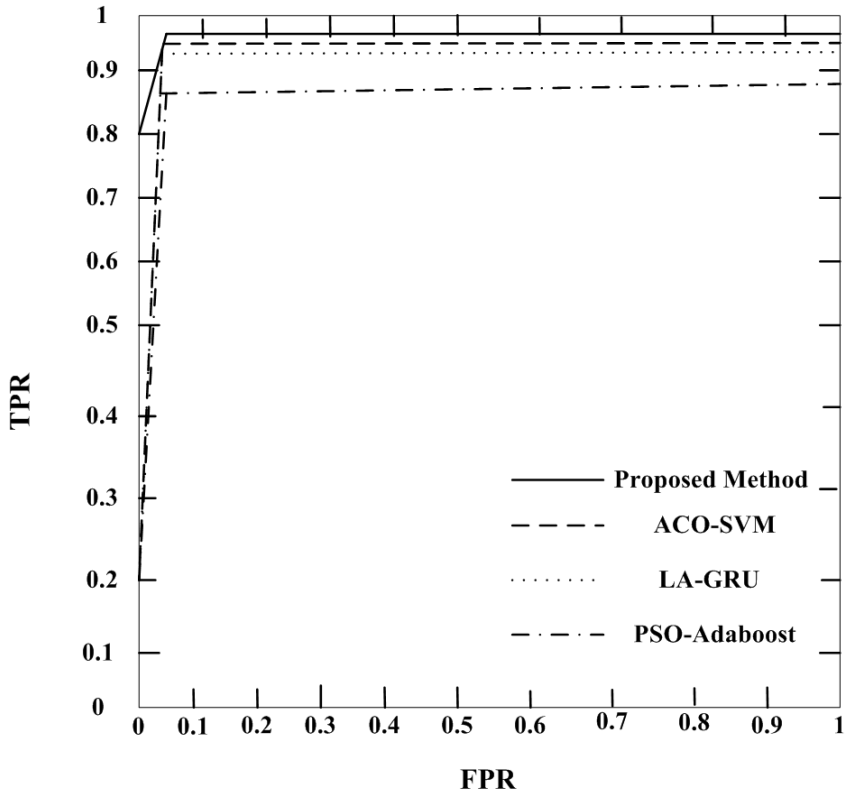

FIGURE 3. AUC results of different methods on pima dataset.

rate, the AUC value of the PSO-Adaboost method is 0.797, which is significantly lower than that of the rest classification methods. The proposed method achieves an AUC value of 0.967, which is significantly better than all other methods. The results show that for datasets with few marginalized samples, the proposed F-SMOTE algorithm adopts the centralization idea for preprocessing optimization, which can significantly improve the imbalance of datasets. And the proposed method improves the extraction ability of features with different depths through a stacked LSTM structure, thereby greatly improving the classification performance of severely imbalanced datasets.

5. **Conclusion.** In this paper, an imbalanced big data classification algorithm based on improved SMOTE and stacked LSTM is proposed to solve the problems of low classification accuracy and easy to fall into local optimum for imbalanced big data classification. In the proposed method, the dataset is firstly preprocessed with the improved SMOTE techniques to obtain a balance between the majority class and the minority class; then the preprocessed dataset is classified through a stacked LSTM network. The experimental results verify that the proposed method can deal with extremely unbalanced big datasets and obtain highly accurate classification results. In addition, there are still many noise samples in the data processed by the proposed F-SMOTE algorithm. In Future, we will

