NETWORK ANOMALY DETECTION BASED ON LATE FUSION OF SEVERAL MACHINE LEARNING ALGORITHMS

Tran Hoang Hai[1], Le Huy Hoang[1], and Eui-nam Huh[2]

[1]School of Information and Communication Technology,
Hanoi University of Science and Technology, Vietnam
[2]Department of Computer Science and Engineering,
Kyung Hee University, Yongin, Korea

ABSTRACT

Today's Internet and enterprise networks are so popular as they can easily provide multimedia and e-commerce services to millions of users over the Internet in our daily lives. Since then, security has been a challenging problem in the Internet's world. That issue is called Cyberwar, in which attackers can aim or raise Distributed Denial of Service (DDoS) to others to take down the operation of enterprises Intranet. Therefore, the need of applying an Intrusion Detection System (IDS) is very important to enterprise networks. In this paper, we propose a smarter solution to detect network anomalies in Cyberwar using Stacking techniques in which we apply three popular machine learning models: k-nearest neighbor algorithm (KNN), Adaptive Boosting (AdaBoost), and Random Decision Forests (RandomForest). Our proposed scheme uses the Logistic Regression method to automatically search for better parameters to the Stacking model. We do the performance evaluation of our proposed scheme on the latest data set NSL-KDD 2019 dataset. We also compare the achieved results with individual machine learning models to show that our proposed model achieves much higher accuracy than previous works.

KEYWORDS

Network Security, Intrusion Detection System, Anomaly Detection, Machine Learning.

1. INTRODUCTION

Network Intrusion Detection System (N-IDS) plays an extremely important role in security management which can support network administrators about unusual behaviors where a traffic flow might be an intrusion, attack, or normal traffic flow. Currently, network administrators apply some solutions such as firewalls to prevent some unwanted traffics. However, network managers must conduct manual detection. In the traditional rule-based N-IDS, the rules are usually pre-defined by the security experts and need to be updated regularly [1,2]. The advantage of rule-based N-IDS is better known for attack detection. [3]. Therefore, we propose a smart N-IDS which can capture network traffic, analyze, and detect network anomalies automatically. With the rapid development of machine learning models, several methods have been proposed to build a knowledge system on the IDS system [4-6], where abnormal traffic can be detected and prevented automatically. Another type of N-IDS based on statistical analysis analyses the statistical behavior of users to find abnormal behaviors [7]. We believe that a knowledge system based on the latest development of machine learning models to combat the risks is extremely important [8-10]. Some related works based on statistical methods [11] and Bayers algorithm [12] are typical representative algorithms in this field. An expert system is currently the most

feasible solution which uses artificial intelligence to solve problems in a field that requires human expertise[10]. The application of machine learning algorithms is a breakthrough that provides us an efficient tool to apply N-IDS in practice. Sincethe publication of the KDD99 data set, there have been several works on using machine learning for anomaly detection which has different characteristics, and efficiency/ accuracy level [13]. The stacking technique is an ensemble learning [14, 15] that takes advantage of different machine learning algorithms to include the predictions of those models for a better one. Stacking is based on the latest classification algorithms [16-17] in which several machine learning algorithms can be used simultaneously. Stacking aggregates different models to derive a better one, thus reducing the probability of false predictions and improving accuracy. Moreover, stacking does not require complex algorithms implemented on the system. In this paper, we propose a matrix that each algorithm will be assigned to an element of that matrix, so when the proposed system processes a final calculation, the model using this matrix factor multiplied by each algorithm prediction to get the final result. The problem is how to choose good coefficients to have a better result. So, we propose to use a logistic algorithm [16-17] to select parameters for our proposed model. The rest of the paper is organized as follows. Section 2 gives an overview of related works and NSL-KDD2019 and other data sets. Section 3 presents our proposed model for network anomaly detection. Section 4 introduces a performance comparison between our proposed model with individual machine learning algorithms. Finally, conclusions are given in Section 5.

2. RELATED WORKS

2.1. Network Anomaly Detection Data Sets

In previous works, there are several data sets to evaluate the performance of machine learning for N-IDS such as DARPA98, KDD CUP 99 [30], CICIDS2017 [31], and NSL-KDD which usually classify by packet-based or flow-based data [26]. Labeled data sets are very important which is used to train and evaluate the anomaly-based N-IDS. The DARPA 1998/99 is the most popular data sets created at the MIT Lincoln Lab which includes various kinds of attacks like DoS, buffer overflow, port scans, or rootkits [27-28]. However, this data set does not reflect the actual traffic because it was simulated in the Lab.KDD CUP 99 is one of the most popular data sets for anomaly detection which contains basic attributes about TCP connections and high-level attributes like the number of failed logins without IP addresses [29]. However, this data set contains too many redundant and duplicate data, About 78% of the data is duplicated in the training data, and 75% in the testing data. NSL-KDD 2019 is the up-to-date data set we chose for testing the model since it has a lot of improvement compared to KDD CUP 99. NSL-KDD data set was created as an optimized version of KDD'99 from the University of New Brunswick [25]. The complete data set NSL-KDD 2019 is an up-to-date data set which contains new types of attacks without duplicates from the KDD'99 data set. This resulting data set contains about 150,000 data points and is divided into predefined training and test subsets which are KDDTest+, KDDTest-21, KDDTrain+, KDDTrain+_20Percent where KDDTest-21 and KDDTrain + _20Percent are subsets of KDDTest+ and KDDTrain+. KDDTrain+ is considered training data and KDDTest+ is considered testing data. KDDTest-21 is a subset of the testing data which removes the most difficult data records (point 21). KDDTrain _20Percent is a subset of the training data where the number of records equal to 20% of the total number of records in the training data. In other words, the records in KDDTest-21 and KDDTrain+_20Percent are included in testing and training data and no records exist in both data sets at the same time which makes the evaluation of anomaly detection more accurate.

