Hindawi Publishing Corporation
Applied Computational Intelligence and Soft Computing
Volume 2012, Article ID 850160, 20 pages
doi:10.1155/2012/850160

Review Article

The Use of Artificial-Intelligence-Based Ensembles for Intrusion Detection: A Review

Gulshan Kumar[1] and Krishan Kumar[2]

[1] *Department of Computer Application, Shaheed Bhagat Singh State Technical Campus, Ferozepur, Punjab 152004, India*
[2] *Department of Computer Science & Engineering, Punjab Institute of Technology, Kapurthala, Punjab 144601, India*

Correspondence should be addressed to Gulshan Kumar, gulshanahuja@gmail.com

Received 4 April 2012; Accepted 11 July 2012

Academic Editor: Farid Melgani

In supervised learning-based classification, ensembles have been successfully employed to different application domains. In the literature, many researchers have proposed different ensembles by considering different combination methods, training datasets, base classifiers, and many other factors. Artificial-intelligence-(AI-) based techniques play prominent role in development of ensemble for intrusion detection (ID) and have many benefits over other techniques. However, there is no comprehensive review of ensembles in general and AI-based ensembles for ID to examine and understand their current research status to solve the ID problem. Here, an updated review of ensembles and their taxonomies has been presented in general. The paper also presents the updated review of various AI-based ensembles for ID (in particular) during last decade. The related studies of AI-based ensembles are compared by set of evaluation metrics driven from (1) architecture & approach followed; (2) different methods utilized in different phases of ensemble learning; (3) other measures used to evaluate classification performance of the ensembles. The paper also provides the future directions of the research in this area. The paper will help the better understanding of different directions in which research of ensembles has been done in general and specifically: field of intrusion detection systems (IDSs).

1. Introduction

The threat of Internet attacks is quite real and frequent so this has increased a need for securing information on any network on the Internet. The objective of information security includes confidentiality, authentication, integrity, availability, and nonrepudiation [1]. The set of activities that violates security objectives is called intrusion. Thus secure information requires the phases that provide (1) protection: automatic protection from intrusions; (2) detection: automatic detection of intrusions; (3) reaction: automatic reaction or alarm when system is intruded; (4) recovery: repair or recovery of loss caused due to intrusion [2]. Out of these phases, the perfect detection of an intrusion is the most important. As only after correct detection of intrusion, correct reaction and recovery phase of information security can be implemented. In the literature, many IDSs have been developed implementing various techniques from different disciplines like statistical techniques, AI techniques, and so forth. Some IDSs have been developed based on single-classification technique while other IDSs (called hybrid/ensemble IDS) implement more-than-one-classfication technique. Ensemble-based IDSs have many advantages over the IDS implementing single technique (refer to Section 2). Many researchers have proposed different ensembles for ID by exploiting the different characteristics of weak classifiers and datasets. To cover various aspects of ensembles, many researchers proposed different taxonomies for ensembles. Keeping the advantages of AI-based techniques over other techniques and ensembles, many researchers proposed AI-based ensembles for ID. However, there exists no comprehensive review of taxonomies of ensembles (in general) and AI-based ensembles for intrusion detection (ID) (in specific).

The objective of this paper is threefold. First objective is to present an updated review of ensembles and their

taxonomies in general for supervised classification. Second objective is to present an updated review of different AI-based ensemble/hybrid classifiers proposed for ID during last decade and compare them by set of evaluation metrics which derives from (1) architecture & approach followed; (2) different methods utilized in different phases of ensemble learning; (3) other measures used to evaluate classification performance of the ensembles. Third objective is to highlight research gaps and directions in developing efficient ensemble for ID.

Paper Overview. The rest of paper is organized as follows. Section 2 highlights the state of art, need, advantages, and disadvantages of AI-based techniques and their ensembles for ID. Section 3 lists the reasons and benefits for combining multiple base classifiers. Various taxonomies proposed in the literature are presented in Section 4. The section also describes various methods used at different levels to generate ensembles. Section 5 highlights various AI-based ensembles proposed for ID during the last decade. Related studies are compared by various evaluation metrics. Finally, Section 6 concludes the paper and presents future research directions.

2. Intrusion Detection

An intrusion detection system (IDS) defined as "an effective security technology, which can detect, prevent and possibly react to the computer attacks" is one of the standard components in security infrastructures. It monitors target sources of activities, such as audit and network traffic data in computer or network systems and then deploys various techniques in order to provide security services. The main objective of IDS is to classify intrusive and nonintrusive network activities in an efficient manner. The process of intrusion detection involves the tasks: (1) data acquisition/collection; (2) data Preprocessing & feature selection; (3) model selection for data analysis; (4) classification and result analysis [3]. To handle these tasks, IDS comprises of different modules for efficient ID. The modules are network to monitor, Data collection & storage unit, Data analysis & processing unit, and signal [4, 5] as depicted in Figure 1.

Based upon these modules, IDSs can be categorized into different classes like host-based IDS (HIDS) versus network-based IDS (NIDS), misuse- or signature-based IDS versus anomaly-based IDS, Passive IDS versus Active IDS, and so forth [5]. HIDSs are developed to monitor the activities of a host system and state, while NIDSs monitor the network traffic for multiple hosts. HIDSs and NIDSs have been designed to perform misuse detection and anomaly detection. Anomaly based ID allows detecting unknown attacks for which the signatures have not yet been extracted [2]. In practice, anomaly detectors generate false alarms due, in large part, to the limited data used for training and to the complexity of underlying data distributions that may change dynamically over time. Since it is very difficult to collect and label representative data to design and validate an anomaly detection system, its internal model of normal behavior will tend to diverge from the underlying data distribution. Further details can be studied in [4, 5].

Since the first introduction, IDSs have been evaluated using a number of different ways based upon evaluation datasets [6]. Various features of IDS can be evaluated, which may range from performance and correctness to usability. However, in the literature most tests that have been performed have mainly focused on measuring the accuracy and effectiveness of IDS, that is, the false alarm rate and the percentage of attacks that are successfully detected. Several other metrics are utilized by researchers to measure the performance of IDS. These metrics can be divided into three classes: threshold, ranking, and probability metrics [7, 8]. Threshold metrics include classification rate (CR), F-measure (FM), and cost per example (CPE). It is not important how close a prediction is to a threshold, only if it is above or below threshold. The value of threshold metrics lies in range from 0 to 1. Ranking metrics include false-positive rate (FPR), detection rate (DR), precision (PR), and area under ROC curve (ROC). The value of ranking metrics lies in range from 0 to 1. These metrics depend on the ordering of the cases, not the actual predicted values. As long as ordering is preserved, it makes no difference. These metrics measure how well the attack instances are ordered before normal instances and can be viewed as a summary of model performance across all possible thresholds. Probability metrics include root mean square error (RMSE). Value of RMSE lies in range from 0 to 1. The metric is minimized when the predicted value for each attack class coincides with the true conditional probability of that class being normal class. Generally, these metrics are computed from confusion matrix.

The performance of IDS is generally evaluated based upon audit data containing mix of legitimate traffic and attacks. The IDSs are evaluated by comparing the true-positive rate (i.e., the percentage of attacks that were correctly recognized) and the false-positive rate (i.e., the percentage of legitimate traffic flagged as an attack). Many researchers tried to collect evaluation audit data. The important audit data available as benchmarked dataset are (1) DARPA evaluation dataset collected at MIT Lincoln Laboratory in year 1998, 1999, and 2000 [9]; (2) KDD cup 1999 dataset [10]; (3) UNM dataset of systems calls [11]; (4) DEFCON 9 capture the flag (CTF) dataset [12]; (5) ITOC dataset [13]; (6) many more datasets collected for realistic evaluation from specific organization. The details can be further explored in [7]. KDD cup 1999 dataset is most popular publically available evaluation benchmarked dataset. But, the dataset is critically discussed in the literature for being nowadays outdated due to the type of attacks and background traffic used and for the methodology implemented for building it [14].

However, in real world, intrusion detection process involves processing of high dimensions of network & system data. Processing of high-dimensional data for ID is highly computationally expensive. This cause may lose real-time capability of IDS. The computation overhead may be reduced by applying feature reduction techniques, which can be further explored in [15, 16]. The distribution of data is also dynamically changing with passage of time having new

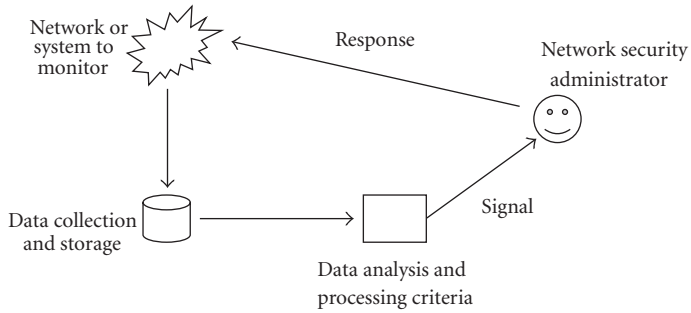

Figure 1: Architecture of IDS.

patterns of novel attacks. Nonavailability of signatures of novel attacks in databases leads to high false alarm rate and low detection accuracy. In fact, practitioners as well as researchers have observed that IDSs can easily trigger thousands of alarms per day, upto 99% of which are false positives (i.e., alarms that were mistakenly triggered by benign events) [17]. Most of attacks are likely to generate multiple related alarms. This flood of mostly false alarms makes it very difficult to identify the hidden true positives (i.e., those alarms that correctly flag attacks) [17]. Current intrusion-detection systems do not make it easy for network administrator to logically group related alerts. Another problem with current intrusion detection is scalability as it is difficult to achieve large-scale deployment [18]. The most appealing way to reduce false alarms is to develop better IDSs that generate fewer false alarms. The process of reducing the false alarms is very challenging because the false alarms are the result of many problems. The major problems includes (1) lack of suitable training dataset; (2) significant real-time requirements; (3) ambiguous events which cannot be decided to constitute intrusion easily (for example, failed login); (4) inherent problem of writing correct patterns for ID; (5) current IDSs do not properly aggregate and correlate the alarms that lead to flood of false alarms for network administrator [18].

These causes require IDS to be faster, flexible (instead of strict thresholds), and adaptive (instead of fixed rules), and dynamic learning of new patterns and aggregate logically corelated false alarms to identify root cause of alarms. The efficient IDS would have to address all these issues including reduction of false positives and fast processing of high volume of network traffic and be adaptive to changing environment for novel attacks.

Various classification techniques (classifiers) from different disciplines have been applied to detect the intrusions efficiently. Examples of these techniques include statistical techniques, artificial-intelligence- (AI-) based techniques, and its subfield techniques [5, 19, 20]. Here AI-based techniques include techniques like decision-tree-based techniques, rule-based techniques, data mining-techniques, genetic-algorithm-based techniques, machine-learning techniques (Neural network, SVM, Bayesian network, etc.), pattern recognition techniques and so forth. [5]. Recent advances in the field of AI led many researchers to apply AI-based techniques for ID successfully. Potential benefits of AI-based techniques over other conventional techniques includes (1) flexibility (versus threshold definition of conventional techniques); (2) adaptability (versus specific rules of conventional techniques); (3) pattern recognition (and detection of new patterns); (4) fast computing (faster than humans); (4) learning capabilities [21]. AI-based techniques help meet the following research issues on IDSs [22]: (1) these techniques has the capability of learning by example that helps to generalize from a representative set of examples and allows detecting new types of intrusion; (2) with learning by example approaches attack "signatures" can be extracted automatically from labeled traffic data, thus allowing to overcome the subjectivity of human interpretation of intrusive behavior, the latter being implemented in many current IDSs; (3) learning by example approaches is able to adapt to new threats. The major difference between AI-based and traditional IDSs is that only AIs can learn new rules automatically, whereas in traditional systems the security administrator must add new rules for each new attack type or each new allowed program. In AI-based systems, it is possible to train the system by examples rather than rules.

