IEICE TRANS. INF. & SYST., VOL.E91–D, NO.4 APRIL 2008

PAPER

Modeling Network Intrusion Detection System Using Feature Selection and Parameters Optimization

Dong Seong KIM[†a)] ***and*** **Jong Sou PARK**[†b)], ***Members***

SUMMARY Previous approaches for modeling Intrusion Detection System (IDS) have been on twofold: improving detection model(s) in terms of (i) *feature selection* of audit data through wrapper and filter methods and (ii) *parameters optimization* of detection model design, based on classification, clustering algorithms, *etc.* In this paper, we present three approaches to model IDS in the context of feature selection and parameters optimization: First, we present Fusion of Genetic Algorithm (GA) and Support Vector Machines (SVM) (*FuGAS*), which employs combinations of GA and SVM through genetic operation and it is capable of building an optimal detection model with only selected important features and optimal parameters value. Second, we present Correlation-based Hybrid Feature Selection (*CoHyFS*), which utilizes a filter method in conjunction of GA for feature selection in order to reduce long training time. Third, we present Simultaneous Intrinsic Model Identification (*SIMI*), which adopts Random Forest (RF) and shows better intrusion detection rates and feature selection results, along with no additional computational overheads. We show the experimental results and analysis of three approaches on KDD 1999 intrusion detection datasets.

key words: *intrusion detection system, genetic algorithm, support vector machines, filter method, random forest, feature selection, parameters optimization, network security*

1. Introduction

As the amount of information which is interconnecting within networks has been increased tremendously, network security is becoming more essential. Among many security methods for protecting network systems such as firewalls and access control, Intrusion Detection System (IDS) plays a vital role in network security field. The main purpose of IDS is to inspect all inbound and outbound network activity and identify suspicious patterns that may indicate a network or system attack from someone attempting to compromise a system. IDS should guarantee high detection rates with minimum overheads to figure out intrusion detection model and process audit data. Previous approaches for modeling IDS have been on twofold: improving detection model(s) in terms of (i) *parameters optimization* of detection model design, based on classification, clustering algorithms, *etc.* and (ii) *feature selection* of audit data through wrapper and filter methods. First, the former case, many studies have been proposed on intrusion detection model based on various kinds of classification algorithms, clustering algorithms, and soft computing techniques, including Artificial Neural Networks (ANN), Hidden Markov Model (HMM) [1], Support Vector Machines (SVM) [6]–[8], [17], *etc.* The works has focus on optimization of parameters in their approaches. For instance, in case of ANN, it is essential to optimize the number of hidden layers, threshold functions so as to minimize the classification error (i.e. intrusion detection rates). Second, the latter case, many studies have tried to figure out important features or feature sets in order to not only minimize overhead of detection model but also maximize detection rates. In terms of feature selection, several researches have proposed identifying important intrusion features through wrapper and filter approaches. The wrapper method exploits a machine learning algorithm to evaluate the goodness of features or feature set. It provides better performance of selecting suitable features, since it employs performance of learning algorithm as an evaluation criterion. On the other hand, the filter method does not use any machine learning algorithm to filter out the irrelevant and redundant features rather it utilizes the underlying characteristics of the training data to evaluate the relevance of the features or feature set by some independent measures such as distance measure, correlation measures, consistency measures [3], [4]. Even though a number of feature selection techniques have been utilized in the fields of data-mining, however, there are very few analogous studies on intrusion detection paradigm [15], [17], [20], [29].

The above context, in this paper, we present three approaches to model IDS: (i) Fusions of GA and SVM (FuGAS) (ii) Correlation-based Hybrid Feature Selection (CoHyFS) and (iii) Simultaneous Intrinsic Model Identification (SIMI). First, FuGAS employs combinations of GA and SVM through genetic operation and it is capable of building an optimal detection model with only selected important features and optimal parameters value of intrusion detection model. FuGAS has two main problems: long training time to build intrusion detection models, and unstable feature selection results. Second, CoHyFS utilizes a filter method in conjunction of GA for feature selection in order to reduce long training time. In addition, CoHyFS shows stable feature selection results. However, CoHyFS shows a small degradation in detection rates and is complicated to implement. Third, SIMI utilizes Random Forest (RF), which has been shown similar or better intrusion detection rates, comparable to FuGAS and CoHyFS, along with no additional computational overheads. The experimental results on KDD 1999 intrusion detection datasets show the feasibility and comparisons of three approaches

Manuscript received February 26, 2007.
Manuscript revised August 30, 2007.

[†]The authors are with the Network and Embedded Security Lab, Korea Aerospace University, Republic of Korea.
a) E-mail: dongseong@gmail.com
b) E-mail: jspark@kau.ac.kr

DOI: 10.1093/ietisy/e91–d.4.1050

in terms of detection rates, feature selection and parameters optimization.

