P2P Traffic Identification Using Support Vector Machine and Cuckoo Search Algorithm Combined with Particle Swarm Optimization Algorithm

Zhiwei Ye(✉), Mingwei Wang, Chunzhi Wang, and Hui Xu

School of Computer Science,
Hubei University of Technology, Wuhan, China
weizhiye121@163.com

Abstract. As peer-to-peer (P2P) technology booms lots of problems arise such as rampant piracy, congestion, low quality etc. Thus, accurate identification of P2P traffic makes great sense for efficient network management. As one of the optimal classifiers, support vector machine (SVM) has been successfully used in P2P traffic identification. However, the performance of SVM is largely dependent on its parameters and the traditional tuning methods are inefficient. In the paper, a novel hybrid method to optimize parameters of SVM based on cuckoo search algorithm combined with particle swarm optimization algorithm is proposed. The first stage of the proposed approach is to tune the best parameters for SVM with training data. Subsequently, the SVM configured with the best parameters is employed to identify P2P traffic. In the end, we demonstrate the effectiveness of our approach on-campus traffic traces. Experimental results indicate that the proposed method outperforms SVM based on genetic algorithm, particle swarm optimization algorithm and cuckoo search algorithm.

Keywords: P2P traffic identification · Support vector machine · Cuckoo search algorithm · Particle swarm optimization algorithm

1 Introduction

A peer-to-peer (P2P) system is a self-organizing system of equal, autonomous entities designed for the shared usage of distributed resources in a networked environment without central services [1, 2]. It is popular around the world as a new mode of Internet application, which has been widely used in download, instant message, PPTV, distributed computation and so on. More and more network bandwidth has been occupied by P2P applications, it is estimated that more than 70 % of the whole traffic is P2P in Internet [1]. As P2P technology booms a lot of problems arise, for instance, decentralized networks introduce new security issues as they are designed so that each user is responsible for controlling their data and resources. Moreover, Peer-to-peer networks, along with almost all network systems, are vulnerable to unsecured and unsigned codes that may allow remote access to files on a victim's computer or even compromise the entire network. A user may encounter harmful data by downloading a file that was originally uploaded as a virus disguised in an exe, mp3, or any other file type due to the lack of an administrator maintaining the list of files being distributed. Network security problem has currently become more critical. Trojan horses, worms, DDoS attacks and network abuse can damage network resources and bring inconvenience to the society and people's lives. As a result, it is very important to identify the P2P traffic for the QoS guarantee in the plan and design of network, much effort has been made on this topic, in general, the existing approaches mainly could be divided into following categories [3–15].

(1) Port number based approaches. They are based on TCP/UDP port number. Destination port number of datagram header source is used to identify common traffic. However, with the development of the P2P technology, the accuracy of the technology is gradually decreased, mainly due to the following reasons: (a) camouflage port technology, some P2P applications use the well-known port number of the application (for example, port 80) to camouflage the function port; (b) user-defined port technology, some P2P applications allow users to manually configure the port number, which makes port number no longer fixed; (c) random port technology, some P2P applications using random port allocation technology, making the port number of P2P applications become unpredictable [3, 4].

(2) Signature based approaches. In order to overcome the disadvantages of the above approach, a more trusty technique is to inspect the packet payload. This approach improves the accuracy of traffic identification greatly. Bleul et al. [5] design a simple, effective and flexible P2P flow identification system based on signature-matching in [6]. Young J. Won et al. [7] improve the approach of signature-matching, and prove that signature matches the five packets to the network stream can identify the network traffic. However, some limitations still remain, firstly, most P2P protocols are proprietary and reverse engineering is needed. Secondly, they cannot deal with brand-new applications that use unknown P2P protocols. Thirdly, they are unable to detect P2P traffic encrypted, even when only protocol headers are encrypted. Thus, signature based approaches are no longer suitable for traffic identification.

(3) Traffic behavior based approaches. It uses the information obtained from IP headers. Hence, traffic behavior based approaches could overcome the disadvantages of signature based techniques and port number based techniques. It will not be affected by the payload encryption and can scale high to speed links. But the approach cannot do an accurate classification for P2P traffic due to the approach is only applied to analysis of traffic records, so it is not real time and efficient [8–10].