Many researchers applied and evaluated AI-based techniques using different evaluation datasets for ID. They reported many challenges related to AI-based techniques and dataset for ID. The challenges related to techniques include: (1) no single-classification technique is capable enough to detect all classes of attacks to acceptable false alarm rate and detection accuracy [23, 24]; (2) some of the existing techniques fall into local minima, for global minima, these techniques are computationally expensive; (3) existing techniques are not capable to model correct hypothesis space of problem [25]; (4) some existing techniques are unstable in nature such as neural networks that show different results with different initializations due to the randomness inherent in the training procedure; (5) different techniques trained on the same data may not only differ in their global performances, but they also may show strong local differences. Each technique may have its own region in the feature space where it performs the best [26]. Related evaluation dataset challenges include (1) lack of sufficient amount of quality training data; (2) class imbalance of training data causes the results of classifiers to be biased towards majority class.

In order to solve these problems, many researchers utilized AI-based ensembles for ID successfully. They proved

that AI-based ensembles can improve detection performance over a single technique/classifier [27–29]. The concept of ensemble is to employ multiple base classifiers and their individual predictions are combined in some way to obtain reliable and more accurate predictions [17, 25]. Employment of ensemble and combination of multiple predictions are mainly motivated by three aspects which characterize the intrusion detection domain [3]: (a) relevant information may be present at multiple abstraction levels, (b) the information may be collected from multiple sources, and (c) this information needs to be represented at the human level of understanding.

No doubt, AI-based ensemble/hybrid classifiers improved the performance over single classifier [23, 28–45]. But, still some research issues exist. The major issues include diversity among the base classifiers, ensemble size, computational overhead, input feature space, and combining strategy.

3. Ensemble Classifiers

The ensembles involve the employment of multiple base classifiers and combine their predictions to obtain reliable and more accurate predictions. Dietterich [25] listed three specific reasons for benefits of ensembles: statistical, computational, and representational reason. Other reasons for combining different classifiers include [26] the following. (1) a designer may have access to a number of different classifiers, each developed in a different context and for an entirely different representation/description of the same problem. An example is the identification of persons by their voice, face, and handwriting. (2) Some times more than a single training set is available, each collected at a different time or in a different environment. These training sets may even use different features. (3) Different classifiers trained on the same data may not only differ in their global performances, but they also may show strong local differences. Each classifier may have its own region in the feature space where it performs the best. (4) Some unstable classifiers like neural networks show different results with different initializations due to the randomness inherent in the training procedure. Instead of selecting the best network and discarding the others, one can combine various networks. Different combination strategies can be applied either to classify combination or to correlate alerts [3]. The former approach involves the use of different classifiers to take a unique decision about the data pattern typically related to a single network packet whereas the later approach is mainly aimed at providing a high-level description of the detected pattern/attack by using the outputs of different classifiers/IDS.

Ensemble helps to meet the following challenges of ID (refer to Section 2 for ID challenges). (1) Ensemble comprise of multiple weak classifiers instead of single classifier. The multiple classifiers complement weaknesses of each other and hence improve the performance. (2) Ensembles use the combined knowledge to model the hypothesis of the problem upon different subset of dataset or feature subspace. The combined knowledge of weak classifiers helps to improve the performance even in lack sufficient amount of quality training data. (3) Since ensemble use the multiple classifiers, so it helps to find the global solution that leads to reduce the false alarm rate and increase the detection accuracy. (4) Unstable base classifiers help to generate the diverse set of base classifiers for efficient ensemble. (5) Classifiers trained with same dataset showing different performance help to maintain diversity among the base classifiers.

In nutshell, by using combined knowledge of multiple classifiers trained by different dataset and exploiting different characteristics of problems, ensembles are capable to improve the performance (in terms of increased detection accuracy and decreasing false-positive rate) even in lack of sufficient amount of quality training data. But, computations of multiple predictions in ensembles increase computational overhead. Many researchers and practitioners advocate ensemble classifiers by keeping following points in mind. (1) The availability of enormous computational power (to cope up computational overhead of ensemble classifiers); (2) lack of quality training data for realistic evaluation; (3) improved performance (over single classifier) of ensembles. An important factor affecting the efficiency of ensemble is the diversity among base classifiers [46–48]. The diversity in ensembles refers to different errors made by different base classifiers on data records. In order to produce diverse classifiers, researchers used two types of methods: (1) *Implicit*; (2) *Explicit* [47, 49]. Implicit methods do not involve any direct measure of diversity whereas explicit method does. Kuncheva and Whitaker [50] proposed different metrics to measure diversity. The diversity in ensembles can be obtained by using different (1) starting point in hypothesis space; (2) set of accessible hypotheses; (3) traversal of hypothesis space [47]. The general observation in ensemble construction is to combine multiple diverse classifiers.

By keeping the benefits of AI techniques and performance enhancement by using ensemble approach in mind (refer to Section 2), AI-based ensembles have been successfully applied to improve the performance of classifier in many fields (e.g., finance [51]), bioinformatics [52], medicine [53], information security [28, 33, 54] information retrieval [55], and so forth, many researchers report that ensembles often outperform the individual best base classifier [5, 33, 47, 51, 56–58]. They proposed different concepts to describe the improved performance, reduced generalization error, and successful applications of ensembles to different fields over individual classifier. For example, Allwein et al. [59] interpreted the improved performance in the framework of large margin classifiers, Kleinberg in the reference of Stochastic Discrimination theory [60], and Breiman in the terms of the bias variance analysis [61].

In spite of much research on AI-based ensembles, many research questions still remain unanswered, for example, how many base classifiers should be combined, how base classifiers should be combined, how to generate diverse set of base classifiers, how instances of training dataset should be partitioned to generate base classifiers, how feature space should be partitioned and in particular for ID quality training dataset, and so forth.

4. Taxonomy

Ensemble learning process has three phases: (1) ensemble generation; (2) ensemble selection; (3) ensemble integration. *Ensemble generation* is homogenous if the same induction algorithm is used to generate all the classifiers of the ensemble, otherwise it is said to be heterogeneous. In ensemble generation phase, a pool of diverse base classifiers are generated. This can be done by using (1) different initialization parameters of base classifiers; or (2) different subsets of feature space (feature level); or (3) different data subsets (data level) to train the base classifiers. *Ensemble selection* requires selection of classifiers from pool of diverse base classifiers. Here different methods can be utilized to combine the pool of base classifiers obtained in ensemble generation phase: (1) combine all base classifiers; (2) combine smaller subsets according to clustering; (3) combine reduced sets of base classifiers that exceed specific thresholds on performance (i.e., overproduce and choose strategy) [33]. *Ensemble integration* involves the combination of predictions of set of base classifiers selected in ensemble selection phase. It can use two different methods: (1) combination (also called fusion); or (2) selection [46]. The combination method consists in the combination of the predictions obtained by the different classifiers in the ensemble to obtain the final prediction. The selection approach, specially its dynamic form, selects one (or more) classifier from the ensemble according to the prediction performance of these classifiers on similar data from the validation set. The ensemble integration phase involves many strategies to combine multiple predictions because these strategies have performance variability when tackling different problems. According to No Panacea Theorem for classifier combination, there is always a situation in which, under certain hypotheses, a combination strategy gives very bad performance [62]. This proves that there is neither a perfect combination strategy, nor one generally outperforming each other. This statement can be used as theoretical guideline for a malicious user to disrupt or evade the system, once combination strategy implemented is known to them.

In nutshell, we may have different feature sets, different training sets, different classification methods, or different training sessions, all resulting in a set of classifiers, whose outputs may be combined, with the hope of improving the overall classification performance [26]. If this set of classifiers is fixed, the problem focuses on the ensemble integration phase. It is also possible to use a fixed combiner and optimize the set of input classifiers; the problem focuses on the generation and selection phases.

Keeping in view, the popularity and successful applications of ensembles in different fields, various methods are proposed in the literature for creating ensembles. Researchers have proposed different taxonomies to categorize ensembles. Since research of ensemble is continuously evolving, there is no existing taxonomy that covers every aspect of ensembles. The literature review of important taxonomies is as follows.

Jain et al. [26] grouped different classifier combination schemes into three main categories according to their architecture: (1) parallel; (2) cascading (or serial combination); (3) hierarchical (tree like). In the parallel architecture, all the individual classifiers are invoked independently, and their results are then combined by a combiner. Most combination schemes in the literature belong to this category. In the gated parallel variant, the outputs of individual classifiers are selected or weighted by a gating device before they are combined. In the cascading architecture, individual classifiers are invoked in a linear sequence. The number of possible classes for a given pattern is gradually reduced as more classifiers in the sequence have been invoked. For the sake of efficiency, inaccurate but cheap classifiers (low computational and measurement demands) are considered first, followed by more accurate and expensive classifiers. In the hierarchical architecture, individual classifiers are combined into a structure, which is similar to that of a decision tree classifier. The tree nodes, however, may now be associated with complex classifiers demanding a large number of features. The advantage of this architecture is the high efficiency and flexibility exploiting the discriminant power of different types of features. Using these three basic architectures, we can build even more complicated classifier combination systems. He listed eighteen different methods for classifier combination and divided them into different categories according to their trainability, adaptivity, and requirement on the output of individual classifiers. The combination methods include (1) voting; (2) sum, mean, median; (3) product, min, max; (4) generalized ensemble; (5) adaptive weighting; (6) stacking; (7) Borda count; (8) logistic regression; (9) class set reduction; (10) Dempster Shafer; (11) fuzzy integral; (12) mixture of local experts (MLE); (13) hierarchical MLE; (14) associative switch; (15) bagging; (16) boosting; (17) random space; (18) neural tree.

Sharkey [63] proposed a three-dimensional taxonomy that include (1) selection or combination of the multiple base classifiers; the ensemble member can be competitive or cooperative: in competitive mode, a single member is selected to provide the final prediction whereas in cooperative mode the predictions of all members are combined; (2) methods based or not on the direct combination of base classifier outputs; there can be top-down or bottom-up ensembles, in top-down mode, the combination method is based on something other than the individual predictions, whereas bottom-up techniques take predictions of the members into account in their combination method, bottom up approach is further divided into two methods, namely, fixed methods (e.g., voting) and dynamic methods (e.g., stacking); (3) methods based on pure ensembles or modular systems. The pure ensemble systems combine a set of classifiers, each of which solves the same original task. On the other hand, the purpose of modular systems is to break down a complex problem into several subproblems so that each learning algorithm either solves a different task or trained by different training set. The taxonomy proposed by Sharkey [63] was further extended by Rokach [64]. He proposed taxonomy based on combiner usage, classifier dependency, diversity generation, ensemble size, and the capability of ensemble methods to be applied with different base learning algorithms.