2. Proposed Approaches

2.1 Fusion of GA and SVM (FuGAS)

In this section, we illustrate the Fusion of GA [5], [22], [23] and SVM, named FuGAS. The previous researches on feature selection and parameters optimization in IDS have been performed in a separated way. In other words, feature selection and parameters optimization of detection model were separately performed. This causes inconsistency to figure out best intrusion detection model. FuGAS performs both feature selection and parameters optimization through genetic operation. The overall flow of FuGAS is depicted in Fig. 1. There are two dataset: training set and testing set. The training set is again separated into two set: learning set and validation set. The learning set is used to build detection models. The validation set is used to test the built detection models. And the testing set has additional instances, which are not appeared in the training set. The testing set is used to evaluate whether built detection models are able to cope with novel instances.

The features of audit data and parameters value of a kernel function in SVM are encoded as chromosomes, as depicted in Fig. 2.

In Fig. 2, a chromosome consists of two parts: The first part, expressed as n bits, represents total number of features of audit data and expressed a binary gene string. '1' represents that the feature is selected, on the other hands, '0' represents that the feature is not selected. The second part represents parameters value of kernel function in SVM, expressed as multi-valued gene string. '2' means RBF kernel function, '0.1' means a starting value, '0.2' means ending value. In the example, the granularity is '0.005'. It represents that the value changes 0.005 for each GA iteration. These values are changeable according to the experimental setting.

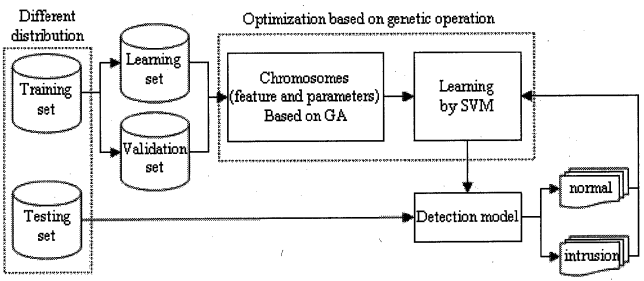


Fig. 1 The overall flow of FuGAS.

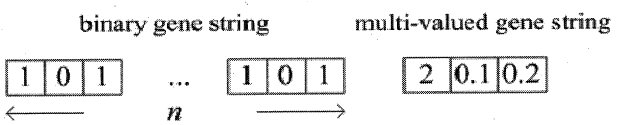


Fig. 2 Structure of a chromosome used in genetic operation.

The generic operation of GA is as follows: GA builds new chromosomes as shown in Fig. 2, and searches the *optimal detection model* based on the fitness function value (i.e. detection rates in IDS, mostly common in GA) obtained from SVM. The SVM is used to evaluate the performance of a detection model represented by a chromosome. In SVM, n-way cross-validation is used to prevent over-fitting problems [26], and the detection rates from n tests are averaged to obtain a fitness function value (i.e. detection rates). This procedure is iterated until it satisfies pre-defined fitness function value or it reaches to final iteration (i.e. final generation in GA). As the result of iteration, in FuGAS, it is cable of obtaining the optimal set of features as well as the optimal parameters for a kernel function in SVM.

2.2 Correlation Based Hybrid Feature Selection (CoHyFS)

Figure 3 illustrates the overall flow of CoHyFS. There are two dataset: Training set and testing set. In CoHyFS, training dataset is segregated into three sets: feature-selection set, model building set, and validation dataset. Feature-selection dataset is passed through the Correlation-based Hybrid Feature Selection process, which results in a set of selected features. The model building dataset is then used to build the intrusion detection models, using the selected features. And the intrusion detection models are evaluated by the validation set. The intrusion detection models are again evaluated by testing dataset to verify whether the intrusion detection models are able to cope with novel attacks and normal patterns.

CoHyFS also utilizes GA to generate feature subset and to figure out important features. It adopts different strategy in order to decrease the training and testing time, compared to FuGAS, described in Sect. 2.1. CoHyFS only encodes features into a chromosome. The structure of a chromosome representing a feature vector is depicted in Fig. 4.