(4) Machine learning based approaches. In essence, P2P traffic identification is a problem belongs to pattern recognition, which is to classify traffic into P2P and non-P2P. Naturally various classification approaches based on statistical characteristics have been applied to P2P traffic identification [11–15]. For example, Bayesian methods are conducted to identify the P2P traffic in [11, 12], Refs. [12, 13] employed Neural Network to identify P2P traffic. But these methods have some disadvantages, firstly, due to the learning rate is fixed, Neural Network requires a longer training time and the learning and memory ability of neural network are unstable. The prior probability of

Bayesian method is difficult to estimate and the assumption of independence between features is difficult to exist. Support vector machine (SVM) is a powerful machine learning approach for classification and regression problems of small samples and high dimensions, which was initially presented by Vapnik in the last decade of the 20th century based on statistical learning theory and structural risk minimization principle [16]. It has been widely used for security identity and image processing.

However, performance of support vector machine (SVM) is largely dependent on its parameters. In the procedure of classification by SVM, penalty factor C and kernel parameter σ have great effect on the performance of classification. How to get the best parameters is a hot topic in SVM. The common used approaches for tuning the best parameters for SVM are grid search approach, genetic algorithm (GA), particle swarm optimization algorithm (PSO) and so on. However, there are many shortcomings in these approaches. For instance, for grid search approach has to bear a heavy computational burden. GA has poor local search ability and is prone to premature convergence; moreover, it is relatively time-consuming and low efficiency search in the later stage of evolution [17]. PSO has advantages of easy implementation, robustness to control parameters and computation efficiency, and has been successfully applied to optimization problem [18–21]. But it is easy to fall into local optimization. Cuckoo search (CS) algorithm is a newly proposed population based stochastic global search algorithm by Yang [22]. CS can better converge to the optimal solution, but its convergence rate is not well [23]. By using the advantages of CS and PSO, a hybrid optimization algorithm of PSO and CS (shorten to CS_PSO) was proposed by Fan Wang [24]. The algorithm can improve the performance of both Cuckoo search and particle swarm optimization. Hence, in the paper, CS_PSO is proposed to handle with the parameters of SVM and used in P2P traffic identification.

The rest of the paper is organized as follows. Section 2 describes basic principle and relative work of SVM parameter optimization. Section 3 gives a brief description about hybrid Cuckoo Search algorithm and Particle Swarm Optimization algorithm. The proposed CS_PSO based SVM model is detailed in Sect. 4. Section 5 presents the implementation of the proposed model and simulation result. Finally the conclusion is drawn in Sect. 6.

2 The Basic Principle of Support Vector Machines

Support Vector Machines (SVM) is a machine learning approach based on the statistical theory, which could find the optimal solution of the classification results with limited information about a small sample dataset. SVM is able to avoid the shortcomings of many machine learning approaches, such as large sample data sets required, appropriate model established for specific problems [25]. SVM use the idea of kernel function to transform nonlinear problem into linear problem, and reduce the complexity of the algorithm [26]. By kernel transformation and optimization, the optimal problem could be converted into an extremism problem of quadratic convex function. In theory, the approach of SVM is bound to get the global optimal solution.

2.1 The Basic Theory of Support Vector Machines

SVM is development of the optimal surface in condition of linearly separable. The idea can be described by 2 dimensional cases. We can see the optimal hyper-plane as Fig. 1.

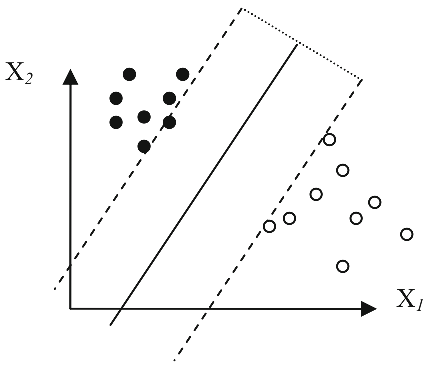

Fig. 1. Representation of hyper planes

In Fig. 1, maximum-margin hyper-plane and margins for an SVM trained with samples from 2 classes. Samples on the margin are called as support vectors [25].