Kuncheva [46] proposed a basic taxonomy for generating a diverse pool of classifiers. She proposed that diverse

classifiers can be generated by using various methods at four different levels, namely, (1) combination level; (2) classifier level; (3) feature level; (4) data level. At combination level, she emphasized the combination rules to accumulate multiple classifiers, distinguishing between fusion methods that combine the outputs of the base classifiers and selection methods, by which a single classifier is selected among the set of available base classifiers. Classifier level may consider different models and may design base learners for specific ensemble methods. At third level, different subsets of features can be used for the classifiers. Finally, different data subsets, so that each base classifier in the ensemble is trained on its own data, can be used to build up the committee of learning machines. She also proposed that there are two types of methods to develop ensembles.

(1) *Decision optimization*: it refers to methods to choose and optimize the combiner for a fixed ensemble of base classifiers. This method corresponds to level A (combination level as described above),

(2) *Coverage optimization*: it refers to methods for creating diverse base classifiers assuming a fixed combiner. This method corresponds to level B, C, and D.

Researchers also proposed that there are trainable and nontrainable ensembles. Trainable ensembles need additional training to create the ensemble (either during the base classifier training or after all base classifiers is trained) [33]. On the other hand, nontrainable ensembles do not need training after the base classifiers have been induced [55, 65].

Witten and Frank [66] provided four methods to generate multiple models: (1) bagging; (2) boosting; (3) stacking; (4) error-correcting code.

Bishop [67] proposed five methods for combining the individual classifiers. The methods are Bayesian model averaging, committees, boosting, tree-based models, and conditional mixture models. Boosting is further divided into two types: (1) minimizing exponential error; (2) error functions for boosting.

Marsland [68] suggested boosting, bagging, and the mixture of experts method as methods for ensembles.

Alpaydin [69] proposed seven methods of combining multiple learning algorithms: (1) voting; (2) error-correcting output codes; (3) bagging; (4) boosting; (5) mixtures of experts (6) stacked generalization; (7) cascading.

Langin and Rahimi [31] proposed three different strategies to combine base classifiers, namely, (1) consecutive combination: a consecutive combination uses methods in order, first one, and then the next; (2) ensemble combinations: an ensemble combination has methods which run in parallel with an additional method at the end which provides a single output from multiple potential outputs; (3) hybrid combinations: a hybrid combination is an offspring of two different parents which implies an interaction of some sort as opposed to being consecutive or parallel. A hybrid strategy can loop back and forth multiple times between methods or can embed one method within another method.

In this study, we adopted and presented the taxonomy proposed by Kuncheva [46] and additional aspects borrowed from Jain et al. [26]. The basic reason for adopting this taxonomy is its simplicity, popularity, and it covers basic aspects for building diverse pool of base classifiers. The author highlighted that diverse pool of classifiers can be generated by using different methods at four levels. The levels are as follows.

4.1. Combination Level. This level focuses on ensemble integration phase of ensemble learning process. Here, predictions of base classifiers are combined in some way to improve the performance of ensemble. The researchers proposed that there are three main ways in combining classifiers, namely, fusion, selection, and Mixture of expert systems [33]. In classifier fusion, each ensemble member is supposed to have knowledge of the whole feature space. Here, each member is trained by the same dataset with all features. To determine final prediction of ensemble, the combiner applies some method to combine the predictions of ensemble members in certain way to get final ensemble prediction, for example, average or majority vote (most popular) method. In classifier selection, each ensemble member is supposed to know well a part of the feature space and be responsible for objects in this part. Here, each member is trained by different dataset. Ensemble output is determined by one classifier.

In nutshell, fusion based combination methods combine all the outputs of the base classifiers, whereas selection based combination methods try to choose the best classifiers among the set of the available base classifiers. Fusion strategy generally falls under coverage optimization, whereas selection strategy falls under decision optimization [33, 46].

4.1.1. Fusion-Based Combination Methods. These methods combine the predictions of the base classifiers to determine ensemble prediction. The major methods proposed in literature are described below.

(i) *Majority Voting Method.* In majority voting ensemble, each base classifier votes for specific class and the class that collects majority of vote is predicted as ensemble final prediction [70–72].

(ii) *Threshold Plurality Vote Method.* This method is further generalization of majority vote method proposed by Xu et al. [73]. This method works by imposing a threshold on the number of votes to select the class, we may move from a unanimity vote rule, by which a class is chosen if all the base classifiers agree on that particular class label, for intermediate cases are considered by moving the threshold of votes, at the expenses of some possible unclassified instances.

(iii) *Naïve Bayes Decision Method:* This method assumes the conditional independence between classifiers. The method selects the class with the highest posterior probability computed through the estimated class conditional probabilities and Bayes' theorem [74, 75].

(iv) *Fuzzy Theory Method.* Many researchers proposed fuzzy set theory to combine base classifiers using fuzzy aggregation connectives to determine ensemble

prediction [76, 77]. Fuzzy combination methods are effective as they measure the strength of every subset of classifiers. Thus to determine the class of any unclassified instance is the decision of ensemble which is based upon competence of every subset of based classifiers [50].

(v) *Decision Template Method.* The main concept of decision template is to compare a prototypical answer of ensemble for prediction of class of a given instance. The method may use different similarity measure to evaluate the matching between matrix of classifiers output and matrix of templates. The method can be applied to combine multiple classifier predictions to determine ensemble prediction [78].

(vi) *Metalearning Method.* The method employs a second level of combiner to fuse the predictions of base classifiers for determining final ensemble prediction, for example, stacking. In stacking, the predictions of base classifiers are fed to an intermediate combiner to perform trained combinations of predictions of base classifiers [79]. Another example of metalearning method is to use an arbiter or a combiner to finalize recursively in a hierarchically structured input space on the basis of the predictions made by the base classifiers. The objective of this method is to offer an alternative classification when base classifiers disagree (arbiter tree) or to combine the predictions of base classifiers by learning their relationships with the correct class labels (combiner trees) [80, 81].

(vii) *Hierarchically Structured Method.* The methods are in general characterized by a two step approach: first, at learning of the classes as a set of independent classification problems; second, at combination of the predictions by exploiting the associations between classes that describe the hierarchy. These methods ensure an increment in precision with respect to other flat methods, but this is achieved at cost of the overall recall [82, 83].

(viii) *Boolean Combination (BC) Methods.* Boolean functions especially the conjunction AND and disjunction OR operations have recently been investigated to combine predictions of different classifiers within the ROC space [2]. These methods were shown to improve performance. The methods are based upon assumption that the classifiers are conditionally independent and their respective ROC is smooth and proper. Khreich et al. [2] proposed an iterative Boolean combination (IBC) method to efficiently fuse the predictions from multiple classifiers. The IBC efficiently exploits all Boolean functions applied to the ROC curves and requires no prior assumptions about conditional independence of classifiers or convexity of ROC curves.

4.1.2. Selection-Based Combination Methods. These methods try to choose the best classifiers among the set of the available base classifiers. The final prediction of ensemble is the prediction of selected base classifier or fused prediction of subset of base classifiers as described in aforementioned text. In order to design selection-based ensemble classifier, individual base classifier and its competence on specific input and selection approach must be decided [46]. Important methods proposed in the literature are described in the following section.

(i) *The Test and Select Method.* This method describes a greedy method which adds a new classifier to ensemble if it reduces the squared error [71]. The method can be assisted by different optimization approaches like genetic algorithms [84].

(ii) *Cascading Classifiers Method.* In this method, different base classifiers are employed sequentially to unclassified instance and confidence level of first classifier is recorded. If its level is high enough then its prediction is the ensemble final prediction. Otherwise prediction of next available base classifier is required. This process is recursively repeated [85].

(iii) *Dynamic Classifier Selection Method.* This method measure the competence of each base classifier to determine the prediction of ensemble classifier. The competence of base classifier can be determined dynamically either by using prior information about base classifiers or by posterior information produced by them in terms of their predictions [86, 87]. Limitation of this method is that the measurement of competence of base classifiers is computationally expensive.

(iv) *Clustering-Based Selection Method.* This method employs clustering technique to search subset of base classifiers which perform similar predictions about the unclassified instance. Then the method selects a model from each cluster to select subset of available base classifiers. These methods also help to improve diversity of ensemble [17, 45, 88]. Many researchers also used these methods to reduce the false alarms by correlating similar alarms [18, 88]. These methods are employed to analyze root cause of false alarms [17].

(v) *Statistical Selection Method.* Statistical method can be employed to heterogeneous ensemble. The method selects those base classifiers which perform better performance than others. Then the method combines the selected base classifiers through majority voting method (described in previous section) [89].

4.1.3. Mixture of Expert Systems. This method is a general method similar to ensemble selection [90]. In this method, the recombination of the base classifiers is governed by a supervisor classifier. Supervisor classifier selects the most suitable member of the ensemble on the basis of the available input data. Two additional components are incorporated in mixture of expert's model: (1) a gating network; (2) selector. Gating network receives the same input vectors as the classifiers in the ensemble, but its function is to calculate probabilities for each classifier as to how competent they are

to classify the given input. These probabilities, accompanied by the predictions of each of the classifiers, are passed on to the selector, which then determines the final output. These probabilities can be used to stochastically select the expert, or to choose the expert according to a winner-takes-all paradigm, or as weights to combine the outputs of the multiple base classifiers [33, 46].

Discussion 1. The different methods cited in the above section can be summarized in Table 1. Fusion is the simples and popular method to combine different base classifiers. This method works on the assumption that all base classifiers are of same importance but practically it may not be true, whereas in selection method, generally, only one classifier is chosen to label an unclassified instance. Thus selection method requires that the ensemble classifier is further trained to obtain mechanisms for deciding which base classifier should be chosen to label a given unclassified instance [33]). Selection is guaranteed by design to give at least the same training accuracy as the best individual classifier. However, the model might overtrain, giving a deceptively low training error. To guard against overtraining we may use confidence intervals and nominate a classifier only when it is significantly better than the others [46].

4.2. Classifier Level. This level focuses on ensemble selection phase of ensemble learning process. It determines which base classifiers are used to constitute the ensemble prediction. Many researchers investigated the combination of base classifiers at this level very advantageous particularly for ID [23, 28–30, 32–44]. It is supported due to fact that different base classifiers perform differently upon different categories of intrusions (e.g., DoS, Probe, U2R, R2L, etc.). Selection of base classifiers may be done from pool of classifiers, which are trained using different induction algorithms (called heterogeneous ensembles) or the same induction algorithm (called homogeneous ensembles). Many researcher generated ensemble by selecting heterogeneous base classifiers [23, 24, 27–29, 36]. For example, Mukkamalla et al. [27] studied the SVM, ANNs (artificial neural networks), LGPs (linear genetic programs), and MARSs (Multivariate Adaptive Regression Splines) for classification of KDD dataset into five classes. As these classifiers obtained better performance over the others to detect different classes of intrusion in terms of detection accuracy, attack severity, training & testing time (scalability). The author reported that classifier combination improves the performance of system. Similar approach of multiclassifier systems is also advocated by Sabhnani and Serpen [23] by combining three different machine learning techniques, namely, an ANN, k-means clustering, and a Gaussian classifier. However, they do not provide further implementation details about training the classifiers, nor about determining output of the ensemble. However, they proved that the classifier combination approach improved the classification rates. Many techniques generate a pool of homogeneous base classifiers, for examples, genetic algorithms. Although in ensembles, the combined knowledge is important, it is very computationally expensive to combine a large number of classifiers from the whole population [33]. So, efficient selection and combination of smaller set of base classifiers help to reduce computational overhead without significant lose in the performance. The smaller subset of classifiers may be selected according to clustering [91] or by selecting the classifiers whose performance exceeds specific threshold values. However, in the general literature on classifier combination, it is observed that there is no evidence supporting the use of base classifiers of the same type or different types [33, 46].