2.2. Recent Machine Learning Algorithms for Anomaly Detection

In this paper, we choose three following machine learning algorithms: k-nearest neighbor algorithm (KNN) [18], Adaptive Boosting (AdaBoost) [19], and Random Decision Forests (RandomForest) [20] for our proposed model. The motivation is that those algorithms are recent works on applying machine learning to N-IDS and they provide better results and lower processing time compared to others. In [18], the authors propose to use the PCA-fuzzy Clustering-KNN method which ensemble of Analysis of Principal Component and Fuzzy Clustering with K-Nearest Neighbor feature selection techniques to detect anomalies. In [19], the authors proposed the AdaBoost algorithm for N-IDS which provides competitive performances compared with other works on the old KDDCUP1999 data set. In [20], the authors proposed a model for N-IDS using the Random Forest classifier which performs well compared to other traditional classifiers. In [21], A. Ahmim and colleagues proposed a new N-IDS model that includes various classification methods based on 'decision tree' and 'rule-base' which are REP Tree, JRip, and Forest PA using the CICIDS 2017 data set. In particular, the first two classification models take input data as network data set characteristics and classify them into normal and anomalous groups. The third classification method uses the characteristics of the original data and combines with the data that has been processed by two previous methods to classify each specific type of attack. Therefore, their proposed model achieved an identification rate of up to 94,475% and reduced the error warning rate to 1,145%, which it is more efficient than Naive Bayes, Random Forest, Support Vector Machine (SVM). With the same CICIDS 2017 data set, in [22], D. Aksu and colleagues proposed a model using the fisher score algorithm to select 30 from 80 characteristics/ features of network flow. Then, the authors apply to machine learning algorithms such as SVM, KNN, and Decision Tree (DT) algorithms where the results seem very promising. In the NSL-KDDdata set [23-24], the authors applied several feature selection algorithms which are J48, SU, and Random Forest algorithm to classify four network attacks. The attacks are Dos, Probe, R2L, U2R on KDD99. In [18], the authors applied Analysis of Principal Component (APC) and Fuzzy Clustering with KNN to reduce the data dimension of the data set then they passed the data via fuzzing clustering algorithm to detect anomaly traffic or not. Their proposed model is quite satisfactory when the accuracy of DoS is up to 94.23%.

2.3. K-nearest neighbors algorithm (KNN)

KNN is one of the simplest supervised-learning algorithms and a non-parametric method used for classification and regression proposed by Thomas Cover [32-33]. The algorithm does not learn anything from training data and all calculations are done when it needs to predict the results of new data.). In KNN classification, the output is a class membership is classified by a majority vote of its neighbors. The point is assigned to the class most common among its K nearest neighbor points. In [34], the authors proposed a technique to improve the accuracy of KNN where it provides weight for each point being considered. Further points play smaller roles in the classification of that point and vice versa. In [35], the authors proposed a TCM-KNN model (Transductive Confidence Machines for K-Nearest Neighbours) on the KDD'99 data set. In general, anomalies can be identified using a genetic weighted KNN based on a classifier [36]. In [37], the authors studied and compared the performances of various clustering algorithms for anomaly detection: k-Means; improved k-Means; k-Medoids; EM clustering and distance-based outlier detection algorithms. In [39], the authors proposed two hierarchical IDS frameworks using Radial Basis Functions (RBF) which can detect network intrusions in real time using the old KDD Cup 1999 Data.

2.4. Adaptive Boosting Algorithm (AdaBoost)

Adaptive Boosting (AdaBoost) is a machine learning meta-algorithm proposed by Yoav Freund and Robert Schapire [38]. AdaBoost is classified as a boosting class because it aims to convert weak classification algorithms, correct previous algorithm errors then finally get a strong classifier. In this paper, we apply the Decision Tree (DT) algorithm [40] in AdaBoost because of the following motivation:

- Since AdaBoost does not work well with linear algorithms, then we try to apply DT since it is not a linear algorithm.
- DT algorithm shows good consistency; therefore, we do not need to adjust any parameters.
- DT algorithm is built quickly, therefore we expect to combine several algorithms together, so the training process is not overloaded.

2.5. Random Forest (RF)

Random Forests (RF) is an ensemble learning method for classification, regression which constructs a multitude of decision trees at training time, and outputting which is classification or regression of the individual trees [41,42]. Following [43], RD uses multiple decision trees for layering. The algorithm assumes that if a sample is layered by multiple decision trees, whichever layer is chosen by most trees, then this sample will be assigned to that class. In [20, 44], several authors show that the RF model applied in N-IDS is efficient with low false alarm rate and high detection rate. For better accuracy, RF uses a Bootstrapping process. This is a statistical resampling technique that involves random sampling of a data set with replacement [45]. In addition, to make sure the decision trees are different, RF will randomly skip a few attributes when building a decision tree. In this case, if the best attribute is not selected, the next attribute will be selected to build the tree. This process is called attribute sampling.

2.6. Logistic Algorithm

The logistic algorithm estimates the parameters of a logistic model (a form of binary regression) whose method is quite flexible compared to other linear algorithms. This algorithm show several advantages in [46] which are (a) significance tests of the model against the null model, (b) the significance test of each predictor, (c) descriptive and inferential goodness-of-fit indices, (d) and predicted probabilities Based on the distribution of data points, we predict what the probability of being in a group is, the output we can take a coefficient matrix to calculate the probability of the points to be grouped in a class. We will use this feature to calculate parameters for what we call automatic Stacking. In the next section, we will propose a Stacking model to combine KNN, AdaBoost, and RF to increase accuracy level of N-IDSm, and logistic play an important role in this model.

3. PROPOSED MODEL

3.1. Late Fusion Approach to Anomaly Detection

With the implementation of individual ML algorithms, we can get the accuracy of each model. However, we would like to increase the accuracy. One method is extracting important features that are relevant to attacks [47]. Another approach is to modify traditional ML algorithms to increase accuracy. This process is very costly and time-consuming, and somehow the results seem not general. Therefore, we propose to combine the results of KNN, RF, and AdaBoost to increase the overall accuracy of the mode. Our simple Stack model will work as following:

*Final result = x * result (KNN) + y * result (AdaBoost) + z * result (RandomForest);*

The problem with our proposed Stacking model is how to choose [x, y, z] parameters for better accuracy. Therefore, we apply a logistic algorithm because the nature of logistics is to classify data points. The logistic curve is the probability that the data to be labeled is labeled 1 or 0 when the data points are the same but different stratification. For example, a data point when tested with the KNN algorithm is classified into layer 1, AdaBoost is assigned to layer 0, and Random Forest is assigned to layer 1, the logistic will give us a probability of this data point is how many percent of the labels 0 and 1 are assigned. This technique helps to improve the accuracy of our proposed model.

