HIERARCHICAL SVM CLASSIFIER FOR INTRUSION DETECTION SYSTEM

Warhamni Jani @ Mokhtar[1] and Azizi Abdullah[2]

Faculty of information Science and Technology, The National University of Malaysia

[1]*warhamni@gmail.com,* [2]*azizia@ukm.edu.my*

Abstract: The development of intrusion detection system (IDS) model to classify various types of attacks was proposed in this study using Support Vector Machine (SVM) with binary tree hierarchical arrangement. This IDS model was developed to classify NSL-KDD dataset into one of the five main classes: Normal, Probe, DoS, U2R and R2L. The main performance measure for IDS is the ability to classify multiple attack classes and achieving low false alarm rate. One-versus-One or (OVO) method is a popular method used in solving multi-class problems. However, the main problem of multiple classes using OVO techniques is the level of ambiguity, which receive the same number of votes as well as the detection was done barely in surface level without considering other conditions. This hierarchical method tests at each level using different SVM classifiers. The SVM, originally a binary classifier, was developed for multi-class classifier. Pre-processing methods involve are mapping attributes, preparing data in SVM format, normalization and feature selection. The SVM hierarchical method is proposed based on the highest value of the test model between the OVO binary class method, one-to-all (OVA) binary class as well as multiple OVO classes. The order of the model is determined by the priority of order from higher to lower according to the level of detection obtained from all experiments. At each level of the hierarchy, one class is removed and classification is resumed for the remaining classes using the next classifier. By comparison it was proven that using multi-class hierarchical model was able to provide average accuracy up to 90.98% compared with 41.01% for standard multi-class model (OVO). The number of false alarm rate also decreased from 5.77 for standard multi-class model compared to 0.5 for hierarchical multi-class model.

***Keywords*:** Support Vector Machine, Intrusion Detection System, Hierarchy classifier, NSL-KDD.

1.0 INTRODUCTION

Intrusion Detection System (IDS) is an important tool used with other components in network defense. The significance of IDS is to identify threats in the network whether the risks are coming from external or internal threats. According to Veal (2005), the activities observed by IDS include suspicious activities such as unauthorized access attempts, manipulations and disruptions of computer system capabilities performed by intruders such as viruses, worms, probes, attacks and abuse of systems program (Nguyen et al. 2012). Lee and Stolfo (2000) also emphasize that IDS needs to be the exact system, able to adapt and expand its use to systematically and automatically regulate the network. According to Panetta (2017) network security experts agree that the main focus should change from avoiding threats to detection and response.

IDSs are built using signature-based or anomaly-based detection (Lee and Stolfo 2000; Alma 2012; Cheng & Syu 2015). Signature-based detection identifies through the attack pattern found in IDS database while anomaly-based detection is based on profile of attack built. Any match will enable alerts by IDS and appropriate actions will be performed based on the settings. The signature-based IDS model works just like a virus detection program that identifies suspicious activity based on a match in the database. The signature-based IDS model able to identify the attack accurately and promptly but has difficulty on detecting activity other than the existing match which eventually increase the false positive rate. Meanwhile, an anomaly-based IDS model has the advantage of identifying an unknown attack but having difficulty building a suitable model for legitimate activity due to an increase in the level of false warnings especially from unique legitimate activities.

There are several methods used in the development of IDS. One of the method was based on network host information study such as the time user's access the system and what resources user retrieved. A simple statistical method can be performed to check user activity type whether it matches the model in the database. The disadvantages of this method are human activities are uniques and significantly changes. The focus was changed from the user type to the set of user's behavior. While IDS based on network information is more focused on packets sent in the network rather than a set of human behaviors. The information sent is more concise and involves between host connections and servers for example network flow such as the number of packets sent, number of bits being exchanged and so on.

Due to its ability to identify an unprecedented attack, an anomaly-based IDS model becomes the choice of researchers. In 1980, James P. Anderson was studying ways to improve computer security and monitoring at user locations (Bruneau 2001). His research uses account audit file to detect unauthorized access. He then suggested that a model be constructed from the user's normal behavioral statistics so that 'impersonators' with different

behavior from the normal profile can be detected. His study has pioneered the initial steps of the construction of intrusion detection and developed the original idea of anomaly detection. The development then booming with combination of various techniques such as statistics including Bayesian analysis, as well as data mining (Lee & Stolfo 2000). Lee & Stolfo (2000) builds the IDS framework using data mining algorithms to calculate activity patterns from audit data system and extract predicted features based on the pattern. To enhance the IDS learning capabilities, machine learning (ML) techniques are also used to transfer the role of detection from human to the system.

In order to get good results, this study focuses on multi-class classification methods for network data that has multiple simultaneous attacks at a time. A quality IDS will be able to detect various types of attacks and not only focusing on common and popular attack types. Plus, if the attack type is small in quantities, a reliable IDS should be able to detect these types of attacks like U2R, a dangerous attacks. There are two standard methods of multi-class testing which is One-versus-One (OVO) and One-versus-All (OVA). This two main methods in classifying class are (a) taking into account all the data in an optimization like OVA or (b) constructing multiple binary classifiers like OVO (Vural & Dy 2004). The main issue of multi-class data is that uneven data numbers can affect detection accuracy as larger amounts of data dominate the final decision. Given that the problem is still persistent, this study aims to get better detection rate methods for multiple attack classes.

In order to obtain a system that has zero dependency on humans, IDS anomaly gives many advantages over signature (Singh & Nene 2013). This study focuses on anomaly-based IDS. The detection will consider all attack classes in NSL-KDD data set. There are many methods that have been proposed for IDS anomaly but according to Horng et al. (2011) the decision tree has been proven to have good performance. Next, SVM is an effective ML method and is capable of giving accurate results compared to other methods (Li et al., 2011). SVM is also easier to use than neural networks (Hsu et al. 2010).

Although the OVA method is a popular method, it experiences some heuristic problems (Bishop 2006). First, the value of trust gained can vary between binary classifier. Secondly, even if division between classes is balanced for training data, binary classification learning is still able to see disproportionate divisions because the negative set is usually greater than the positive set. The OVA method also has a weakness especially for data that has small amount in numbers like U2R. Large amounts of data will usually dominate the decision. Meanwhile, through the OVO method, many classes will take time as many cross-tests are needed before the exact model is obtained. In addition, the detection is done particularly on the surface only and no involvement of any other condition to consider. The OVO method also faces ambiguity problems where some input space areas are likely to receive the same vote (Bishop 2006). The same number of votes will make the decision making more difficult.

Next, this study focuses on addressing the problem of OVO standard classification through the construction of multi-class hierarchical classification. The combination of binaries and SVM in each level of hierarchy is guided by the strategy to get the decision by filtering one by one class at each level. Filters in each level helps classification produces accurate results. Each attack class will be tested with an SVM classifier according to the hierarchical multi-class with order of priority.