The structure of a chromosome is simple than that of it in FuGAS. In FuGAS, both features and parameters' value are encoded as a chromosome. This causes the

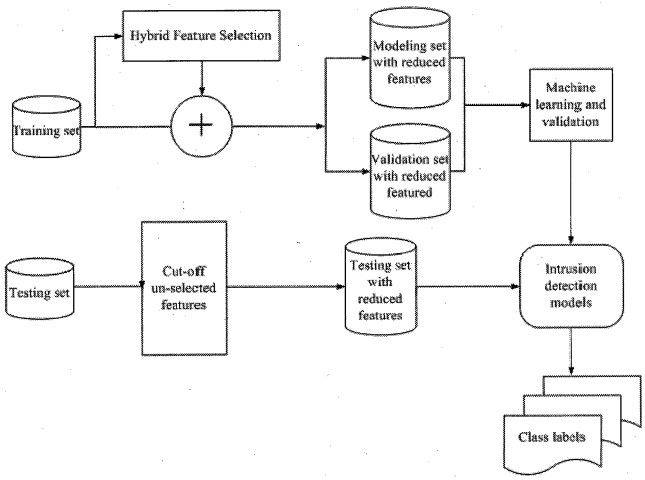


Fig. 3 The overall flow of CoHyFS.

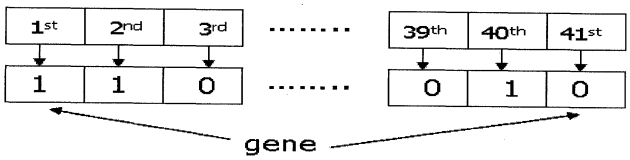


Fig. 4 Structure of a chromosome representing a feature vector.

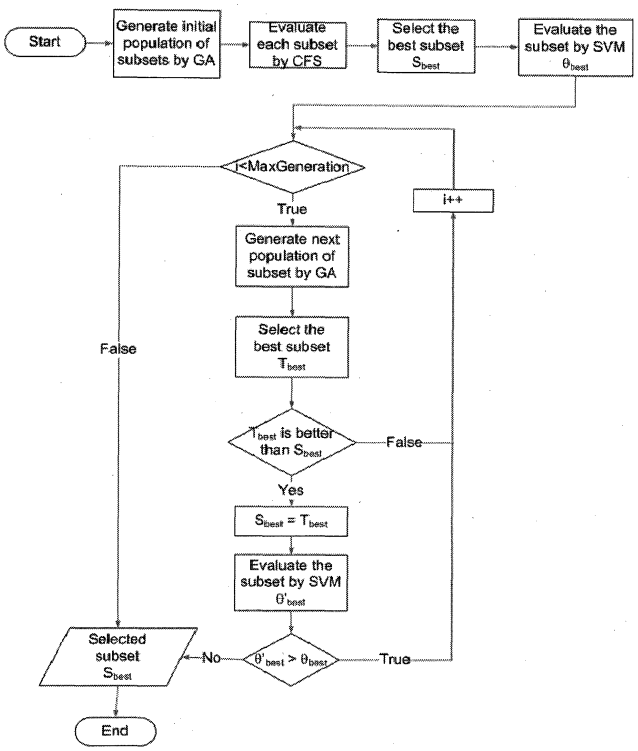


Fig. 5 A flow chart of correlation-based hybrid feature selection.

computational overheads in GA iteration in terms of training and testing time, due to cross validation in FuGAS. CoHyFS accordingly only focus on figuring out important features, based on correlation based hybrid feature selection, depicted in Fig. 5.

CoHyFS is a crafted combination of Correlation based Feature Selection (CFS) [18] and SVM. GA is used to generate subsets of features from given feature set. The algorithm takes full feature set as input and returns the optimal subset of feature after being evaluated by CFS and SVM. Each chromosome represents a feature vector. The length of the chromosome is 41 genes, in the dataset, where each gene (bit) may have values 1 or 0 which indicates whether corresponding feature is included or not in the feature vector, respectively. Like every stochastic algorithm, the initial population of chromosomes is generated randomly. Merit of each chromosome is calculated by CFS. The chromosome having highest merit, γ_{best} represents the best feature subset, S_{best} in population. This subset is then evaluated by SVM classification algorithm, and the value is stored in θ_{best} which represents metric of evaluation. Here we have chosen intrusion detection rates as a metric (a fitness function value in FuGAS) although a complex criterion such as a combination

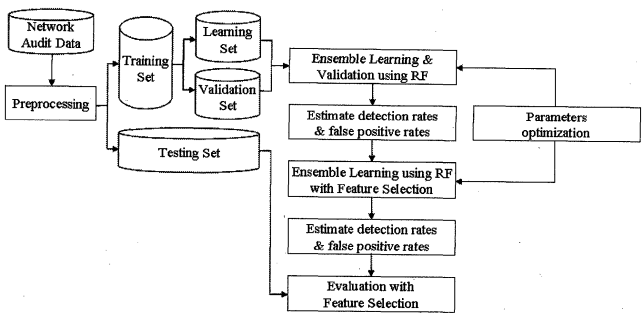


Fig. 6 The overall flow of SIMI.

of detection rate and false positive rate or a rule based criterion like [30] could be used. Then, genetic operations, selection, crossover and mutation, are performed and a new population of chromosomes is generated. In each generation, best chromosome or feature subset is compared by previous best subset, S_{best}. If newer subset is better than previous one, it is assigned as the best subset. This subset is then evaluated by SVM. If new detection rate is higher than previous one, this value is to θ_{best} and algorithm goes forward. Otherwise, the S_{best} is returned as the optimal subset of features. The algorithm stops if better subset is not found in next generation or when maximum number of generation is reached.

