UNIVERSITY OF MAIDUGURI
FACULTY OF ENGINEERING SEMINAR SERIES
VOLUME 8, MARCH 2017

A Hybridized SVM-kNN-*pdAPSO* Approach to Intrusion Detection System

E.G. Dada
Department of Computer Engineering, Faculty of Engineering, University of Maiduguri, Maiduguri, Borno State, Nigeria.
gbengadada@unimaid.edu.ng; +2349084222298

Abstract
Several cases of service attacks on major Internet sites have shown us, no open computer network is immune from intrusions. The wireless ad-hoc network is particularly vulnerable due to its features of open medium, dynamic changing topology, cooperative algorithms, lack of centralized monitoring and management point, and lack of a clear line of defense. The traditional way of protecting networks with firewalls and encryption software is no longer sufficient and effective. The field of machine learning has proved that the hybridization of classifiers usually has a better performance than individual ones. This paper proposes a new approach that hybridizes different classifiers for better accuracy in detection of intrusion attacks. The result of experiments conducted shows that the fusion of Support Vector Machine, k-Nearest Neighbor, and Primal-Dual Particle Swarm Optimization produced better classification accuracy than each of the singular classifiers on the KDD99 dataset.

Keywords: Intrusion Detection System, k-Nearest neighbor, Support Vector Machine, Primal Dual, Particle Swarm Optimization

1.0 Introduction

Intrusion detection is a very important topic of network security that has received much attention according to Patcha and Park (2007). The threats posed to computer users by computer related malicious code or intrusion attacks is on the increase every day. According to Lee, Stolfo, and Mok (1999), KDD data set can be used for evaluating a comprehensive set of pattern recognition and machine learning. Hybridized methods were proved to experimentally and theoretically possess better accuracy than using any one classifier algorithm alone. The success of the hybridized method for intrusion classification greatly depends on the diversity in the outputs of its different classifiers, as well as on the choice of method to combine these outputs into a single one (Chen and Wong, 2014).

Over the past twenty years, many research has been done much to advance the state-of-the-art in this increasingly important area. According to Curtis and Carver (2000), research prototypes and commercial IDS and IRS built during this period now number nearly 100. Jianxiao and Lijuan (2008) presented an Intrusion Detection technique that combines static agent and mobile agent, Host-based IDS and Network-based IDS and a distributed intrusion detection system model based on agents. A new pattern matching algorithm is proposed by Zhang (2009), first sequence letters in the pattern string from low appearance probability to high appearance probability in natural English, and then match one by one according to the algorithm thereby reducing the comparison times. Syarif *et al.* (2012) used naïve Bayes, J48 (decision tree), JRip (rule induction) and iBK (nearest neighbor) to solve the intrusion detection problem by improving the accuracy and reduce its false positive

rates. Bahri et al. (2011) proposed a hybrid technique called Greedy-Boost using the KDD 1999 data set. Bukhtoyarov et al. (2014) proposed a probabilistic method to design base neural network classifiers known as probability-based generator of neural networks structures (PGNS) to the network intrusion detection problem. Cordeiro and Pappa (2011) proposed the PSO-WV technique that used the PSO algorithm for weighting classifications coming from different data views.
In this work, Six k-nearest neighbor (k-NN) were trained and six support vector machine (SVM) on the same dataset. The output of the training is feed into the *pdAPSO* algorithm. The final decision is reached by combining the opinions of the different SVM-kNN-pdAPSO hybrid approach.

2.0 Methodology

2.1 System framework

In line with the objectives stated earlier, the SVM, k-NN, and *pdAPSO* were used in training and testing the system. Fig. 1 illustrates the entire process. For ease of comprehension, the system framework was divided into the following four phases: Kdd99 data pre-processing, data classification using SVM algorithm, data classification using k-NN algorithm, and data classification using pdAPSO algorithm.