2.0 Machine Learning Method Application In Classification

There are various ML techniques applied in building of IDS models such as Support Vector Machine (SVM), Random Forest and Bat Algorithm (BA). Enache and Sgarciu (2015) propose an anomaly-based IDS model that has a pre-processing phase for feature selection using information obtained while for detection using SVM classifier. This study used the advantages of Swarm Intelligence (SI) algorithm, the Bat (BA) algorithm. The model constructed then tested on NSL-KDD data sets of 9566 records and is divided into two files namely training and testing. Better results were obtained by comparing with other methods with 99.15% detection with a false warning rate of 0.019. This study also states that the SVM algorithm deficit is dependent on the correct parameter input from the user.

Hence Hasan et al. (2014) builds two types of classification models that are first based on SVM and the second based on Random Forests (RF). Experimental test results show that both models are effective. SVM gives more precise classification than RF but are time consuming. While RF is capable of delivering similar results to SVM, it is actually will be much faster if the parameters of the model are supplied. The set data used is KDD'99 that has been cleared from redundant data so that the classifier does not lean over to the frequent record. RF technique produces many classification points. Each tree is constructed with a different sample of the original data using the classification tree algorithm. After the forest is created, an object to classify will be placed for each tree. Each tree will then cast vote class for the object. The highest voting is the final result. For the SVM model, the RBF kernel is selected and the grid search technique is used to get the best model. The accuracy of the SVM model is 92.99 compared to 91.41 for RF. The time taken by RF is 10.62 minutes compared to 44.14 minutes for SVM.

2.1 Support Vector Machine

For the purpose of this study, SVM was selected as machine learning (ML) techniques. The ML method using Support Vector Machine (SVM) has been selected based on its ability to properly and accurately detect the attack. SVM is a learning algorithm derived from statistical learning theory (Calix & Sankaran 2013; Schwenker 2000). SVM is one of the popular and useful ML methods for data classification (Hsu et al., 2010) developed by Cortes and Vapnik (1995) for use in solving the pattern detection problem besides the nearest neighbor classifier. SVM has been recognized as the State-of-the-Art, a modern and latest tool in classifying applications in pattern recognition (Mohd Rizal Kadis 2016; Azizi Abdullah 2010; Boswell 2002; Cortes & Vapnik 1995) and texts, handwriting recognition and bioinformatical analysis (Pervez & Farid, 2014). This algorithm is used to perform binary classification or classification of two SVM classes, but is easily developed for multi-class classification. MSV is a classification technique that involves data division into two sets of data ie training and testing (Azizi Abdullah 2010). SVM's main idea is to define the optimum separation space of hyperplane as the dividing line separates Class +1 from Class -1 by maximizing the largest margin between the two closest points (Calix & Sankaran 2013; Azizi Abdullah 2010). Hyperplane is built with the boundaries specifying the data entered. The points that are at the boundary are known as the supporting vector and the midline between the margins is the optimal line of hyperplane. Figure 1.1 shows margin position, hyperplane and support vector.

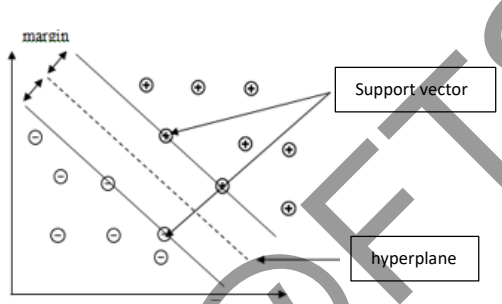


Figure 1.1 Margin, hyperplane and support vector position in SVM

SVM's original concept is to separate the hyperplane between two separate classes in a straight line (linear) where a class is labeled negatively (-1) while the opposite class has positive label (+1). The best hyperplane is by getting the maximum thickness of the margin; the boundary distance between the two classes. Schwenker (2000) states that the greater the margin, the higher the ability to generalize for the separation of hyperplane.

2.2 Parameter Optimization

The parameter optimization process is able to improve classification performance. There are two methods used to get parameter optimization that is grid-search and cross-validation verification. There are two parameters needed for the RBF kernel; C and γ values. The value of both is unknown before the test runs however there is a way to get the best value for both of these parameters. The use of grid-search to get the value C and γ using cross-validation techniques is recommended. In the *k*-cycle cross-validation technique, the training data is divided into the same subset of *k* (Hsu et al. 2010). Subsequently, a subset is tested using a classifier which has been tested on the remaining k-1 subset. Therefore, the prediction for each data in the entire training data is performed and the percentage of cross-validation is accurately categorized. The cross-validation process avoids overfitting problems, which is a model error that occurs when a model attempts to make an accurate guess at a limited set of data points.It is important to tuning parameter C to ensure the best step in the SVM that minimizes the structure's risk. Grid-search search is a traditional method of determining parameter optimizations that perform individual searches to completion by subset of the predefined parameters of the chosen learning algorithm. For SVM classifier using the RBF kernel there are two main parameters that need tuning in order to produce good performance for unknown data C and γ parameters. Grid-search then trains MSV with matching C and γ so it achieves the best classification performance.

Cross-validation is used to obtain the expected performance of a model's generalization by selecting the best parameter. Among the main purposes of cross-validation are (a) as a testing technique that will give results that are not favorable to any expectation of generalization which may result in overfitting. Next, it is also (b) a step to choose the appropriate model. The parameters obtained (the best value of C and γ) will be reused for the training data model. Next, the model obtained will be used on test data. In cross-validation, the data set is divided into random *k*-fold numbers by the same amount. If the value of k = 10. Training data is randomly broken into 10 subsets. A subset is set as a test set while the remaining nine subsets are considered as training data. The cross-validation process is repeated ten times and the accuracy of the classification is measured by the average of the test results (Li et al., 2012). For LIBSVM, there is a grid.py program that performs a grid-search best training parameters for the set of vector feature supplied. The program also uses cross-validation techniques to anticipate the accuracy of any combination of parameters on a certain scale and thus help select the best parameter.

2.3 Classification using SVM

For classification using SVM, there are two types of classification namely binary and multi-class classification.

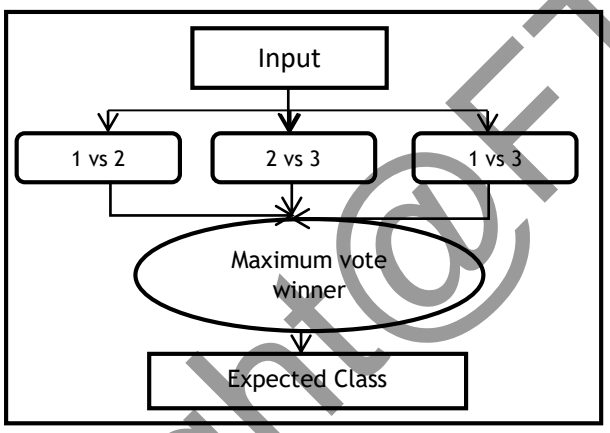
a. Binary classification

This method is used when there are only two classes for the data. Classifier attempts to classify unknown data to two groups. However, this binary classification can be expanded to multiple classes for example one-versus-one or one-versus-all test if there are more than two classes exists in the dataset (Azizi Abdullah 2010).

b. Multi-class classification

If there are multiple classes in a dataset, it is important to classify the *N* Class data to its correct class. There are four methods identified for this classification as per below:

i. **One-versus-One (OVO)** –For this approach, it uses maximum votes and each one is differentiate with two types of class (Azizi Abdullah 2010). The number of classes is calculated based on formula *N* (*N*-1) / 2 class model. For example, if N = 5, the number of class models is 10. Each model is trained with +1 for the actual class and -1 for the otherwise. The data set is tested on each model and the most voted class is considered a winner. The difference with OVA model is that more models need to be built and the performance measure is via maximum draws by considereing the results from all models. However the amount of records selected only for the class involved and does not require all classes for each binary test. According to Li et al. (2008) OVO provides better performance if accurate classification is produced. The disadvantage of this method is when the number of classes is too large. For example, if N = 20 then the binary class number to be trained is N (N-1) / 2 = 190. The following Figure 1.2 shows the OVO concept for multi-class.