2.3 Simultaneous Intrinsic Model Identification (SIMI)

In this section, we present simultaneous intrinsic model identification (SIMI) depicted in Fig. 6. SIMI performs feature selection and parameter optimization simultaneously without any additional overheads, non-likely to FuGAS and CoHyFS. SIMI adopts Random Forest (RF) which is a stage-of-the-art data mining algorithm comparable to SVM [32].

The preprocessed network audit data is divided into two datasets; training and testing set. The training set is further separated into learning set and validation set. Although we do not need to perform cross-validation to get a balanced estimate of generalization error since RF is robust against over-fitting [32], [33], we adopt n-fold cross validation to minimize that. The learning set is used to generate classifiers and aggregate their results based on RF and find out variable importance of each feature of network audit data and optimal parameters for RF simultaneously. These classifiers can be considered as detection models in IDS. The validation set is used to compute detection rates according to estimating error rates which is Out-Of-Bag (OOB) errors in RF. Feature selection is performed by eliminating the irrelevant features which are low ranked in the ranking of variable importance. In other words, we only select top m numbers of important feature and optimize both of *mtry* and *ntree*. In next steps, therefore, only important features that have more effect on classification and optimal parameters are used to build detection models and evaluate by testing set with respect to detection rates. Our approach enables one to identify intrinsic model through this procedure.

3. Experiments and Analysis

3.1 Experimental Dataset

We have used the KDD 1999 intrusion detection dataset. The dataset contains a total of 24 attack types that fall into four main categories [12]: DoS (Denial of Service), R2L (unauthorized access from a remote machine), U2R (unauthorized access to root privileges) and probing. The data was preprocessed by extracting 41 features from the tcpdump data in the 1998 DARPA datasets and we have labeled them as f1, f2, f3, f4 and so forth. We have only used DoS type of attacks, since in [31] pointed out that it is not suitable to use U2R and R2L in KDD 1999 dataset to show the generalization of experiments. It doesn't mean that our approach can not deal with dataset, which has small number of instances. We chose only DoS type of attacks out of 4 types of attacks to show the validation of 3 approaches. In machine learning field, it's necessary to use enough data instance, for instance, at least more than 1000 samples a dataset.

3.2 FuGAS

We performed 10-fold cross validation with 2500 samples to reduce over-fitting problem [26]. After completing cross validation (in other word, learning and validation), we perform classification using testing set (corrected.gz) with 2000 samples to evaluate detection model constructed during learning phase. We estimate how well this model copes with novel attacks since there are 14 new attacks in testing set which are not included in training set. In the experimental results, the detection rates designate the accuracy of classification. For GA, we used tournament method for selection process, and set the probability of mutation and crossover, $P_m = 0.015$ and $P_c = 0.8$, respectively.

The result of cross validation is depicted in Fig. 7. The detection rates monotonously increase when all of three kernel functions are used in learning phase. While GA was executed for 20 generations, the learning process using SVM with neural kernel function [16], [17], [29] achieved the highest detection rates among the kernel function compared.

We figured out the number of selected features and optimal parameters in SVM, respectively. These features can be considered as most important features to detect DoS type attacks. The optimal parameters are different according to the kernel function in SVM. Most of the classification problems using SVM, radial kernel is selected and has showed a good performance, however, in our experiments, it indicates that neural kernel function for SVM shows better results. The reason why is that, according to *no free lunch theorem* [26] on machine learning, there is no superior kernel function in general, and the performance of a kernel function rather depends on applications. In addition, Fig. 8 shows that FuGAS guarantees the stability for detection rates. In case of detection model using neural kernel

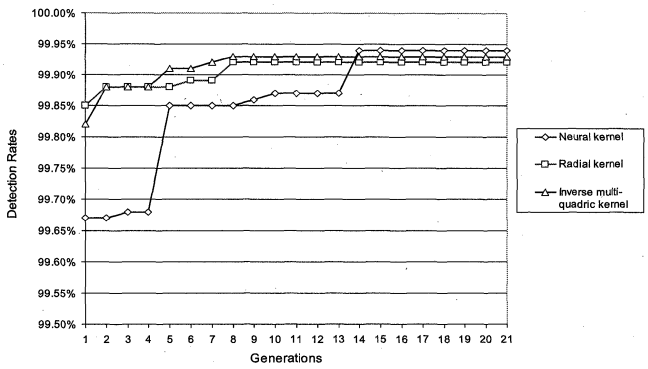


Fig. 7 The results of cross-validation: detection rates versus generations of GA execution.

