Liu et al. EURASIP Journal on Wireless Communications and Networking (2018) 2018:77
https://doi.org/10.1186/s13638-018-1078-5

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

GA-AdaBoostSVM classifier empowered wireless network diagnosis

Xuewen Liu*, Gang Chuai, Weidong Gao and Kaisa Zhang

Abstract

Self-healing is one of the most important parts in self-organizing mobile communication network. It focuses on detecting the decline of service quality and finding out the cause of network anomalies and repairing it with high automation. Diagnosis is a particularly important task which identifies the fault cause of problematic cells or regions. To perform the diagnosis, this paper presents two modified ensemble classifiers by using Support Vector Machine (SVM) with different kernels, i.e., SVM with the radial basis function (RBF) kernel (RBFSVM in short) and SVM with the linear kernel (LSVM in short), as component classifier in Adaptive Boosting (AdaBoost), and we call the two ensemble classifiers as Adaptive Boosting based on RBFSVM (AdaBoostRBFSVM in short) and Adaptive Boosting based on linear kernel (AdaBoostLSVM in short). Different with previous AdaBoostSVM classifiers using weak component classifiers, in this paper, the performance of the classifiers is adaptively improved by using moderately accurate SVM classifiers (the training error is less than 50%). To solve the accuracy/diversity dilemma in AdaBoost and get good classification performance, the training error threshold is regulated to adjust the diversity of classifier, and the parameters of SVM (regularization parameter C and Gaussian width σ) are changed to control the accuracy of classifier. The accuracy and diversity will be well balanced through reasonable parameter adjustment strategy. Results show that the proposed approaches outperform individual SVM approaches and show good generalization performance. The AdaBoostLSVM classifier has higher accuracy and stability than LSVM classifier. Compared with RBFSVM, the undetected rate and diagnosis error rate of AdaBoostRBFSVM decrease slightly, but the false positive rate does reduce a lot. It means that the AdaBoostRBFSVM classifier is indeed available and can greatly reduce the number of normal class samples that have been wrongly classified. Therefore, the two ensemble classifiers based on the SVM component classifier can improve the generalization performance by reasonably adjusting the parameters. To set the parameter values of component classifiers in a more reasonable and effective way, genetic algorithm is introduced to find the set of parameter values for the best classification accuracy of AdaBoostSVM, and the new ensemble classifier is called AdaboostSVM based on genetic algorithm (GA-AdaboostSVM in short) (including AdaboostLSVM based on genetic algorithm and AdaboostRBFSVM based on genetic algorithm). Results show that GA-AdaboostSVM classifiers have a lower overall error than AdaboostSVM classifiers. Genetic algorithm could help to achieve a more optimal performance of the ensemble classifiers.

Keywords: Diagnosis, AdaBoostSVM, GA-AdaboostSVM, Self-organizing networks (SONs)

* Correspondence: xuewen_liu1990@bupt.edu.cn
Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, People's Republic of China

1 Introduction

Over the past few years, the wireless network has undergone great changes. The coexistence of 2G, 3G, LTE/LTE-A, and HetNet architecture makes the wireless network more and more complex. A sharp increase in the traffic demand has forced the operator to increase CAPital EXpenditures (CAPEX) and OPerational EXpenditures (OPEX). In order to reduce operating and maintenance costs, the self-organizing network (SON) [1] has been introduced by 3GPP. Self-organizing networks (SONs), a set of principles and concepts for increasing the automation of mobile networks, automatically choose the network parameters to improve the key performance indicators (KPIs). Three categories, self-configuration, self-optimization, and self-healing [2], have been involved in SONs. Self-configuration includes automatic planning and deployment of the network, such as self-establishing base stations and automatic management during the operation of the base station. Self-optimization refers to adaptively adjusting the parameters of network equipment according to its own operating conditions in order to achieve the goal of optimizing network performance. Self-healing, the ability to automatically recover from failures, includes detection, diagnosis, and recovery. This work is centered in diagnosis which identifies the fault cause of problematic cells or regions.

Recently, some research on network diagnosis has been published [3–6]. However, the number of papers on self-healing is limited due to two major reasons. One of the reasons is that the fault causes and the corresponding KPIs are not recorded when fault occurs. The other reason is that historical data of faults in mobile networks is usually in the hands of operators, and it is usually hard for the scientific community to get. In view of the above problems, some scholars use simulators to simulate faults and corresponding network KPIs, but there is a big difference from the real network settings [7]. In spite of this, many significant projects have been developed, such as the UniverSelf Project [8], the COMMUNE Project [9], and the SELFNET Project [10]. There have been quite a few researches on network diagnosis, most of which apply new concepts and techniques, such as data mining [3, 11], self-organizing maps [4], genetic algorithms [5], fuzzy logic [6], and Bayesian networks [12, 13], to diagnose faults in communication network. But there is little research based on Machine Learning [14–20] for network diagnosis. In this paper, several supervised Machine Learning (ML) techniques, i.e., Support Vector Machine (SVM), Adaptive Boosting based on SVM (AdaBoostSVM), and AdaboostSVM based on genetic algorithm (GA-AdaBoostSVM), have been used for diagnosis in network.

Support Vector Machine (SVM) evolves from the optimal classification of linearly separable cases. The optimal classification surface requires that the classification surface not only correctly separates the two classes (the training error rate is 0), but also makes the classification interval the largest. In order to get a good classification effect, kernel functions were usually used to map the training samples to a high-dimensional feature space. There are many kernel functions, such as linear kernel, radial basis function (RBF) kernel, and polynomial kernel, which were commonly used in the SVM. Among them, two popular kernels used in SVM are the RBF and linear kernels, which respectively have a parameter known as regularization parameter C and Gaussian width σ. The parameters are used to control the model complexity and training error.

Adaptive Boosting (AdaBoost) [21] is one of the ensemble learning algorithms, which improves the performance of the ensemble classifier by improving the accuracy of the weak classifier. The weight coefficients of each classifier are set to be the same before starting the iteration. After each iteration, the weight coefficients of each classifier will be adaptively adjusted according to the classification results. The weights of misclassified samples will be increased; on the contrary, the weights of correctly classified samples will be decreased. Many researches that use Decision Trees [22], Neural Networks [23], or RBFSVM [17] as component classifiers in AdaBoost have been investigated. To the best of our knowledge, there are few researches using linear kernel as component classifiers in AdaBoostSVM. It is well known that there is a dilemma of accuracy/diversity in AdaBoost, which means that the more accurate the two component classifiers, the less disagreement between them. AdaBoost can demonstrate excellent generalization performance only if accuracy and diversity are well balanced. Therefore, how could we balance the accuracy/diversity dilemma in AdaBoostSVM?

In this paper, we try our best to find solutions to the following problems: Can we use the component classifiers based on linear kernel or RBF kernel to get better generalization performance in AdaBoostSVM? If we can, which classifier based on the different kernels could get better performance and why? How could we balance the accuracy/diversity dilemma in AdaBoostSVM? How could we set the parameter values of component classifier in a reasonable and effective way?

As mentioned above, there are two parameters σ and C in Adaptive Boosting based on RBFSVM (AdaBoostRBFSVM) and one parameter C in Adaptive

Boosting based on linear kernel (AdaBoostLSVM) which have to be set beforehand. According to the performance analysis of RBFSVM [17], we know that σ is a more important parameter than C: the performance of RBFSVM mainly depends on the value of σ in the proper range of C. As known in [17], if all RBFSVM component classifiers are set to a single σ, it will result in an unsuccessful AdaBoost process due to the reason that over-weak or over-strong component classifiers may appear. So, in this paper, the proposed AdaBoostRBFSVM method adaptively adjusts the value of σ in the RBFSVM component classifier to obtain a set of moderately accurate RBFSVMs for AdaBoost. Similarly, the C values in LSVM component classifiers are also adaptively adjusted. It means we adjust the accuracy of the classifier by changing the values of the parameters C and σ. Furthermore, as mentioned above, there is a dilemma of accuracy/diversity in AdaBoost. Therefore, we increase the diversity of the classifier by increasing the training error threshold. The greater the training error threshold, the more the weak classifiers satisfying the condition will be obtained, so that the diversity of the classifier will be better.