Sumber: Gu et al. (2016)

ii. **One-versus-All (OVA)** – in contrast to the OVO approach, this method uses the "winner-takes-all" strategy (Azizi Abdullah 2010). This means that, if N = 5, the number of class models is five, which is a model for each class (Li et al. 2008). Each model will be tested with the test data set and the class that gives the highest classification result is considered the winner. The OVA method takes long training times and often the rate of accuracy it produces is lower than OVO. Figure 1.3 gives an overview of one-versus-all concept.

The pseudocod of the learning algorithm for OVA constructed from binary L classification is as follows:

Input:

- *L*, is a learner (binary classifier learning algorithm)
- sample *X*
- label y where $y_i \in \{1, \ldots K\}$ is the label for the sample X_i

Output:

- list f_k classifier for $k \in \{1, \ldots, K\}$

Procedure:

- For every k in {1, ..., K}
- Build a new vector label, z where $z_i = 1$ if $y_i = k$ and $z_i = 0$ or
- Use *L* to *X*, z to get f_k

Decide to match all the classifier to a new sample X and predicts for the label k for which each classifier states the highest value of confidence:

$$\hat{y} = \underset{k \in \{1 \ldots K\}}{\operatorname{argmax}} f_k(x)$$

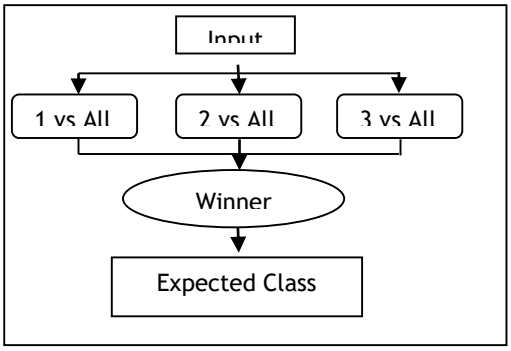


Figure 1.3 One-versus-all concept
Source: Gu et al. (2016)

iii. **Hierarchy or binary tree SVM** - is a different method for solving N-Class problems is by constructing a hierarchy or binary classification tree (Schwenker 2000). Using this method, multi-class classification problems are divided into a series of SVM binary classifier classes arranged in a hierarchy. The arrangement method is root node at the top while the terminal node (leaf) is at the bottom. Each class is presented using the leaves and each node is categorized using binary classification. Li et al. (2008) states the constructed hierarchy must be correctly designed before classification training is carried out. Figure 1.4 shows the general method of hierarchical classification.

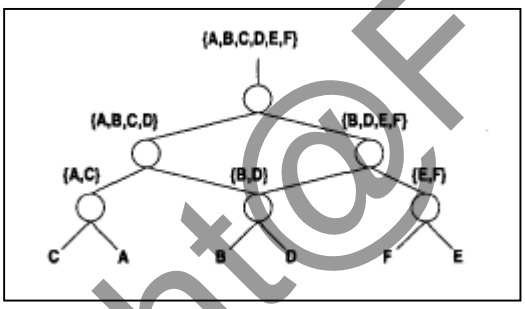


Figure 1.4 General method of Hierarchy classfication
Source: Schwenker (2000)

iv. **Directed acyclic graph SVM (DAGSVM) or open graph without SVM cycle :**- is a hierarchical binary architecture in which DAG is used to combine the value obtained from different one-versus-one classifier introduced by Platt et. al (2000). For *N* class problems, a number of N (N-1) / 2 binary classifier are trained. DAGSVM relies on DAG binary root to make a decision. When the test sample is approaching the leaf node, the final result as shown in Figure 1.5. The binary testing depends on the number of nodes contained in the decision path. According to Wang and Casasent (2006), at each node, one class is excluded from the list.

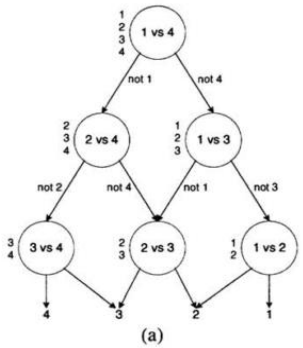


Figure 1.5 DAG make decision for 4 Class where binary classifier (SVM) is used in each node
Source:Platt et al. (2000)

3.0 METHODOLOGY

In this study, there are four main activities conducted. Figure 1.6 below shows the activity.

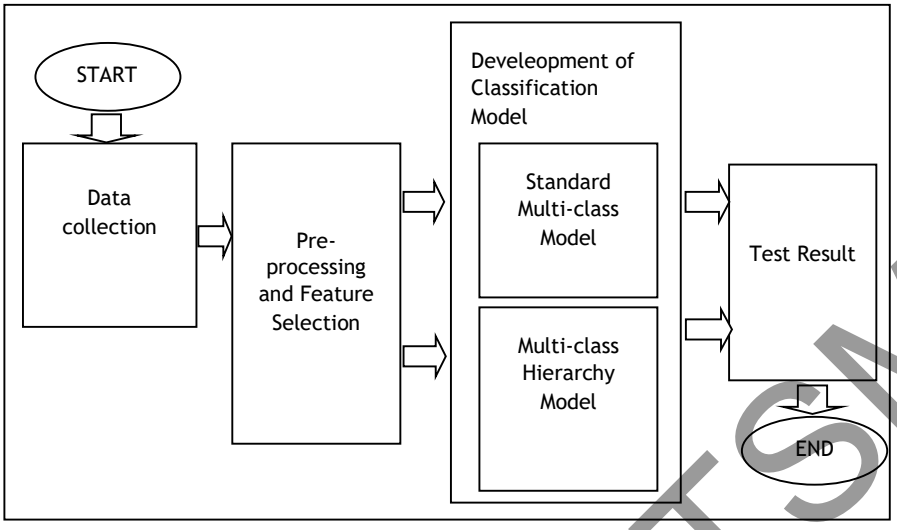


Figure **Error! No text of specified style in document.**.6 Summary of Methodology

a. Step 1: Data Collection

The data to be tested is NSL-KDD dataset. This data is used to detect intrusions and is provided from earlier certified data ie KDD Cup '99 (Chen & Syu 2015). The NSL-KDD data has been upgraded from the original data where some improvements like redundant data cleaning. This data contains 41 features and 1 label. The data structure and features are similar to KDD Cup 1999 datasets. There are 5 main data classes which is 1 normal and the remaining 4 are attack data.