4.3. Feature Level. This level focuses on ensemble generation phase of ensemble learning process. Here, pool of classifiers is generated by using different feature subsets of dataset for the training of the base classifiers. Basic reason behind this level is to improve the computational efficiency of the ensemble and to increase the accuracy [46]. By reducing the number of input features of the base classifiers, we can gap the effects of the classical curse of dimensionality problem that characterize high-dimensional and sparse data [102]. Many feature selection techniques for ensemble classifiers are proposed in the literature which can be further investigated in [15, 16, 103].

4.4. Data Level. This level focuses on ensemble generation phase of ensemble learning process. Here, different data subsets are used to train the pool of base classifiers. This level decides which data subset is used to train each base classifier. Most of popular ensemble methods proposed and implemented in the literature utilize data level. These methods are used to generate different training sets and a learning algorithm, which can be applied to the obtained subsets of data in order to produce multiple hypotheses. Various methods have been proposed in literature as described below.

(i) *Bagging.* Bagging (bootstrap aggregating) is originally proposed by Breiman [61]. The method is dependent on the instability of the base classifiers. The instability of base classifiers refers to sensitivity to configuration of base classifier and/or training data. Bagging creates individual classifiers for its ensemble by training each classifier on a random redistribution of the training dataset. Each classifier's training set is generated by randomly drawing, with replacement, N examples—where N is the size of the original training dataset; many of the original examples may be repeated in the resulting training set while others may be left out. Each individual classifier in the ensemble is generated with a different random sampling of the training set. Final prediction of ensemble is generated by fusing different predictions of individual base classifiers. Generally fusion of predictions is performed by using majority voting method. However, this is not always possible due to the size of the dataset. Therefore, the different training subsets are sampled with replacement (bootstrap replicates) from the original training set. Bagging works well if classifier predictions of base classifiers were independent and classifiers had the same individual accuracy, and then

Table 1: Summary of ensembles.

Optimization level	Ensemble learning phase	Ensemble level	Strategy adopted	Method employed
Decision optimization	Ensemble integration	Combination level	Fusion	Majority voting method [70–72]
				Threshold plurality vote method [73]
				Naïve Bayes method [74, 75]
				Fuzzy theory method [76, 77]
				Decision template method [78]
				Metalearning method [79]
				Hierarchically structured method [82, 83]
			Selection	Boolean combination method [2]
				The test and select method [71]
				Cascading classifiers method [85]
				Dynamic classifier selection method [86, 87]
				Clustering-based selection method [17, 45, 88, 91]
				Statistical selection method [89]
			Mixture of expert systems	Stochastic selection method [46]
				Winner-takes-all method [46]
				Weighting method [46]
Coverage optimization	Ensemble selection	Classifier level	Homogenous	Clustering-based selection method [17, 45, 88, 91]
				Threshold-based selection method [86]
			Heterogeneous	—
	Ensemble generation	Feature level	Feature selection/reduction	Random subspace method [46]
				The input decimation method [90]
				Genetic algorithms [92]
				Markov blanket BN [28]
				Principal component analysis [93]
				Information theory [16]
		Data level	Resampling	Bagging [61]
				Wagging [94]
				Random forest [95]
				Boosting [96]
				Stacking [79]
			Output code method	One per class (OPC) [97]
				Pairwise coupling [98]
				Correcting classifiers [99]
				Pairwise coupling correcting classifiers [99]
				Error-correcting output coding [100]
				Data-driven ECOC [101]

the majority vote is guaranteed to improve on the individual performance [46].

(ii) *Wagging*. Wagging method is a variant of bagging. This method based on a nonuniform probability to extract instances from the training dataset [94]. While in bagging each instance is drawn with equal probability from the available training dataset, in wagging each instance is extracted according to a weight stochastically assigned.

(iii) *Random Forest (RF)*. This method is a version of bagging which comprised of decision trees (DTs) [95]. Just like bagging, each DT is trained on different random sampling from dataset, or by sampling from the feature set, or from both. The predictions are combined by a majority vote. The performance of RF is comparable to AdaBoost, but is more robust to noise [57, 95].

(iv) *Boosting*. Boosting [96] is popular meta-algorithm for generating ensemble [33]. It is a meta-algorithm which can be viewed as a model averaging method and one of the most powerful learning ideas introduced in the last twenty years [104]. In this method, the ensembles are populated one classifier at the time. Each classifier is trained on selective subset of data from the original dataset. For the first base classifier, the data is selected uniformly. For successive classifiers, the sampling distribution is continuously updated so that instances that are more difficult to classify are selected more often than those that are easy to classify. This method places the highest weight on the examples most often misclassified by the previous base classifier. In this way the base classifier focuses on the hardest instances. Then the boosting algorithm combines the base rules taking a weighted majority vote of the base classifiers which are based on the accuracy of the classifiers [46].

Major difference in bagging and boosting is that in bagging, the resampling of the training set is not dependent on the performance of the earlier classifiers, whereas Boosting attempts to produce new classifiers that are better able to predict examples for which the current ensemble's performance is poor.

(v) *Stacking*. This (also called stacked generalization) is a way of combining multiple classifiers using the concept of a metalearner [79]. Unlike bagging and boosting, stacking may be utilized to combine classifiers of different types. The method involves the following steps: (1) split the training dataset into two disjoint subsets; (2) train several base classifiers on the first part; (3) test the base classifier on the second part; (4) using the predictions from step (3) as the inputs, and the correct responses as the outputs, train a higher level classifier. Note that steps (1) to (3) are the same as cross-validation, but instead of using a winner-takes-all approach, the base learners are combined, possibly non linearly.

(vi) *Output Code Method*. Output code method works by manipulating the coding of classes in multiclass classification problems. Here ensembles are designed to partially correct errors performed by the base classifiers by exploiting the redundancy in the bit-string representation of the classes [25, 105]. More correctly, output coding (OC) methods decompose a multiclass problem in a set of two-class subproblems and then recompose the original problem combining them to achieve the class label. An equivalent way of thinking about these methods consists in encoding each class as a bit string (named codeword) and in training a different two-class base classifier in order to separately learn each codeword bit. When the classifiers are applied to classify new points, a suitable measure of dissimilarity between the codeword computed by the ensemble and the codeword classes is used to predict the class (e.g., Hamming distance) [100]. Various decomposition schemes have been proposed in the literature: in the one-per-class (OPC) decomposition [97], pairwise coupling (PWC) decomposition [98], the correcting classifiers (CC) and the pairwise coupling correcting classifiers (PWC-CC) [99]. Error-correcting Output Coding (ECOC) [100], and data-driven ECOC [101].

(vii) *Troika*. Troika is an improvement to stacking proposed by Menahem et al. [106]. The method involves three stages to combine the classifiers. In the first stage it combines all base classifiers using specialist classifiers which have a dichotomous model. Second stage contains k metaclassifiers which are used to learn the prediction characteristics of the specialist classifiers. Each metaclassifier is in charge of one class only and will combine all the specialist classifiers which are able to classify their own particular class. The third stage contains only one classifier: the super classifier. The goal of this stage is to produce Troika's final prediction. The inputs of the super classifier are the outputs produced by the metaclassifiers from the previous stage. In the training phase, the super classifier learns the conditions which enable one or more of the metaclassifiers to predict correctly or incorrectly. The super classifier's output is a vector of probabilities (one value for each class) which form the final decision of the Troika ensemble scheme. The authors reported superior performance of Troika over other stacking methods.

Discussion 2. Which is better bagging or boosting? Many researchers compared the two methods including some large-scale experiments [56, 57, 107, 108]. The general consent is that boosting reaches lower testing error. Boosting methods have been crowned as the most accurate available off-the-shelf classifiers on a wide variety of datasets [107]. But, it is observed that boosting methods are sensitive to noise and outliers, especially for small datasets [46, 56, 107]. Bagging is effective with noisy data efforts on noisy data whereas boosting is quite sensitive to noise [57]. Another benefit of bagging methods is that they are parallel in nature in both the training and classification phases, whereas the boosting method is sequential in nature [33].

Details of ensembles are summarized in Table 1.

5. AI Based Ensembles for ID

Many researchers employed AI-based ensembles and hybrid approaches to improve performance of IDS. The focus is

on the combination of classifiers and correlating the alerts to reduce the alarms for network security administrator [3]. Combination of classifiers involves development of ensemble at generation and selection phases of learning, whereas ensemble integration phase involve combination of different predictions of multiple classifiers. In the following paragraphs, we presented important AI-based ensembles studies proposed in the last decade and compared them by various evaluation metrics.

Giacinto and Roli [45] proposed an approach based on multiple classifier systems for ID. The approach is based on the motivation that human experts use different feature sets to detect different kinds of attacks. They generated different neural network-based classifiers by training them using different feature subsets of KDD cup 99 dataset, namely, intrinsic, content, and traffic features. The predictions of trained classifiers are fused together to produce final prediction of ensemble by using the methods like the majority voting rule, the average rule, and the belief function. They found that these multistrategy techniques, particularly the belief function, performed better than all three neural nets individually. The overall performance was also comparable to or better than a single neural net trained on the entire feature set; however, the single neural net did a better job identifying previously unseen attacks. Similar experiments are also performed by Didaci et al. [22].

Sabhnani and Serpen [23] proposed a multi-classifier approach to detect intrusions. They utilized different classifiers, namely, an ANN, k-means clustering, and a Gaussian classifier to classify different classes of intrusions by using KDD 1999 dataset. Multiple classifiers were generated by training from all features of training dataset. The classifiers obtained highest accuracies on different categories of intrusions are used to detect corresponding category of intrusions. They reported that classifier combination results the improvement of classification performance. They reported that probability of detection of 88.7%, 97.3%, 29.8%, and 9.6% with 0.4% false alarm rate for Probe, DoS, U2R, and 0.1% for R2L attack classes, respectively.

Chebrolu et al. [28] proposed a hybrid approach to detect intrusions. They utilized Bayesian networks (BNs) and classification and regression trees (CARTs) and their ensemble to generate hybrid system. They empirically proved that CART performed best for Normal, Probe, and U2R and the ensemble approach worked best for R2L and DoS. The heterogeneous ensemble was generated by training the individual classifiers from reduced KDD cup 99 dataset. In the ensemble approach, the final output was decided as follows: each classifier's output is given a weight (0-1 scale) depending on the generalization accuracy. If both classifiers agree then the output is decided accordingly. If there is a conflict then the decision given by the classifier with the highest weight is taken into account. By using hybrid approach, the authors reported that Normal, Probe, and DOS could be detected with 100% accuracy and U2R and R2L with 84% and 99.47% accuracies, respectively.