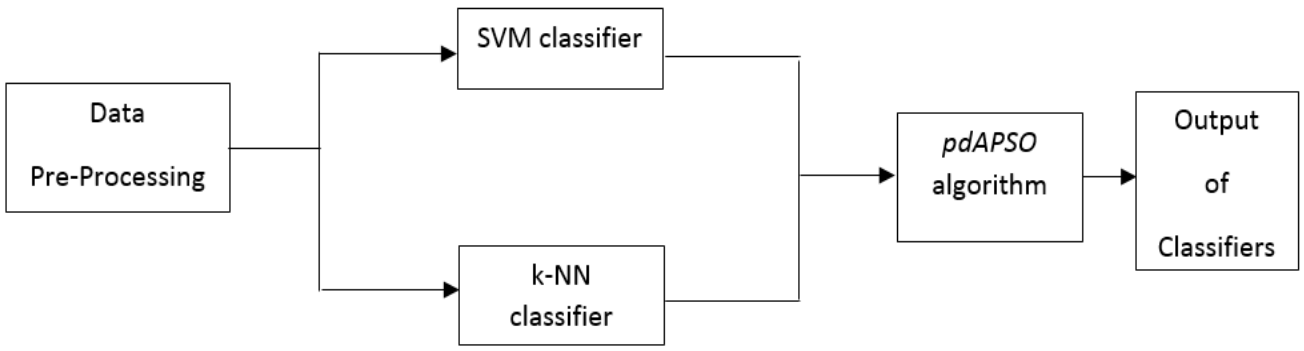

Fig. 1: System Framework

2.2. Dataset used for experiments

For the purpose of this research work, the knowledge discovery and data mining 1999 (KDD99) dataset was used for the experiments. The dataset contains hundreds of thousands of connection records. KDD dataset covers four major categories of attacks: Probing attacks (information gathering attacks), Denial-of-Service (DoS) attacks (deny legitimate requests to a system), user-to-root (U2R) attacks (unauthorized access to local super-user or root), and remote-to-local (R2L) attacks (unauthorized local access from a remote machine). Elkan (2000) described the attributes in the KDD datasets that they have all forms - continuous, discrete, and symbolic, with significantly varying resolution and ranges.

Experiments was conducted for this study using training, validation and testing samples were drawn from five datasets downloaded from the KDD 99 dataset. In order to compare efficiency of proposed algorithms, we adopt the accuracy of classification as the yardstick for measuring the performance of the system. Where accuracy of classification is the proportion of the test data correctly classified by an algorithm. A classifier's accuracy is the number of correctly classified observations C_i of class i divided by the total number of observations s. As depicted in Fig. 2, each of the algorithms consists of five binary classifiers. Each binary classifier B_i is used to classify instances of class i from the set of observations $O = O_1, O_2, \ldots, O_s$. Therefore, we determine accuracy A_i of each binary classifier according to Eq. (1).

$$A_i = \frac{C_i}{S} \tag{1}$$

All the data were gotten from (KDD99) dataset, and can be categorized into three sets: Training data, testing data, and validation data. The KDD99 dataset is divided into normal traffic and four classes of attacks (Araujo *et al*, 2010). Table 1 provides information about the number of observations in training and testing datasets for each class. This experiment used five data sets, taken from the training and testing KDD99. As shown in Table 2, all five samples are of the same size, but contain different observations selected at random from the training and testing datasets represented in Table 1.

Table 1: Number of observations in KDD99.

Connection type	Training dataset	Testing dataset
Normal	97,278	60,593
DoS	391,458	229,853
Probe	4107	4166
R2L	1126	6,347
U2R	52	70
Total	494,021	311,029

Table 2: Dataset size used in experiments.

	Normal	DoS	Probe	R2L	U2R	Total
Training	5000	4107	5000	52	1126	15,285
Testing	7500	3124	7500	52	3750	21,926
Validation	2500	1042	2500	18	1250	7310

2.3 Data Pre-processing

Data in the real world is in their raw state. This means that they are incomplete (lacking attribute values, lacking certain attributes of interest, or containing only aggregate data. KDD99 data includes three symbolic features that are incompatible with the proposed classification algorithms such as protocol type, service, and flag. Pre-processing operation was performed on the data in two steps: Data mapping and identification of State. This means that each record in the data belongs to one of five major classes: Normal, DoS, Probe, U2R, and R2L. The values for each state are mapped to a numeric value.