b. Step 2: Pre-processing and Feature Selection

The data will then undergo the initial process of data preparation to the appropriate format. An attribute mapping process will be performed to convert the data in alphabetical form into numeric form. Next, convert the data into SVM format and normalization will be implemented in this step. For the purpose of getting prominent features, the data will go through SVM test to get the number of conforming features in the next experiment. Three feature sets, 13 (network features), 15 (host features) and 41 (overall) features are provided for this test. Next, the number of features with the highest accuracy results will be used.

c. Step 3: Development of Classification Model

There were several experiments carried out for the development of classification model. Experiments were done through the LIBSVM program using the RBF kernel. Standard multi-class models OVO and OVA will be developed and tested through multiple experiments to get the order of priority for attack class. Detailed description of the process carried out in the experiment will be described in the next step. Based on priority order, a multi-class hierarchical classification model is built. Construction of multi-class hierarchical models will be carried out after experiments using LIBSVM compared to the methods used by Horng et al. (2011) which is using a hierarchical algorithm before testing with MSV.

d. Step 4: Test Result

Hence, comparative tests between standard class classifier (OVO only) and multi-class hierarchical will be implemented to identify which methods give accurate detection and lower false alarm levels. Classification of multi-class hierarchical conducted is tested to obtain the conclusion whether the technique affects the level of accuracy in order to improve detection performance. Calculation and comparison processes are provided for both models. Conclusions were made to summarize the findings obtained during the study.

3.1 Performance Metrics

To measure the performance level of the model developed in this study, performance measures need to be used. The main reason for the use of measurements is to obtain standard results and able to make comparisons of learning algorithms developed with methods used by other researchers (Azizi Abdullah 2010). For the purposes of this study, the model performance is tested through the accuracy (K) level, the detection rate (P) and the false warning (AP) achieved (Mohd Rizal Kardis 2016, Parsaei et al 2016). The developed models are formed using the appropriate probabilities to ensure all factors are taken into account. Models that provide high detection results are considered to be a better model than others. However, the value of AP needs to be lower before the model is considered good and appropriate.

Table 1.1 shows the confusion matrix as pillar to calculation construction to obtain K, P level and AP. The performance of the model is presented visually through a confusion matrix. The confusion matrix is the square matrix and the number shown in the diagonal is the exact classification number and otherwise the wrong classification. The confusion matrix read is through columns and lines, for each column is the expectation while the line represents the actual category of data. Through it, according to Azizi Abdullah (2010), one of the benefits of using a confusion matrix is it is easy to see which class is accurately detected and vice versa by classifier.

Table 1.1 Confusion Matrix

Category		Expectation	
		Normal	**Attack**
Actual	Normal	TP	FP
	Attack	FN	TN

For the purposes of abbreviations in Table 1.1 are as follows:
- True Positive (TP) is the original value is true and successfully detected as true
- True Negative (TN) is the original value is false and successfully detected as incorrect
- False Positive (FP) the original value is true but detected as false
- False Negative (FN) the original value is flase but detected as true.

$$\text{Accuracy Level (K)} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.1)$$

$$\text{Detection Level (P)} = \frac{TP}{TP + FP} \quad (3.2)$$

$$\text{False Alarm (AP)} = \frac{FP}{FP + TN} \quad (3.3)$$

Based on the precision level eq*uation (K)*, derived based on the number of correct detection for each class and divided by the amount of data. Meanwhile, the detection level is obtained through the accurate number of detectable TP which is divided by the total detection for the attack that is the amount of FP and TP. And lastly, False Alarm (APs) are obtained through the normal amount of data detected as attacks divided by TN and FP counts.

3.2 Feature Selection

The feature selection is an important step in pre-processing. To ensure that studies are capable of providing a high level of detection, several tests for prominent feature points have been implemented. Experiments conducted take into account the number of features involved. This matchs opinion by Kang and Kim (2016) which state that the performance of the intrusion detection system relies heavily on the number of features selected in the context of accuracy and efficiency. If there are more features are involved so the detection process will take time and vice versa. However, the main objective still focusing on achieving higher detection levels between the number of features selected.

In order to get the most prominent features in this study, there are several methods of selection of features used as discussed. For this purpose, this study applies the findings of the Staudemeyer & Omlin (2014) study which uses the distribution histogram method, the baseline plot and the decision tree to get really powerful features and represent each type of attack category. Through these methods, a feature set that is really important and useful for each type of attack category can be determined. Table 1.2 below lists the relevant features for each type of attack.

Table 1.2 List of relevant feature for each attack category.

Bil	Attack Category	Most relevant feature in data set
1	DoS (Network)	3, 4, 5, 6, 8, 23, 29, 36, 38, 39, 40
2	Probe (Network)	2, 5, 29, 33, 34, 35
3	R2l (Host)	1, 3, 5, 6, 10, 24, 32, 33, 35, 36, 37, 38, 39, 41
4	U2R (Host)	5, 6, 10, 14, 17, 33

In this study, the features are determined by the group of either network groups or host groups by type of attack. DoS and Probe are network categories while R2L and U2R are host category attacks. The features identified in above table are then combined into a class of category; either host or network attacks. In addition, each recurring feature will only be calculated once to facilitate the group identification process such as Table 1.3 below.

Table 1.3 List of relevant feature based on network and host category.

Bil	Network Attack	Host Attack
1	protocol_type (2)	duration (1)
2	service (3)	service (3)
3	flag (4)	src_bytes(5)
4	src_bytes (5)	dst_bytes (6)
5	dst_bytes (6)	hot (10)
6	same_srv_rate (29)	num_file_creations (17)
7	dst_host_srv_count (33)	srv_count (24)
8	dst_host_same_srv_rate (34)	dst_host_count (32)
9	dst_host_diff_srv_rate (35)	dst_host_srv_count (33)
10	dst_host_same_src_port_rate (36)	dst_host_diff_srv_rate (35)
11	dst_host_serror_rate (38)	dst_host_same_src_port_rate (36)
12	dst_host_srv_serror_rate (39)	dst_host_srv_diff_host_rate (37)
13	dst_host_rerror_rate (40)	dst_host_serror_rate (38)
14		dst_host_srv_serror_rate (39)
15		dst_host_srv_serror_rate (41)

Next step, a set of data is provided according to group features, 13 for network attack types and 15 for host attacks and 41 for all features. The data set will go through an experimental process in SVM. The feature set that provides the highest classification value will be selected for the next experiment.