Abraham and Thomas [43] proposed an ensemble of DT, SVM, and hybrid system consisting of DT and SVM. The classifiers are generated by using training on KDD99 dataset. They observed in the experiments that different models provided complementary information about the patterns to be classified. The final prediction of ensemble is computed based upon highest score of base classifiers. The score of classifiers is computed by weights assigned according to training performance and their individual predictions. So, for a particular instance to be classified if all of them have different opinions, then their scores are considered. The classifier having the highest score is declared as winner and used to predict the final output of the ensemble. They reported 100% detection of Probe attack class and 99.92%, 68%, and 97.16% detection of DoS, U2R, and R2L attack classes', respectively, using ensemble approach.

Kruegel et al. [109] proposed a multimodel approach that uses a number of different anomaly detection techniques (Bayesian technique) to detect attacks against web servers and web-based applications. The multimodels help to reduce the vulnerability of the detection process with respect to mimicry attacks. The system works by analyzing client queries that reference server side programs. Different models are generated by using a wide range of different features of client queries. The system derives automatically the parameter profiles associated with web applications (e.g., length and structure of parameters) and relationships between queries (e.g., access times and sequences) from the analyzed data. The system takes as input web server log files that conform to the common log format (CLF) and produces an anomaly score for each web request. The task of a model is to assign a probability value to either a query as a whole or one of the query's attributes. This probability value reflects the probability of the occurrence of the given feature value with regards to an established profile. Based on the model outputs, a query is either reported as a potential attack or as normal. This decision is reached by calculating a number of anomaly scores: one for the query itself and one for each attribute. A query is reported as anomalous if at least one of these anomaly scores is above the corresponding detection threshold. The anomaly score is calculated using a weighted sum of model's output and its probability value. The system was tested on data gathered at Google, Inc. and two universities in USA and Europe, showing promising results. However, they used anomaly detection technique (Bayesian technique) to model attribute inputs without taking into account typical semantic differences between classes of characters (alphabetic, numeric, and non-alphanumeric), which usually determine their meaning. Moreover, the authors definitely did not exploit the power of such a model, because they rounded every nonzero probability value to one. Finally, they assumed that the training set is without attacks, by filtering it with a signature-based IDS, in order to throw out at least known attacks.

Similar approach was also proposed by Corona et al. [110]. Here, the authors addressed the problem related to the presence of noise (i.e., attacks) in the training set. The proposed model composed of a set of (independent) application-specific modules. Each module, composed by multiple HMM ensembles, is trained using queries on a specific web application and, during the operational phase, outputs a probability value for each query on this web

application. Furthermore, a decision module classifies the query as suspicious (a possible attack) or legitimate, applying a threshold to this probability value. Thresholds are fixed independently for each application-specific module.

Perdisci et al. [88] proposed a clustering-based fusion module to combine multiple alarms that help to reduce the volume of alarms produced by IDSs. The produced meta-alarms provide the system administrator with a concise high-level description of the attack. They suggested assigning different alarms to predefine set of attack classes, called meta-alarms. Many definitions exist to evaluate similarity between an alarm and a meta-alarm. In fact, the distance between an alarm and a meta-alarm is defined in terms of correlation between them, which is in turn defined as an application of a distance function to features characterizing each of the raised alarms.

Hwang et al. [42] proposed a 3-tier hybrid approach to detect intrusions. First tier of system is a signature-based approach to filter the known attacks using black list concept. Second tier of system is anomaly detector that uses the white list concept to distinguish the normal and attack traffic that has by passed first tier. The third tier component of system uses the SVM to classify the unknown attack traffic into five classes, that is, Normal, Probe, DoS, U2R, and R2L. KDD dataset was used to train and test the system. They claimed 94.71% detection accuracy with 3.8% of false alarm rate of old as well as new attacks.

Chen et al. [41] suggested a hybrid flexible neural-tree-based IDS based on flexible neural tree, evolutionary algorithm, and particle swarm optimization (PSO). They focus on improving the ID performance by reducing the input features and hybrid approaches for combining base classifiers. The classifiers are generated by using different feature subset of training dataset. They proved empirically that result of the proposed method is improved. They performed experiments by using 41 features and 12 features of KDD dataset. They reported 98.39%, 98.75%, 99.70%, and 99.09% detection of Probe, DoS, U2R, and R2L attack classes using 41 features of KDD dataset.

Khan et al. [40] suggested a hybrid of SVM and clustering to cut down the training time. A hierarchical clustering algorithm is engaged to establish boundary points in the data that best separate the two classes. These boundary points are used to train the SVM. This is an iterative process, in which the SVM is trained on every new level of cluster nodes in the tree that is being built. Iteratively, support vectors are calculated and the SVM is tested against a stopping criterion to determine if a desirable threshold of accuracy has been achieved. Otherwise the iterative process continues. The authors reported 91%, 97%, 23%, and 43% detection of Probe, DoS, U2R, and R2L attack classes.

Toosi and Kahani [39] proposed IDS by using neurofuzzy classifiers to classify KDD cup 99 dataset into five classes, namely, Normal, Probe, DoS, U2R, and R2L. The proposed system includes two layers. In the first layer, there are five ANFIS modules which are trained to explore the intrusive activity from the input data. Each ANFIS module belongs to one of the classes in the dataset each providing an output which specifies the degree of relativity of the data to the specific class. Second, a fuzzy inference module, based on empirical knowledge, is employed to make the final decision for recognition. The fuzzy inference module implements nonlinear mappings from the outputs of the neurofuzzy classifiers of the pervious layer to the final output space which specifies if the input data are normal or intrusive. The genetic algorithm is used to optimize the structure of neurofuzzy engine. A great time consuming of the system may be a big problem. The authors reported 84.1%, 99.5%, 14.1%, and 31.5% detection of Probe, DoS, U2R, and R2L attack classes.

Yan and Hao [111] presented ensemble of neural network for ID based upon improved MOGA (improvement of NSGA-II). They used improved MOGA to select relevant feature subsets of dataset. Selected subsets of feature are used to train accurate and diverse base classifiers. Final ensemble is constructed by using ensemble selection method. They reported improvement of ID in terms of detection rate and false-positive rate over other related approaches. The authors reported 98.96%, 99.98%, 99.95%, and 98.51% detection of Probe, DoS, U2R, and R2L attack classes with 0.38%, 0.03%, 0.11%, and 8.91% false-positive rate, respectively.

Xiang et al. [36] proposed a hierarchical hybrid system involving multiple-level hybrid classifier, which combines the supervised decision tree classifiers and unsupervised Bayesian clustering to detect intrusions. It was able to achieve a higher true positive rate than previously reported in the literature on the original training and test sets of the KDD Cup 99 dataset. However, this was at the expense of a higher false-positive rate.

Hu et al. [112] suggested an AdaBoost algorithm-based ensemble which in turn uses decision stump as base classifiers. They utilized continuous and categorical features separately without any forced conversion. The proposed system was evaluated using KDD cup 99 dataset. They reported 90.04%–90.88% of detection rate with false alarm rate of 0.31%–1.79%. The proposed system suffers from limitation of incremental learning. It requires continuous retraining for changing environment.

Cretu et al. [113] proposed a micromodel-based ensemble of anomaly sensors to sanitize the training data. Here, different models are generated to produce provisional labels for each training input, and models are combined in a voting scheme to determine which parts of the training data may represent attacks. The models are trained by partitioning the original training dataset.

Zainal et al. [35] proposed heterogeneous ensemble of linear genetic programming (LGP), adaptive neural fuzzy inference system (ANFIS), and random forest (RF) for ID. Base classifiers are generated by using class-specific features of KDD cup 99 dataset. They utilized rough-discrete particle swarm optimization (Rough-BPSO) to select significant features for specific class. Final ensemble prediction is the weighted voting of base classifiers. They empirically proved that by assigning proper weights to classifiers in ensemble approach improves the detection accuracy of all classes of network traffic than individual classifier.

Menahem et al. [106] proposed metalearning-based approach. They utilized multiple classifiers and tried to

exploit their strengths. They used C4.5 decision tree [114], Naïve Bayes [115], k-NN clustering [116], VFI-voting feature intervals [34], and OneR [117] classifiers as base classifiers over five malware datasets. Each classifier belongs to different family of classifiers. They proposed to partition the original dataset into two subsets. The first subset is reserved to form the metadataset and the second subset is used to build the base-level classifiers. This classifier (Metaclassifier) combines the different predictions into a final one. They improved the classifiers performance by using Troika [106] over other stacking methods. Troika combines base classifiers in three stages: specialists level, metaclassifiers, and super classifier. In order to conclude which ensemble performs best over multiple datasets, they followed the procedure proposed in [118].

Wang et al. [32] proposed an approach, called FC-ANN, based on ANN and fuzzy clustering, to solve the problem and help IDS achieve higher detection rate, less false-positive rate, and stronger stability. They used fuzzy clustering technique to generate different homogeneous training subsets from heterogeneous training set, which are further used to ANN models as base models. Finally, a metalearner, fuzzy aggregation module, was employed to aggregate these results. They reported the improvement of proposed approach over BPNN and other well-known methods such as decision tree and the Naïve Bayes in terms of detection precision and detection stability.

Khreich et al. [2] proposed an iterative Boolean combination (IBC) technique for efficient fusion of the responses from any crisp or soft detector trained on fixed-size datasets in the ROC space. The proposed technique applies all Boolean functions to combine the ROC curves corresponding to multiple classifiers. It requires no prior assumptions, and its time complexity is linear with the number of classifiers. They generated HMMs models as base classifiers by training them using different number of HMM states and random initializations. They applied multiple HMM to dataset and final prediction is computed by exploiting all Boolean functions applied to the ROC curves. At each iteration, the proposed technique selects those combinations that improve the ROC convex hull and recombines them with the original ROC curves until the ROC convex hull ceases to improve. The results of computer simulations conducted on both synthetic (UNM intrusion detection dataset of system calls) and real-world host-based intrusion detection data indicate that the IBC of responses from multiple HMMs can achieve a significantly higher level of performance than the Boolean conjunction and disjunction combinations, especially when training data are limited and imbalanced. However, IBC does not allow to efficiently adapt a fusion function over time when new data becomes available, since it requires a fixed number of classifiers. The IBC technique was further improved as incremental Boolean combination (incrBC) by the authors [119]. The incrBC is a ROC-based system to efficiently adapt ensemble of HMM (EoHMMs) over time, from new training data, according to a learn-and-combine approach without multiple iterations. Given new training data, a new pool of HMMs is generated from newly acquired data using different HMM states and initializations. The responses from these newly trained HMMs are then combined with those of the previously trained HMMs in ROC space using the incremental Boolean combination (incrBC) technique.

Govindarajan and Chandrasekaran [30] presented hybrid architecture of multilayer perceptron and radial basis function and their ensemble for ID. Different ensemble members were generated by training from reduced dataset. The final outputs were decided as follows: each classifier's output is given a weight (0-1 scale) depending on the generalization performance during the training process. If both classifiers agree then the output is decided accordingly. If there was a conflict then the decision given by the classifier with the highest weight is taken into account. They showed that the performance of the proposed method is superior to that of single usage of base classification methods. Additionally it has been found that ensemble of multilayer perceptron is superior to ensemble of radial basis function classifier for normal behavior and reverse is the case for abnormal behavior. They reported that the proposed method provides significant improvement of prediction accuracy in ID.