Suppose given some training data D, a set of n points of the form:

$(x_i, y_i), i = 1, 2, \ldots, n, x \in R^n, y \in \{1, -1\}$, where the y_i is either 1 or -1, indicating the class to which the point x_i belongs. Each x_i is a p-dimensional real vector. Hyper-plane could be written as (1) [25]:

$$\omega \cdot x - b = 0. \tag{1}$$

If the training data are linearly separable, the general form of the discriminant function can be stated as (2) [25]:

$$f(x) = \omega \cdot x + b \tag{2}$$

Subject to constraints:

$$y_i[(\omega \cdot x_i) + b] - 1 \geqslant 0, i = 1, \ldots, n \tag{3}$$

b is the classification threshold and ω is one-dimensional coefficient vector of the separating hyper-plane in the high-dimensional feature space. Its classification principle can be stated as below [25].

$$\min = \frac{1}{2} \|\omega\|^2 \tag{4}$$

The optimal hyper-plane problem may be converted into convex quadratic programming optimization dual problem. The dual problem may be expressed as (5) [25]:

$$\max \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j (x_i \cdot x_j) \tag{5}$$

Subject to constraints:

$$a_i \geqslant 0, i = 1, \ldots, n \tag{6}$$

$$\sum_{i=1}^{n} a_i y_i = 0 \tag{7}$$

a_i is Lagrangian multiplier, because the problem is about quadratic function optimization, we can get a unique solution. If a_i^* is the optimal solution, so

$$\omega^* = \sum_{i=1}^{n} a_i^* y_i x_i \tag{8}$$

a_i^* is support vector.

If the training set is not linearly separable, the optimal hyper-plane cannot separate 2 types completely, in order to seek a certain balance between the empirical risk and the promotion of performance, the relaxation factor (ε) is introduced that allows the presence of misclassified samples. By adding the penalty factor (c) in minimizing objective, the objective function can be expressed as (9) [25]:

$$\Phi(\omega, \varepsilon) = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n} \varepsilon_i \tag{9}$$

Subject to constraints:

$$y_i[(\omega \cdot x_i) + b] \geqslant 1 - \varepsilon_i, i = 1, \ldots, n \tag{10}$$

With $0 < \varepsilon_i < 1$, if the sample points x_i are classify correct; else $\varepsilon_i \geqslant 1$

The optimal hyper-plane problem can be transformed into convex quadratic programming optimization dual problem. The dual problem could be expressed as follows:

$$\max \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j (x_i \cdot x_j) \tag{11}$$

$$s.t.\, 0 \leqslant x_i \leqslant C, i = 1, \ldots, n \tag{12}$$

$$\sum_{i=1}^{n} a_i y_i = 0 \tag{13}$$

2.2 SVM Parameters Optimization

As there are many non-linear problems in practice SVM utilizes kernel function transform to solve the problem. In 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create nonlinear classifiers through applying the kernel trick to maximum-margin hyper planes [26]. This permits the algorithm to fit the maximum-margin hyper plane in a transformed feature space. The transformation may be nonlinear and the transformed space may be high dimensional; thus though the classifier is a hyper plane in the high-dimensional feature space, it may be nonlinear in the original input space. Hence, the optimization problem could be represented as below.

$$\max Q(a) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j K(x_i, x_j) \tag{14}$$

The most used Common Kernel functions is RBF kernel function as shown below.

$$K(x_i, x_j) = \exp(\frac{-||x_i - x_j||^2}{\sigma^2}) \tag{15}$$

The parameters σ in kernel function reflect the characteristics of training data and greatly affect the generalization ability of the SVM. Penalty actor c determines the trade-off cost between minimizing the training error and minimizing the model's complexity, whether the value is too big or small will reduce the generalization of SVM. In the paper, the proposed approach aims to optimize the kernel parameters and reduce the number of support vectors efficiently.

3 Hybrid Cuckoo Search and Particle Swarm Optimization

3.1 Overview of Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) is a population-based stochastic search algorithm first proposed by Kennedy and Eberhart [18] in 1995. PSO has two notable features different from other evolutionary algorithms. Firstly, PSO is a stochastic evolutionary algorithm that does not incorporate survival of the fittest, and all individuals are retained as members of the population through the course of the run. Secondly, there are no conventional evolutionary operators such as crossover and mutation in PSO. It is easy to implement with quick search speed and widely applied to optimization problems that are partially irregular, noisy, change over time, etc. [18–21]. PSO is based on iteration model. In the procedure of optimization, every particle's fitness value is determined by the value of corresponding optimization function. Each particle has two types of information: the particle own flying experience denoted with historical best position of individual particle (p_{id}) and flying experience of particle swarm denoted with historical best position of particle swarm (p_{gd}). Particle and groups cooperate with each other, the relationship between particle movement and group activities are coordinated. Given the population of particles is n, every particle's location is represented as $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, particle's speed is $V_i = (v_{i1}, v_{i2}, \ldots, v_{id})$. Particle's velocity and location is adjusted as (16–17).