The performance of the ensemble classifier depends on the parameters value of each component classifier. How to set the parameter value of the component classifier in a reasonable and effective way is a very important issue. Genetic algorithm is a method of searching for the optimal solution, which is largely used in search and optimization problems. In this paper, genetic algorithm is proposed to find the set of parameter values for the optimal performance of the ensemble classifiers.

In this paper, two modified ensemble classifiers, i.e., AdaBoostRBFSVM and AdaBoostLSVM, were employed for root cause analysis in network by using the cases from [6]. The cases for training and validation were generated by the real LTE network. Each case includes the information on Cause-KPI (key performance indicators) relations, which will be used for training and validating the model generated by AdaBoostSVM with different kernels. By using genetic algorithm to optimize parameters and control training error threshold, a good balance on accuracy/diversity will be achieved. Since SVM and AdaBoost were originally designed for binary classifier, in this paper, OAO (One Against one) approach was used for classifiers to generate a multi-classifier to train the model. The results show that the two proposed algorithms based on AdaBoostRBFSVM and AdaBoostLSVM can automatically diagnose different classes of network anomalies with high accuracy, low diagnosis error rate, low false positive rate, and low undetected rate. Genetic algorithm is used to find the set of parameter values for the optimal accuracy of the ensemble classifier. GA-AdaboostSVM classifiers outperform AdaboostSVM classifiers with a lower overall error. Therefore, the genetic algorithm could help the AdaBoostLSVM classifier to obtain the optimal performance.

The main contributions of this paper are as follows:

1. Proposed two modified ensemble learning algorithms using LSVM and RBFSVM as component classifier, i.e., AdaBoostLSVM and AdaBoostRBFSVM, to improve wireless network troubleshooting performance.
2. Proposed a new method to solve the accuracy/diversity dilemma in AdaBoost to obtain the optimal performance of the AdaBoostSVM.
3. Genetic algorithm is used to get the best classification accuracy of AdaBoostSVM.
4. The diversity of the AdaBoostSVM is regulated by changing the training error threshold.

2 Problem formulation

There are three main tasks in the process of troubleshooting: detection, diagnosis, and recovery. This work is centered in diagnosis with the cases provided in [6]. The following sections provide the knowledge necessary to understand the diagnosis system, such as performance metrics, fault causes, and related KPIs.

2.1 KPIs

KPI is an indicator that reflects network performance. The statistics and calculations of abnormal KPI value which is lower or higher than a certain threshold can reflect the network performance of a cell or part of region. In this paper, seven common KPIs including Retainability, Handover Success Rate (HOSR), Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ), Signal to Interference Noise Ratio (SINR), Distance, and Average throughput were calculated for later diagnosis.

2.2 Fault causes

A network failure can cause an abnormality in the KPI indicator. The causes of mobile network failure can generally be divided into three categories, including coverage, mobility, and interference. In this paper, the data set is provided in [6], which is generated by the real LTE network. Six fault causes were selected, such as excessive uptilt (EU), excessive downtilt (ED), reduction in cell power (RP), coverage hole (CH), mobility, and intersystem interference (II). For more explanations about KPIs and fault causes in this paper, please refer to reference [6].

2.3 Performance metrics

The accuracy of the case diagnosis is utilized to assess the diagnostic performance of the system. The higher the correct rate is, the better the system performance will be. Seven metrics were calculated to evaluate the performance of the diagnosis system.

(a) Diagnosis error rate (E_d): The proportion of misdiagnosed cases in the total number of cases. It shows the accuracy of the classifier.
(b) Undetected rate (E_u): The proportion of fault cases diagnosed as normal cases in the total number of fault cases. It shows the reliability of the classifier.
(c) False positive rate (E_{fp}): The ratio of normal cases diagnosed as fault cases to the total number of normal cases. It shows the availability of the classifier.
(d) Total error rate (E_p): The sum of diagnosis error rate (DER) and Undetected rate (UDR). It is given by $E_p = E_d + E_u$.
(e) Overall error (E): The probability that misdiagnosis occurs. It is given by $E = P_n \cdot E_{fp} + P_p \cdot E_p$, where P_n and P_p are the percentage of normal and fault cases in the validation set, respectively.
(f) Complementary of the Positive Predictive Value (P_{fp}): The probability that a given positive diagnosis is a false positive, which indicates the importance of a low false positive rate. High P_{fp} makes the system unreliable because too many of the fault cases that are diagnosed are not real. It is given by $P_{fp} = \frac{P_n \times E_{fp}}{P_n \times E_{fp} + P_p \times (1-E_u)}$.
(g) Confusion matrix: The confusion matrix is used to compare the mapping probabilities between the classification result and the true value. Each column of the confusion matrix represents a prediction category of data, and each row represents the true category of data.

3 Fault management based on Machine Learning

3.1 Support Vector Machine

SVM is a dichotomous model whose main idea is to find the separating hyper-plane that can correctly classify the training set and maximize the geometric interval. The decision function of SVM can be expressed as $f(\mathbf{x}) = <\mathbf{w}, \quad \phi\ (\mathrm{x})> + \ b$, where $\phi(\mathbf{x})$ represents the mapping of the input sample x to a high-dimensional space [20]. <·, ·> denotes the dot product in the feature space. The optimal $\mathbf{w}$ and b can be solved by solving the following formula:

$$\text{minimize} : g(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\boldsymbol{\xi}_i \tag{1}$$

$$\text{subject to} : y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b) \geq 1 - \boldsymbol{\xi}_i, \quad \boldsymbol{\xi}_i \geq 0, \tag{2}$$

where $\boldsymbol{\xi}_i$ is the ith slack variable and C is the regularization parameter. According to the Wolfe dual form, the above minimization problem can be written as:

$$\text{minimize} : W(\alpha) = -\sum_{i=1}^{N}\alpha_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{3}$$

$$\text{subject to} : \sum_{i=1}^{N} y_i \alpha_i = 0, \quad \forall i : 0 \leq \alpha_i \leq C, \tag{4}$$

where α_i is a Lagrange multiplier which corresponds to the sample $\mathbf{x}_i$ and $k\langle\cdot,\cdot\rangle$ and $k(\cdot,\cdot)$ are kernel functions mapping all input vectors into an appropriate feature space $k(\mathbf{x}_i, \mathbf{x}_j) = \langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle$. The linear kernel function is expressed as $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$, and the RBF kernel function is expressed as $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)$. By applying the kernel function, the sample is mapped linearly to the high-dimensional feature space. In this space, the optimal separating the hyper-plane is constructed via SVM. Platt's sequential minimal optimization (SMO) [19] has been widely used for solving the SVM problem. SMO is a fast iterative algorithm, which decomposes a large QP (quadratic programming) problem into several QP sub-problems of the minimum size. Each QP sub-problem has only two variables. For this small QP sub-problem, the analytic solution can be found, so that the training speed gets faster.

3.2 AdaBoost

AdaBoost is one of the ensemble learning algorithms, which improves the performance of the ensemble classifier by boosting the accuracy of the weak classification classifier. After each iteration, the weight of classifier will be changed according to the classification results. If the classification result is wrong, the weight will be increased; otherwise, the weight will be reduced. The bigger the training error is, the smaller the weight will be. Finally, all the classifiers will be linearly combined to compose the final classifier.

3.3 Genetic algorithm

Genetic algorithm (GA) is a computational model that simulates the biological evolutionary process of natural selection and genetics of Darwin's biological evolution [24, 25]. It is a method of searching for the optimal solution by simulating the natural evolutionary process. According to the principle of survival of the fittest, the genetic algorithm first generated an initial population of potential solution sets and then evolved generation after generation to get better and better approximate solution. At each generation, individuals were selected based on the fitness of individuals in the problem domain. Crossover and mutation were used to generate individuals that represented new potential solutions. The flow chart of GA is shown in Fig. 1.