Muda et al. [120] proposed a combined approach of clustering and classification. The clustering is performed by using K-means algorithm to form groups of similar data in earlier stage. Next, in second stage, clustered data is classified by attack category using Naïve Bayes classifier. They reported the better performance of proposed hybrid approach over single Naïve Bayes classifier over KDD 1999 dataset. But the proposed method suffers from limitation that it is unable to detect similar attacks like U2R and R2L.

These related studies can be compared by following a set of evaluation metrics which derives from: (1) architecture & approach followed; (2) different methods utilized in different phases of ensemble learning; (3) other measures used to evaluate classification performance of the ensembles as depicted in Table 2. Here, architecture of system can be parallel, cascading, or hierarchical [35], the classifiers can be combined by ensemble- or hybrid-combining approach. Ensemble level refers to different levels (combination level, classifier level, feature level, or data level as proposed in [46]) used in different ensemble learning phases (ensemble generation, ensemble selection, and ensemble integration). Diversity among the base classifiers can be measured by implicit or explicit methods [47, 49]. In order to evaluate the performance, different performance metrics can be computed based upon benchmarked datasets.

6. Discussion

Over the past decade, ID based upon ensemble approaches has been a widely studied topic, being able to satisfy the growing demand of reliable and intelligent IDS. In our view, these approaches contribute to intrusion detection in different ways. These approaches combine complementary multiple classifiers. They use combined knowledge to meet the challenges of ID-like high false alarm rate,

Table 2: Comparison of AI based ensembles for ID.

Study	Architecture	Combining approach	Ensemble learning phase and ensemble level			Combining method employed	Metric	Dataset	Diversity	Base classifier
			Generation	Selection	Integration					
Giacinto and Roli [45]	Parallel	Ensemble	Feature level	—	Fusion	Majority voting, average rule, belief function	Error rate, FPR, cost	KDD 99	Implicit	NN
Sabhnani and Serpen [23]	—	Hybrid	—	Classifier level	—	Multi-classifiers method	DR, FPR	KDD 99	—	NN, KM, GC
Chebrolu et al. [28]	Parallel	Ensemble	—	Classifier level	Selection	Weighting method	CA	KDD 99	Implicit	BN, CART
Abraham et al. [43]	Parallel	Ensemble	Feature level	Classifier level	Selection	Weighting method	CA	KDD 99	Implicit	DT, SVM
Kruegel et al. [109]	Parallel	Ensemble	Feature and data level	—	Fusion	Score-and probability-based method	FPR	Real world dataset	Implicit	BN
Perdisci et al. [88]	—	Ensemble	—	—	Fusion	Clustering	—	Real world dataset	—	—
Hwang et al. [42]	Cascading	Hybrid	—	—	—	Consecutive combination	DR, FPR	KDD 99	—	SVM
Chen et al. [41]	Hierarchical	Hybrid	Feature level	—	—	Multi-classifiers method	DR, FNR, FPR	KDD 99	—	FNT
Khan et al. [40]	Cascading	Hybrid	—	—	—	Clustering + classification	CA, training time, FP, FN	KDD 99	—	SVM, clustering
Toosi and kahani [39]	Parallel	Ensemble	—	Classifier level	Fusion	Fuzzy theory method	CA, DR, FPR, CPE	KDD 99	Implicit	NN, fuzzy logic
Yan and Hao [111]	Parallel	Ensemble	Feature level	—	Selection	—	DR, FPR	KDD 99	Implicit	NN
Xiang et al. [36]	Cascading	Hybrid	Data level	Classifier level	—	Clustering + classification	TP, FP	KDD 99	—	DT, BC
Cretu et al. [113]	Parallel	Ensemble	Data level	—	Fusion	Voting method	FP, TP	Real world data	—	Anagram, Payl
Hu et al. [112]	Parallel	Ensemble	Feature level	—	—	Mixture of expert systems	DR, FAR, computation time	KDD 99	Implicit	DS
Corona et al. [110]	Parallel	Ensemble	Feature and data level	—	Fusion	Threshold probability method	FPR, DR	Real world dataset	Implicit	HMM
Zainal et al. [35]	Parallel	Ensemble	Feature level	Classifier level	Fusion	Weighted voting method	CA, TP, FP	KDD 99	Implicit	LGP, ANFIS, RF
Menahem et al. [106]	Parallel	Ensemble	Data level	Classifier level	Fusion	Meta learning	CA, area under the ROC curve, training time	Real-time network traffic	Implicit	DT, NB, K-NN, VFI, OneR
Wang et al. [32]	Parallel	Ensemble	Data level	—	Fusion	Meta learning	Precision, recall, F-measure	KDD 99	Implicit	NN, fuzzy logic, clustering

Table 2: Continued.

Study	Architecture	Combining approach	Ensemble learning phase and ensemble level			Combining method employed	Metric	Dataset	Diversity	Base classifier
			Generation	Selection	Integration					
Khreich et al. [2]	Parallel	Ensemble	—	—	Fusion	Iterative Boolean combination method	ROC space	UNM dataset, real world dataset	Implicit	HMM
Govindarajan and Chandrasekaran [30]	Parallel	Ensemble	Data level	—	Fusion	Weighted method	CA	Immune system dataset from University of New Mexico	Implicit	MLP, RBF
Muda et al. [120]	Cascading	Hybrid	Data level	—	—	Clustering + classification	CA, DR, FPR	KDD 99	—	KM, NB

Abbreviations—NN: neural network; KM: K-means clustering; GC: Gaussian classifier, BN: Bayesian network; CART: classification and regression trees; DT: decision tree; SVM: support vector machine; FNT: fuzzy neural tree; BC: bayesian clustering; DS: decision stump; LGP: linear genetic programming; ANFIS: adaptive neural fuzzy inference system; RF: random forest; NB: Naïve Bayes; K-NN: K-nearest neighbor; VFI: voting feature intervals; MLP: multilayer perceptron; RBF: radial basis function; HMM: hidden Markov model.

low detection accuracy, and better performance in lack of sufficient amount of quality training dataset. The results of ensemble approach are proved to be improved than single best classifier. Researchers focused heterogeneous as well homogeneous ensembles. Heterogeneous ensembles exploit the characteristics of different classifiers to improve the results over single classifier. It is supported by the fact that different base classifiers perform differently upon different categories of intrusions (e.g., DoS, Probe, U2R, R2L, etc.) [8]. The performance variation of various classifiers for different intrusions may be described by two aspects. The first aspect is the different design principles of classifiers which work to optimize different parameters. For example, SVM is designed to minimize structural risk based upon statistical theory whereas ANN to minimize empirical risk in which classification function is derived by minimizing the mean square error over the training dataset. The second aspect is that detection of intrusions depends upon specific features of dataset. But, availability of irrelevant and redundant feature affects detection performance of classifiers. Homogeneous ensembles focus on different features of training dataset and/or different training subsets and/or other ways to generate diverse base classifiers. Applications of the AI-based ensembles revealed that they have pros and cons. Hence, ensemble approach couples the base classifiers together in a way that they supplement each other favorably. The resulting synergy has been shown to be an effective way for building IDSs with improved performance in terms of detection accuracy and false-positive rate. It is observed that successful employment of ensemble for ID depends upon many factors including size of training dataset, modification of dataset for training of different base classifiers, choice of accurate and diverse base classifiers, ability of base classifiers to detect different intrusions, and choice of level for generating ensemble of base classifiers, for example, combination, classifier, feature, or data level. It may be concluded that by considering appropriate base classifiers, training sample size & combination method, the performance of hybrid classifier/ensemble can be improved.

We compared the related AI-based ensemble studies for ID in terms of variety of aspects as shown in Table 2. Majority of research works described here were trained & tested on KDD cup 1999 dataset. Since these works are evaluated in different environments using different training and test datasets extracted from KDD cup 1999 dataset, these studies cannot be critically analyzed based upon these reported results. But, it is clear from results presented in Section 4 that all researches did not perform well for minority attack classes of U2R and R2L attacks. The reason may be either class imbalance in training dataset or 11 attack types in these two classes only appear in the test dataset, not the training set, and they constitute more than 50% of the data. However, ensembles utilized the combined knowledge of multiple classifiers to improve the performance in these minority attacks classes. But, still there is a need to generate diverse set of base classifiers that perform well on majority and minority attack classes. Many researchers proposed use of population-based approaches to generate diverse set of classifiers.

Although some promising results have been achieved by current AI-based ensembles to IDSs, there are still challenges that lie ahead for researchers in this area. First and foremost, high-quality benchmark datasets for network intrusion detection are needed. The KDD 99 derived from DARPA 1998 & 1999 datasets are main benchmarks used to evaluate the performance of network intrusion detection systems. However, they are suffering from a fatal drawback: failing to realistically simulate a real-world network [102, 121]. An IDS trained and tested on these datasets may demonstrate unacceptable performance in real environments. In order to validate the evaluation results of IDS on a simulated dataset, one has to develop a methodology to quantify the similarity of simulated and real network traces. KDD cup 1999 dataset and its original form possess some special features, such as huge volume, high dimension, and highly skewed data distribution. Similar properties are not generally found in other benchmark dataset, so they have been usually used for challenging and evaluating learning algorithms in both supervised and unsupervised mode. However, this purpose of dataset is also under criticism [121]. The major criticisms are that DARPA datasets include irregularities, like differences in the TTL for attacks versus normal traffic, so that even a basic IDS could attain an excellent performance [121] and the KDD99 training and test datasets have dissimilar target hypotheses for U2R and R2L classes [23]. Therefore, using these datasets only is not sufficient to reveal the efficiency of a learning algorithm. So, new and good quality benchmark datasets need to be developed for realistic evaluation of IDS. While developing new dataset, payload, and temporal locality information along with header information may be considered and beneficial for realistic evaluation of IDS.

Second challenge to tackle in AI-based ensembles is the enormous amount of audit data that make it difficult to build effective IDS. The processing of enormous amount of data increases computational overhead and causes delay in detection of intrusions. The delay in detection of intrusions leads to loss of real-time capability of IDS. Many researchers suggested the use of feature selection/reduction techniques [16, 122]. These techniques help remove irrelevant and redundant features and identify appropriate features for intrusion detection. The reduction in features reduces the amount of audit data for effective IDS. A focus on feature reduction/selection technique is highly recommended to reduce computational overhead of ensembles. Some researchers proposed to use a distributed environment in which each node is assigned a part of dataset. An ensemble method was used to fuse or select predictions.

Thirdly, an important feature of IDS is the capability to adapt to dynamic behavior of intrusive and normal traffic. If the IDSs are not flexible enough to cope with behavioral changes, detection performance will noticeably decline. AI-based techniques and their ensembles can help to address this important issue, but still only a small number of researchers have focused it so far.

Most of the methods discussed in this paper have their roots in the ensemble learning process. The process has three phases, namely, generation of base classifiers,

selection of classifiers from base classifiers, and integration of different predictions from selected classifiers. The paper clearly shows that some researchers have applied their knowledge to address different issues at these phases for intrusion detection. But still there is a need to focus more on issues of each phase of ensemble learning. It is expected that new discoveries and a deepened understanding of different techniques suitable for different phases in ensemble learning of ID problem will be the subject of future work.