2.4 SVM classifier

The support vector machine (SVM) is a supervised learning method that generates input-output mapping functions from a set of labeled training data. The mapping function can be either a classification function, i.e., the category of the input data, or a regression function. For classification, nonlinear kernel functions are often used to transform input data to a high-dimensional feature space in which the input data become more separable compared to the original input space. Maximum-margin hyperplanes are then created. The model thus produced depends on only a subset of the training data near the class boundaries. SVM is originally an implementation of Vapnik's Structural Risk Minimization (SRM) principle (Kuang et al., 2014), which is known to have low generalization error or equivalently does not suffer much from overfitting to the training data set.

2.5 k-NN Classifier

The K-nearest neighbour algorithm is a (k-NN) is a simple and effective method for classifying objects based on closest training examples in the feature space. K-Nearest Neighbors (KNN) classification divides data into a test set and a training set. If there are ties for the Kth nearest vector, all candidates are included in the vote. Consider a set of observations and targets $(x_1, y_1), \ldots, (x_n, y_n)$, where observations $x_i \in \mathbf{R}^d$ and targets $y_i \in \{0, 1\}$; then for a given i, k-NN rates the neighbours of a test sequence among the training sample, and uses the class labels of the nearest neighbours to predict the test vector class. In k-NN, the Euclidean distance is often used as the distance metric to measure the similarity between two vectors (points):

$(x_i, y_i), \ldots, (x_n, y_n)$ where observations $x_i \in \mathbf{R}^d$ and targets $y_i \in \{0, 1\}$,

$$d^2(x_i, x_j) = \|x_i - x_j\|^2 = \sum_{k=1}^{d}(x_{ik} - x_{jk})^2 \qquad (2)$$

where $(x_i, y_j) \in \mathbf{R}^d$, $x_i - (x_{i1}, x_{i2}, \ldots, x_{id})$

2.6 Primal-Dual Asynchronous Particle Swarm Optimisation (pdAPSO) Algorithm

The primal dual asynchronous PSO (*pdAPSO*) works by first allowing the primal dual section of the algorithm to optimize the particles that have been randomly introduced into the search space. The asynchronous PSO then accept the result of the primal dual algorithm and optimize it to get the optimal solution. The asynchronous PSO computes the personal best ($pbest_{i,m}$) or the global best ($gbest_{i,m}$) of a particle. The velocity and position of the particles are updated immediately after computing their fitness. The Primal Dual provides a better balance between exploration and exploitation, preventing the particles from experiencing premature convergence and been trapped in local minima easily. For detail information on the Primal Dual APSO the reader is referred to the previous work Dada (2016).

3.0 Results and Discussion

Experiments were conducted, and the *pdAPSO* was used to optimize the selected dataset that was chosen for the purpose of our experiment. From the result of our experiment, the *pdAPSO* converged optimally for each of the dataset that represent the different attack categories that we are considering. Examples of datasets used include normal, DoS, probe, R2L, and U2R. There was no case of premature convergence or particles of the *pdAPSO* been trapped in the local optimal. Depicted in Fig. 3 is the graph of convergence of *pdAPSO* algorithm towards optimal solution, for each of our five categories of attacked considered in this study.