3.2. Detail Implementation

To evaluate the Stacking model, we propose two steps need to be done as follows:

1. **Pre-processing data:** In Figure 1, we process the data (digitalization) from the NSL-KDD 2019 data set since it has some features including string data. Then, we divide this data set into three subsets which are Training; Validation; and Testing. After that, we need to bring data to a certain range to avoid an imbalance between features. Specifically, we use min-max scaling (sometimes called min-max normalization) which is calculated based on min, max value of training data [55]. Then, we have three data subsets as in Figure 1.
2. **Proposed Stacking model:** In Figure 2 is the illustration of our proposed Stacking model. In the model, Training and Validation data are inputs for the three machine learning algorithms which we apply KNN, RandomForest, AdaBoost. Y_pred_KNN is the predicted output from KNN, Y_pred_AdaBoost is the predicted output from AdaBoost and Y_pred_RandomForest is the predicted output of the RandomForest algorithm model respectively. In 2nd step, we combine the predictions of three algorithm models into one, we suppose following:
   Y_pred_KNN = $[y_{00}, y_{01}, y_{02}, \ldots, y_{0N}]$
   Y_pred_AdaBoost = $[y_{10}, y_{11}, y_{12}, \ldots, y_{1N}]$
   Y_pred_RandomForest = $[y_{20}, y_{21}, y_{22}, \ldots, y_{2N}]$

After obtaining results from the individual algorithm model, we sum up the results to have a matrix Xi = [y0i, y1i, y2i,]. We put X into the logistic regression algorithm where each row in the matrix is a sample and the label will be the label of the data validation. After that, the accuracy of the individual model and the whole model is calculated. Some parameters we use in the model for three machine learning algorithms are:

n_neighbor = 5 with KNN which is the number of neighbors to use by default for neighbors queries;

maxdepth = 1 and n_estimators = 100 with AdaBoost which is the maximum number of estimators at which boosting is terminated. In case of perfect fit, the learning procedure will stop early;

n_estimators = 100 with Random Forest which is the number of trees in the forest.

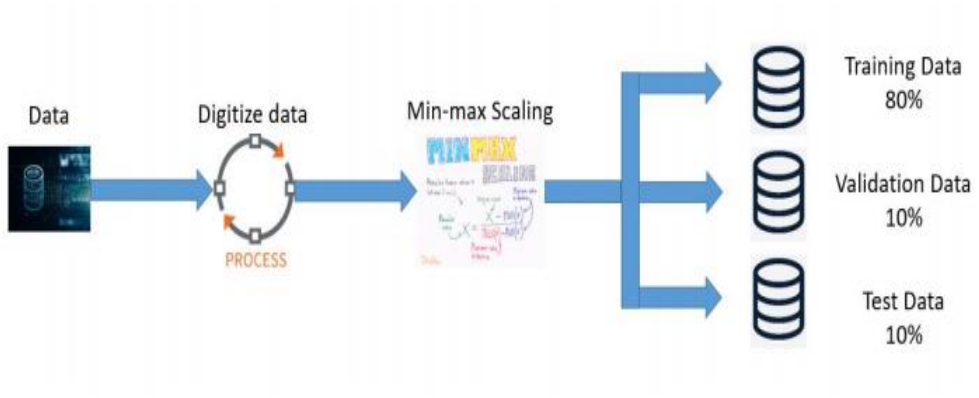

Figure 1. Pre-processing NSL-KDD 2019 data

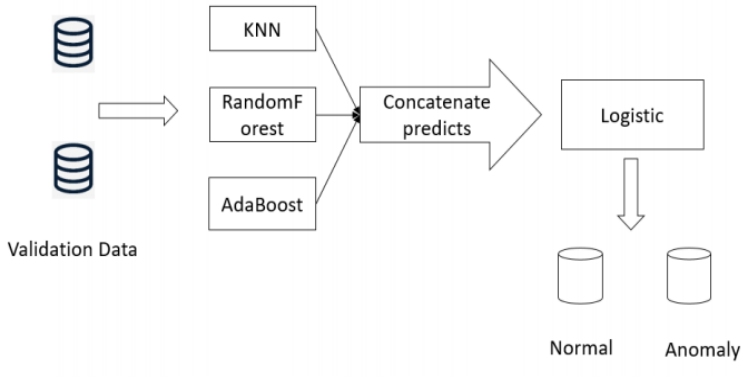

Figure 2. Proposed Stacking model

3.3. Accuracy Calculation of the Model

The simplest and most common performance metric is Accuracy. This metric simply calculates the ratio between the number of correctly predicted points and the total number of points in the test data set. The accuracy is calculated as the following:

$$accuracy = \frac{number\ of\ right\ prediction}{number\ of\ data}$$

For example, if we test on 1000 data, the number of times the model correctly predicts the other 1000 data is 912, which means that the accuracy of the model on that data set is approximately 91.2%. However, the accuracy in some cases has some drawbacks or only provides the general view of network attack. Instead of calculating the correct number of projections, we calculate the number of correct predictions on each label, specifically:

$$accuracy\ (from\ label\ 0) = \frac{number\ of\ right\ prediction\ label\ 0}{number\ of\ data}$$

Data: The number of data input in the test is 1000, there are 4 types of labels {0,1,2,3}

- *Label 0*: predicts exactly 250; the number of labels 0 are 300 which occupies 30% of the data

- *Label 1*: predicts exactly 0, the number of labels 1 are 100 which accounts for 10% of data
- *Label 2*: predicts exactly 150, the number of labels 2: 200 makes up 20% of the data
- *Label 3*: predicts exactly 300, the number of labels 3: 400 makes up 40% of the data

Finally, the accuracy level is 70%. Moreover, we obtain the accuracy levels of individual (Label 0) is 83.33%; accuracy(Label 1) is 0%; accuracy(Label 2) is 75%; and accuracy(Label 3) is 75% respectively. Accuracy on each label will affect general accuracy according to the amount of data that label occupies in the data set. Again, we can calculate the general accuracy by individual Label accuracy as follows:

$$Accuracy = 83.33\% * 30\% + 0\% * 10\% + 75\% * 20\% + 75\% * 40\% = 70\%;$$

We change the testing data with the data distribution on the labels appropriately, we can achieve very different levels of accuracy. For example, we assume that the data has two labels in which one is 100% accurate and the other is 70%. So, the general accuracy will be 100% * 50% + 70% * 50% = 85%. However, sometimes it is difficult for us to obtain data uniformly, so we need better metrics to evaluate the models which are Precision; Recall; and F1 score (or F-score) [49]. In statistics, the F-score is a measure of a test's accuracy [50].

$$Precision = \frac{TP}{TP + FP}$$

and

$$Recall = \frac{TP}{TP + FN}$$

and

$$\frac{2}{F1} = \frac{1}{Precision} + \frac{1}{Recall}$$

In order to clarify the meaning of those parameters, we have some following definitions on True Positive; True Negative; False Positive; and False Negative which are usually used in N-IDS anomaly detection [51].