7. Conclusive Remarks

AI-based techniques and their ensembles are presently attracting considerable attention from the research community for intrusion detection. Their features, such as flexibility, adaptability, new pattern recognition, fault tolerance, learning capabilities, high computational speed, and error resilience for noisy data, fit the prerequisite of building effective IDS. Ensemble approach imitates our second nature to look for several opinions before making an essential decision. The basic principle is to evaluate several individual pattern classifiers, and integrate them in order to reach a classification that is better than the one obtained by each of them separately.

Our overview focused on supervised AI-based ensemble for intrusion detection proposed in last decade, since historically these were the first to be studied and applied to several application domains. More precisely, in this paper a general taxonomy, distinguishing between decision and coverage optimization ensembles, important AI-based ensembles for intrusion detection proposed in last decade has been described, considering the different ways supervised base classifiers can be generated or combined together.

However, the practice of AI-based classifiers reveals that each of them has advantages and disadvantages for intrusion detection. Ensemble has the power to combine the strengths of these classifiers in such a way that their disadvantages will be compensated, thus offering better solutions. We therefore included ensemble learning as a topic in this paper. The findings of research work in each study are systematically summarized and compared, which allows us to clearly identify existing research challenges for intrusion detection and underline research directions. It is expected that this paper can serve as a practical channel through the maze of the literature.

References

[1] J. McCumber, "Information system security: a comprehensive model," in *Proceedings of the 14th National Computer Security Conference*, Baltimore, Md, USA, 1991.

[2] W. Khreich, E. Granger, A. Miri, and R. Sabourin, "Iterative Boolean combination of classifiers in the ROC space: an application to anomaly detection with HMMs," *Pattern Recognition*, vol. 43, no. 8, pp. 2732–2752, 2010.

[3] I. Corona, G. Giacinto, C. Mazzariello, F. Roli, and C. Sansone, "Information fusion for computer security: state of the art and open issues," *Information Fusion*, vol. 10, no. 4, pp. 274–284, 2009.

[4] S. Axelsson, "Research in intrusion detection system—a survey," Tech. Rep. CMU/SEI, 1999.

[5] G. Kumar, K. Kumar, and M. Sachdeva, "The use of artificial intelligence based techniques for intrusion detection—a review," *Artificial Intelligence Review*, vol. 34, no. 4, pp. 369–387, 2010.

[6] C. Kruegel, F. Valeur, and G. Vigna, *Intrusion Detection and Correlation, Challenges and Solution, Advances in Information Security*, Springer, 2005.

[7] R. Caruana and A. Niculescu-Mizil, "Data mining in metric space: an empirical analysis of supervised learning performance criteria," in *Proceedings of the 10th ACM SIGMOD International Conference on Knowledge Discovery and Data Mining (KDD-2004)*, pp. 69–78, ACM Press, August 2004.

[8] G. Kumar and K. Kumar, "AI based supervised classifiers an analysis for intrusion detection," in *Proceedings of the International Conference on Advances in Computing and Artificial Intelligence (ACAI '11)*, pp. 170–174, ACM Digital Library, Chitkara, India, July 2011.

[9] J. W. Haines, R. P. Lippmann, D. J. Fried, E. Tran, S. Boswell, and M. A. Zissman, "DARPA intrusion detection system evaluation: design and procedures," Tech. Rep., MIT Lincoln Laboratory, 1999.

[10] KDDCup, "The Third International Knowledge Discovery and Data Mining Tools Competition," 1999, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[11] UNM dataset, http://www.cs.unm.edu/immsec/systemcalls.htm.

[12] DEFCON 9, http://ictf.cs.ucsb.edu/data/defcon_ctf_09/.

[13] ITOC dataset, http://www.itoc.usma.edu/research/dataset/.

[14] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln laboratory," *ACM Transactions on Information and System Security*, vol. 3-4, pp. 262–294, 2000.

[15] G. Kumar, K. Kumar, and M. Sachdeva, "An empirical comparative analysis of feature reduction methods for intrusion detection," *International Journal of Information and Telecommunication*, vol. 1, pp. 44–51, 2010.

[16] G. Kumar and K. Kumar, "An information theoretic approach for feature selection," *Security and Communication Networks*, vol. 5, pp. 178–185, 2012.

[17] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Transactions on Information and System Security*, vol. 6, no. 4, pp. 443–471, 2003.

[18] H. Debar and A. Wespi, "Aggregation and correlation of intrusion-detection alerts, recent advances in intrusion detection," *Lecture Notes in Computer Science*, vol. 2212, pp. 85–103, 2001.

[19] A. Patcha and J. M. Park, "An overview of anomaly detection techniques: existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[20] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[21] M. C. Ponce, "Intrusion detection system with artificial intelligence," in *Proceedings of the FIST Conference*, Universidad Pontificia Comillas de Madrid, 2004, edition: 1/28 .

[22] L. Didaci, G. Giacinto, and F. Roli, "Ensemble learning for intrusion detection in computer networks," in *Proceedings*

of the 8th Conference of the Italian Association of Artificial Intelligence (AIAA '02), Siena, Italy, 2002.

[23] M. Sabhnani and G. Serpen, "Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context," in *Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications (MLMTA '03)*, pp. 209–215, June 2003.

[24] M. Panda and M. R. Patra, "A comparative study of data mining algorithms for network intrusion detection," in *Proceedings of the 1st International Conference on Emerging Trends in Engineering and Technology (ICETET '08)*, pp. 504–507, IEEE Computer Society, July 2008.

[25] T. G. Dietterich, "Ensemble methods in machine learning," in *Proceedings of the Multiple Classifier Systems. First International Workshop (MCS '00)*, J. Kittler and F. Roli, Eds., vol. 1857 of *Lecture Notes in Computer Science*, pp. 1–15, Cagliari, Italy, 2000.

[26] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[27] S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms," *Journal of Network and Computer Applications*, vol. 28, no. 2, pp. 167–182, 2005.

[28] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Computers and Security*, vol. 24, no. 4, pp. 295–307, 2005.

[29] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems," *Journal of Network and Computer Applications*, vol. 30, no. 1, pp. 114–132, 2007.

[30] M. Govindarajan and R. M. Chandrasekaran, "Intrusion detection using neural based hybrid classification methods," *Computer Networks*, vol. 55, no. 8, pp. 1662–1671, 2011.

[31] C. Langin and S. Rahimi, "Soft computing in intrusion detection: the state of the art," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, no. 2, pp. 133–145, 2010.

[32] G. Wang, H. Jinxing, M. Jian, and H. Lihua, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6225–6232, 2010.

[33] V. Engen, *Machine learning for network based intrusion detection [Ph.D. thesis]*, Bournemouth University, June 2010.

[34] G. D. Guvenir, "Classification by voting feature intervals," in *Proceedings of the European Conference on Machine Learning*, pp. 85–92, 1997.

[35] A. Zainal, M. A. Maarof, and S. M. Shamsuddin, "Ensemble classifiers for network intrusion detection system," *Journal of Information Assurance and Security*, vol. 4, pp. 217–225, 2009.

[36] C. Xiang, P. C. Yong, and L. S. Meng, "Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees," *Pattern Recognition Letters*, vol. 29, no. 7, pp. 918–924, 2008.

[37] N. B. Anuar, H. Sallehudin, A. Gani, and O. Zakari, "Identifying false alarm for network intrusion detection system using hybrid data mining and decision tree," *Malaysian Journal of Computer Science*, vol. 21, no. 2, pp. 101–115, 2008.

[38] F. Gharibian and A. A. Ghorbani, "Comparative study of supervised machine learning techniques for intrusion detection," in *Proceedings of the 5th Annual Conference on Communication Networks and Services Research (CNSR '07)*, pp. 350–358, Washington, DC, USA, May 2007.

[39] A. N. Toosi and M. Kahani, "A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers," *Computer Communications*, vol. 30, no. 10, pp. 2201–2212, 2007.

[40] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," *The International Journal on Very Large Data Bases*, vol. 16, no. 4, pp. 507–521, 2007.

[41] Y. Chen, A. Abraham, and B. Yang, "Hybrid flexible neural-tree-based intrusion detection systems," *International Journal of Intelligent Systems*, vol. 22, no. 4, pp. 337–352, 2007.

[42] T. S. Hwang, T.-J. Lee, and Y.-J. Lee, "A three-tier IDS via data mining approach," in *Proceedings of the 3rd Annual ACM Workshop on Mining Network Data (MineNet '07)*, pp. 1–6, June 2007.

[43] A. Abraham and J. Thomas, "Distributed intrusion detection systems: a computational intelligence approach," in *Applications of Information Systems to Homeland Security and Defense*, H. Abbass and D. Essam, Eds., pp. 105–135, Idea Group, New York, NY, USA, 2005, chapter 5.

[44] Z. S. Pan, S. C. Chen, G. B. Hu, and D. Q. Zhang, "Hybrid neural network and C4.5 for misuse detection," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, pp. 2463–2467, November 2003.

[45] G. Giacinto and F. Roli, "An approach to the automatic design of multiple classifier systems," *Pattern Recognition Letters*, vol. 22, no. 1, pp. 25–33, 2001.

[46] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, Wiley-Interscience, New York, NY, USA, 2004.

[47] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: a survey and categorisation," *Journal of Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[48] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[49] E. K. Tang, P. N. Suganthan, and X. Yao, "An analysis of diversity measures," *Machine Learning*, vol. 65, no. 1, pp. 247–271, 2006.

[50] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.

[51] W. Leigh, R. Purvis, and J. M. Ragusa, "Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support," *Decision Support Systems*, vol. 32, no. 4, pp. 361–377, 2002.

[52] A. C. Tan, D. Gilbert, and Y. Deville, "Multi-class protein fold classification using a New Ensemble Machine Learning Approach," *Genome Informatics*, vol. 14, pp. 206–217, 2003.

[53] P. Mangiameli, D. West, and R. Rampal, "Model selection for medical diagnosis decision support systems," *Decision Support Systems*, vol. 36, no. 3, pp. 247–259, 2004.

[54] R. Moskovitch, Y. Elovici, and L. Rokach, "Detection of unknown computer worms based on behavioral classification of the host," *Computational Statistics and Data Analysis*, vol. 52, no. 9, pp. 4544–4566, 2008.

[55] R. P. W. Duin, "The combining classifier: to train or not to train?" in *Proceedings of 16th International Conference on Pattern Recognition (ICPR' 02)*, pp. 765–770, Quebec City, Canada, 2002.

[56] E. Bauer and R. Kohavi, "Empirical comparison of voting classification algorithms: bagging, boosting, and variants," *Machine Learning*, vol. 36, no. 1, pp. 105–139, 1999.

[57] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139–157, 2000.

[58] R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "A comparison of decision tree ensemble creation techniques," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 173–180, 2007.

[59] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," *Journal of Machine Learning Research*, vol. 1, no. 2, pp. 113–141, 2001.

[60] E. M. Kleinberg, "On the algorithmic implementation of stochastic discrimination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 5, pp. 473–490, 2000.

[61] L. Breiman, "Bias, variance and arcing classifiers," Tech. Rep. TR 460, Statistics Department, University of California, Berkeley, Calif, USA, 1996.