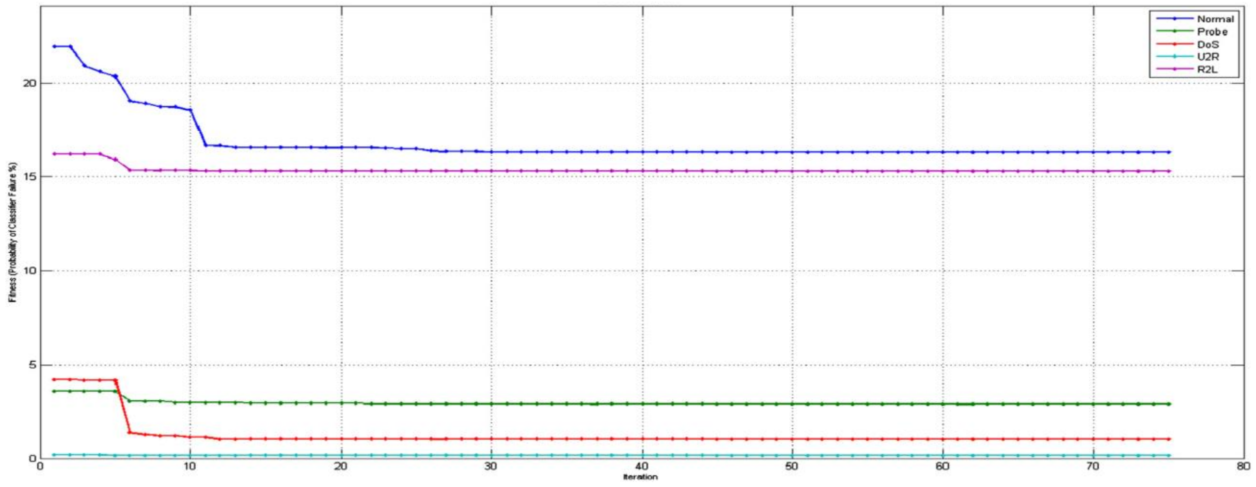

Fig. 3. Convergence of *pdAPSO*

Table 3: Mean result of 5 datasets

Classifier	Normal	DoS	Probe	R2L	U2R
SVM 1	76.212%	96.299%	93.9912%	98.5825%	84.5462%
SVM 2	76.518%	95.229%	96.9701%	99.4846%	83.8420%
SVM 3	79.731%	94.792%	98.0965%	99.6142%	83.7937%
SVM 4	76.972%	94.401%	97.9521%	99.6789%	83.4562%
SVM 5	73.359%	93.654%	95.2101%	99.7419%	83.4115%
SVM 6	70.401%	92.311%	94.0982%	99.7528%	83.1743%
k-NN 1	81.1667%	96.2729%	97.6329%	99.666%	83.2364%
k-NN 2	79.8540%	95.8880%	96.5283%	99.7118%	83.2674%
k-NN 3	78.5606%	95.7776%	95.3316%	99.7710%	83.2099%
k-NN 4	76.6487%	95.8407%	93.3157%	99.7920%	82.8222%
k-NN 5	76.6734%	95.9591%	92.7429%	99.7948%	82.8158%
k-NN 6	76.9698%	96.1087%	92.7091%	99.7948%	82.8003%
pdAPSO	84.1814%	96.7170%	98.8567%	99.8023%	84.8629%

The Figs. 4 – 6 are the graphs depicting the percentage of accuracies of SVM, k-NN and the *pdAPSO* algorithm for dataset 1 respectively.

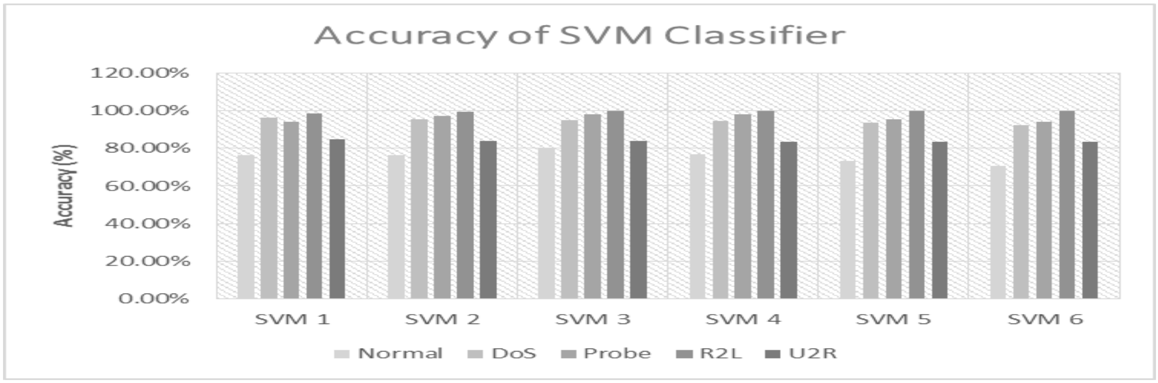

Fig. 4. Accuracy of SVM

The result depicted in fig. 4 above shows that SVM have highest accuracy of about 97% for R2L attacks for each of the datasets considered. The lowest accuracy was recorded for Normal dataset.