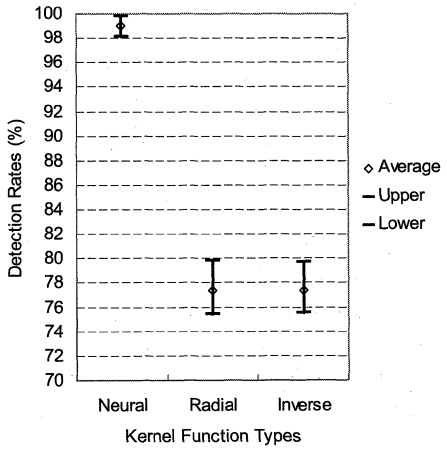


Fig. 8 Detection rates for each kernel function with feature selection in testing phase.

function can provide at least 98% detection rates. And it also shows the possibility that can cope with novel attacks since it shows detection rates more than 98% during testing phase. Because testing set includes 14 additional attack types which are not included in training set. In other words, our method has the potential of detecting previously unknown DoS types of attacks. And FuGAS provides higher detection rates than the approaches that only adopt SVM for IDS [8], [28]–[30].

However, there are two main problems in FuGAS, First, it consumes long time to perform training and testing. Refering Figs. 9 and 10, it takes more time to perform training and testing with all features. FuGAS uses whole features as well as parameter values. Second, the selected features are not stable and only feature set are selected. For instance, assuming that feature set, {f1, f4, f5} are selected, we can not figure out which feature, among three feature, is more important than others.

3.3 CoHyFS

For CoHyFS, we have used open source WEKA [13] library for SVM and CFS algorithm. For implementing our algorithm, we have modified several classes of WEKA library

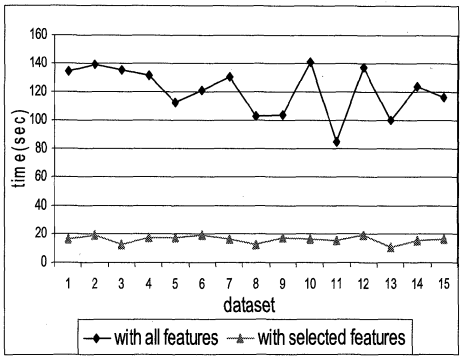


Fig. 9 Model training time vs. dataset index.

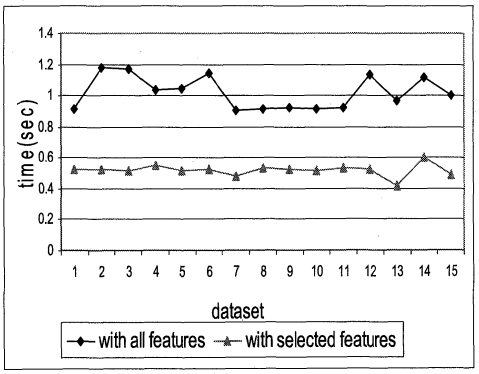


Fig. 10 Model testing time vs. dataset index.

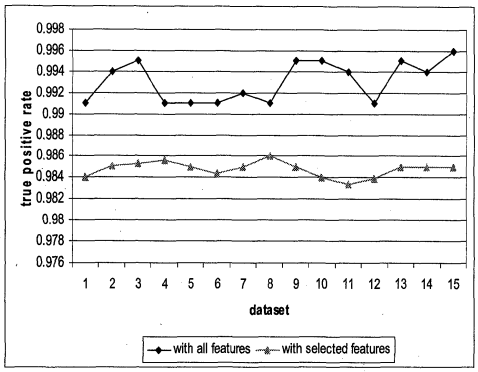


Fig. 11 Detection rates vs. dataset index.

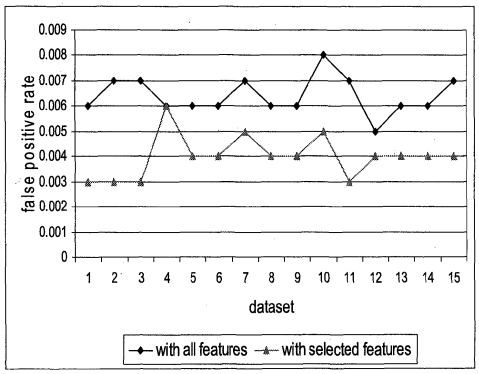


Fig. 12 False positive rates vs. dataset index.

such as "weka.attributeSelection.GeneticSearch".