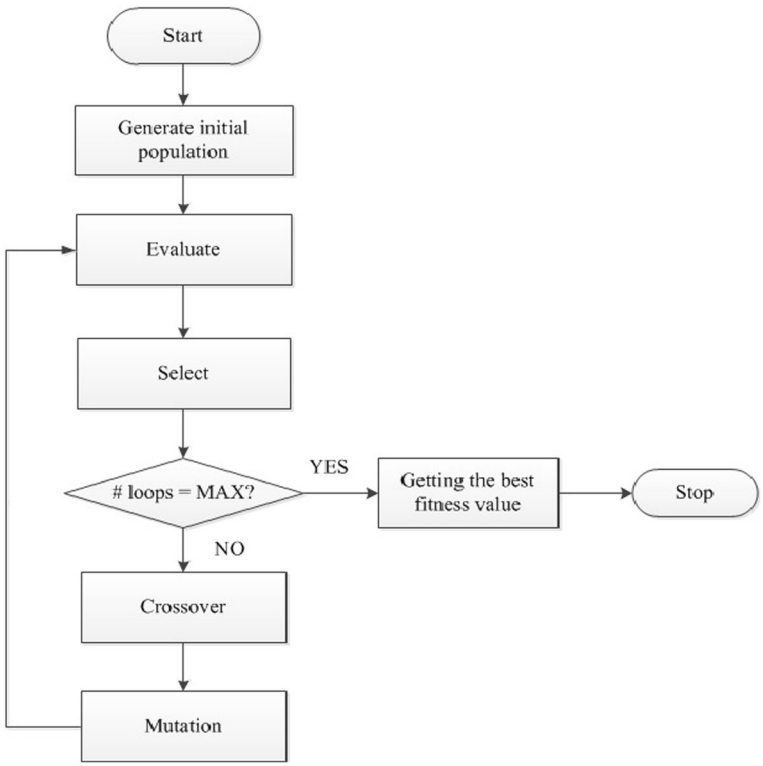

Fig. 1 Flowchart of the genetic algorithm

3.4 Proposed algorithm: AdaBoostRBFSVM

In this part, RBFSVM classifier was employed as component classifier in AdaBoost. Before the AdaBoost iterations, it is the most important problem on setting the σ and C values for these RBFSVM component classifiers. According to RBFSVM performance analysis in [26], we know that σ is a more important parameter that affects the performance of a classifier than C. If a roughly suitable C is given, the performance of RBFSVM classifier is largely determined by the σ. It is known that setting a too large σ will get a too weak RBFSVM component classifier. On the contrary, a too small σ will make the RBFSVM component classifier too strong to boost it. As known in [17], giving all RBFSVM component classifiers a single σ value, the boosting process will be unsuccessful. Therefore, in this paper, the σ value will be adaptively adjusted to obtain a set of moderately accurate RBFSVM component classifiers. AdaBoostRBFSVM can be described as follows (Algorithm 1):

Firstly, weak RBFSVM classifiers are generated by setting a large σ value, and the weights of training samples are initialized to the same value.

Then, the weak RBFSVM classifiers with an initial σ value are trained on the weighted training set. The training error of RBFSVM is calculated, on which based different operations are performed. If the training error is more than the threshold ε_{th}, the σ value will be decreased slightly by σ_{step} and go back to step 3. Otherwise, the weights of RBFSVM classifiers will be set, and the weights of training samples will also be updated to calculate the training error for the next iteration. Slightly decreasing the σ value, we can prevent the new RBFSVM from being too strong for the current weighted training samples. Different from the AdaBoostSVM in [17] with the fixed training error value ($\varepsilon_{th} = 0.5$), in this paper, we adjust the diversity of the classifier by changing the training error threshold. The greater the training error threshold, the more the weak classifiers satisfying the condition will be obtained, so that the diversity of the classifier will be better. Therefore, by reasonably adjusting the values of σ and ε_{th}, the accuracy/diversity dilemma can be balanced and the optimal parameter configuration of the classifier is obtained.

Furthermore, the weights of training samples will be adaptively adjusted by the classified results, i.e., component classifiers with lower training errors will gain greater weights, and component classifiers with higher training errors will get smaller weights. This process will finish when the σ is less than the given minimal value.

Finally, AdaBoost makes a linear combination of all component classifiers into a single final hypothesis f.

Algorithm 1: AdaBoostRBFSVM

1. Input: The training samples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_N,y_N)\}$; the initial σ, σ_{ini}; the minimal σ, $\sigma_{\min}$; the step of σ, σ_{step}; The regularization parameter C, and the training error threshold ε_{th}.
2. Initialize: The weights of training samples w_i^1 are initialized to the same value, where $w_i^1=1/N\ (i=1,2,\ldots,N)$.
3. Do while($\sigma > \sigma_{\min}$)

(1) The RBFSVM component classifier, h_t, is trained based on the weighted training set.

(2) The training error of h_t is calculated by the formula: $\varepsilon_t = \sum_{i=1}^{N} w_i^t, y_i \neq h_t(\mathbf{x}_i)$.

(3) If $\varepsilon_t > \varepsilon_{th}$, decrease σ value by σ_{step} and go to step 3.

(4) Set weight for the component classifier h_t: $\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$.

(5) Update the weights of training samples: $w_i^{t+1} = \frac{w_i^t \exp\{-\alpha_t y_i h_t(\mathbf{x}_i)\}}{C_t}$,

$i = 1,\ldots,N$, where C_t is a normalization constant, and $\sum_{i=1}^{N} w_i^{t+1} = 1$.

4. Output: $f(\mathbf{x}) = sign(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}))$.

3.5 Proposed algorithm: AdaBoostLSVM

This section aims at employing LSVM as component classifier in AdaBoost. Similar to the AdaBoostRBFSVM, it is important to set the C value for these LSVM component classifiers during the AdaBoost iterations. It is known that the value of C represents the importance of outliers to the classifier. The larger C represents more attention will be paid to the outliers, which means that they cannot be easily ignored. Increasing the value of C can always achieve the correct classification of the training samples, but this will lead to over-fitting and bad generalization performance. On the contrary, continuously decreasing the value of C will result in under-fitting. Obviously, if all LSVM component classifiers were set to a single C value, the boosting process will be unsuccessful. Therefore, in this paper, a set of moderately accurate LSVM component classifiers will be obtained by adaptively adjusting the C value. AdaBoostLSVM can be described as follows (Algorithm 2):

Firstly, weak LSVM classifiers are generated by setting a small C value, which means that the LSVM classifiers have weak learning ability, and the weights of training samples are initialized to the same value.

Then, LSVM with this C is trained in as many cycles as it can get in less than a training error threshold ε_{th}. Otherwise, this C value is increased slightly to enhance the learning capability of LSVM to help it achieve less than the training error threshold ε_{th}. Similar to AdaBoostRBFSVM, we adjust the diversity of the classifier by changing the training error threshold. Through regulating the values of C and ε_{th} reasonably, we can balance the accuracy/diversity dilemma and get the optimal parameter configurations of classifier.

Furthermore, the weights of training samples will be adaptively adjusted by the classified results. This process continues until the C is increased to the given maximal value.

Finally, AdaBoost makes a linear combination of all component classifiers into a single final hypothesis f.

3.6 Proposed algorithm: GA-AdaBoostSVM

The principle of AdaBoost is to linearly combine multiple component classifiers into ensemble classifier. The value of parameters (C and σ) plays a big role in the performance of the component classifier during the AdaBoost iterations. Different values of parameters will get different component classifiers, resulting in different performance of the ensemble classifier. Therefore, the performance of the ensemble classifier depends on the parameter value of each component classifier. Although the AdaBoostSVM algorithm can achieve good classification performance, it needs to set the value of C_{ini}, C_{step}, σ_{ini}, and σ_{step} in advance. Therefore, how to set the parameter values of component classifier in a reasonable and effective way is a very important issue. Genetic algorithm is a method of searching for the optimal solution, which is largely used in search and optimization problems. In this paper, genetic algorithm is used to find the optimal set of parameter values of the ensemble classifier. The GA-AdaBoostSVM

Algorithm 2: AdaBoostLSVM

1. Input: The training samples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_N,y_N)\}$; the initial C, C_{ini}; the maximal C, $C_{\max}$; the step of C, C_{step}; The training error threshold ε_{th}.
2. Initialize: The weights of training samples w_i^1 are initialized to the same value, where $w_i^1=1/N\ (i=1,2,\ldots,N)$.
3. Do while($C \le C_{\max}$)

   (1) The LSVM component classifier, h_t, is trained based on the weighted training set.

   (2) The training error of h_t is calculated by the formula: $\varepsilon_t = \sum_{i=1}^{N} w_i^t, y_i \ne h_t(\mathbf{x}_i)$.

   (3) If $\varepsilon_t > \varepsilon_{th}$, increase C value by C_{step} and go to step 3.

   (4) Set weight for the component classifier h_t: $\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$.

   (5) Update the weights of training samples: $w_i^{t+1} = \frac{w_i^t \exp\{-\alpha_t y_i h_t(\mathbf{x}_i)\}}{C_t}$,

$i = 1,\ldots,N$, where C_t is a normalization constant, and $\sum_{i=1}^{N} w_i^{t+1} = 1$.

4. Output: $f(\mathbf{x}) = sign(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}))$.

---

algorithm is showed in Algorithm 3. With different kernel functions, the GA-AdaboostSVM is abbreviated as GA-AdaboostLSVM and GA-AdaboostRBFSVM.

Algorithm 3: GA- AdaBoostSVM

1. Input: The training samples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_N,y_N)\}$; The training error threshold ε_{th}. The population size (p_{size}), population number (p_{num}), mutation rate (p_m), crossover rate (p_c) and maximum iterations (n_{max}).
2. Initialize: Initialize the individuals of each population by defining random values to the genes.
3. Do while(loops< n_{max})
   (1) Evaluate the fitness value of each population with **AdaBoostSVM** algorithm.
   (2) Select the population by roulette wheel to compose the population sets for the next generation.
   (3) Perform crossover by single-point crossover operator according to crossover rate.
   (4) Perform mutation according to mutation rate.
4. Output: The best fitness value.

3.7 Multi-classifier based on binary classification

Since SVM and AdaBoost were originally designed for binary problems, several methods were proposed to extend binary classifier to solve multi-classification problems. One approach is to decompose the multi-classification problem into multiple binary classification problems, and then, the classification result of each binary classifier is combined to obtain the final classification result. There are several commonly used multi-classification methods based on binary classifier, such as OAA (One Against All), OAO (One Against one), and DAG (directed acyclic graph) [27–29].

The OAA method is to classify the samples of one category into one class, and the rest of the samples are classified as another one. In this way, samples of k categories construct k classifiers. The classification result is to classify the unknown sample into the class with the maximum value of the classification function. The advantage of this method is that for the k classification problem, only k binary classifiers need to be trained, so the number of the classification functions (k) obtained is less, and the classification speed is relatively fast. The disadvantage is that it will cause imbalances in the categories, which greatly affect classification accuracy. Therefore, it is not very practical.

The OAO method is to design a classifier between any two classes of samples, so $k(k-1)/2$ classifiers need to be designed for samples of k classes. The classification result is to classify the unknown sample into the class with the maximum value of the classification function. The advantage of this method is that the training accuracy is relatively high, but the classification speed is slow and it takes high cost.

Similar to the OAO method, the DAG method also needs to construct $k(k-1)/2$ binary classifiers and obtain the corresponding decision functions of these classifiers. However, in classification, the DAG method is classified by constructing a "binary directed acyclic graph" with a root node. The "binary directed acyclic graph" has $k(k-1)/2$ internal nodes and k leaf nodes. Each internal node corresponds to a binary classifier, and each leaf node corresponds to a class.

For OAO and DAG methods, OAO is generally considered to be slightly more accurate than DAG for the same training time, but the testing time of DAG is slightly lower or the same. The most commonly used multi-classification methods are the OAO and OAA methods, but the OAO method is more suitable for practical applications. Therefore, in this paper, OAO approach was used for each binary classifier to train the multi-classification model. The algorithm of multi-classifier based on binary classification is showed in Algorithm 4.

4 Evaluation

4.1 Case study

In this work, the training and validation cases are provided in [6]. The training data is a set of cases following the format described in Fig. 2. A case is a vector consisting of multiple KPIs and corresponding fault cause.

4.2 Experimental design

There are 550 cases in the training set and 4009 cases in the validation set. The distribution of fault causes in the training set is showed in Fig. 3, and the sample distribution of the validation set is comparable to that of the training set. All the algorithms were firstly trained with the training cases, and afterward, these were tested with the validation cases, and five performance metrics (E_d, E_u, E_{fp}, E, and P_{fp}) were calculated. To compare the advantages of the proposed algorithms with other algorithms in generalization performance, the same performance metrics were calculated by other algorithms in the same training and validating set. In order to

---

Algorithm 4: Generating multi- classifier

1. Input: The training samples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_N,y_N)\}$; the testing samples $\{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_O,y_O)\}$; choose the machine learning algorithm (SVM, AdaBoostSVM or GA-AdaBoostSVM) and set the parameters.
2. Initialize: Set the number of classes M, and combine any two classifiers to be one group. The total number of different group is W, and W equal to $M(M-1)/2$.
3. Do for $i = 1,\ldots,W$
   (1) A binary classifier is trained.
   (2) Do for $j = 1,\ldots,O$
   Each testing sample will be classified by the trained binary classifier. The result of the sample j being classified by the classifier i is recorded as C_j^i, where $C_j^i = 1,\ldots,M$.
4. Do for $j = 1,\ldots,O$
   The classified result is judged by a "Max wins" strategy, on which based a majority voting scheme.

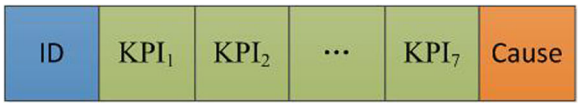

Fig. 2 Format of data

improve the accuracy of the classification results, this paper tests the validation set 100 times and takes the average. The main parameters of the algorithm for testing and evaluation are shown in Table 8 (Appendix).

4.3 Results and discussion

4.3.1 Evaluation based on LSVM

The parameter C has a great influence on the performance of the LSVM classifier. If C is small, the classifier will be under-fitting. On the contrary, the classifier will be over-fitting. So, we tested the performance of the classifier with different C values and got the optimal C values. The diagnosis error rate (DER), undetected rate (UDR), false positive rate (FPR), Overall error (OE), and Complementary of the Positive Predictive Value (PFP) with different C values were shown in Fig. 4. It can be seen that as the C value increases, the five metrics are reduced firstly and then increased and maintained in a relatively stable range. It obtains the minimum OE and PFP value at $C = 0.5$.

4.3.2 Evaluation based on RBFSVM

We know that the parameter C and σ have a great influence on the performance of the RBFSVM classifier. Given a roughly suitable C, the performance of the RBFSVM classifier is largely determined by the σ value which also influences the complexity of classifier. With a larger σ, the complexity of classifier often decreases and it gets bad classification performance. Conversely, the complexity of classifier increases and good classification performance will achieve a small σ value. Several performance metrics with different σ values are shown in Fig. 5 where C is set to be 1. It can be seen that the minimum OE will be obtained when $\sigma = 3$. Compared with LSVM, the UDR of RBFSVM was significantly higher. It means that the LSVM classifier can decrease the number of minority class samples that are misclassified.

4.3.3 Evaluation based on AdaBoostLSVM

It is known that the good generalization performance cannot be gotten by using a too large or too small value of C. Simply applying a single C to all LSVM component classifiers cannot lead to successful AdaBoost due to the over-fitting or under-fitting situations encountered in the Boosting process. So, in this section, the proposed AdaBoostLSVM approach adaptively adjusts the C value in LSVM component classifiers to obtain a set of moderately accurate LSVMs for AdaBoost. In order to increase the diversity of the classifier, we change the training error threshold from 0.01 to 0.5 and several performance metrics with different ε_{th} and C were calculated. Table 1 shows several performance metrics with different ε_{th} on the optimal C values. It can be seen that we could get the optimal parameter configurations of classifier through regulating the values of C and ε_{th} reasonably. Figure 6 shows the best performance metrics with optimal C values and training error threshold ε_{th}. Compared with LSVM, all performance metrics of AdaBoostLSVM have a significant improvement. The overall error is reduced to 6.8% and the complementary of the Positive Predictive Value is reduced to 4.1%. It means that the AdaBoostLSVM classifier has higher accuracy and stability than LSVM. Therefore, the ensemble classifier based on LSVM component classifier could boost the generalization performance through regulating the values of C and ε_{th} reasonably.

Table 2 shows the normalized confusion matrix of AdaBoostLSVM method with optimal parameter values. The diagonal of the matrix represents the diagnosis success rate of each problem in the system. It can be seen that more than half accuracy can be obtained for each problem, which illustrates the diagnosis system is indeed available. The normal cases have the highest diagnosis success rate, which demonstrates the diagnostic system has high availability with a lower FPR. However, the diagnostic accuracy of II, CH, and TLHO is relatively low, and the probability that each fault is misdiagnosed as normal is higher than the probability of being misdiagnosed as another fault, which respectively corresponds to a high DER and UDR.

4.3.4 Evaluation based on AdaBoostRBFSVM

According to the previous analysis, we know that σ is a more important parameter compared to C: the

ID	Retainability	HOSR	RSRP	RSRQ	SINR	Throughput	Distance	Fault Cause
1	0.85	0.61	-66.73	-20.04	10.84	148.69	0.71	1
2	0.91	0.8	-61.82	-19.07	15.24	205.65	0.6	2
...	...	...	...	...	...	...	...	...

Fig. 3 Distribution of the collected faults in training set

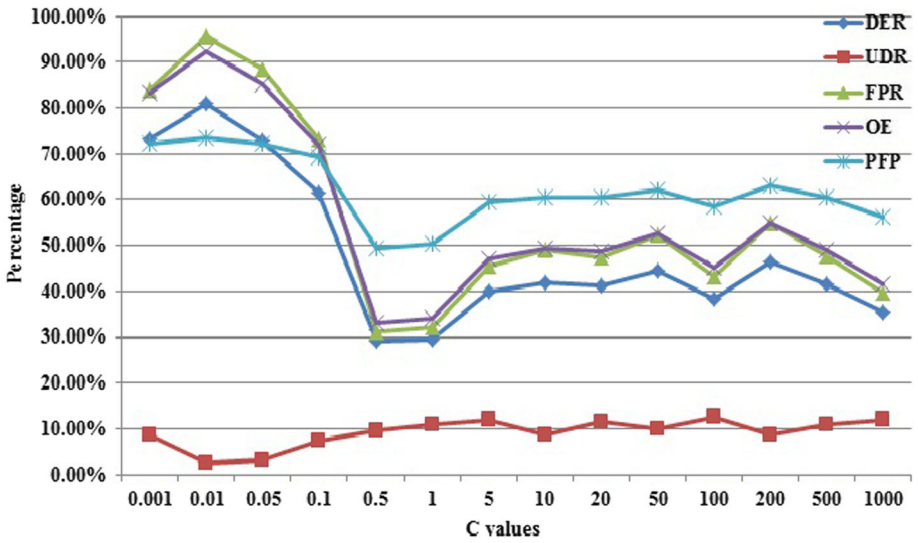

Fig. 4 The performance metrics of LSVM with different *C* values

performance of classifier is largely determined by σ. For comparison with RBFSVM, we tested several performance metrics with different σ values when C was set to be 1. In order to increase the diversity of the classifier, we changed the training error threshold from 0.01 to 0.5 and several performance metrics with different ε_{th} and σ were calculated. Table 3 shows several performance metrics with different ε_{th} on the optimal σ values. It can be seen that we could get the optimal parameter configurations of classifier through regulating the values of σ and ε_{th} reasonably. Figure 7 shows the best performance metrics with optimal σ values and training error threshold ε_{th}.

False positive rate (FPR) suggests the ability to filter out normal cases. High FPR indicates that the diagnostic systems are not available because of the high probability of false positives. Compared with RBFSVM, although the UDR and DER are only slightly reduced, the FPR does reduce a lot. It means the AdaBoostRBFSVM classifier is indeed usable and could largely reduce the number of normal cases being misclassified. Furthermore, the OE and PFP also decrease, indicating that the AdaBoostRBFSVM classifier has higher accuracy and reliability compared to RBFSVM. Compared with AdaBoostLSVM, AdaBoostRBFSVM shows a significantly higher UDR. It means that the AdaBoostLSVM classifier, which is the same as LSVM, also can decrease the number of minority class samples that are misclassified.

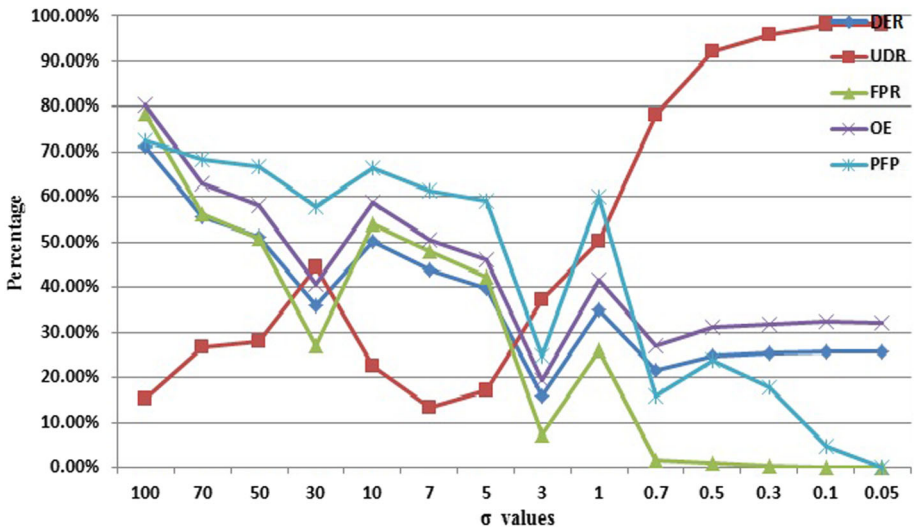

Fig. 5 The performance metrics of RBFSVM with different σ values

Table 1 Several performance metrics of AdaBoostLSVM classifier with different ε_{th} on the optimal C values

ε_{th}	DER	UDR	FPR	*C_optimal*	OE	PFP
0.01	0.419	0.19	0.407	0.3	0.459	0.587
0.05	0.0645	0.16	0.0129	3	0.068	0.041
0.1	0.0713	0.196	0.0105	1	0.077	0.035
0.2	0.072	0.137	0.03	1	0.076	0.089
0.3	0.0998	0.186	0.0444	0.5	0.107	0.133
0.4	0.106	0.178	0.057	0.5	0.116	0.166
0.5	0.0997	0.144	0.0645	3	0.111	0.176

Table 4 illustrates the normalized confusion matrix of AdaBoostRBFSVM method with optimal parameter values. Compared with Table 2, the diagnosis success rate of normal cases is increased, but the diagnostic accuracy of other problems is decreased a lot. The diagnostic accuracy of CH is reduced to 35.92%, which shows bad diagnosis performance. Furthermore, with a high UDR and DER, a significant increase appears in the probability that each fault is misdiagnosed as normal or other faults. Generally, the performance of AdaBoostRBFSVM is worse than that of AdaBoostLSVM.

4.3.5 Evaluation based on GA-AdaBoostSVM

The difference between this algorithm and the traditional genetic algorithm is that the best parameter we find in this paper is a set of numerical values rather than a single numerical value. Therefore, in the initial population stage, different individuals in multiple populations are randomly assigned different numerical sizes. The individual sets in each population represent a set of potential optimal solutions. At each generation, the fitness value of each population was calculated and used to select the population for the next generation by roulette wheel selection method. In order to prevent the solution set from falling into local optimal, crossover and mutation were used to generate populations that represented new sets of solutions. The basic process of genetic algorithm is summarized as follows:

1. Initial population: Each population is a possible solution to the problem. Each individual of the population is randomly selected and coded as binary bits. For the GA-AdaBoostRBFSVM classifier, C and σ values are both coded as binary bits. Only C value is coded as binary bits in GA-AdaBoostLSVM. In this paper, multiple populations were introduced to obtain sets of parameter values that optimize the objective function. Each population contains the same number of individuals representing different parameter values. In this paper, 100 populations are initially generated,

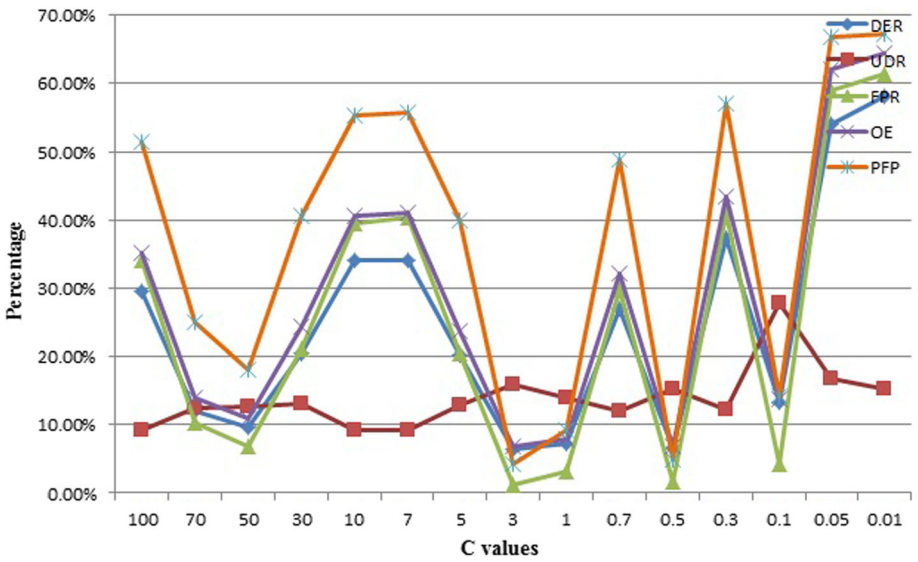

Fig. 6 The performance metrics of AdaBoostLSVM with optimal ε_{th} value ($\varepsilon_{th} = 0.05$)

Table 2 Normalized confusion matrix of AdaBoostLSVM method with optimal parameter values

Predicted cause								
Real cause		ED	CH	II	TLHO	EU	RP	Normal
	ED	**0.9009**	0.0047				0.0094	0.0849
	CH		**0.6213**	0.0679	0.0194			0.2912
	II		0.0377	**0.6698**	0.0566			0.2358
	TLHO		0.0245	0.0196	**0.7009**	0.0343		0.2205
	EU			0.0424		**0.8396**	0.0141	0.1037
	RP	0.0048			0.0144		**0.8509**	0.1298
	Normal	0.0006	0.0020	0.0010	0.0087	0.0003		**0.9871**

and each population includes 15 randomly generated individuals.

2. Evaluation: The fitness value of each population determines whether the population will survive and reproduce in future generations, which is decided by fitness function. In this paper, the overall error (OE) is used as fitness function.
3. Selection: Population with better fitness has greater probability to be selected to compose the population sets for the next generation. A selection by roulette wheel is used to choose the population sets for the next generation in this paper.
4. Crossover: Crossover refers to the operation of generating a new individual by replacing and reorganizing parts of two parental individuals. By crossing, the search power of genetic algorithms is dramatically increased. Single-point crossover operator is implemented to perform the crossover in this paper. The crossover rate is set to be 0.8.
5. Mutation: Mutation refers to the variation of certain gene values of individual strings to increase the population diversity. The mutation rate is set to be 0.1.

The parameters of genetic algorithm can be seen in Table 5. Figures 8 and 9 show the overall error variation of the GA-AdaboostLSVM and GA-AdaboostRBFSVM classifiers at different training error thresholds, respectively. As can be seen, when the number of iterations is greater than 120, the overall error no longer changes.

Figure 10 shows the minimum overall error rate of different classifiers at different training error thresholds. From this figure, we can find that, compared with AdaboostSVM, GA-AdaboostSVM has better classification performance, which can make the classifier get a lower overall error, no matter what the threshold is set to be. It can be seen that the GA-AdaboostLSVM and GA-AdaboostRBFSVM classifier will get the minimum OE separately when ε_{th} is set to be 0.05 and 0.3, which more illustrates the validity of our previous point of view: the accuracy/diversity dilemma can be solved through reasonable parameter adjustment strategy which will be more reasonable and effective by using genetic algorithm.

Tables 6 and 7 respectively illustrate the normalized confusion matrix of GA-AdaBoostLSVM and GA-AdaBoostRBFSVM method with optimal parameter values. Compared with Tables 2 and 4, the diagnosis success rate of each fault cause in Tables 6 and 7 is increased significantly at the expense of a slight drop in FPR. What is more, a significant decrease appears in the probability that each fault is misdiagnosed as normal or other faults and low DER and UDR are obtained. In comparison with Tables 6 and 7, we can see that the diagnosis success rate of each fault cause in Table 6 is higher than that of Table 7 with a slight decreasing on FPR. It demonstrates that the GA-AdaBoostLSVM classifier, with low UDR and DER and almost the same FPR, has better classification

Table 3 Several performance metrics of AdaBoostRBFSVM classifier with different ε_{th} on the optimal σ values

ε_{th}	DER	UDR	FPR	σ_optimal_optimal	OE	PFP
0.01	0.124	0.334	0.0107	10	0.127	0.043
0.05	0.124	0.364	0.006	5	0.132	0.029
0.1	0.126	0.348	0.0119	30	0.132	0.049
0.2	0.122	0.348	0.0082	50	0.128	0.034
0.3	0.121	0.344	0.00686	7	0.1265	0.028
0.4	0.131	0.369	0.0073	30	0.1359	0.0318
0.5	0.1207	0.346	0.00832	5	0.128	0.0348

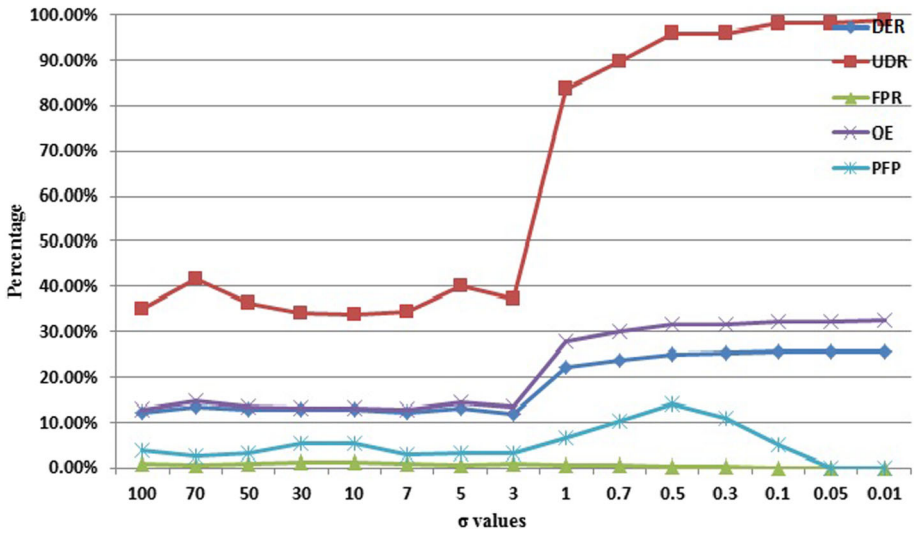

Fig. 7 The performance metrics of AdaBoostRBFSVM with optimal ε_{th} value ($\varepsilon_{th} = 0.3$)

performance than the GA-AdaBoostRBFSVM classifier in this sample set.

5 Conclusions

In conclusion, two multi-classification diagnosis systems based on AdaBoostRBFSVM and AdaBoostLSVM have been presented for mobile network self-diagnosis. Both of the two diagnosis systems can automatically detect and diagnose different classes of network anomalies with good performance. Before testing the performance of proposed approaches, the performance of individual LSVM and RBFSVM was tested firstly to find the suitable range of parameters. Then, the AdaBoostRBFSVM and AdaBoostLSVM approaches were employed to perform the diagnosis. The result shows that the two proposed approaches outperform individual SVM approaches and show good generalization performance. The AdaBoostLSVM classifier has higher accuracy and stability than LSVM classifier. Compared with RBFSVM, the UDR and DER of AdaBoostRBFSVM are only slightly reduced, but the FPR does reduce a lot. It means the AdaBoostRBFSVM classifier is indeed usable and could largely reduce the number of normal class samples being misclassified. Through some parameter-adjusting strategies, we can tune the distributions of accuracy and diversity over these component classifiers to achieve a good balance. Therefore, the ensemble classifier based on SVM component classifier could boost the generalization performance through regulating the parameters reasonably. In order to get a more accurate and effective classifier, genetic algorithm is used to make more reasonable adjustments to the classifier parameters.

In this paper, we did not consider the effect of imbalanced data on the classifier performance. So, in the next step, we will consider some data balancing methods [22], such as random oversampling, under-sampling, and synthetic minority oversampling technique (SMOTE), to reduce the impact of data imbalance on diagnostic performance.

Table 4 Normalized confusion matrix of AdaBoostRBFSVM method with optimal parameter values

Predicted cause								
Real cause		ED	CH	II	TLHO	EU	RP	Normal
	ED	**0.5707**	0.0471	0.0188			0.0707	0.2924
	CH		0.3592	0.0582	0.0388	0.0097		**0.5339**
	II	0.0471	0.0660	**0.4245**	0.0754			0.3867
	TLHO		0.0392	0.0539	**0.4362**	0.0441		0.4264
	EU		0.0189	0.0189		**0.6839**	0.0141	0.2641
	RP	0.0144			0.0192		**0.7019**	0.2788
	Normal	0.0013	0.0010	0.0016	0.0023		0.0003	**0.9932**

Table 5 The parameters of genetic algorithm

Population size (p_{size})	Population number (p_{num})	Mutation rate (p_m)	Crossover rate (p_c)	Maximum iterations (n_{max})
100	15	0.1	0.8	200

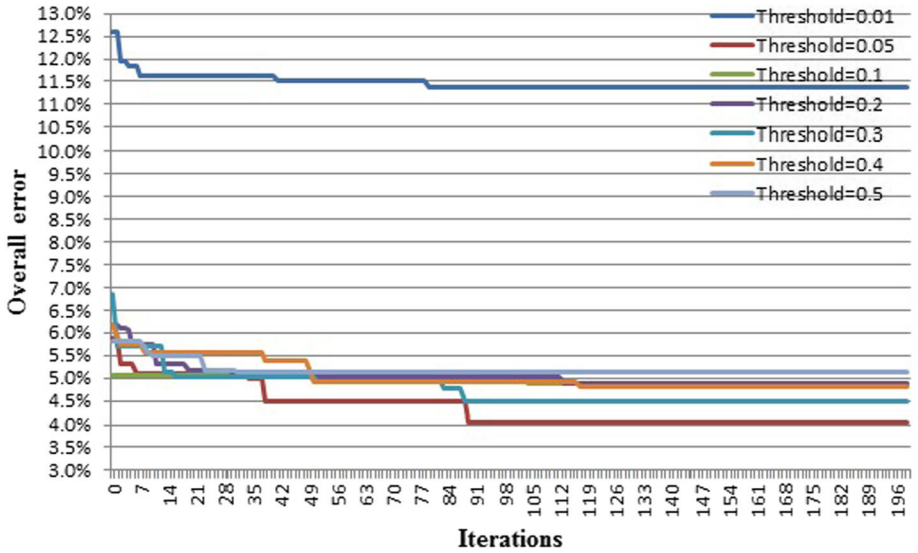

Fig. 8 The overall error variation of the GA-AdaboostLSVM classifier at different training error thresholds

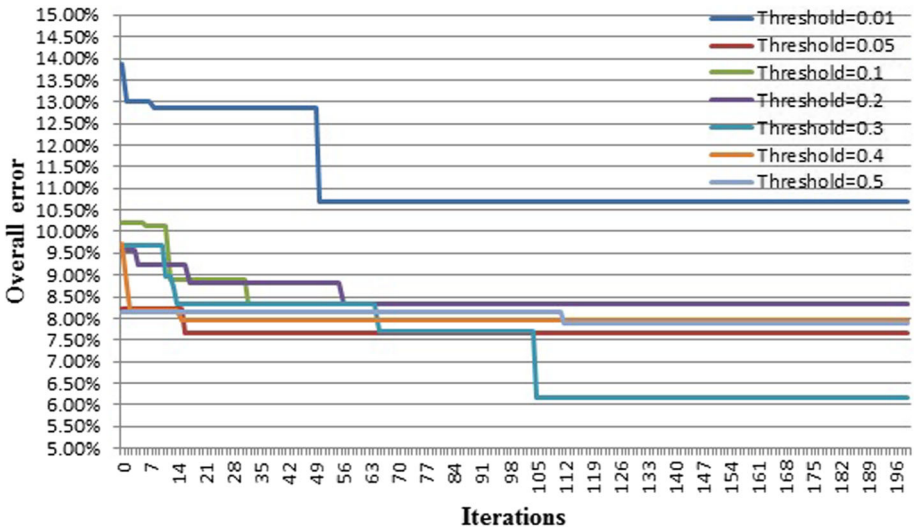

Fig. 9 The overall error variation of the GA-AdaboostRBFSVM classifier at different training error thresholds

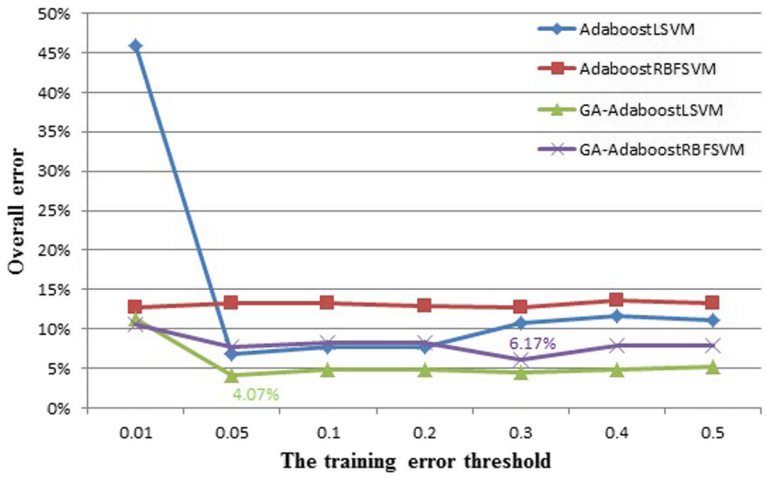

Fig. 10 The minimum overall error rate of different classifiers at different training error thresholds

Table 6 Normalized confusion matrix of GA-AdaBoostLSVM method with optimal parameter values

Predicted cause								
Real cause		ED	CH	II	TLHO	EU	RP	Normal
	ED	**0.9245**					0.0047	0.0707
	CH		**0.8155**	0.0291		0.0097		0.1456
	II		0.0283	**0.8301**	0.0188			0.1226
	TLHO	0.0098		0.0098	**0.8774**	0.0049		0.0980
	EU			0.0047		**0.9481**		0.0471
	RP	0.0144					**0.9326**	0.0528
	Normal	0.0010	0.0026	0.0033	0.0053	0.0006	0.0006	**0.9861**

Table 7 Normalized confusion matrix of GA-AdaBoostRBFSVM method with optimal parameter values

Predicted cause								
Real cause		ED	CH	II	TLHO	EU	RP	Normal
	ED	**0.7311**	0.0801	0.0094	0.0141		0.0660	0.0991
	CH		**0.5922**	0.0776	0.1456		0.0097	0.1747
	II	0.0283	0.0566	**0.6415**	0.0754			0.1981
	TLHO		0.0686	0.0294	**0.6715**	0.0833		0.1471
	EU		0.0377	0.0283	0.0330	**0.7924**		0.1084
	RP	0.0144	0.0192		0.0288	0.0144	**0.8076**	0.1153
	Normal	0.0027	0.0003	0.0030	0.0020		0.0010	**0.9908**