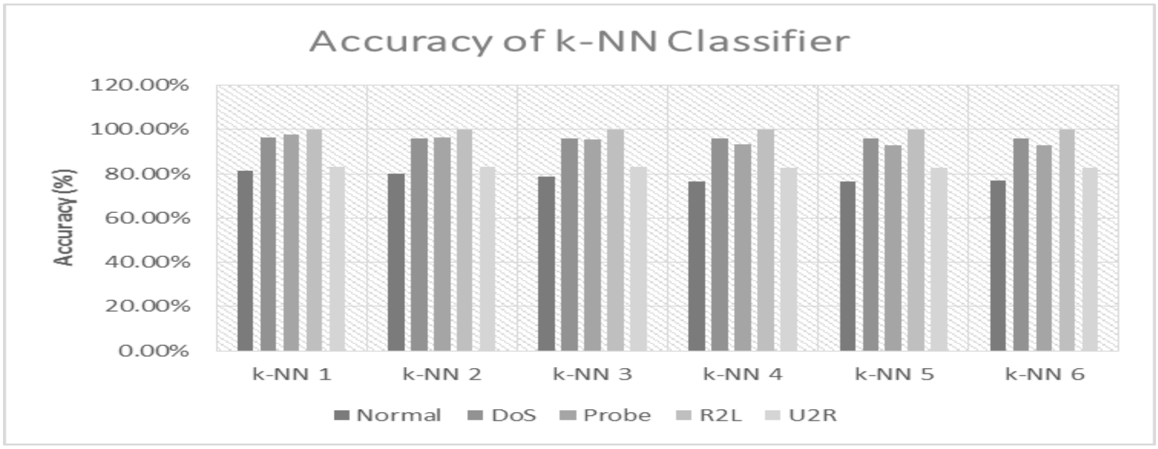

Fig. 5: Accuracy of K-NN classifier

The result for k-NN classifier is in figure 5 above. The k-NN have highest accuracy of about 98% for R2L attacks for each of the datasets considered. The lowest accuracy was recorded for Normal and U2L datasets.

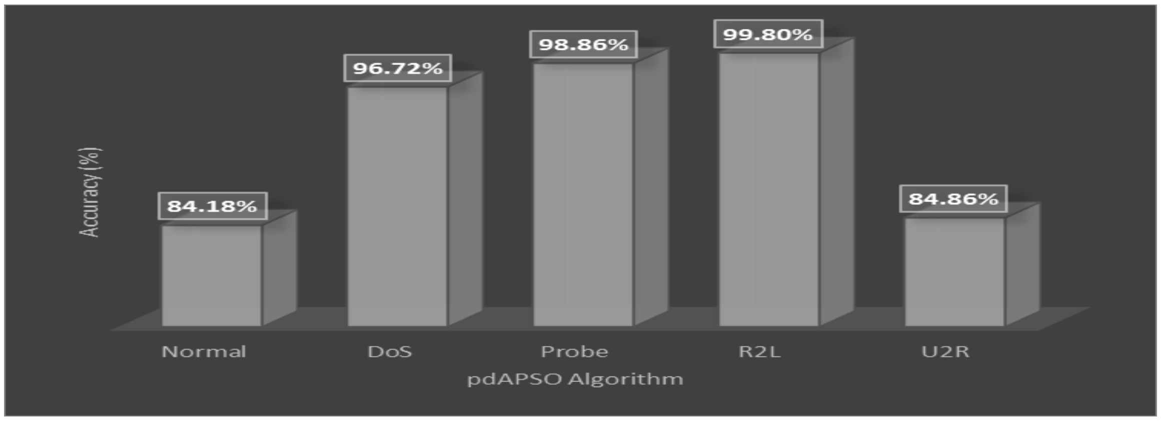

Fig. 6: Accuracy of *pdAPSO* algorithm

The result depicted in fig. 6 above shows that pdAPSO has highest accuracy of about 98% for R2L attacks for each of the datasets considered, and closely followed by prober attack.

4.0 Conclusion

This study explores the use of hybrid SVM, k-NN, and *pdAPSO* algorithms to detect intrusion. It has been established from the experimental results that classification accuracy can be enhanced by hybridizing different classifiers. The paper compared the performance of the three classifiers using the statistical results obtained from the KDD99 datasets. The SVM, k-NN and *pdAPSO* hybrid approach outperform the other algorithms compared by generating classification accuracy of 98.55%.

References

A. Patcha and J. M. Park (2007). An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51, 2007.

Aziz N. A. A., and Ibrahim Z (2012). Asynchronous Particle Swarm Optimization for Swarm Robotics, International Symposium on Robotics and Intelligent Sensors (IRIS 2012). *Procedia Engineering*, Vol. 41 pp. 951 – 957.

C. Elkan (2000), "Results of the KDD'99 Classifier Learning", SIGKDD Explorations, ACM SIGKDD.

Curtis A. and Carver Jr (2000). Intrusion Detection System: A Survey. Department of Computer Science, Texas A & M University, College Station, TX 77843-3112, USA

Dada Emmanuel Gbenga (2016). Primal-Dual Asynchronous Particle Swarm Optimization (pdAPSO) Algorithm for Self-Organized Flocking of Swarm Robots. *IOSR Journal of Computer Engineering* (IOSR-JCE) e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 18, Issue 6, Ver. II (Nov. - Dec. 2016), pp. 59-71. Available at www.iosrjournals.org

Dada, E. G., and Ramlan, E. I. (2016). pdAPSO: The Fusion of Primal-Dual interior point method and particle swarm optimization algorithm. *Malaysian Journal of Computer Science*. Status (In Press).

E. Bahri, N. Harbi, H.N. Huu (2011). Approach based ensemble methods for better and faster intrusion detection, in: *Computational Intelligence in Security for Information Systems, Springer*, pp. 17–24.

F. Kuang, W. Xu, S. Zhang (2014). A novel hybrid KPCA and SVM with GA model for intrusion detection, *Appl. Soft Comput.* 18, 178–184.

http://kdd.ics.uci.edu/databases/kddcup99.

I. Syarif, E. Zaluska, A. Prugel-Bennett, G. Wills, (2012). Application of bagging, boosting and stacking to intrusion detection, in: Machine Learning and Data Mining in Pattern Recognition, Springer, 2012, pp. 593–602.

Jianxiao Liu and Lijuan Li (2008). Distributed Intrusion Detection System Based on Agents. 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application.

N. Araujo, R. de Oliveira, E. W. Ferreira, A. Shinoda, B. Bhargava (2010). Identifying important characteristics in the KDD99 intrusion detection dataset by feature selection using a hybrid approach, in: 2010 IEEE 17th International Conference on Telecommunications (ICT), IEEE, 2010, pp. 552–558.

V. Bukhtoyarov, V. Zhukov (2014). Ensemble-distributed approach in classification problem solution for intrusion detection systems, in: *Intelligent Data Engineering and Automated Learning-IDEAL* 2014, Springer, pp. 255–265.

W. Lee, S. J. Stolfo, K. W. Mok (1999). A Data Mining Framework for Building Intrusion Detection Models, IEEE Symposium on Security and Privacy, Oakland, California, pp. 120-132.

Y. Chen, M. L. Wong, H. Li (2014). Applying Ant Colony Optimization to configuring stacking ensembles for data mining, *Expert Syst. Appl*. 41 (6), 2688–2702.

Z. Cordeiro Jr., G.L. Pappa (2011). A PSO algorithm for improving multi-view classification, in: 2011 IEEE Congress on Evolutionary Computation (CEC), *IEEE*, pp. 925–932.