For feature selection in CoHyFS, we have selected a data set randomly from 15 datasets and applied our algorithm which was described in previous section. We have applied 10 fold cross validation to achieve low generalization error and to determine the intrusion detection rate. The optimal subset selected has shown 99.56% detection rate. The indices of feature selected are f1, f6, f12, f14, f23, f24, f25, f31, f32, f37, f40 and f41. The dimension of feature vector is reduced from 43 to 12 that is a significant gain while the detection rate is above 99%.

The Figures from 9 to 12 show the comparisons between different performance indicators. Figure 9 shows that the dramatic reduction of model building time with reduced features as expected because the feature selection process has cut the 70% of total number of features. Testing time depicted in Fig. 10 also accedes with model training time, compared to FuGAS.

For selected features, though the detection rate is lower than that of having full features the decrement is very small, in other words, around 0.83% in average (see Fig. 11). But the significant performance enhancement was achieved in the reduction of false positive rate (see Fig. 12), which is 37.5% in average. For all above experiments, we used polynomial kernel of exponent 1 and c = 1 that are default value for SVM in WEKA. If we optimize SVM in more, the intrusion detection rates would be improved. It is noteworthy that we have not taken any measure of optimizing the kernel and SVM parameters as our main goal is to investigate that how hybrid feature selection reduced the computational resource while maintaining the detection and false positive rate within tolerable range.

Enhancement of detection rate and optimization between false positive and detection rate can be improved further by parameter tuning, exploiting better kernel function and improving classification algorithm [10], [16].

3.4 SIMI

In order to evaluate SIMI, RF version (R 2.2.0) and MDS algorithm in open source R-project [34] is used to perform several experiments. There are only two parameters in RF to be optimized; the number of variables in the random subset at each node (*mtry*) and the number of trees in the forest (*ntree*). To get the best classification rates, that is, the best detection rates, it is essential to optimize both two parameters. This is considered as parameters optimization. Fortunately, we could get the optimal value of *mtry* using tuneRF() function provided in randomForest package of R-project and it turned out *mtry* = 6. In case of *ntree*, there is no specific function that figures out the optimal value of it. Thus, we got the optimal value of *ntree* by choosing the *ntree* value that has high and stable detection rates. We assume that 350 trees are enough to be the maximum value to evaluate our approach and detection rates are determined

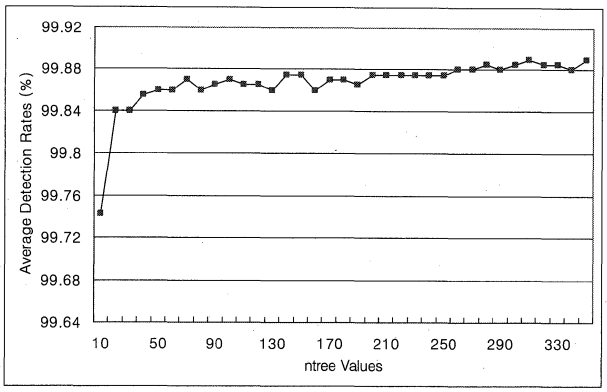


Fig. 13 Average detection rates vs. ntree values.

Table 1 Top 5 important features.

Feature	Properties	Average variable importance
*f*23	number of connections to the same host as the current connection in the past two seconds	0.4023
*f*6	number of data bytes from destination to source	0.3318
*f*24	number of connections to the same service as the current connection in the past two seconds	0.3172
*f*3	network service on the destination, e.g., http, telnet, etc.	10.3163
*f*5	number of data bytes from source to destination	0.2973

by equation "1–OOB errors". The experimental results for determination of the optimal value of *ntree* are described in Fig. 13.

According to Fig. 13, average detection rates of RF turned out the highest value when *ntree* = 310. As the result of experiments, we set two optimized parameter values; *mtry* = 6, *ntree* = 310. After optimizing two parameters, feature selection of network audit data was carried out employing the feature selection algorithm supported by RF. We ranked features thorough the average variable importance of each feature as the results of 10-fold with 2000 samples. As the results, feature important of each individual feature were decided. The importance value of each feature varies and we rank features with respect to their average importance values of cross validation experiments.

We partially show the top 5 important features and their properties in Table 1. Our approach showed reasonable context information for each important feature. For instance, f23 represents "number of connections to the same host as the current connection in the past two seconds" property and f6 represents "number of data bytes from destination to source" and so on.