3.3 Multi-class classification using Support Vector Machine Hierarchy Classifier

Classification techniques using hierarchy are used by some researchers, for example Nashat & Abdullah (2010) which provide detailed class hierarchical construction in the study of food color checks using Wilk's λ and SVM analysis. Meanwhile, Xiao and Cheng (2015) use the OVA and OVO methods to develop the SVM hierarchy based on multi-class classification to classify based on bus status. The study uses GPS Guandong's smart data traffic and is processed by PCA as well as the RBF kernel to test data samples. Data is also calculated using Euclidean distance between Class. Hassan and Damper (2010) used the SVM binary classification method extended to multi-class classification to identify speech-based emotions. The study applies two standard classifier, one-versus-one and one-versus-all to develop a hierarchical classification model in which each classification assigns members to each class for three types of public data sets. The dataset used is the popular data set for acted types are EMO-DB, DES and Serbian. All datasets were tested using binary classification methods one- versus -one (OVO), one-versus -all (OVA), Directed Acyclic Graph (DAG) and Unbalanced Decision Tree (UDT).

The model of the study was developed through several experiments conducted to identify the best methods of modeling the entire model and determining the classifier for each level of hierarchy. The data will be tested using OVA and OVA Binary and Multi-class OVO techniques and the best results of the test obtained from the test will determine the priority order of the classifier to construct hierarchical classification. Figure 1.7 below gives a detailed of the determination of the classifier model for each level and hence the completion construction of the hierarchy.

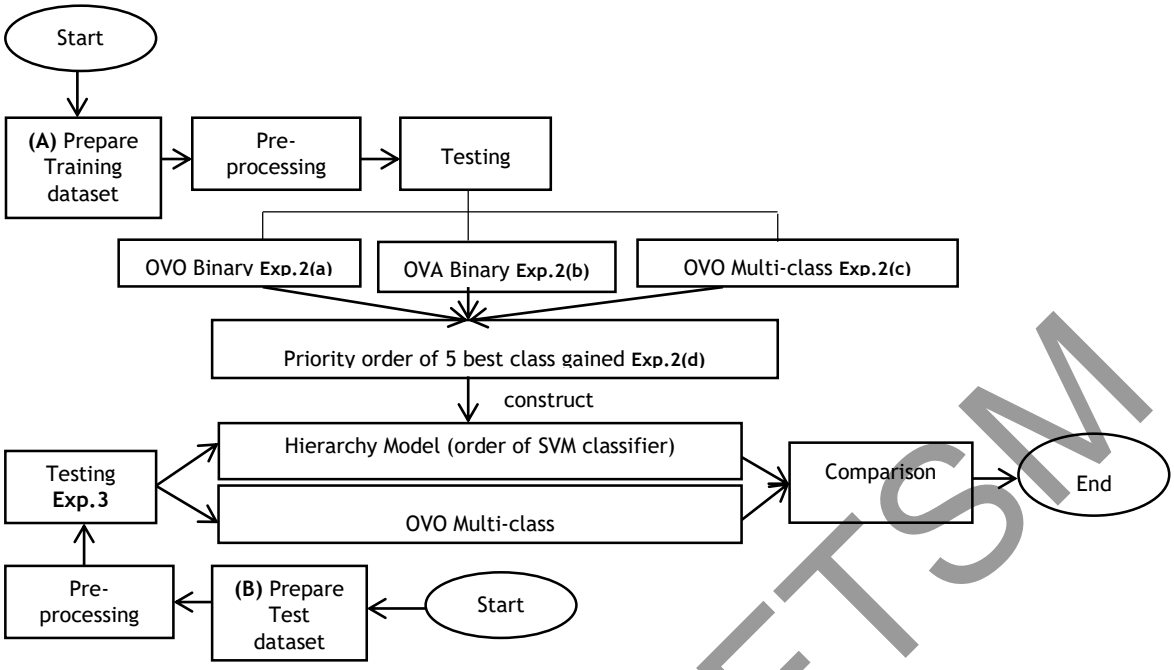


Figure **Error! No text of specified style in document.**.7 Multi-Class Hierarchy Model Flow.

4.0 EXPERIMENT DESIGN AND RESULT

In this study, three experiments were conducted. Experiment I is to get the appropriate feature value for the data. Experimental II provides a clear overview of attack data tested based on one-to-all and one-to-one for binary models and the construction of multi-class hierarchy models. For Experiment III, tests were performed using test data to compare the performance of multi-class OVO models with multi-class hierarchical models.

4.1 Experiment 1

The main objective of the test is to observe the detection level for different feature inputs and get the best number of features that provide high average accuracy. The record used in this experiment is set of training data. The experiment run at this level will use three sets of feature, the first is 13 feature for network attack type, 15 for host attack and 41 for all features. Next, each set of features will be tested repeatedly before and after by using the best parameter that will then produces six classifier. The classification technique used is multi-class OVO using LIBSVM with RBF kernel.

Table 1.4 Percentage accuracy breakdown by type of attack based on feature set

No	Type of attack	Accuracy level (K)		
		13 feature	**15 feature**	**41 feature**
1	Normal	30.77%	31.05%	31.03%
2	U2R	0.23%	0.1%	0.255%
3	R2L	6.02%	6.16%	6.113%
4	DoS	31.10%	31.15%	31.16%
5	Probe	31.10%	31.07%	31.13%
	Average of accuracy %	**99.22%**	**99.53%**	**99.69%**

Table 1.4 lists the breakdown of percentage accuracy obtained from the three feature sets for each type of attack. Feature 13 is a feature of the network attack type and based on observations, the value of Probe and DoS is 31.10% for both class while for the features 15 is 31.15% and 31.07% repectively. While feature 15 is host attack type feature and notice that U2R detection value is less than the use of 13 features that is 0.23% compared to 0.1%. However there was a slight increase in the percentage of detection value obtained for R2L class from 6.02% to 6.16%. Based on Table 1.6, the highest average percentage of detection is by features 41 followed by features 15 and lastly features 13. Therefore, the number of features for the training and test datasets for all subsequent experiments will use 41 features based on this findings.

4.2 Experiment 2

The main objective of the second stage test is to evaluate the performance of the classification model between the standard multi-class classification model (OVO and OVA) and the construction of the multi-class hierarchical classification model. There are three types of tests to run that is OVO binary, multi-class OVO and binary OVA before the construction of multi-class hierarchy. Data is sets according to the specified test. Records used are from training data sets. There are 10 classifier for OVO and five classifier for OVA using a collection of 41 features (results from Experiment I). The result will helps to construct priority order based on the type of attack and supply the order in hierarchical classification model for Experiment 3.

Experiment 2(a)
In this experiment, data are divided into 5 major classes. Each class will then paired with the other class resulting in 20 testing pairs as in Table 1.5. Each of these class pairs is tested using LIBSVM with the RBF kernel. This method is an OVO testing method but is generated manually. The highest value of classification for each major class tested will be selected.

Table 1.5 Accuracy of detection percentage obtained from Binary class OVO testing.