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1 [p_{id} - x_{id}(t)] + c_2 r_2 [p_{gd} - x_{id}(t)] \tag{16}$$

$$levy(\lambda) \sim u = t^{-\lambda}, 1 < \lambda < 3 \tag{17}$$

In the above expressions, $p_g d$ is historical best position in the whole group, p_{id} is particle's current historical best position, c_1 and c_2 are acceleration constants, commonly set as $c_1 = c_2 = 2$, r_1 and r_2 are random real number drawn from $U(0,1)$, ω is inertial weight.

3.2 Cuckoo Search Algorithm

Cuckoo search (CS) is an optimization algorithm developed by Xin-she Yang and Suash Deb [22]. It was inspired by the obligate brood parasitism of some cuckoo species which lay their eggs in the nests of other host birds [22]. Some host birds may find these extraneous eggs, these eggs will be discarded or the nest will be abandoned and a new nest will be built elsewhere. CS idealized such breeding behavior and has been applied for various optimization problems. CS is based on three idealized rules: (1) Each cuckoo lays a egg at a time, and dumps it in a randomly selected nest; (2) The optimal nests with high quality of eggs (solutions) will be kept up to the next generation; (3) The number of available host nests is a constant number, and a host can discover an alien egg with a probability $p_a \in [0,1]$. In this case, the host bird can either throw the egg away or give up the nest and build a completely new nest in a new location [22].

In CS algorithm, each egg in nest represents a solution, and a cuckoo egg denotes a new solution. The goal is to use the new and potentially better solutions to substitute a not so good solution in the nests. Suppose in the D-dimensional space, the nest's position is $X_i = (x_{i1}, x_{i2}, \ldots, x_{id}), i = 1, 2, \ldots, n$, local best solution is $p_t = (p_{i1}, p_{i2}, \ldots p_{id})$ and global best solution is $p_g = (p_{g1}, p_{g2}, \ldots p_{gd})$. The nest's position is updated according to following formula.

$$x_i(t+1) = x_i(t) + a \oplus levy(\lambda) \tag{18}$$

$$levy(\lambda) \sim u = t^{-\lambda}, 1 < \lambda < 3 \tag{19}$$

Here a is step size related to the scales of the problem of interest and $a > 0$. In iteration, a new solution is generated from scratch with a certain probability p_a and this solution is inserted in place of the lowest fit solution. This handles the problem of convergence on local optimal solutions efficiently. The pseudo code of CS is as below.

```
Objective Function $f(x)$ ,  $X = (x_1,...,x_d)^T$
Initialize the population of n host nests $x_i$
while (t<Max number of iterations)
Get a cuckoo randomly/generate a solution by levy flights
and then evaluate its quality/fitness $F_i$
  Choose a nest among n(say, j) randomly
if ($F_i > F_j$)
Replace j by the new solution
end
A fraction ($p_a$) of worse nests are abandoned and new so-
lutions are built
Keep best solutions
Rank the solutions and find the current best
end while
Postprocess results and visualization
```

3.3 Hybird Cuckoo Search and Particle Swarm Optimization Algorithm

CS_PSO is a hybrid algorithm which combines PSO and CS. It persists in high searching efficiency of PSO and inherits the ability of obtaining high grade solution in the whole searching space. Thus the hybrid algorithm enhances the searching precision and optimizing ability of the primitive methods. The basic thought of CS-PSO is that at an iteration it first carries out PSO to get a group of optimal positions, then the acquired optimal positions are used as the initial positions of CS and the CS is run until meets the terminal condition. The pseudo code of CS-PSO is as below.

```
Objective Function $f(x)$ ,  $X = (x_1,...,x_d)^T$
Initialize the population of n particle swarm $x_i$
While (t<Max number of iterations)
  Evaluate every particle’s fitness.
Get particle’s local position and global position by PSO
algorithm.
Using particle’s local position and global position as
initial population and get the new local position and
global position by CS algorithm
end while
Keep the global best solution
```

4 The Proposed CS_PSO Based SVM Model

4.1 The Idea of SVM Parameters Optimization

The performance of SVM mainly referred to the generalization ability. The penalty factor c and kernel function parameters σ exert a considerable influence on the generalization ability of SVM. The paper proposes to adopt CS_PSO algorithm to optimize parameters automatically. The main idea of applying CS_PSO to tune the best parameters pair (C and σ) of SVM is as following.

Each position vector of the CS_PSO stands for a candidate parameters pair for SVM. The initial population is generated with N number of solutions and each solution is a D-dimension vector, here D is set as 2 that each solution represents 2 parameters. x_i is the i-th particle position in the population which denotes a candidate parameter pair and its fitness can be measured by predefined fitness function, with the defined movement rules, the algorithm will run and until it terminates and output the best position as the optimal parameters for SVM.

4.2 Objective Function for Parameter Optimization

The goal of optimizing parameters for SVM is to utilize optimized procedures that explore a finite subset of the possible values to find the parameters that minimize the generalization error or maximize the correct classification rate. Thus, in the paper, the objective function of parameter optimization for SVM is the correct classification rate on the training data set.

4.3 The Implementation of the Proposed Method

The processes of the proposed CS_SPO for SVM parameters method are as follow, which mainly could be divided into 2 parts [21].

1. Direct Use of SVM

RBF kernel function is taken as kernel function in the paper. After the relevant parameters have been selected, RBF kernel function could be applied to any distribution of samples. The generalization ability of SVM algorithm is largely dependent on a set of parameters. The parameters needs to optimized are: RBF kernel parameter and the estimated accuracy. The n-fold approach is used to estimate the generalization ability. The data set was randomly divided into a one-second of a set (training set) and one-second of a set (testing set). The basic step is as follows [21]:

Step1. Input the sample training set, and set a group of parameters $\{c, \sigma\}$.
Step2. Train SVM based on the parameters. Calculate the cross validation error and obtains its object.
Step3. Test the SVM using object obtained from step 2.
Step4. Repeat the above step 25 times and find the average testing accuracy.

2. The proposed CS_PSO based SVM model

The procedure for describing proposed CS-PSO_SVM is as follows:

Step1. Initialize population size, inertia weight of PSO and parameters of CS and PSO,
Step2. Train SVM on particles.
Step3. Evaluate each particle's fitness value. Take the cross validation error of the SVM training set as fitness value.
Step4. Compare the fitness values and calculates the local optimal solution and global optimal solution.
Step5. Update the velocity and position of the each particle by Eqs. (16–17)
Step6. Take the local optimal solution and global optimal solution as the initial nest in cuckoo search.
Step7. Train SVM on nest.
Step8. Evaluate each nest's fitness value. Take the cross validation error of the SVM training set as fitness value.
Step9. Compare the fitness values and calculates the local optimal solution and global optimal solution.
Step10. Update the position of nest by Eqs. (18–19)
Step11. Repeat the step 2–10 until fitness function converges or the maximal number of iteration is reached.
Step12. The global best solution is input into SVM for classification.

5 Experimental Evaluation and Discussion

To show effectiveness of the proposed method, some real campus P2P traffic are used to evaluate the performance of the proposed CS-PSO based SVM for traffic identification of P2P; moreover, the performance is compared with some well known algorithms, that is GA-SVM, PSO-SVM. Moreover, the proposed method is compared with CS_SVM too. The main parameters used for these approaches are: the initial population for four algorithms is the same, that is 20, and all these algorithms will terminate after being executed 50 times. Moreover, the crossover rate for GA is 0.4 and mutation rate for GA is 0.01. And for PSO, $C_1 = 1.5$ and $C_2 = 1.7$, the value of inertia weight is set as 1. For CS, Pa = 0.25.

5.1 Data for Experiment

The base truth datasets were established manually from campus network. It was obtained from several host running P2P applications and several host running non-P2P applications. This experiment uses nearly 1386 samples, half of the samples are training set and the rest is testing set. P2P sample is positive class while non-P2P sample is negative, there are 968 positive samples and 418 negative samples.

5.2 P2P Traffic Features and Preprocessing

8 features are extracted and used in the paper, which are listed out as following, the time stamp, the time interval, the instantaneous flow rate, the packet length, time period, source port, destination port, the average flow rate per second. As is known that normalization is particularly useful for classification algorithms based on distance measurements, therefore, after collecting network traffic, we have used Min-max normalization to perform a linear transformation on the original data. Assume that *minA* and *maxA* are the minimum and maximum values of an attribute A. Min-max normalization maps a value, v, of A to v' in the range $[new\_minA;\ new\_maxA]$ by computing (20).

$$v' = \frac{v - \min_A}{\max_A - \min_A}(new\_\max_A - new\_\min_A) + new\_\min_A \tag{20}$$

5.3 The Methods of Classification Evaluation

In order to test and evaluate the algorithms, the k-fold validation is adopted in the paper. That is, the data are divided into k subsets of approximately equal size. Each time, one of the k subsets is used as the test set and the other k-1 subsets form the training set. Performance statistics are calculated across all k trials. This provides a good indication of how well the classifier will perform on unseen data. In this experiment, k is set as 5. We have used the following quantity of result evaluation.

$$Identification\ Rate = \frac{T_P + T_N}{T_P + \mathrm{T}_N + \mathrm{F_P} + \mathrm{F_N}} \tag{21}$$

The meaning of parameters in (21) is as below. $\mathrm{T_P}$ (True Positive): In case of test sample is positive and it is identified as positive, it is considered as a true positive; $\mathrm{T_N}$ (True Negative): in case of test sample is negative and it is identified as negative, it is considered as a true negative. FP (False Positive), in case of test sample is negative and it is identified as positive, it is considered as a false positive. FN (False Negative) in case of test sample is positive and it is identified as negative, it is considered as a false negative. The aim of any classification technique is to maximize the number of correct classification given by True Positive samples (TP) and True Negative samples (TN), whereas minimizing the wrong classification given by False Positive (FP) and False Negative (FN).

The range of parameters C and σ is from 2^{-10} to 2^{10} on the dataset. For these methods are stochastic algorithms all these methods have been run 30 times in order to make the results more reliable and impartial. Accuracy results of identification of 30 times are summarized in Table 1. Furthermore, some typical convergence procedures of these methods are displayed in Figs. 2, 3, 4 and 5.

In Table 1, BestAcc stands for the highest accuracy in 30 times test, WorstAcc is the lowest accuracy in 30 times test, and the average accuracy of 30 times test is expressed with AverAcc. According to Table 1, it is observed the proposed CS-PSO

Table 1. Running results with four approaches by 30 time

Approach	BestAcc	WorstAcc	AverAcc
GA + SVM	84.4156	81.3531	83.05918
CS + SVM	84.8485	80.8081	82.80906
PSO + SVM	84.7042	82.1068	83.46803
Proposed	85.2814	82.5397	83.78549