Then, we carried out several times of experiments with

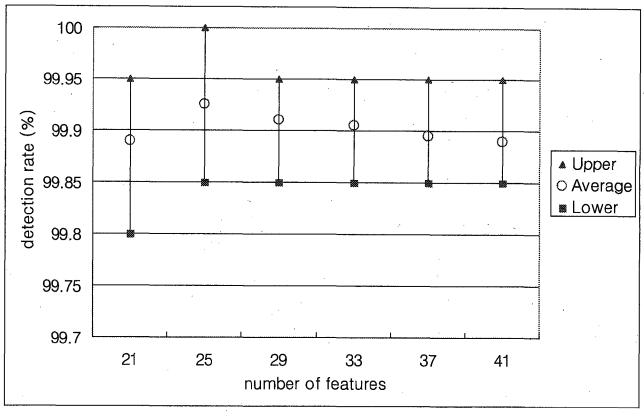


Fig. 14 Detection rates and total number of selected features.

Table 2 The comparison between our approach and previous approaches.

Approach	Detection rates	Feature selection		Parameters optimization	
		Method	result	method	
FuGA	99.85%	GA	Optimal feature set	GA	Optimal parameters for Kernel function in SVM
CoHyFS	98.4%	Filter method with GA	Optimal feature set	N/A	Default value of SVM
SIMI	99.87 %	RF	Individual feature importance/ *m* features remain	*mtry* and *ntree*	Optimal RF

elimination of irrelevant features and measure detection rates. The experimental results are depicted in Fig. 14.

3.5 Comparison and Discussion

In Table 2, we present comparison results. SIMI showed higher detection rates than others. Even though the detection rates is slightly high than others, SIMI only used selected important features and training and testing time is faster than others. Although both FuGAS and CoHyFS have showed "optimal feature set", they didn't show the numeric value as the variable importance of each feature. SIMI is able to get individual feature importance so that only important individual features can be used. We need to calculate computational complexity and compare it to other approaches. But this is out of scope of this paper because both FuGAS and CoHyFS utilize GA [31].

4. Conclusions

In this paper, we have present three approaches to model lightweight Intrusion Detection System. FuGAS employed a fusion of GA and SVM, which perform both feature selection and parameters optimization and showed better detection rates than the approaches, only adopts SVM. FuGAS,

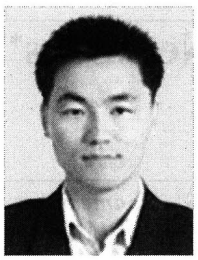
however, have additional computational overhead due to training and testing time. To cope with this, CoHyFS have been proposed, which is based on a filter method in conjunction with GA. CoHyFS shows a little degradation in detection rates but the degradation is marginal and showed faster training and testing time with stable important feature sets. Both FuGAS and CoHyFS showed the implication to IDS, it has still room for further enhancement so that SIMI has been proposed. SIMI showed that IDS based on RF is easily built without additional overhead compared to FuGAS and CoHyFS. We carried out several experiments on KDD 1999 dataset and the results showed that feature selection and parameters optimization are able to help one to model and implement IDS.

Acknowledgements

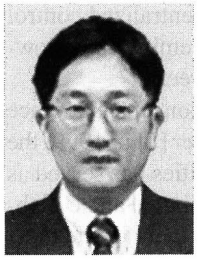
The authors are grateful to anonymous reviewers for their invaluable comments to revise the manuscript. We would like to thank Sang Min Lee, Khaja M. Sazzad, and Ha-Nam Nguyen for their help. This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2007-013-D00095).