6 Appendix

Table 8 Parameter values

	C	σ	ε_{th}
LSVM	(0.001,0.01,0.05,0.1,0.5,1,5,10,20,50,100,200,500,1000)	–	–
RBFSVM	1	(0.01,0.05,0.1,0.3,0.5,0.7,1,3,5,7, 10,30,50,70, 100)	–
AdaBoostLSVM	(0.01,0.05,0.1,0.3,0.5,0.7,1,3,5,7,10,30,50,70, 100)	–	(0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5)
AdaBoostRBFSVM	1	(0.01,0.05,0.1,0.3,0.5,0.7,1,3,5,7, 10,30,50,70, 100)	(0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5)
GA-AdaBoostLSVM	15 values were randomly extracted from (0.01, 100).	–	(0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5)
GA-AdaBoostRBFSVM	15 values were randomly extracted from (0.01, 100).	15 values were randomly extracted from (0.01, 100).	(0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5)

Abbreviations

3GPP: Third Generation Partnership Project; AdaBoost: Adaptive Boosting; AdaBoostLSVM: Adaptive Boosting based on linear kernel; AdaBoostRBFSVM: Adaptive Boosting based on RBFSVM; AdaBoostSVM: Adaptive Boosting based on SVM; CAPEX: Capital Expenditures; CH: Coverage hole; DAG: Directed acyclic graph; DER: Diagnosis error rate; ED: Excessive downtilt; EU: Excessive uptilt; FPR: False positive rate; GA: Genetic algorithm; GA-AdaboostLSVM: AdaboostLSVM based on genetic algorithm; GA-AdaboostRBFSVM: AdaboostRBFSVM based on genetic algorithm; GA-AdaboostSVM: AdaboostSVM based on genetic algorithm; HetNet: Heterogeneous network; HOSR: Handover Success Rate; II: Intersystem interference; KPIs: Key performance indicators; LSVM: SVM with the linear kernel; LTE: Long Term Evolution; LTE-A: LTE-Advanced; ML: Machine Learning; OAA: One Against All; OAO: One Against one; OE: Overall error; OPEX: Operational Expenditures; PFP: Complementary of the Positive Predictive Value; RBF: Radial basis function; RBFSVM: SVM with the RBF kernel; RP: Reduction in cell power; RSRP: Reference Signal Received Power; RSRQ: Reference Signal Received Quality; SINR: Signal to Interference Noise Ratio; SMOTE: Synthetic minority oversampling technique; SONs: Self-organizing networks; SVM: Support Vector Machine; UDR: Undetected rate

Acknowledgements

This work was funded by the National Science and Technology Major Project: No. 2018ZX03001029-004.

Availability of data and materials

The datasets supporting the conclusions of this article were collected from reference [11].

Authors' contributions

WDG conceived and designed the study. XWL performed the simulation experiments. KSZ wrote the paper. GC reviewed and edited the manuscript. All authors read and approved the final manuscript.

Authors' information

Xuewen Liu is currently working toward a Ph.D. degree with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 18 January 2018 Accepted: 11 March 2018
Published online: 06 April 2018

References

1. 3GPP. (2012). Telecommunication Management; Self-Organizing Networks (SON); Concepts and Requirements. Next Generation Mobile Networks (NGMN) Alliance, ts 32.500 edn
2. Self-Organizing Networks (SON), *Concepts and Requirements Version 12.1.0, 3GPP TS*, vol 32 (2014), p. 500
3. EJ Khatib, R Barco, P Muñoz, et al, Knowledge Acquisition for Fault Management in LTE Networks[J]. Wirel. Pers. Commun. **95**, 1–20 (2017)
4. A Gómez-Andrades, P Muñoz, I Serrano, et al., Automatic root cause analysis for LTE networks based on unsupervised techniques[J]. IEEE Trans. Veh. Technol. **65**(4), 2369–2386 (2016)
5. EJ Khatib, R Barco, A Gómez-Andrades, et al., Diagnosis based on genetic fuzzy algorithms for LTE self-healing[J]. IEEE Trans. Veh. Technol. **65**(3), 1639–1651 (2016)
6. A Gómez-Andrades, P Muñoz, EJ Khatib, et al., Methodology for the design and evaluation of self-healing LTE networks[J]. IEEE Trans. Veh. Technol. **65**(8), 6468–6486 (2016)
7. Rezaei S, Radmanesh H, Alavizadeh P, et al. Automatic fault detection and diagnosis in cellular networks using operations support systems data[C]// NOMS 2016–2016 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2016:468–473.
8. Univerself. (2012). Univerself project. http://www.univerself-project.eu/.
9. COMMUNE. (2012). Commune (Cognitive Network Management Under Uncertainty).
10. SELFNET. (2015). Selfnet Project. https://selfnet-5g.eu/.
11. EJ Khatib, R Barco, A Gómez-Andrades, et al., Data mining for fuzzy diagnosis systems in LTE networks[J]. Expert Syst. Appl. **42**(21), 7549–7559 (2015)
12. Iacoboaiea O, Sayrac B, Jemaa S B, et al. SON conflict diagnosis in heterogeneous networks[C]//Personal, Indoor, and Mobile Radio Communications (PIMRC), 2015 IEEE 26th Annual International Symposium on. IEEE, 2015: 1459–1463.
13. R Barco, V Wille, L Díez, M Toril, Learning of model parameters for fault diagnosis in wireless networks. Wireless Netw. **16**(1), 255–271 (2010)
14. J Moysen, L Giupponi, *A Reinforcement Learning Based Solution for Self-Healing in LTE Networks[C]//Vehicular Technology Conference*. IEEE, 2014:1–6
15. L Flores-Martos, A Gomez-Andrades, R Barco, et al, *Unsupervised System for Diagnosis in LTE Networks Using Bayesian Networks[C]//Vehicular Technology Conference*. IEEE, 2015:1–5
16. P Casas, A D'Alconzo, P Fiadino, et al, *Detecting and Diagnosing Anomalies in Cellular Networks Using Random Neural Networks[C]//Wireless Communications and Mobile Computing Conference*. IEEE, 2016:351–356
17. X Li, L Wang, E Sung, AdaBoost with SVM-based component classifiers[J]. Eng. Appl. Artif. Intell. **21**(5), 785–795 (2008)
18. H He, EA Garcia, Learning from imbalanced data[J]. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009)
19. Platt J. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines[J]. 1998.

20. Xuewen Liu, Gang Chuai, Weidong Gao, Yifang Ren and Kaisa Zhang. Diagnosis Based on Machine Learning for LTE Self-Healing[M]// The Proceedings of the Sixth International Conference on Communications, Signal Processing, and Systems. Springer International Publishing (Accepted).
21. RE Schapire, Y Singer, Improved boosting algorithms using confidence-rated predictions. Mach. Learn. **37**(3), 297–336 (1999)
22. TG Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization[J]. Mach. Learn. **40**(2), 139–157 (2000)
23. H Schwenk, Y Bengio, Boosting neural networks[J]. Neural Comput. **12**(8), 1869–1887 (2000)
24. D Yang, Z Liu, T Shu, et al., An improved genetic algorithm for multiobjective optimization of helical coil electromagnetic launchers[J]. IEEE Transactions on Plasma Science **PP**(99), 1–7 (2017)
25. CZ Cooley, MW Haskell, SF Cauley, et al., Design of sparse Halbach magnet arrays for portable MRI using a genetic algorithm[J]. IEEE Trans. Magn. **PP**(99), 1–12 (2017)
26. G Valentini, TG Dietterich, Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods[J]. J. Mach. Learn. Res. **5**(Jul), 725–775 (2004)
27. C.-Wei Hsu, C.-Jen Lin, A comparison of methods for multiclass support vector machines Neural Networks, IEEE Trans. on, vol. 13, no. 2, pp. 415-425, 2002.
28. G Madzarov, D Gjorgjevikj, Multi-class classification using support vector machines in decision tree architecture. EUROCON **2009**, 288–295 (2009)
29. HJ Rong, GB Huang, YS Ong, *Extreme learning machine for multi-categories classification applications[C]// IEEE International Joint Conference on Neural Networks*. IEEE, 2016:1709–1713