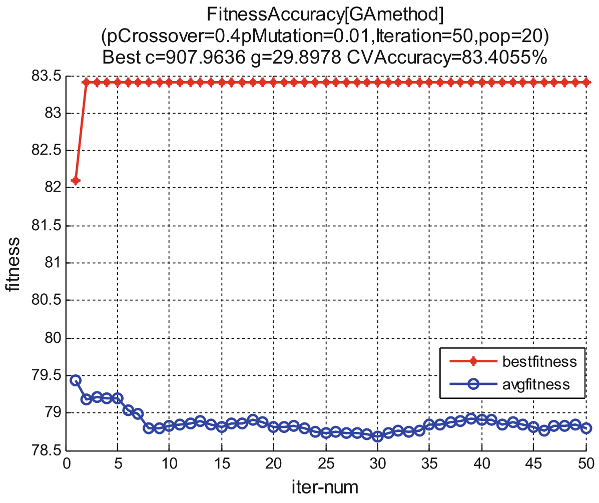

Fig. 2. Classified by GA based SVM

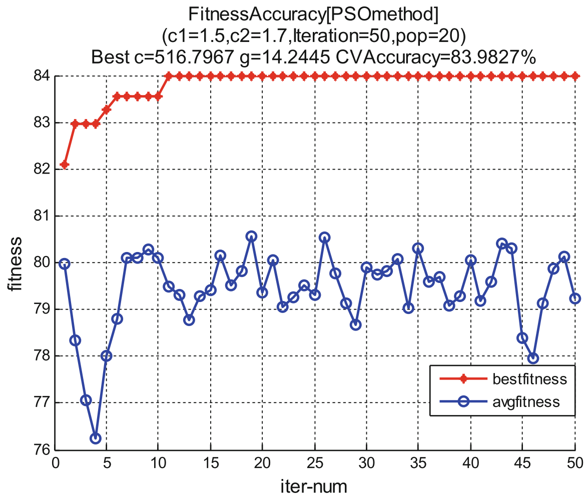

Fig. 3. Classified by PSO based SVM

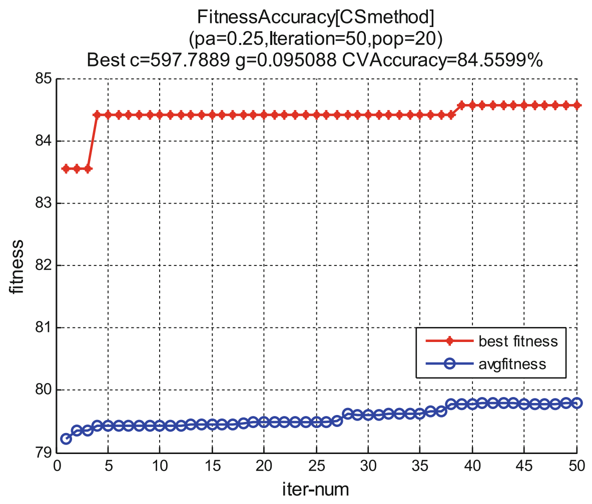

Fig. 4. Classified by CS based SVM

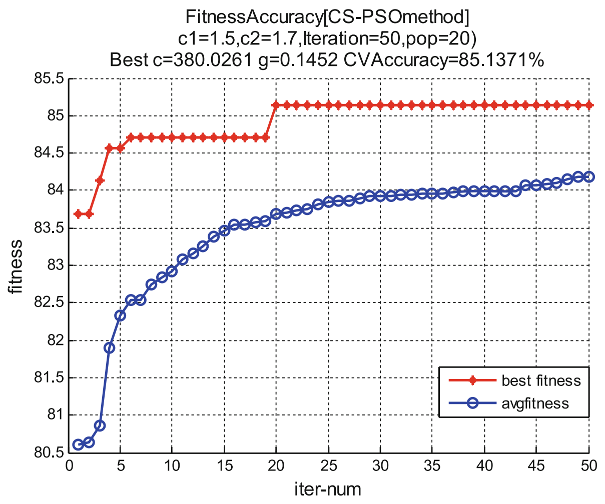

Fig. 5. Classified by CS-PSO based SVM

based SVM outperforms the other methods in all these data. It has the optimal BestAcc, WorstAcc and AverAcc. For example as for the highest accuracy, the CS-PSO based SVM is 85.2814 %, at the same time, the highest accuracy of the rest methods is less than 85 %; moreover, the CS-PSO based SVM has the highest average accuracy among these methods. It demonstrates that the performance of the proposed CS-PSO based SVM model is superior to GA-SVM, PSO-SVM and CS-SVM. Further, It is worth noting that as for the bestACC, the CS based SVM is better than PSO and GA based SVM but inferior to CS_PSO, because the terminal condition of the paper is max iteration and which is set only 50, as a result, though it could obtain fair good bestACC, its average correct rate is less than GA and PSO. Hence, by combining with PSO, the hybrid algorithm takes advantages of both PSO and CS, which could have a good balance of search efficiency and time consuming. Furthermore, Figs. 2, 3, 4 and 5 display that convergence rate of GA-SVM and PSO-SVM, which is very fast, CS is rather slow, in later of iteration, the best fitness of GA and PSO is almost no longer improved. Meanwhile, the best fitness of the proposed approach increases along with iterations, it illustrates that CS-PSO could avoid local optimal solution at the most extent. Lastly, the average fitness of CS-PSO is higher than other algorithm, it illustrates that performance of CS-PSO based SVM is more robust than GA based SVM, CS based SVM and PSO based SVM, which is more suitable for tuning parameters of SVM and P2P traffic identification.

6 Conclusion

In sum, a P2P traffic identification approach is developed based on SVM and CS_PSO in the paper. That is, for obtaining good tuning parameters of SVM, a CS_PSO method which combines Cuckoo Search algorithm and Particle Swarm Optimization algorithm is employed for parameter optimization of SVM. The proposed approach has been test on P2P dataset and compared with several existing approaches such as GA and PSO based SVM. The experimental results indicate that the proposed CS_PSO algorithm outperforms GA, PSO and CS based SVM and is feasible to optimize the parameters for SVM, which confirms that the new CS_PSO based SVM model can obtain ideal results. The schema is generic in identifying P2P applications. Our future work is to extend our approach to distinguish different P2P application flows by using ensemble learning SVM.

Acknowledgment. This work is supported by Natural Science Foundation of China (No. 41301371, 61170135 and 61202287), the Emergency Management Program for National Natural Science Foundation of China (No. 61440024), Doctor Fund of Hubei University of technology (BSQD13081 BSQD12032).