[62] R. Hu and R. I. Damper, "A "No Panacea Theorem" for classifier combination," *Pattern Recognition*, vol. 41, no. 8, pp. 2665–2673, 2008.

[63] A. Sharkey, "Types of multi-ney systems," in *Multiple Classifier Systems, Third International Workshop (MCS '02)*, F. Roli and J. Kittler, Eds., vol. 2364 of *Lecture Notes in Computer Science*, pp. 108–117, 2002.

[64] L. Rokach, "Taxonomy for characterizing ensemble methods in classification tasks: a review and annotated bibliography," *Computational Statistics and Data Analysis*, vol. 53, no. 12, pp. 4046–4072, 2009.

[65] M. S. Kamel and N. M. Wanas, "Data dependence in combining classifiers," in *Proceedings of 4th International Workshop on Multiple Classifier Systems (MCS '03)*, T. Windeattand and F. Roli, Eds., vol. 2709 of *Lecture Notes in Computer Science*, pp. 1–14, Guildford, UK, 2003.

[66] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, The Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann, San Francisco, Calif, USA, 2nd edition, 2005.

[67] C. M. Bishop, *Pattern Recognition and Machine Learning, Information Science and Statistics*, Springer, New York, NY, USA, 2006.

[68] S. Marsland, *Machine Learning: An Algorithmic Perspective*, Chapman & Hall/CRC Machine Learning & Pattern Recognition, CRC Press, Boca Raton, Fla, USA, 2009.

[69] E. Alpaydin, *Introduction to Machine Learning, Adaptive Computation and Machine Learning*, The MIT Press, Cambridge, Mass, USA, 2nd edition, 2010.

[70] F. Kimura and M. Shridhar, "Handwritten numerical recognition based on multiple algorithms," *Pattern Recognition*, vol. 24, no. 10, pp. 969–983, 1991.

[71] M. P. Perrone and L. N. Cooper, "When networks disagree: ensemble methods for hybrid neural networks," in *Artificial Neural Networks for Speech and Vision*, R. J. Mammone, Ed., pp. 126–142, Chapman & Hall, London, UK, 1993.

[72] L. Lam and C. Y. Suen, "Application of majority voting to pattern recognition: an analysis of its behavior and performance," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 27, no. 5, pp. 553–568, 1997.

[73] L. Xu, A. Krzyzak, and C. Y. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[74] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," *Machine Learning*, vol. 29, no. 2-3, pp. 103–130, 1997.

[75] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, John Wiley & Sons, New York, NY, USA, 2nd edition, 2001.

[76] S. B. Cho and J. H. Kim, "Combining multiple neural networks by fuzzy integral for robust classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, no. 2, pp. 380–384, 1995.

[77] A. Verikas, A. Lipnickas, K. Malmqvist, M. Bacauskiene, and A. Gelzinis, "Soft combination of neural classifiers: a comparative study," *Pattern Recognition Letters*, vol. 20, no. 4, pp. 429–444, 1999.

[78] M. Re and G. Valentini, "Integration of heterogeneous data sources for gene function prediction using decision templates and ensembles of learning machines," *Neurocomputing*, vol. 73, no. 7–9, pp. 1533–1537, 2010.

[79] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[80] P. K. Chan and S. J. Stolfo, "On the accuracy of meta-learning for scalable data mining," *Journal of Intelligent Information Systems*, vol. 8, no. 1, pp. 5–28, 1997.

[81] T. Hothorn and B. Lausen, "Bundling classifiers by bagging trees," *Computational Statistics and Data Analysis*, vol. 49, no. 4, pp. 1068–1078, 2005.

[82] Y. Guan, C. L. Myers, D. C. Hess, Z. Barutcuoglu, A. A. Caudy, and O. G. Troyanskaya, "Predicting gene function in a hierarchical context with an ensemble of classifiers," *Genome Biology*, vol. 9, supplement 1, article S3, 2008.

[83] G. Obozinski, G. Lanckriet, C. Grant, M. I. Jordan, and W. S. Noble, "Consistent probabilistic outputs for protein function prediction," *Genome Biology*, vol. 9, supplement 1, article S6, 2008.

[84] W. B. Langdon and B. F. Buxton, "Genetic programming for improved receiver operating characteristics," in *Proceedings of the 2nd International Conference on Multiple Classifier System*, J. Kittler and F. Roli, Eds., pp. 68–77, Cambridge, UK, 2001.

[85] E. Alpaydin and C. Kaynak, "Cascading classifiers," *Kybernetika*, vol. 34, no. 4, pp. 369–374, 1998.

[86] G. Giacinto and F. Roli, "Dynamic classifier fusion," in *Proceedings of the Multiple Classifier Systems. First International Workshop (MCS '00)*, J. Kittler and F. Roli, Eds., vol. 1857 of *Lecture Notes in Computer Science*, pp. 177–189, Springer, Cagliari, Italy, 2000.

[87] E. M. Dos Santos, R. Sabourin, and P. Maupin, "A dynamic overproduce-and-choose strategy for the selection of classifier ensembles," *Pattern Recognition*, vol. 41, no. 10, pp. 2993–3009, 2008.

[88] R. Perdisci, G. Giacinto, and F. Roli, "Alarm clustering for intrusion detection systems in computer networks," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 4, pp. 429–438, 2006.

[89] G. Tsoumakas, L. Angelis, and I. Vlahavas, "Selective fusion of heterogeneous classifiers," *Intelligent Data Analysis*, vol. 9, no. 6, pp. 511–525, 2005.

[90] R. A. Jacobs, "Methods for combining experts' probability assessments," *Neural Computation*, vol. 7, no. 5, pp. 867–888, 1995.

[91] X. Yao and M. Md. Islam, "Evolving artificial neural network ensembles," *IEEE Computational Intelligence Magazine*, vol. 3, pp. 31–42, 2008.

[92] M. Y. Su, K. C. Chang, H. F. Wei, and C. Y. Lin, "Feature weighting and selection for a real-time network intrusion detection system based on GA with KNN," *Intelligence and Security Informatics*, vol. 5075, pp. 195–204, 2008.

[93] J. Xiao and H. Song, "A novel intrusion detection method based on adaptive resonance theory and principal component analysis," in *Proceedings of the International Conference on Communications and Mobile Computing (CMC '09)*, pp. 445–449, January 2009.

[94] R. Valentini, *Ensemble Methods: A Review*, CRC press, 2001.

[95] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[96] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the 30th International Conference on Machine Learning*, pp. 148–156, San Francisco, Calif, USA, 1996.

[97] R. Anand, K. Mehrotra, C. K. Mohan, and S. Ranka, "Efficient classification for multiclass problems using modular neural networks," *IEEE Transactions on Neural Networks*, vol. 6, no. 1, pp. 117–124, 1995.

[98] T. Hastie and R. Tibshirani, "Classification by pairwise coupling," *The Annals of Statistics*, vol. 26, no. 1, pp. 451–471, 1998.

[99] M. Moreira and E. Mayoraz, "Improved pairwise coupling classification with correcting classifiers," in *Proceedings of the 10th European Conference on Machine Learning*, C. Nedellec and C. Rouveirol, Eds., vol. 1398 of *Lecture Notes in Computer Science*, pp. 160–171, Berlin, Germany, 1998.

[100] T. G. Dietterich and G. Bakiri, "Error—correcting output codes: a general method for improving multiclass inductive learning programs," in *Proceedings of the 9th AAAI National Conference on Artificial Intelligence*, pp. 572–577, 1991.

[101] J. Zhou, H. Peng, and C. Y. Suen, "Data-driven decomposition for multi-class classification," *Pattern Recognition*, vol. 41, no. 1, pp. 67–76, 2008.

[102] J. Friedman and P. Hall, "On bagging and nonlinear estimation," Tech. Rep., Statistics Department, University of Stanford, Palo Alto, Calif, USA, 2000.

[103] L. I. Kuncheva, F. Roli, G. L. Marcialis, and C. A. Shipp, "Complexity of data subsets generated by the random subspace method: an experimental investigation," in *Multiple Classi_er Systems. Second International Workshop (MCS '01)*, J. Kittler and F. Roli, Eds., pp. 349–358, Cambridge, UK, 2001.

[104] M. Sewell, "Ensemble Learning," Research Note RN/11/02, UCL department of computer science, 2011.

[105] E. Mayoraz and M. Moreira, "On the decomposition of polychotomies into dichotomies," in *Proceedings of the XIV International Conference on Machine Learning*, pp. 219–226, Nashville, Tenn, USA, July 1997.

[106] E. Menahem, L. Rokach, and Y. Elovici, "Troika—an improved stacking schema for classification tasks," *Information Sciences*, vol. 179, no. 24, pp. 4097–4122, 2009.

[107] L. Breiman, "Arcing classifiers," *The Annals of Statistics*, vol. 26, no. 3, pp. 801–849, 1998.

[108] G. Valentini, *Ensemble methods based on bias-variance analysis [Ph.D. thesis]*, University of Genova, Genova, Italy, 2003.

[109] C. Kruegel, G. Vigna, and W. Robertson, "A multi-model approach to the detection of web-based attacks," *Computer Networks*, vol. 48, no. 5, pp. 717–738, 2005.

[110] I. Corona, D. Ariu, and G. Giacinto, "HMM-web: a framework for the detection of attacks against web applications," in *Proceedings of the IEEE International Conference on Communications (ICC '09)*, June 2009.

[111] Y. Yan and H. Hao, "An ensemble approach to intrusion detection based on improved multi-objective genetic algorithm," *Journal of Software*, vol. 18, no. 6, pp. 1369–1378, 2007.

[112] W. M. Hu, W. Hu, and S. Maybank, "AdaBoost-based algorithm for network intrusion detection," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 38, no. 2, pp. 577–583, 2008.

[113] G. F. Cretu, A. Stavrou, M. E. Locasto, S. J. Stolfo, and A. D. Keromytis, "Casting out demons: sanitizing training data for anomaly sensors," in *Proceedings of the IEEE Symposium on Security and Privacy (SP '08)*, pp. 81–95, IEEE Computer Society, May 2008.

[114] J. R. Quinlan, *C4.5 Programs for Machine Learning*, Morgan Kaufmann, San Mateo, Calif, USA, 1997.

[115] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 338–345, 1995.

[116] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991.

[117] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine Learning*, vol. 11, no. 1, pp. 63–91, 1993.

[118] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[119] W. Khreich, E. Granger, A. Miri, and R. Sabourin, "Adaptive ROC-based ensembles of HMMs applied to anomaly detection," *Pattern Recognition*, vol. 45, no. 1, pp. 208–230, 2012.

[120] Z. Muda, W. Yassin, M. N. Sulaiman, and N. I. Udzir, "A K-Means and Naive Bayes learning approach for better intrusion detection," *Information Technology Journal*, vol. 10, no. 3, pp. 648–655, 2011.

[121] M. V. Mahoney and P. K. Chan, "An analysis of the 1999 DARPA/Lincoln laboratory evaluation data for network anomaly detection," Tech. Rep. CS-200302, Computer Science Department, Florida Institute of Technology, 2003.

[122] G. Kumar and K. Kumar, "A novel evaluation function for feature selection based upon information theory," in *Proceedings of the IEEE International Conference on Electrical and Computer Engineering (CCECE '11)*, pp. 000395–000399, Niagara Falls, Canada, May 2011.