References

[1] D. Ourston, S. Matzner, W. Stump, and B. Hopkins, "Applications of hidden Markov models to detect multi-stage network attacks," Proc. 36th Hawaii Int. Conf. on System Sciences, p.334, 2003.
[2] M.A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," Proc. 17th Int'l Conf. on Machine Learning, pp.359–366, 2000.
[3] M. Dash, H. Liu, and H. Motoda, "Consistency based feature selection," Proc. Fourth Pacific Asia Conf. on Knowledge Discovery and Data Mining (PAKDD), pp.98–109, Kyoto, Japan, 2000.
[4] H. Almuallim and T.G. Dietterich, "Learning Boolean concepts in the presence of many irrelevant features," Artif. Intell., vol.69, nos.1-2, pp.279–305, 1994.
[5] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.
[6] M. Fugate and J.R. Gattiker, "Anomaly detection enhanced classification in computer intrusion detection," Lecture Notes in Computer Science, vol.2388, Springer-Verlag, Berlin Heidelberg, 2002.
[7] B. Nguyen, "An application of support vector machines to anomaly detection," Research in Computer Science—Support Vector Machine, Report, Fall 2002.
[8] W. Hu, Y. Liao, and V. Vemuri, "Robust support vector machines for anomaly detection in computer security," Proc. Int. Conf. on Machine Learning and Applications 2003, pp.168–174, CSREA Press, 2003.
[9] J. Cannady, "Artificial neural network for misuse detection," Proc. 1998 National Information System Security Conf., pp.443–456, Arlington, Oct. 1998.
[10] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," Machine Learning Journal, vol.46, pp.131–159, 2002.
[11] K. Duan, S.S. Keerthi, and A.N. Poo, "Evaluation of simple performance measures for tuning SVM hyperparameters," Neurocomputing, vol.51, pp.41–59, 2003.
[12] KDD Cup 1999 Data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[13] Open Source WEKA Project, http://www.cs.waikato.ac.nz/ml/weka/index.html
[14] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," IEEE Trans. Knowl. Data Eng., vol.17, no.3, pp.1–12, 2005.
[15] A.H. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," Proc. 2003 Int. Symposium on Applications and the Internet Technology, pp.209–216, IEEE Computer Society Press, 2003.
[16] D. Kim, H.-N. Nguyen, S.-Y. Ohn, and J. Park, "Fusions of GA and SVM for anomaly detection in intrusion detection system," Lecture Notes in Computer Science, vol.3498, pp.415–420, Springer Verlag, 2005.
[17] V. Vapnik, The Nature of Statistical Learning Theory, Springer, Berlin Heidelberg, New York, 1995.
[18] U. Fayyad and K. Irani, "Multi-interval discretization of continuous attributes as pre-processing for classification learning," Proc. 13th Int. Join Conf. on Artificial Intelligence, pp.1022–1027, 1993.
[19] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, Numerical recipes in C, Cambridge University Press, Cambridge, 1988.
[20] S. Chebrolu, A. Abraham, and J.P. Thomas, "Data reduction and data classification in an intrusion detection system," Proc. South Central Information Security Symposium, Texas, USA, 2004.
[21] M. Sabhnani and G. Serpen, "On failure of machine learning algorithms for detecting misuse in KDD intrusion detection data set," Journal of Intelligent Data Analysis, vol.8, no.4, pp.403–415, 2004.
[22] D.E. Goldberg, Genetic Algorithms in Search, Optimization & Machine Learning, Addison Wesley, 1989.
[23] M. Mitchell, Introduction to genetic Algorithms, 5th printing, MIT Press, 1999.
[24] Z. Michalewicz, Genetic Algorithms + Data structures = Evolution Programs, 3 re rev. and extended ed., Springer-Verlag, 1996.
[25] N. Cristianini and J.S. Taylor, An introduction to support vector machines and other kernel-based learning methods, Cambridge University Press, 2000.
[26] R.O. Duda, P.E. Hart, and D.G. Stork, Pattern Classification, 2nd ed., Wiley Interscience Inc., 2001.
[27] T. Joachims, "Making large-scale SVM learning practical," in Advances in Kernel Methods — Support Vector Learning, ed. B. Scholkopf, C. Burges, and A. Smola, chapter 11, MIT Press, 1999.
[28] M. Fugate and J.R. Gattiker, "Anomaly detection enhanced classification in computer intrusion detection," LANL Tech Report, LA-UR-02-1149, 2002.
[29] D. Kim and J. Park, "Network-based intrusion detection with support vector machines," ICOIN 2003, Lecture Notes in Computer Science, vol.2662, Springer-Verlag, 2003.
[30] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," Proc. IEEE Int. Joint Conf. on Neural Networks, pp.1702–1707, 2002.
[31] B. Rylander, Computational Complexity and the Genetic Algorithm, Ph.D. Thesis, University of Idaho, 2001.
[32] L. Breiman, "Random forest," Mach. Learn., vol.45, pp.5–32, 2001.
[33] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, Classification and Regression Trees, Chapman and Hall, New York, 1984.
[34] The R Project for Statistical Computing, http://www.r-project.org/



Dong Seong Kim received the B.S. degrees in Electronic Engineering from Korea Aerospace University, Republic of Korea in 2001. And he received M.S. and Ph.D degree in Computer Engineering from Korea Aerospace University, Republic of Korea in 2003, 2008, respectively. And he was a visiting research associate in University of Maryland at College Park in 2007. His research interests are in all area of network security, ubiquitous computing security.



Jong Sou Park received the M.S. degree in Electrical and Computer Engineering from North Carolina State University in 1986. And he received his Ph.D in Computer Engineering from The Pennsylvania State University in 1994. From 1994–1996, he worked as an assistant Professor at The Pennsylvania State University in Computer Engineering Department and he was president of the KSEA Central PA, Chapter. He is currently a full professor in Computer Engineering Department, Korea Aerospace University. His main research interests are information security, embedded system and hardware design. He is a member of IEEE and he is an executive board member of the Korea Institute of Information Security and Cryptology and Korea Information Assurance Society.