References

1. Steinmetz, R., Wehrle, K.: Peer-to-Peer Systems and Applications. Springer, Berlin (2005)
2. http://en.wikipedia.org/wiki/Peer-to-peer
3. Constantinou, F., Mavrommatis, P.: Identifying known and unkonwn peer-to-peer traffic. In: Fifth IEEE International Symposium on Network Computing and Application, pp. 93–102. IEEE Press, Cambridge, MA (2006)
4. Kim, M.-S., Kang, H.-J., Hong, J.W.: Towards Peer-to-Peer Traffic Analysis Using Flows. In: Brunner, Marcus, Keller, Alexander (eds.) DSOM 2003. LNCS, vol. 2867, pp. 55–67. Springer, Heidelberg (2003)
5. Bleul, H., Rathgeb, E.P.: A Simple, efficient, and flexible approach to measure multi-protocol peer-to-peer traffic. In: Lorenz, P., Dini, P. (eds.) Networking - ICN 2005. LNCS, vol. 3427. Springer, Heidelberg, pp. 606–616 (2005)

6. Sen, S., Spatscheck, O., Wang, D.: Accurate, scalable in-network identification of P2P traffic using application signatures. In: WWW2004, pp. 512–521. ACM Press, New York (2004)
7. Won, Y.J., Park, B.-C, Ju, H.-T.: A hybrid approach for accurate application traffic identification. In: 4th IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services, 2006, pp. 1–8. IEEE Press, Canada (2006)
8. Xu, K., Zhang, M., Ye, M., Chiu, D.M., Wu, J.: Identify P2P traffic by inspecting data transfer behavior. Comput. Commun. **33**, 1141–1150 (2010)
9. Keralapura, R., Nucci, A., Chuah, C.N.: A novel self-learning architecture for P2P traffic classification in high speed networks. Comput. Netw. **54**, 1055–1068 (2010)
10. Liu, B.: A semi-supervised clustering approach for P2P traffic classification. J. Netw. **6**(3), 424–431 (2011)
11. Moore, A.W., Zuev, D.: Internet traffic classification using Bayesian analysis techniques. In: Proceedings of ACM Sigmetrics, pp. 50–60 (2005)
12. Jin, F., Duan, Y.: A P2P flow identification model based on Bayesian network. In: 7th International Conference Wireless Communications, Networking and Mobile Computing (WiCOM), 2011, pp. 1–4. IEEE Press, Wuhan (2011)
13. Chen, H., Hu, Z., Ye, Z.:. Research of P2P traffic identification based on neural network. In: IEEE Conference on Computer Network and Multimedia Technology, 2009, pp. 18–20. IEEE Press, Wuhan (2009)
14. Chen, H., Zhou, X., You, F., Xu, H., Wang, C., et al.: A SVM approach for P2P traffic identification based on multiple traffic mode. J. Netw. **5**(11), 1381–1388 (2010)
15. Zheng, J., Xu, Y.: Identification of network traffic based on support vector machine. In: 2010 3rd International Conference on Advanced Computer Theory and Engineering, pp. 286–291. IEEE Press, Chengdu (2010)
16. VapNik, V.N.: The Nature of Statistical Learning Theory. Springer, New York (1995)
17. Eberhart, R.C., Shi, Y.: Comparison between genetic algorithms and particle swarm optimization. In: Proceedings of 1998 7th Annual Conference on Evoluationary Programming, vol. 1447, pp. 611–616. Springer, Berlin, Heidelberg (1998)
18. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942–1948. IEEE Press, Piscataway (1995)
19. Esmin, A., Torres, G., Zambroni, A.: A hybrid particle swarm optimization applied to loss power minimization. IEEE Trans. Power Syst. **20**(2), 859–866 (2005)
20. Ting, T.O.: A novel approach for unit commitment problem via an effective hybrid particle swarm optimization. IEEE Trans. Power Syst. **21**(1), 11–418 (2006)
21. Parimala, R.: Feature selection using a novel particle swarm optimization and It's variants. I. J. Inf. Technol. Comput. Sci. **5**, 16–24 (2012)
22. Yang, X., Deb, S.: Cuckoo search via levy fights. In: 2009 World Congress on Nature and Biologically Inspired Computing (NaBIC 2009), India, pp. 210–215. IEEE Press, Coimbatore (2009)
23. Civicioglu, P., Besdok, E.: A conceptual comparison of the cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. Artif. Intell. Rev. **39**, 315–346 (2013)
24. Wang, F., Luo, L., He, X., Wang, Y.: Hybird optimization algorithm of PSO and Cickoo search. In: IEEE 2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), pp. 1172–1175. IEEE Press, Deng Leng (2011)
25. http://en.wikipedia.org/wiki/Support_vector_machine
26. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: 5th Annual ACM Workshop on COLT, pp. 144–152. ACM Press, New York (1992)