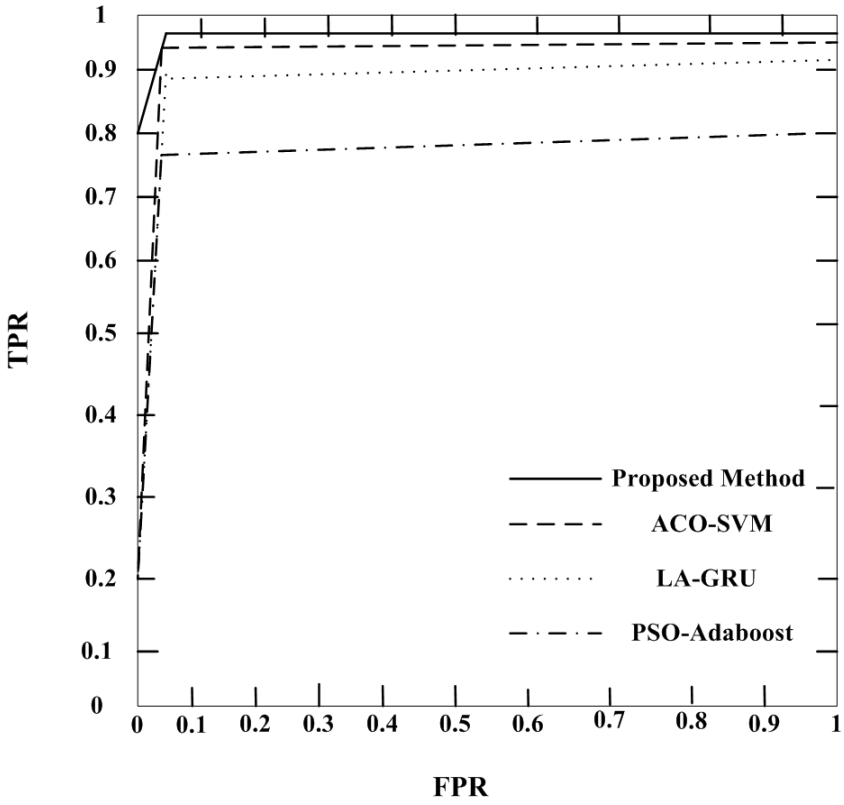

FIGURE 4. AUC results of different methods on pop_failures dataset.

focus on combining undersampling technology with resampling technology, with oversampling techniques to balance the dataset, and then use specific undersampling algorithms to remove noisy samples and outliers in the dataset to enhance the data re-balancing performance.

Data Availability. The data used to support the findings of this study are included within the article.

Conflicts of Interest. The author declares that there is no conflict of interest regarding the publication of this paper.

Funding Statement. This work is supported by National Social Science Fund Military Science Youth Project (2019-SKJJ-C-064)

REFERENCES

[1] W.-A. Günther, M.-H.-R Mehrizi, M. Huysman, F. Feldberg, "Debating big data: A literature review on realizing value from big data," *The Journal of Strategic Information Systems*, vol. 26, no. 3, pp. 191-209, 2017.

[2] N.-D. Feng, T.-Y. Wu, Y.-Q Liang, "A deep dynamic neural network model and its application for ECG classification," *Journal of Intelligent & Fuzzy Systems*, vol. 43, no. 2, pp. 2147-2154, 2022.

[3] J.-M.-T Wu, L. Sun, G. Srivastava, N. Herencsar, "A long short-term memory network stock price prediction with leading indicators," *Big Data*, vol. 9, no. 5, pp. 343-357, 2021.

[4] J.-M.-T Wu, Z. Li, N. Herencsar, C.-W. Lin, "A graph-based CNN-LSTM stock price prediction algorithm with leading indicators," *Multimedia Systems*, vol. 27, no.3, pp. 1-20, 2021.

[5] J.-L. Leevy, T.-M. Khoshgoftaar, R.-A. Bauder, S. Naeem, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, pp. 1-30, 2018.