No	Dataset Combination		Total Data	Number of correct detection		Best Parameter		Detection Percentage
	A	B		A	B	C	γ	
1	DoS-versus-Probe		10 000	5000	5000	32 768	0.008	**100%**
2	DoS-versus-U2R		5052	4999	52	2048	0.031	99.98%
3	DoS-versus-R2L		5995	4998	992	2048	0.0005	99.92%
4	DoS-versus-Normal		10 000	4997	4998	32	0.5	99.95%
5	NORMAL-versus-U2R		5052	4999	35	2048	0.008	99.64%
6	NORMAL-versus-DoS		10 000	4998	4997	32	0.5	**99.95%**
7	NORMAL-versus-Probe		10 000	4995	4992	128	0.125	99.87%
8	NORMAL-versus-R2L		5995	4994	994	512	2	97.15%
9	PROBE-versus-DoS		10 000	5000	5000	32768	0.008	**100%**
10	PROBE-versus-R2L		5995	5000	995	128	0.031	**100%**
11	PROBE-versus-U2R		5052	4999	45	512	0.0001	99.84%
12	PROBE-versus-Normal		10 000	4990	4995	128	0.125	99.85%
13	U2R-versus-Dos		5052	52	4999	2048	0.031	**99.98%**
14	U2R-versus-Probe		5052	45	4999	512	0.0001	99.84%
15	U2R-versus-Normal		5052	35	4999	2048	0.008	99.64%
16	U2R-versus-R2L		1047	25	995	32 768	0.0005	97.42%
17	R2L-versus-Probe		5995	995	5000	128	0.031	**100%**
18	R2L-versus-U2R		1047	995	25	32 768	0.0005	97.42%
19	R2L-versus-Normal		5995	994	4994	512	2	99.88%
20	R2L-versus-DoS		5995	992	4998	2048	0.0005	99.93%

İt showed that the DoS versus Probe and Probe versus R2L models produced the highest percentage of 100% each. This shows that the model is capable of classify accurately. In addition, it may be noted that the lowest percentage model is the U2R versus R2L model of 97.42%.

Experiment 2(b)
This test tests one-to-all binary classes. In this experiment, the data is divided into 5 main classes and testing is done between one Class with the remaining of the other class for example DoS versus joint class(Probe + U2R + R2L + Normal). These sets are then tested using LIBSVM with RBF kernel and the data are marked with 0 for the primary class and 1 for the combination class. In this experiment, data is tested in OVA. Table 1.6 provides comparison of the precision level results obtained with the use of best parameters. Refer to the table, DoS attack category gave the highest result of accuracy using the best value of parameters C and γ 99.99%. Therefore, DoS gets the highest priority order while secondly, Probe gives 99.907% accuracy and 99.907% for R2L. Next U2R earns 99.83% and lastly Normal with 99.58%.

Table 1.6 Detection accuracy percentage for OVA binary class testing

Class	Best C value	Best γ value	Accuracy Level (K) with the best valueof parameter C and γ
Normal	128	0.125	**99.58%**
U2R	512	0.00195	**99.83%**
R2L	128	2	**99.91%**
DoS	512	0.125	**99.99%**
Probe	128	0.125	**99.91%**

Experiment 2(c)
For this experiment, data is divided into 5 main classes and marked using values of 0 to 4 for each class. Test conducted using LIBSVM with RBF kernel with multi-class OVO. Table 1.7 shows the confusion matrix obtained from OVO classification and Table 1.8 provides level of accuracy for each class.

Table 1.7 Confusion matrix on accuracy obtained from OVO classification.

Category	Normal	U2R	R2L	DoS	Probe
Normal	**4980**	1	12	1	6
U2R	7	**41**	4	0	0
R2L	14	0	**981**	0	0
DoS	0	0	0	**5000**	0
Probe	4	0	0	0	**4996**

Table 1.8 Accuracy level for multi-class OVO testing

Category	Accuracy	Accuracy Percentage
Normal	4980/5000	**99.60%**
U2R	41/52	**78.85%**
R2L	981/995	**98.59%**
DoS	5000/5000	**100.00%**
Probe	4996/5000	**99.92%**

The DoS model produced the highest percentage of 100% (5000/5000) accurately. The second highest model is Probe with 99.92% (4996/5000). This shows the two models are able to classify precisely. In addition, note that the model with lowest percentage is the U2R model which is about 78.85% (41/52).

Order of Priority 2(d)
Based on the results obtained from the three experiments namely (a) binary class OVO, (b) binary class OVA and (c) multi-class OVO, preference arrangement is made to obtain order according to the highest level of accuracy followed by subsequent accuracy such as Table 1.9.

Table 1.9 Comparison Summary of Detection Rate between binary class OVO, binary class OVA and multi-class OVO

Class	Binary Class OVO	Priority order	Binary Class OVA	Priority order	Multi-class OVO	Priority order
Normal	99.95%	3	99.58%	5	99.60%	3
U2R	99.98%	2	99.83%	4	78.85%	5
R2L	100%	1	99.91%	3	98.59%	4
DoS	100%	1	99.99%	1	100.00%	1
Probe	100%	1	99.91%	2	99.92%	2

Furthermore, it conclude that the DoS classifier model provides the highest percentage based on test from binary class OVO, binary class OVA and multi-class OVO followed by Probe classifier. The third level is R2L based on the highest accuracy obtained during binary class OVO and the third highest for binary class OVA. Next, U2R and Normal are the lowest class in priority order based on binary class OVO and binary class OVA. Based on the accuracy of the detection value obtained from all the tests, summarized the best priority order as follows:

Table 1.10 Final priority order level of detection based on the best order by class

Class	Priority order
DoS	1
Probe	2
R2L	3
U2R	4
Normal	5

4.3 Experiment 3

The main objective of this experiment is to evaluate the performance of the standard multi-class classification (OVO) and hierarchy using records in test dataset. This test focused on the model ability to detect new types of attacks that are not found the previous training data. The record used is the test dataset. The model used is the standard classification model (OVO) and the multi-class hierarchy model developed from Experiment II. The comparison is done with the result obtained from multi-class OVO classification method with multi-class hierarchy experiment with test datasets. Next, Figure 1.8 is the proposed model for multi-class hierarchical classification by excluding a class at each level initiated according to priority order defined in Table 1.10.

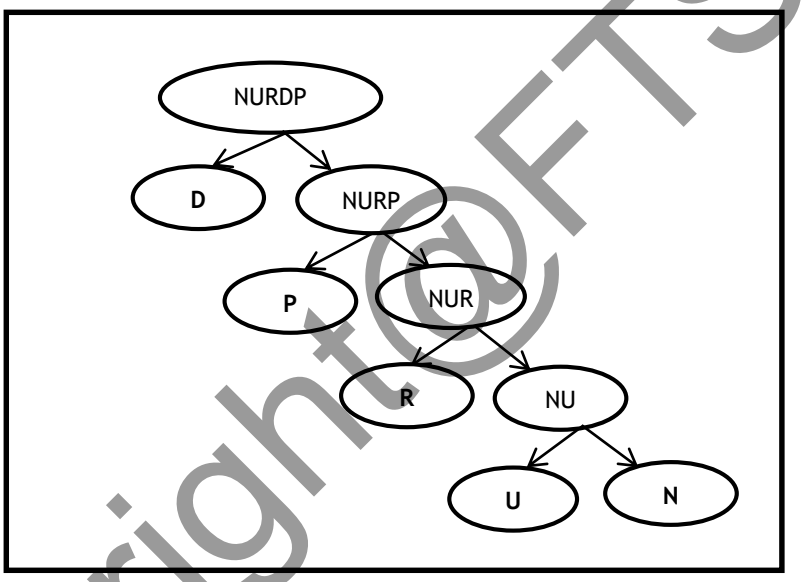


Figure **Error! No text of specified style in document.**.8 The Proposed Model of Hierarchical Multi-class.D=DoS, P=Probe, R=R2L, U=U2R, N=Normal