- True Positive (TP): The number of points of the positive class that is correctly classified as positive.
- True Negative (TN): The number of points of the negative class that is classified as negative.
- False Positive (FP): the number of points of the negative class that is mistakenly classified as positive.
- False Negative (FN): the number of points of a positive class that has been misclassified as negative.

We can see that the Precision parameter demonstrates the ability to predict the X label correctly. We also see that in Precision's formula, the factor that makes Precision increase or decrease is not TP but FP. Therefore, when a high Precision means a small FP or several incorrectly predicted labels to an X label is low. The Recall parameter shows the ability to predict without missing the X label, as well as the Precision, Recall depends on FN, or in other words it depends on the ability of the model to correctly predict the wrong label is X. In evaluating the effectiveness of these models, we always expect both Precision and Recall parameters to be high as much as possible. Unfortunately, there is always a trade-off between two parameters. For example, when a high Precision parameter often entails a lower Recall parameter and vice versa. The reason is if the high Precision parameter means that the model must have very high accuracy to predict the X

label, but the opposite can make the prediction missing the actual data as X label, vice versa. Therefore, we need to use another parameter to synthesize these two parameters by another parameter, F1 score, to evaluate the model for more accuracy and efficiency.

4. PERFORMANCE EVALUATION

In this section, we do a performance evaluation of three individual machine learning models which are KNN, Random Forest, AdaBoost, and our proposed Stacking model on an up-to-date NSL-KDD 2019 data set. The computing environment is Ubuntu 18.0.4 64-bit; CPU Intel i7 8750H chip, 6 cores 12 threads; GPU: Intel UHD Graphic 630 4Gb; 8Gb RAM DDR4 and the Pycharm software [48].

4.1. Evaluation of Individual KNN Model

KNN works by taking a data point and looking at the *'k'* closest data points. The data point is then assigned the label of the majority of the *'k'* closest points, therefore, how to choose a n_neighbor parameter in the individual KNN model is very important for our proposed model. We try several n_neighbor values starting at 1 and ending in 8. In Figure 3, we can see the accuracy level corresponding to the n_neighbor values. With the value of k = 5, the accuracy level almost reaches peak constant Therefore, we choose the value of k = 5 for our proposed Stacking model.

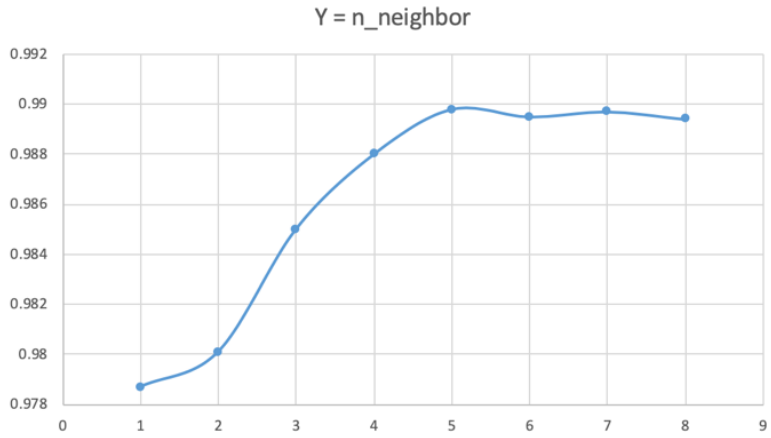

Figure 3. Accuracy level of different *n_neighbor* of KNN model

4.2. Evaluation of Individual Adaboost Model

We select the parameter n_estimators for the AdaBoost model with each value of n_estimators running with k = 5 times, then take the average value. Adaboost will execute with a decision tree with depth = 1. The fundamental parameter n_estimators is used to select the maximum number of trees involved in the classification process. After each classification, the algorithm re-calculates weight for each sample that has been misclassified, and after many calculations, we will get the most accurate model. A higher number of trees that participate in the classification will increase the accuracy of the AdaBoost algorithm. We see in Figure 4, Accuracy is higher when the parameter n_estimators = 100, then the line goes horizontally with accuracy equal to 98.13% on average. Thus, if we further increase in n_estimators can reach a certain threshold or when n_estimators -> ∞, accuracy will converge at near 1. However, the disadvantage of increasing n_estimator too much is not determining the threshold when the algorithm converges.

Figure 5 illustrates the timing processing of AdaBoost with varying n_estimators. In addition, we see that increasing n_estimators will be resource-intensive for training. In this paper, we choose n_estimators = 100.

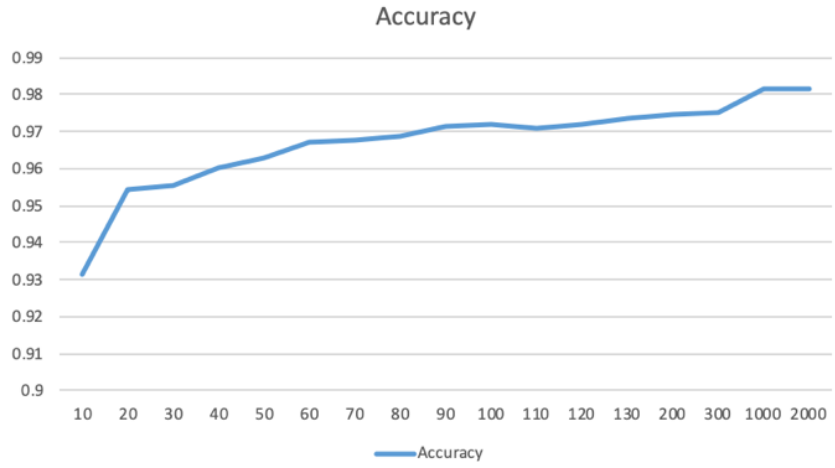

Figure 4. Accuracy level of different n_estimators of AdaBoost model

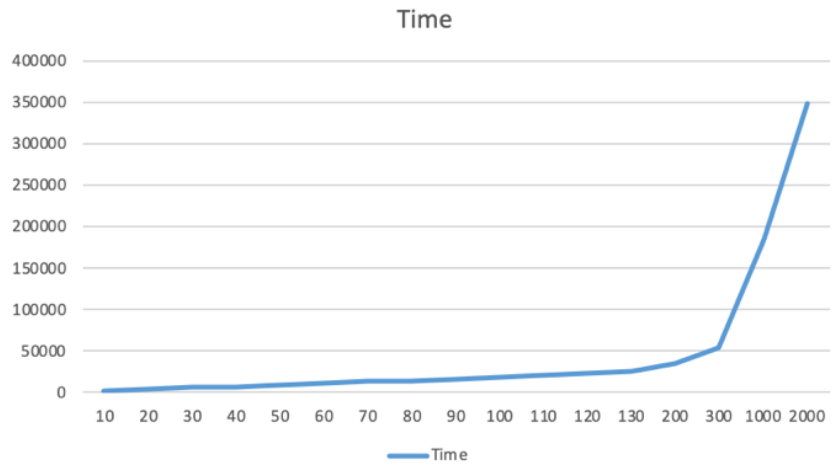

Figure 5. Time implementation estimation of AdaBoost model

4.3. Evaluation on Individual Random Forest Model

For RandomForest, the *n_estimator's* parameter is the number of trees participating in the classification process. For example, if a sample would like to be classified then it will receive votes from other trees. If receiving the largest number of tree votes in a class, the sample will be assigned to this class. In Figure 6, we can see that the RF model obtains peak accuracy when *n_estimator's* equals 100.

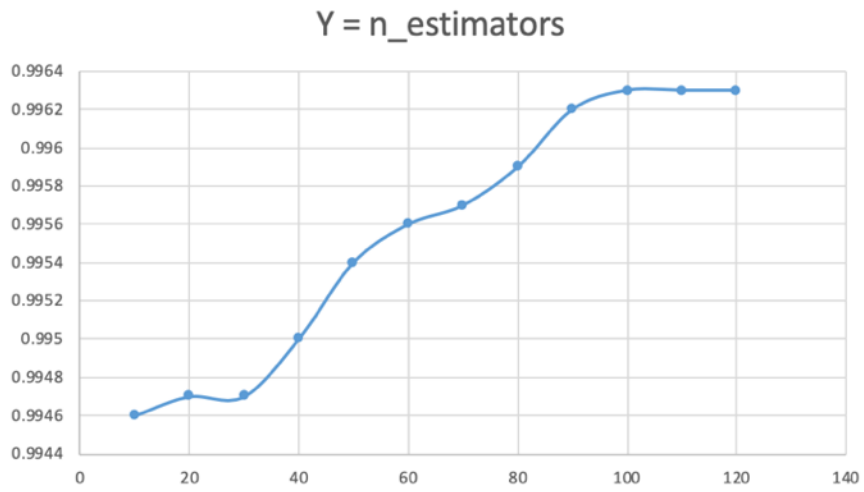

Figure 6. Accuracy level of different *n_estimators* of Random Forest

4.4. Performance Comparisons of Accuracy, Precision, F1-Score and Recall

In Figure 7, the general accuracy obtained from our proposed Stacking model is higher than three individual machine learning algorithms. Accuracy of the proposed Stacking model is 0,6% higher than KNN, 2,5% for AdaBoost, and 0.03% for RandomForest respectively. In Figure 8, we can see that Precision is higher than KNN by 0,5%, AdaBoost by 2,4%, and Random Forest by 0,02% respectively.

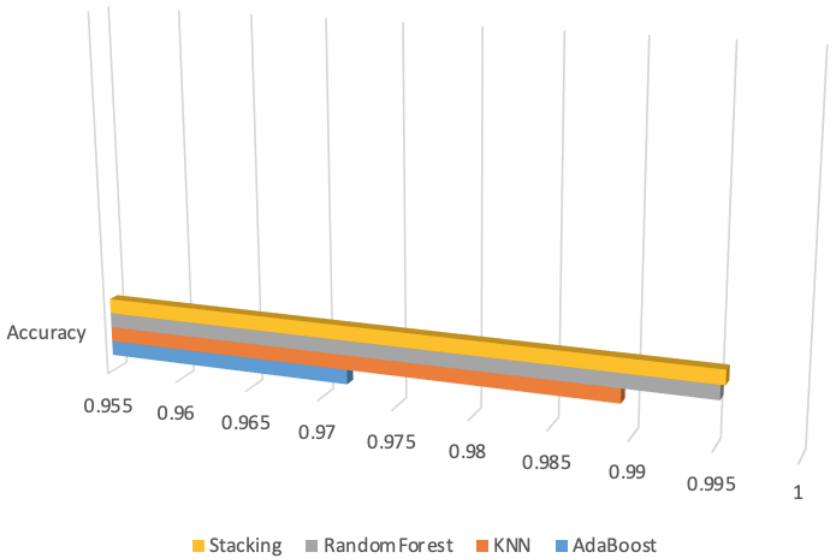

Figure 7. Accuracy level of individual algorithm and Stacking model

In Figure 9, the Recall of our proposed Stacking model is higher than KNN by 0.6%, AdaBoost by 2,2%, and Random Forest by 0.02% respectively. In Figure 10, F1 of our proposed Stacking model is higher than KNN 0,6%, compared to AdaBoost is 2,34% and with Random Forest is 0.03% respectively. We can see that with the NSL-KDD 2019 data set, for three individual ML models, RF gives the most accurate prediction results with the highest Precision. RF algorithms predict the best label compared to the remaining algorithms, and the higher Recall shows that

Random Forest is also less likely to miss labels than the others. The reason might be when choosing the input parameter of the algorithm without considering the max depth parameter, Decision Tree constituting Random Forest will have the depth equal to the depth when browsing the tree with the ID3 algorithm [52] when each Decision Tree is reached. This reason makes Random Forest able to learn more deeply than others. With AdaBoost, we do not set the base_estimator parameter results in the algorithm using a training data set with Decision Tree with max_dept equal to 1. The base estimator from which the boosted ensemble is built that makes the algorithm much faster than Random Forest and provides lower accuracy but in the acceptable range compared to others. Finally, our proposed Stacking model provides significant improvements in Accuracy, Precision, Recall, F1-score which can be applied in practice with a very good performance. The increase in Accuracy indicates that our proposed model is more accurate than other individual models. Higher Recall shows that our model is less likely to miss labels.

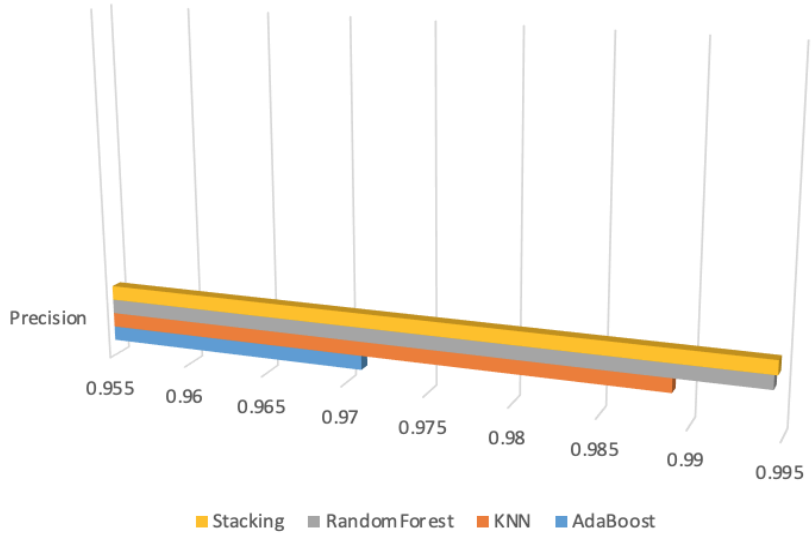

Figure 8. Precision of individual algorithm and Stacking model

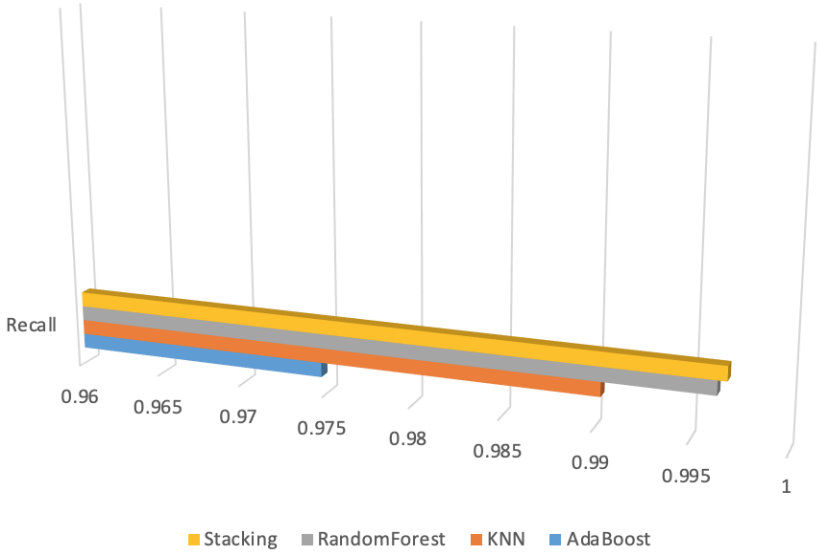

Figure 9. Recall of individual algorithm and Stacking model

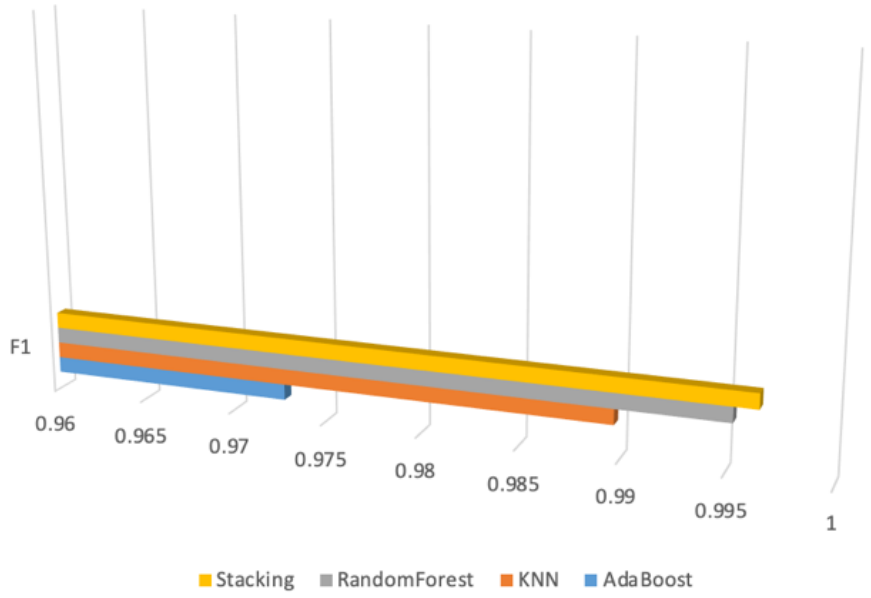

Figure 10. F1-score of individual algorithm and Stacking model

5. CONCLUSIONS

In this paper, we have proposed a Stacking model to combine three machine learning algorithms which are KNN, AdaBoost, and Random Forest for anomaly detection. Moreover, we proposed a Logistic algorithm to automatically select parameters for the stacking model. This model achieves a very high accuracy of 99.64% and significant improvement of Recall, Precision, and F1-score compared to individual machine learning models. A drawback of our proposed Stacking model is that it is more time-consuming than individual machine learning models. However, we consider processing in parallel individual machine learning models on individual dynamic Virtual Machines (VM) by using Apache Hadoop or Spark which we would like to extend in another work. Finally, we hope to integrate deep learning algorithms to increase the accuracy of the overall model and combine time-series training for online, smart, and early detection.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

ACKNOWLEDGMENTS

This research is funded by Hanoi University of Science and Technology (HUST) under a grant entitled “A Network Anomaly Detection Model Based-on Several Machine Learning Models”.

REFERENCES

[1] R, Karthikeyan & Indra, A.. (2010). Intrusion Detection Tools and Techniques –A Survey. International Journal of Computer Theory and Engineering. 2. 901-906. 10.7763/IJCTE.2010.V2.260.

[2] Claude Turner, Rolston Jeremiah, Dwight Richards, Anthony Joseph, A Rule Status Monitoring Algorithm for Rule-Based Intrusion Detection and Prevention Systems, Procedia Computer Science, Volume 95, 2016.

[3] Aditya Chellam, Ramanathan L, Ramani S, Intrusion Detection in Computer Networks using Lazy Learning Algorithm, Procedia Computer Science, Volume 132, 2018, Pages 928-936, ISSN 1877-0509.
[4] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour and H. Janicke, "A Novel Hierarchical Intrusion Detection System Based on Decision Tree and Rules-Based Models," 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), Santorini Island, Greece, 2019, pp. 228-233, doi: 10.1109/DCOSS.2019.00059.
[5] Claude Turner, Rolston Jeremiah, Dwight Richards, Anthony Joseph, A Rule Status Monitoring Algorithm for Rule-Based Intrusion Detection and Prevention Systems, Procedia Computer Science, Volume 95, 2016.
[6] Butun I, Morgera SD, Sankar R (2014) A survey of intrusion detection systems in wireless sensor networks. IEEE Communications Surveys & Tutorials 16(1):266–282.
[7] Xiao, L., Chen, Y. and Chang, C.K. (2014) Bayesian Model Averaging of Bayesian Network Classifiers for Intrusion Detection. Proceedings of the 2014 IEEE 38th Annual International Computers, Software and Applications Conference Workshops, Vasteras, 2014, 128-133.
[8] Panja, B., Ogunyanwo, O. and Meharia, P. (2014) Training of Intelligent Intrusion Detection System using Neuro Fuzzy. Proceedings of 2014 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Las Vegas, 2014.
[9] Vladimir Zwass " Expert system ".
[10] Fuchsberger, A. (2005) Intrusion Detection System and Intrusion Prevention Systems. Information Security Technical Report, 34, 134-139.
[11] Larraga, P., Karshenas, H. and Bielza, C. (2013) A Review on Evolutionary Algorithms in Bayesian Network Learning and Inference Tasks. Information Sciences, 233, 109-125.
[12] I. Seraphim, S. Palit, K. Srivastava and E. Poovammal, "A Survey on Machine Learning Techniques in Network Intrusion Detection System," 2018 4th International Conference on Computing Communication and Automation (ICCCA), Greater Noida, India, 2018, pp. 1-5, doi: 10.1109/CCAA.2018.8777596.
[13] Abdulla Amin Aburomman, Mamun BinIbne Reaz " A novel SVM-kNN-PSO ensemble method for intrusion detection system " - 2015.
[14] Iwan Syarif , Ed Zaluska , Adam Prugel-Bennett , Gary Wills, " Application of Bagging, Boosting and Stacking to Intrusion Detection " - 2012.
[15] Shivang Agarwal, Ravindranath Chowdary, " A-Stacking and A-Bagging: Adaptive versions of ensemble learning algorithms for spoof fingerprint detection" - 2020.
[16] Deeman Yousif Mahmood, " Classification Trees with Logistic Regression Functions for Network Based Intrusion Detection System " - 2017.
[17] H. BENADDI, K. IBRAHIMI and A. BENSLIMANE, "Improving the Intrusion Detection System for NSL-KDD Data set based on PCA-Fuzzy Clustering-KNN," 2018 6th International Conference on Wireless Networks and Mobile Communications (WINCOM), Marrakesh, Morocco, 2018, pp. 1-6, doi: 10.1109/WINCOM.2018.8629718.
[18] W. Hu, W. Hu and S. Maybank, "AdaBoost-Based Algorithm for Network Intrusion Detection," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 38, no. 2, pp. 577-583, April 2008, doi: 10.1109/TSMCB.2007.914695.
[19] Nabila Farnaaz, M.A. Jabbar, "Random Forest Modeling for Network Intrusion Detection System", Procedia Computer Science, Volume 89, 2016, Pages 213-217, ISSN 1877-0509..
[20] Sharafaldin, Iman & Habibi Lashkari, Arash & Ghorbani, Ali. (2018). Toward Generating a New Intrusion Detection Data set and Intrusion Traffic Characterization. 108-116. 10.5220/0006639801080116.
[21] Ahmim, Ahmed & Maglaras, Leandros & Ferrag, Mohamed Amine & Derdour, Makhlouf & Janicke, Helge. (2019). A Novel Hierarchical Intrusion Detection System Based on Decision Tree and Rules-Based Models. 10.1109/DCOSS.2019.00059.
[22] Aksu, Doğukan & Ustebay, Serpil & Aydin, M.Ali & Atmaca, Tulin. (2018). Intrusion Detection with Comparative Analysis of Supervised Learning Techniques and Fisher Score Feature Selection Algorithm. 10.1007/978-3-030-00840-6_16.
[23] Negandhi, Prashil & Trivedi, Yash & Mangrulkar, Ramchandra. (2019). Intrusion Detection System Using Random Forest on the NSL-KDD Data set. 10.1007/978-981-13-6001-5_43.

[24] NSL-KDD data set for network-based intrusion detection systemsAvailable on: http://nsl.cs.unb.ca/KDD/NSLKDD.html, March 2009.
[25] Ring, Markus & Wunderlich, Sarah & Scheuring, Deniz & Landes, Dieter & Hotho, Andreas. (2019). A Survey of Network-based Intrusion Detection Data Sets.
[26] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, et al., Evaluating Intrusion Detection Systems : The 1998 DARPA Offline Intrusion Detection Evaluation, in: DARPA Information Survivability Conference and Exposition (DISCEX), Vol. 2, IEEE, 2000, pp. 12–26.
[27] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, K. Das, The 1999 DARPA Off-Line Intrusion Detection Evaluation, Computer Networks 34 (4) (2000) 579–595.
[28] S. Stolfo, (Date last accessed 22-June-2018). [link] URL http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
[29] M. Tavallaee, E. Bagheri, W. Lu, A. A. Ghorbani, A detailed analysis of the KDD CUP 99 data set, in: IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6.
[30] Tavallaee, Mahbod et al. “A detailed analysis of the KDD CUP 99 data set.” 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (2009): 1-6.
[31] Panigrahi, Ranjit & Borah, Samarjeet. (2018). A detailed analysis of CICIDS2017 data set for designing Intrusion Detection Systems. 7. 479-482.
[32] Altman, Naomi S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression" (PDF). The American Statistician. 46 (3): 175–185. doi:10.1080/00031305.1992.10475879. hdl:1813/31637.
[33] Wang, Lishan. (2019). Research and Implementation of Machine Learning Classifier Based on KNN. IOP Conference Series: Materials Science and Engineering. 677. 052038. 10.1088/1757-899X/677/5/052038..
[34] Mehrnaz Mazini, Babak Shirazi, Iraj Mahdavi; "Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms", Journal of King Saud University - Computer and Information Sciences, Volume 31, Issue 4, 2019, Pages 541-553, ISSN 1319-1578.
[35] Li, Yang & Fang, Binxing & Guo, Li & Chen, You. (2007). Network anomaly detection based on TCM-KNN algorithm. 13-19. 10.1145/1229285.1229292.
[36] M.-Y. Su, "Using clustering to improve the KNN-based classifiers for online anomaly network traffic identification", J. Netw. Comput. Appl., vol. 34, no. 2, pp. 722-730, 2011.
[37] I. Syarif, A. Prugel-Bennett and G. Wills, "Unsupervised clustering approach for network anomaly detection", Proc. 4th Int. Conf. Netw. Digit. Technol. (NDT), pp. 135-145, Apr. 2012.
[38] Y. Freund and R. E. Schapire, "A short introduction to boosting", J. Jpn. Soc. Artif. Intell., vol. 14, no. 5, pp. 771-780, Sep. 1999.
[39] Zhang, Chunlin & Jiang, Ju & Kamel, Mohamed S.. (2005). Intrusion detection using hierarchical neural networks. Pattern Recognition Letters. 26. 779-791. 10.1016/j.patrec.2004.09.045.
[40] Ho, Tin Kam (1995). Random Decision Forests (PDF). Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282. Archived from the original (PDF) on 17 April 2016. Retrieved 5 June 2016.
[41] Ho TK (1998). "The Random Subspace Method for Constructing Decision Forests" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 20 (8): 832–844. doi:10.1109/34.709601.
[42] Biau, Gérard. (2010). Analysis of a Random Forests Model. Journal of Machine Learning Research. 13.
[43] Paulo Angelo Alves Resende and André Costa Drummond. 2018. A Survey of Random Forest Based Methods for Intrusion Detection Systems. ACM Comput. Surv. 51, 3, Article 48 (July 2018), 36 pages. DOI:https://doi.org/10.1145/3178582.
[44] Efron, B. (1979) "Bootstrap methods: Another look at the jackknife", The Annals of Statistics 7 (1): 1-26.
[45] Peng, Joanne & Lee, Kuk & Ingersoll, Gary. (2002). An Introduction to Logistic Regression Analysis and Reporting. Journal of Educational Research - J EDUC RES. 96. 3-14. 10.1080/00220670209598786.
[46] Francisco Sales de Lima Filho, Frederico A. F. Silveira, Agostinho de Medeiros Brito Junior, Genoveva Vargas-Solar, Luiz F. Silveira, "Smart Detection: An Online Approach for DoS/DDoS Attack," Security and Communication Networks, vol. 2019.

[47] Barlas, Panagiotis & Lanning, Ivor & Heavey, Cathal. (2015). A Survey of Open Source Data Science Tools. International Journal of Intelligent Computing and Cybernetics.
[48] J. Davis and M. Goadrich, "The Relationship Between Precision-Recall and ROC Curves", In ICML'06, 2006.
[49] Powers, David M W (2011). "Evaluation: From Precision, Recall and F-Score to ROC, Informedness, Markedness & Correlation". Journal of Machine Learning Technologies. 2 (1): 37–63. hdl:2328/27165.
[50] Bhattacharyya, Dhruba K & Kalita, Jugal. (2013). Network Anomaly Detection: A Machine Learning Perspective.
[51] Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81–106.
[52] J. Song, X. Lu and X. Wu, "An Improved AdaBoost Algorithm for Unbalanced Classification Data," 2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery, Tianjin, 2009, pp. 109-113, doi: 10.1109/FSKD.2009.608.

AUTHORS

Tran Hoang Hai received his B.S degree from Hanoi University of Science and Technology in Vietnam and an M.S degree in Computer Engineering from Kyunghee University, South Korea in 2008. Since then, he has worked at INRIA joint Alcatel-Lucent Bell Laboratory and got his Ph.D. degree in computer science from the University of Rennes 1 (France) in 2012. His interesting research areas are network security, routing, and resource allocation mechanisms in the next-generation Internet, and applied game theory to the communication network. He has published several papers on those issues. He is currently Assistant Professor at the Department of Data Communication & Computer Networks, School of Information & Communication Technology, Hanoi University of Science and Technology, Vietnam.

Le Huy Hoang received his B.S degree in Information Security from Hanoi University of Science and Technology, Vietnam in 2020. His interesting research areas are network security, machine learning, and network intrusion detection system.

Eui-nam Huh earned a B.S. degree from Busan National University in Korea, a master's degree in Computer Science from the University of Texas, the USA in 1995, and a Ph.D. degree from the Ohio University, the USA in 2002. He is the director of the Real-time Mobile Cloud Research Center. He is a chair of the Cloud/BigData Special Technical Committee for the Telecommunications Technology Association (TTA), and a Korean national standards body of ITUT SG13 and ISO/IEC SC38. He was also an Assistant Professor at Sahmyook University and Seoul Women's University, South Korea. He is now a Professor in the Department of Computer Science and Engineering, Kyung Hee University, South Korea. His research interests include cloud computing, screen contents coding (cloud streaming), Internet of Things, distributed real-time systems, security, and big data.