[6] S. Birla, K. Kohli, A. Dutta, "Machine learning on imbalanced data in credit risk," in *7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON, 2016)*. IEEE, pp. 1-6, 2016.
[7] X. Yuan, L. Xie, M. Abouelenien, "A regularized ensemble framework of deep learning for cancer detection from multi-class, imbalanced training data," *Pattern Recognition*, vol. 77, no. 1, pp. 160-172, 2018.
[8] A. Fernández, S. Delrío, N.-V. Chawla, F. Herrera, "An insight into imbalanced big data classification: outcomes and challenges," *Complex & Intelligent Systems*, vol. 3, no. 2, pp. 105-120, 2017.
[9] J. Luengo, D. García-Gil, S. Ramírez-Gallego, S. Garcia, F. Herrera, "Imbalanced data preprocessing for big data," *Big Data Preprocessing*, vol, 47, no. 3, pp. 147-160, 2020.
[10] L. Wang, Y. Wang, Q. Chang, "Feature selection methods for big data bioinformatics: a survey from the search perspective," *Methods*, vol. 111, no. 12, pp. 21-31, 2016.
[11] Y. Cai, D. Li, Y. Wang, "Massive Crime Data Screening Framework based on Downsampling and Compressed Sensing," in *Fourth International Conference on Electronics, Communication and Aerospace Technology (ICECA, 2020)*. IEEE, pp. 509-513, 2020.
[12] T. Singh, R. Khanna, M. Kumar, "Multiclass Imbalanced Big Data Classification Utilizing Spark Cluster," in *12th International Conference on Computing Communication and Networking Technologies (ICCCNT 2021*). IEEE, pp. 1-7, 2021.
[13] N.-V. Chawla, K.-W. Bowyer, L.-O. Hall, W.-P. Kegelmeyer, "SMOTE: synthetic minority oversampling technique," *Journal of Artificial Intelligence Research*, vol.16, no. 3, pp. 321-357, 2002.
[14] H. Han, W.-Y. Wang, B.-H. Mao, "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning," in *Third International Conference on Intelligent Computing*. IEEE, pp. 878-887, 2005.
[15] F.-R. Torres, J.-A. Carrasco-Ochoa, J. F. Martínez-Trinidad, "SMOTE-D a deterministic version of SMOTE," in *Eighth Mexican Conference on Pattern Recognition (MCPR 2016)*. IEEE, pp. 177-188, 2016.
[16] J. Mathew, M. Luo, C.-K. Pang, H.-L. Chan, "Kernel-based SMOTE for SVM classification of imbalanced datasets," in *41st Annual Conference of the IEEE Industrial Electronics Society*. IEEE, pp. 1127-1132, 2015.
[17] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, J.-S. Jhang, "Clustering-based under sampling in class-imbalanced data," *Information Sciences*, vol. 409, no. 10, pp. 17-26, 2017.
[18] K.-W. Li, G.-Y. Zhou, J.-N. Zhai, F.-L. Li, M.-W. Shao, "Improved PSO_AdaBoost ensemble algorithm for imbalanced data," *Sensors*, vol. 19, no. 6, pp. 1476-1487, 2019.
[19] B. Scholkopf, A.-J. Smola, R.-C. Williamson, "New support vector algorithms," *Neural Computation*, pp. 12, no. 5, pp. 1207-1245, 2000.
[20] D. Nurlaily, Irhamah, S.-W. Purnami, H. Kuswanto, "Support vector machine for imbalanced microarray dataset classification using ant colony optimization and genetic algorithm," in *Third AIP Conference Proceedings (AIP 2019)*, IEEE, pp. 79-86, 2019.
[21] C.-B. Lu, H.-F. Ke, G.-Y. Zhang, Y. Mei, H.-H. Xu, "An improved weighted extreme learning machine for imbalanced data classification," *Memetic Computing*, vol. 11, no. 1, pp. 27-34, 2019.
[22] X. Zhang, J. Ran, J. Mi, "An intrusion detection system based on convolutional neural network for imbalanced network traffic," in *7th International Conference on Computer Science and Network Technology (ICCSNT 2019)*. IEEE, pp. 456-460, 2019.
[23] B. Yan, G. Han, "LA-GRU: building combined intrusion detection model based on imbalanced learning and gated recurrent unit neural network," *Security and Communication Networks*, vol. 2018, 6026878, 2018.
[24] S. Kanai, Y. Fujiwara, S. Iwamura, "Preventing gradient explosions in gated recurrent units," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*. IEEE, pp. 1-30, 2017.
[25] S. Kumar, A. Damaraju, A. Kumar, S. Kumari, C.-M. Chen, "LSTM network for transportation mode detection," *Journal of Internet Technology*, vol. 22, no. 4, pp. 891-902, 2021.
[26] M.-E. Wu, J.-H. Syu, C.-M. Chen, "Kelly-based options trading strategies on settlement date via supervised learning algorithms," *Computational Economics*, vol. 59, no. 4, pp. 1627-1644, 2022.
[27] Z. Zhang, "Improved adam optimizer for deep neural networks," in *26th International Symposium on Quality of Service (SQS 2018)*, IEEE, no. 1-2, 2018.