Table 1.11 The comparison of detection levels for Multi-class OVO Model dan Multi-class hierarchy model using training and test data by class

Class	Multi-class OVO Model		Multi-class hierarchy model	
	Training Data	Test Data	Training Data	Test Data
Normal	99.60%	96.89%	99.64%	99.38%
U2r	78.85%	0.00%	99.90%	99.38%
R2l	98.59%	13.72%	99.99%	77.17%
Dos	100%	0.00%	99.64%	88.73%
Probe	99.92%	94.42%	99.85%	90.28%
Average %	**95.39%**	**41.01%**	**99.80%**	**90.98%**

Based on Table 1.11 above, multiple-class hierarchical models gives higher results using training data with 99.80% compared to 95.39% for Multi-class OVO Models. Subsequently, the testing of multi-class hierarchical models using test data provides better Accuracy (P) results compared to the multi-class OVO with 90.98% to 41.01%. U2R and DoS both provide 0% results during testing with test data using multi-class OVO models. This is significant with one of OVO's weaknesses which this method fails to provide accurate generalization especially attack data that have very small numbers like U2R or too big amount data in this case is DoS. In training data set, DoS data have the largest number while U2R is the smallest number between attack class. The classification of multi-class OVO Models performed simultaneously for all five models instead compared of one to one for a hierarchical model. Through the use of the multi-class hierarchical model, better generalizations can be produced as the number of classes is decreasing at each descending level.

5.0 CONCLUSION

Each SVM classifier can only manage binary classification at a time. For the purpose of classification of multiple classes, a combination of several SVM strategies such as OVA, OVO and binary are used. The aim of this study is to test and compare which of the model among Multi-class OVO Models and Multi-class binary tree hierarchy gave better detection. Based on experiments conducted on the NSL-KDD dataset, this proposed model can achieve 90.98% accuracy with a false warning rate of 0.5. There was also increased for U2R and R2L detection although not on DoS and Probe Models. The model was carried out using training data consists of 16 047 records. The total test data is 22 544 used totally with new attack types which was not found in training data. Therefore, this study proposes the use of the SVM hierarchy with inclining binary tree for the classification of multiple classes gives better results than the traditional OVO and OVA classification methods in classifing five types of attacks in NSL-KDD data sets. The removal of one class at each level helps to speed up the classification process and produce better results. This study also contributes to the priority order method for each level of hierarchy. Experiments shows that the conclusion is convincing.

6.0 PROPOSED RESEARCH EXPANSION

For further continuous research, there are several aspects that can be considered to maximize the findings and strengthen the research aspect. There are various methods used in intrusion detection to determine the order of hierarchy eg Wilk's Analysis, Cluster and DAG methods. However, based on several experiments carried out, hierarchical level arrays were determined with the highest yields to the lowest obtained from each experiment. This expands the growth of hierarchical level arrangement concept for future studies. Thus, other aspects that can be study is the combination of features that produce good feature combinations, balanced and comprehensive data for model development and choosing different kernels to get better results. In conclusion, this study proves that the multiple-class hierarchical model of the inclined binary tree is able to provide much better classification results than the standard classification class of OVO and OVA Class.

Acknowledgement

This work has been supported by the Malaysia's Ministry of Higher Education Fundamental Research Grant FRGS/1/2019/ICT02/UKM/02/8.

7.0 REFERENCE

Azizi Abdullah. 2010. Supervised Learning Algorithms for Visual Object Categorization. Tesis PhD, Universiteit Utrecht.

Benabdeslem, K. 2006. Descendant Hierarchical Support Vector Machine for Multi-Class Problems. *International Joint Conference on Neural Networks (IJCNN)*. doi: 10.1109/IJCNN.2006.246868

Bishop, C. M. 2006. Pattern Recognition and Machine Learning. Springer-Verlag New York. Softcover ISBN 978-1-4939-3843-8. http://users.isr.ist.utl.pt/~wurmd/Livros/school/Bishop%20-%20Pattern%20Recognition%20And%20Machine%20Learning%20-%20Springer%20%202006.pdf

Bruneau, G. 2001. The History and Evolution of Intrusion Detection. SANS Institute InfoSec Reading Room. https://www.sans.org/reading-room/whitepapers/detection/history-evolution-intrusion-detection-344.

Calix, R. A. & Sankaran R. 2013. Feature Ranking and Support Vector Machines Classification Analysis of the NSL-KDD Intrusion Detection Corpus. *Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society (FLAIRS Conference) Conference.* Association for the Advancement of Artificial Intelligence (www.aaai.org)

Chen, L.-S. & Syu, J.-S. 2015. Feature Extraction based Approaches for Improving the Performance of Intrusion Detection Systems. *Proceedings of the International MultiConference of Engineers and Computer Scientists 2015 Vol I, IMECS 2015, March 18-20, 2015, Hong Kong.*

Cortes, C. & Vapnik, V. 1995. AT&T Bell Labs., Holmdel, NJ 07733, USA. *Machine Learning, 20, 273-297 (1995). Kluwer Academic Publishers, Boston.*

Denning, D. E. 1987. An Intrusion-Detection Model. *IEEE Transactions On Software Engineering, Vol. Se-13, No. 2, February 1987*

Eid, H. F., Hassanien, A. E., Kim, T.-H. & Banerjee, S. 2010. Linear Correlation-Based Feature Selection for Network Intrusion Detection Model. *Scientific Research Group in Egypt (SRGE).*http://www.egyptscience.net

Enache, A.-C. & Sgârciu, V. 2015. Anomaly Intrusions Detection Based On Support Vector Machines with an Improved Bat Algorithm. 2015 20th International Conference on Control Systems and Computer Science. doi: 10.1109/CSCS.2015.12

Fischer, M. 2014. Resilient Networking: Intrusion Detection. https://www.tk.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_TK/08_IDS_01.pdf. Technische Universitat Darmstadt. [21 September 2017]

Ghose, A. 2017. Support Vector Machine (SVM) Tutorial. https://blog.statsbot.co/support-vector-machines-tutorial-c1618e635e93

Gu, C., Zhang, B., Wan, X., Huang, M. & Zou, G. 2016. The Modularity-based Hierarchical Tree Algorithm for Multi-class Classification. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016 17th IEEE/ACIS International Conference on 30 May-1 June 2016.

Hasan, M. A. M., Nasser, M., Pal, B., & Ahmad, S. 2014. Support Vector Machine and Random Forest Modeling for Intrusion Detection System (IDS). *Journal of Intelligent Learning Systems and Applications. Vol. 6, No. 1(2014) 45-52.* doi: 10.4236/jilsa.2014.61005

Hassan, A. & Damper, R. I. 2010. Multi-class and Hierarchical SVMs for Emotion Recognition. School of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK.

Horng, S.H., Su, M.-Y., Chen, Y.-H., Kao, T.-W., Chen, R.-J., Lai, J.-L. & Perkasa, C.D. 2011. A Novel Intrusion Detection System Based On Hierarchical Clustering And Support Vector Machines. *Expert Systems with Applications 38 (2011) 306-313.* https://doi.org/10.1016/j.eswa.2010.06.066

Hsu, C.-W., Chang, C.-C. & Lin, C.-J. 2010. A Practical Guide to Support Vector Classification. http://www.csie.ntu.edu.tw/~cjlin.

Kang, S.-H. & Kim, K. J. 2016. A Feature Selection Approach To Find Optimal Feature Subsets For The Network Intrusion Detection System. Springer Science+Business Media New York 2016.

Lee, W. & Stolfo, S. J. 2000. A Framework for Constructing Features and Models for Intrusion Detection Systems. *ACM Transactions on Information and System Security, Vol. 3, No. 4, November 2000, Pages 227-261.*

Li, H., Jiao R. & Fan J. 2008. Precision of Multi-class Classification Methods for Support Vector Machines. *Signal Processing, 2008. ICSP 2008. 9th International Conference on 26-29 Oct. 2008.*

Li, Y., Xia, J., Zhang, S., Yan, J. Ai, X. & Dai, K. 2012. An efficient intrusion detection system based on support vector machines and gradually feature removal method. *Expert Systems with Applications, 39(1):424-430.* doi: 10.1016/j.eswa.2011.07.032.

Liao, H.-J., Lin, C.-H. R., Lin, Y.-C. & Tung, K.-Y. 2013. Intrusion Detection System: A Comprehensive Review. *Journal of Network and Computer Applications 36 (2013):16-24.*

Limthong, K. 2013. Real-Time Computer Network Anomaly Detection Using Machine Learning Techniques. *Journal of Advances in Computer Networks, Vol. 1, No. 1, March 2013.*

Mohd Rizal Kadis. 2016. Umpukan Lembut Kluster Sejagat dan Setempat untuk Sistem Pengesanan Pencerobohan: Satu Kajian Perbandingan. Tesis Sarjana Keselamatan Siber. Universiti Kebangsaan Malaysia.

Nashat, S. & Abdullah, M.Z, 2010. Multi-Class Colour Inspection of Baked Foods Featuring Support Vector Machine and Wilk's λ Analysis. *Journal of Food Engineering 101 (2010) 370–380.* doi: 0.1016/j.jfoodeng.2010.07.022

Nashat, S., Abdullah, A., Aramvith, S. & Abdullah, M. Z. 2011. Support Vector Machine Approach to Real-Time Inspection of Biscuits on Moving Conveyor Belt. *Computers and Electronics in Agriculture, 75(1), 147-158.* doi: 10.1016/j.compag.2010.10.010

Nguyen, H. T., Franke, K. & Petrovic, S. 2012. Feature Extraction Methods for Intrusion Detections System. https://www.researchgate.net/publication/231175349_Feature_Extraction_Methods_for_Intrusion_Detection_Systems.

Panetta, K. 2017. 5 trends in cybersecurity for 2017 and 2018. http://www.gartner.com/smarterwithgartner/5-trends-in-cybersecurity-for-2017-and-2018/.

Parsaei, M. R., Rostami S. M. & Javidan, R. 2016. A Hybrid Data Mining Approach for Intrusion Detection on Imbalanced NSL-KDD Dataset. *International Journal of Advanced Computer Science and Applications, Vol. 7, No. 6, 2016.*

Pervez, M. S. & Farid, D. M. 2014. Feature Selection and Intrusion Classification in NSL-KDD Cup 99 Dataset Employing SVMs. *8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA).* doi: 10.1109/SKIMA.2014.7083539.

Platt, J. C., Cristianini, N. & Shawe-Taylor, J. 2000. Large Margin DAGs for Multiclass Classification. *In Advances in Neural Information Processing Systems (pp. 547-553).*doi: 10.1.1.158.4557.

Sahu, S. K., Sarangi, S. & Jena, S. K. 2014. A Detail Analysis on Intrusion Detection Datasets. *Souvenir of the 2014 IEEE International Advance Computing Conference, IACC 2014.* doi: 10.1109/IAdCC.2014.6779523

Sasan, H. P. S., & Sharma, M. 2016. Intrusion Detection Using Feature Selection and Machine Learning Algorithm with Misuse Detection. *International Journal of Computer Science & Information Technology (IJCSIT) Vol 8, No 1, February 2016.*

Schwenker, F. 2000. Hierarchical Support Vector Machines for Multi-Class Pattern Recognition. *Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies.*doi: 10.1109/KES.2000.884111

Singh, J. & Nene, M. J. 2013. A Survey on Machine Learning Techniques for Intrusion Detection Systems. *International Journal of Advanced Research in Computer and Communication Engineering. 2013, Nov,2(11).*

Sridhar, M. S. 2017 . Research Methodology Part 1:Introduction to Research & Research Methodology. ISRO Satellite Centre. https://www.researchgate.net/publication/39168208_Research_Methodology_Part_1_Introduction_to_Research_Research_Methodology

Staudemeyer, R. C. & Omlin, C.W. 2014. Extracting Salient Features for Network Intrusion Detection using Machine Learning Methods. *South African Computer Journal Research Article-SACJ, 53, July 2014.* doi: 10.18489/sacj.v52i0.200.

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. 2009. A Detailed Analysis of the KDD CUP 99 Data Set. *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA 2009).* doi: 10.1109/CISDA.2009.5356528

The UCI KDD Archive 1999. KDD CUP 1999 Data. University of California, Irvine. http://kdd.ics.uci.edu/databases/kddcup99/task.html

Xiao, L. & Cheng, L. 2015. State Classification Algorithm for Bus Based on Hierarchical Support Vector Machine. *2015 8th International Symposium on Computational Intelligence and Design (ISCID).*doi: 10.1109/ISCID.2015.259

Xue, S., Jing, X., Sun, S. & Huang, H. 2014. Binary-Decision-Tree-Based Multiclass Support Vector Machines. *2014 14th International Symposium on Communications and Information Technologies (ISCIT).* doi: 10.1109/ISCIT.2014.7011875.

Wang, Y.-C. F. & Casasent, D. 2006. Hierarchical K-means Clustering Using New Support Vector Machines for Multi-class Classification. *2006 International Joint Conference on Neural Networks Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada July 16-21, 2006.*

Wu, T. 2009. Practical Guide to Support Vector Machines. MPLAB, UCSD. Retrieved from : http://tdlc.ucsd.edu/events/boot_camp_2009/tingfansvm.pdf

Zisserman, A. 2015. Lecture 2: The SVM classifier. Information Engineering, Department of Engineering Science, University of Oxford. http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf