Application of Bagging, Boosting and Stacking to Intrusion Detection

Iwan Syarif [1,2], Ed Zaluska[1], Adam Prugel-Bennett[1], and Gary Wills[1]

[1] School of Electronics and Computer Science, University of Southampton, UK
{is1e08,ejz,apb,gbw}@ecs.soton.ac.uk
[2] Eletronics Engineering Polytechnics Institute of Surabaya, Indonesia
iwanarif@eepis-its.edu

Abstract. This paper investigates the possibility of using ensemble algorithms to improve the performance of network intrusion detection systems. We use an ensemble of three different methods, bagging, boosting and stacking, in order to improve the accuracy and reduce the false positive rate. We use four different data mining algorithms, naïve bayes, J48 (decision tree), JRip (rule induction) and iBK(nearest neighbour), as base classifiers for those ensemble methods. Our experiment shows that the prototype which implements four base classifiers and three ensemble algorithms achieves an accuracy of more than 99% in detecting known intrusions, but failed to detect novel intrusions with the accuracy rates of around just 60%. The use of bagging, boosting and stacking is unable to significantly improve the accuracy. Stacking is the only method that was able to reduce the false positive rate by a significantly high amount (46.84%); unfortunately, this method has the longest execution time and so is inefficient to implement in the intrusion detection field.

Keywords: Intrusion detection system, bagging, boosting, stacking, ensemble classifiers.

1 Intrusion Detection System

Intrusion detection is a process of gathering intrusion-related knowledge occurring in the process of monitoring events and analyzing them for signs of intrusion [1]. There are two basic IDS approaches: misuse detection (signature-based) and anomaly detection. The misuse detection system uses patterns of well-known attacks to match and identify known intrusions. It performs pattern matching between the captured network traffic and attack signatures. If a match is detected, the system generates an alarm. The main advantage of the signature detection paradigm is that it can accurately detect instances of known attacks. The main disadvantage is that it lacks the ability to detect new intrusions or zero-day attacks [16][17].

The anomaly detection model works by identifying an attack by looking for behaviour that is out of the normal. It establishes a baseline model of behaviour for users and components in a computer or network. Deviations from the baseline cause alerts that direct the attention of human operators to the anomalies [17][18]. This system searches for anomalies either in stored data or in the system activity. The main advantage of anomaly detection is that it does not require prior knowledge of an intrusion

P. Perner (Ed.): MLDM 2012, LNAI 7376, pp. 593–602, 2012.

and thus can detect new intrusions. The main disadvantage is that it may not be able to describe what constitutes an attack and may have a high false positive rate [16][17][18]. We will develop a hybrid IDS which combines both misuse detection and anomaly detection system, but this paper focuses on the first technique.

2 Data Mining for IDS

Data mining studies automatic techniques for learning to make accurate predictions based on past observations [2]. In the intrusion detection case, data mining can be used to build a system that can distinguish intrusions or anomalies from normal network traffic. To build this kind of system, the first step is for the machine learning algorithms to learn the training dataset, which contains both normal traffic and intrusions. This learning phase results in a model that can be used to determine whether the network traffic is normal or an intrusion. There are many possible algorithms that can be used in the intrusion detection problem; their performance is measured using accuracy rate and false positive rate. In order to achieve a higher accuracy and lower false positive rate, many data mining researchers have proposed various ensemble learning approaches. It is well known in the data mining literature that the appropriate combination of a number of weak classifiers can yield a highly accurate global classifier [1].

3 Ensemble Classifier

An ensemble classifier is a method which uses or combines multiple classifiers to improve robustness as well as to achieve an improved classification performance from any of the constituent classifiers. Furthermore, this technique is more resilient to noise compared to the use of a single classifier. This method uses a 'divide and conquer approach' where a complex problem is decomposed into multiple sub-problems that are easier to understand and solve.

Ensemble approaches [2][15] have the advantage that they can be made to adapt to any changes in the monitored data stream more accurately than single model techniques. An ensemble classifier has better accuracy than single classification techniques. The success of the ensemble approach depends on the diversity in the individual classifiers with respect to misclassified instances [3]. According to Polikar [4], there are four ways to achieve this diversity, the first is to use different training data to train single classifiers, the second is to use different training parameters, the third is to use different features to train the classifiers and the final one is to combine different types of classifier.

Dietterich [5] reported that there are three main reasons why an ensemble classifier is usually significantly better than a single classifier. Firstly, the training data does not always provide sufficient information for selecting a single accurate hypothesis. Secondly, the learning processes of the weak classifier might be imperfect, and thirdly, the hypothesis space being searched might not contain the true target function while an ensemble classifier can provide a good approximation.

In this paper we evaluated and analyzed three different ensemble classifier techniques, called bagging, boosting and stacking, using various weak classifiers, such as nearest neighbour, decision tree, rule induction and naïve bayes; these were applied on a network intrusion dataset [11][12][13].

3.1 Bagging

Bagging, which means bootstrap aggregation, is one of the simplest but most successful ensemble methods for improving unstable classification problems. For example, weak classifiers, such as decision tree algorithms, can be unstable, especially when the position of a training point changes slightly and can lead to a very different tree. This method is usually applied to decision tree algorithms, but it also can be used with other classification algorithms such as naïve bayes, nearest neighbour, rule induction, etc. The bagging technique is very useful for large and high-dimensional data, such as intrusion data sets, where finding a good model or classifier that can work in one step is impossible because of the complexity and scale of the problem.

Bagging was first introduced by Leo Breiman [6] to reduce the variance of a predictor. It uses multiple versions of a training set which is generated by a random draw with the replacement of N examples where N is the size of original training set. Each of these data sets is used to train a different model. The outputs of the models are combined by voting to create a single output. Details of the bagging algorithm and its pseudo-code were given in [10].

3.2 Boosting

Boosting, which was introduced by Schapire et al.[7], is an ensemble method for boosting the performance of a set of weak classifiers into a strong classifier. This technique can be viewed as a model averaging method and it was originally designed for classification, but it can also be applied to regression. Boosting provides sequential learning of the predictors. The first one learns from the whole data set, while the following learns from training sets based on the performance of the previous one. The misclassified examples are marked and their weights increased so they will have a higher probability of appearing in the training set of the next predictor. It results in different machines being specialized in predicting different areas of the dataset [8].

In this paper, we select an AdaBoost algorithm, which is one of the most widely used boosting techniques for constructing a strong classifier as a linear combination of weak classifiers. The AdaBoost algorithm was first introduced by Freund and Schapire [9] and has been shown to solve many of the practical difficulties of earlier boosting algorithms, since it has solid theoretical foundation and produces very accurate predictions. Details of the boosting algorithm and its pseudo-code were given in [10].

3.3 Stacking

Stacking or *stacked generalization*, is a different technique of combining multiple classifiers. Unlike bagging and boosting, stacking is usually used to combine various different classifiers, e.g. decision tree, neural network, rule induction, naïve bayes, logistic regression, etc. Stacking consists of two levels which are base learner as level-0 and stacking model learner as level-1. Base learner (level-0) uses many different models to learn from a dataset. The outputs of each of the models are collected to create a new dataset. In the new dataset, each instance is related to the real value that

it is suppose to predict. Then that dataset is used by stacking model learner (level-1) to provide the final output [8]. For example, the predicted classifications from the three base classifiers, naïve bayes, decision tree and rule induction can be used as input variables into a nearest neighbour classifier as a stacking model learner, which will attempt to learn from the data how to combine the predictions from the different models to achieve the best classification accuracy. Details of the boosting algorithm and its pseudo-code were given in [10].

4 Experimental Settings

The following section describes the intrusion data sets used in the experiment, the performance metric used to evaluate the proposed system and the experimental settings and its results.

4.1 Intrusion Dataset

One of the most widely used data sets for evaluating intrusion detection systems (IDS) is the DARPA/Lincoln Laboratory off-line evaluation dataset or IDEVAL [11]. IDEVAL is the most comprehensive testset available today and it was used to develop the 1999 KDD Cup data mining competition [12]. In this experiment, we use the NSL-KDD intrusion data, which was provided to solve some problems in KDD'99, particularly that its training and test sets contained a huge number of redundant records with about 78% and 75% of the records being duplicated in the training and test sets, respectively. This may cause the classification algorithms to be biased towards these redundant records and thus prevent it from classifying other records [13].

Table 1. List of intrusions in training and testing data

Intrusions which exist in both training and testing data	Intrusions which only exist in testing data
back, buffer_overflow, ftp_write, guess_passwd, imap, ipsweep, land, loadmodule, multihop, neptune, nmap, phf, pod, portsweep, rootkit, satan, smurf, spy, teardrop, warezclient, warezmaster	apache2, httptunnel, mailbomb, mscan, named, perl, processtable, ps, saint, sendmail, snmpgetattack, snmpguess, sqlattack, udpstorm, worm, xlock, xsnoop, xterm

The intrusion data set consists of forty different intrusions classified into four main categories: DoS (Denial of Service), R2L (Remote to Local Attack), U2R (User to Root Attack) and Probing Attack. The training dataset consists of 25,191 instances and the testing dataset consists of 11,950 instances. The testing data set has many intrusions which do not exist in the training data, as shown in table 1.

4.2 Performance Metric

We use accuracy rate and false positive rate as the performance criteria based on the following metric shown in Table 2 below.

Table 2. Performance metric

		Actual Result	
		Intrusion	Normal
Predicted Result	Intrusion	True Positive (TP)	False Positive (FP)
	Normal	False Negative (FN)	True Negative (TN)

True Positive (TP) is a condition when an actual attack is successfully detected by the IDS and True Negative (TN) is a condition when no attack has taken place and no IDS alert is raised. False Positive (FP) is an alarm/alert that indicates that an attack is in progress when in fact there was no such attack. False Negative (FN) is a failure of IDS to detect an actual attack [19]. The accuracy rate and false positive rate are measured using these following formulas:

$$Accuracy\ rate = \frac{\text{TP+FN}}{\text{TP+TN+FP+FN}}(1), False\ Positive = \frac{FP}{\text{TP+FP}}(2)$$

4.3 Experimental Settings

We apply various data mining algorithms in the misuse detection module in order to find the best method for detecting intrusion based on accuracy, false positives and speed (computation time). We use four single algorithms from the Weka Data Mining Tools: Naïve Bayes, iBK, Jrip and J48, then apply these algorithms into three different ensemble classifiers, which are bagging, boosting and stacking, as shown in Figure 1 below.

These algorithms were executed on a PC with Intel Xeon quad core processors 2.67 GHz and 12 Gb RAM. In the first experiment, we use 10-fold cross validation as a performance measurement while in the second experiment we use testing data which contains many new intrusions.

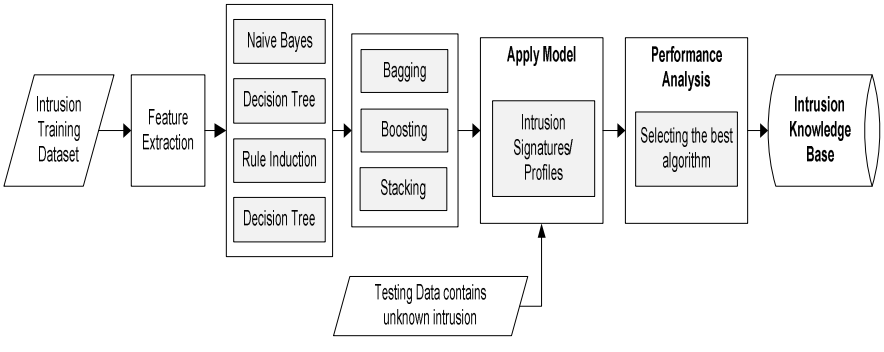

Fig. 1. Misuse detection model

4.3.1. Cross Validation

For performance measurement, we first use the 10-fold cross validation technique, which only needs training data. In 10-fold cross-validation, the original training data

is randomly partitioned into 10 subsamples. Of the 10 subsamples, a single subsample is retained as the validation data for testing the model, and the remaining 9 subsamples are used as training data. The cross-validation process is then repeated 10 times with each of the 10 subsamples used exactly once as the validation data. The 10 results from the folds then can be averaged to produce a single estimate. The results of the first experiment are given in Tables 3 and 4 below.

Table 3. The performance of ensemble classifiers using 10-fold cross validation

Algorithm	Accuracy			False Positive		
	Single	Bagging	Boosting	Single	Bagging	Boosting
Naïve Bayes	89.59%	89.57%	94.56%	10.60%	10.70%	5.30%
iBK	99.44%	99.44%	99.44%	0.60%	0.60%	0.60%
Jrip	99.58%	99.71%	99.73%	0.40%	0%	0.30%
J48	99.56%	99.67%	99.80%	0.40%	0.30%	0.20%

In the stacking method, we use three different algorithms as base learners and an algorithm as a stacking model learner. We use various combinations of naïve bayes, iBK, J48 and JRip. The classifications predicted by the base learners will be used as input variables into a stacking model learner. Each input classifier computes predicted classifications using cross validation from which overall performance characteristic can be computed. Then the stacking model learner will attempt to learn from the data how to combine the predictions from the different models to achieve maximum classification accuracy. The stacking algorithm experiment results are given in the Table 4.

Table 4. The performance of stacking algorithm using 10-fold cross validation

Base Learner	Stacking Model Learner	Accuracy (%)	False Positive (%)
Naïve Bayes	Jrip	99.64%	0.40%
iBK			
J48			
Jrip	Naïve Bayes	99.75%	0.30%
iBK			
J48			
Naïve Bayes	iBK	99.51%	0.50%
J48			
Jrip			
Naïve Bayes	J48	99.63%	0.40%
iBK			
Jrip			

4.3.1.1. Results. Overall, all the algorithms achieved good results, with the highest accuracy being 99.80% and the lowest being 89.59%. Tables 3 and 4 above show that Adaboost when implement with J48 as a weak classifier achieves the highest accuracy, which is 99.80%, with a false positive (FP) rate of 0.30%. On the other hand, the J48 Bagging algorithm achieves the lowest FP rate of 0%. Unfortunately the computation time of the three ensemble classifiers are all very high; the slowest one is stacking followed in turn by boosting and bagging.

Table 5. Accuracy improvement on 10 fold cross validation experiment

Algorithm	Single Classifier	Accuracy Improvement					
		Bagging	%	Boosting	%	Stacking	%
Naïve Bayes	89.59%	89.57%	-0.02%	94.56%	5.55%	99.75%	11.34%
iBK	99.44%	99.44%	0.00%	99.44%	0.00%	99.51%	0.07%
Jrip	99.58%	99.71%	0.13%	99.73%	0.15%	99.64%	0.06%
J48	99.56%	99.67%	0.11%	99.80%	0.24%	99.63%	0.07%

Table 5 and Table 6 show that the use of the bagging, boosting and stacking algorithms did not improve the accuracy significantly. Only the use of boosting and stacking on the Naïve Bayes algorithm were able to improve the accuracy, by 5.55% and 11.22% respectively, while the others showed a less than 1% improvement.

Table 6. False positive reduction on 10 fold cross validation experiment

Algorithm	Single Classifier	False Positive Improvement					
		Bagging	%	Boosting	%	Stacking	%
Naïve Bayes	10.60%	10.70%	-0.94%	5.30%	50.00%	0.30%	97.17%
iBK	0.60%	0.60%	0.00%	0.60%	0.00%	0.50%	16.67%
Jrip	0.40%	0.30%	25.00%	0.30%	25.00%	0.40%	0.00%
J48	0.40%	0.30%	25.00%	0.20%	50.00%	0.40%	0.00%

While the three ensemble algorithms failed to improve the accuracy, they succeed in reducing the false positive rates. Bagging was able to reduce the false positive rate by up to 25% when implemented with Jrip and J48, boosting by up to 50% for Naïve Bayes and J48, and stacking by up to 96.23% for Naïve Bayes.

4.3.2. Testing Data

In the second stage, we implement various single algorithms against the training data set to build an intrusion model then apply this model to the testing data which contains a lot of unknown attacks (see Table 1). The results are given in Tables 7 and 8 below.

4.3.2.1. Results. Overall none of the algorithms in the misuse detection module performed very well in detecting data with a lot of new intrusions. The best accuracy was only 67.90%, which was achieved by the stacking algorithm with iBK as a model learner and three other algorithms (Naïve Bayes, Jrip and J48) as base classifiers. Bagging was only able to improve it by less than 1% in three methods (Naïve Bayes, iBK, J48) while boosting failed to improve any method. The stacking method was able to improve the accuracy to 6.90% (Naïve Bayes) and 8.05% (iBK).

Table 7. Accuracy improvement using testing data experiment

Algorithm	Single Classifier	Accuracy Improvement					
		Bagging	%	Boosting	%	Stacking	%
Naïve Bayes	55.77%	56.10%	0.59%	37.60%	-32.58%	59.62%	6.90%
iBK	62.84%	62.95%	0.18%	20.90%	-66.74%	67.90%	8.05%
Jrip	63.69%	59.40%	-6.74%	18.40%	-71.11%	64.31%	0.97%
J48	63.97%	64.51%	0.84%	18.80%	-70.61%	61.23%	-4.28%

The bagging algorithm failed to reduce the false positive rates in three base classifiers (Naïve Bayes, iBK, JRip) and was only able to reduce it by 1.12% with J48 as a base classifier. Boosting is worse than bagging because it failed to reduce the false positive rates on all four base classifiers.

Table 8. False positive reduction using testing data experiment

Algorithm	Single Classifier	False Positive Improvement					
		Bagging	%	Boosting	%	Stacking	%
Naïve Bayes	34.80%	35.10%	-0.86%	37.60%	-8.05%	18.50%	46.84%
iBK	20.90%	20.90%	0.00%	20.90%	0.00%	17.40%	16.75%
Jrip	18.00%	19.00%	-5.56%	18.40%	-2.22%	16.90%	6.11%
J48	17.90%	17.70%	1.12%	18.80%	-5.03%	19.60%	-9.50%

Stacking algorithm is the only approach which was able to reduce the false positive rates significantly, with a 46.84% reduction on Naïve Bayes, a 16.75% reduction on iBK and a 6.11% reduction on JRip, even though it failed on J48 (-9.50%).

Figure 2 shows that the use of bagging, boosting and stacking significantly increases the execution time. The slowest is stacking followed in turn by bagging and boosting. The stacking method was able to reduce the false positive rate, but it would be too slow to implement in a misuse detection module. The bagging method, especially when applied to the iBK and Naïve Bayes algorithms, did not increase the execution time significantly and only improves the accuracy by 0.18% (iBK) and 0.59% (Naïve Bayes). Furthermore, bagging failed to reduce the false positive rate in either algorithm.

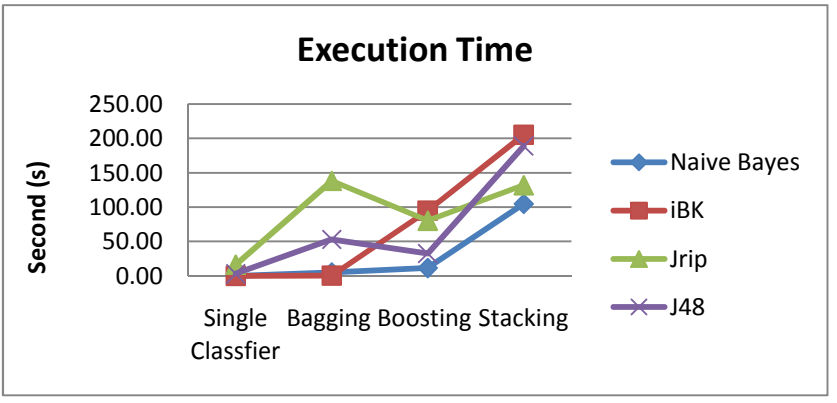

Fig. 2. Execution time comparison for single classifier bagging, boosting and stacking

5 Conclusions

We investigated the possibility of using ensemble algorithms (bagging, boosting and stacking) to improve the performance on network intrusion detection systems. Our experiment shows that a misuse detection module which implements four base classifiers and three ensemble algorithms achieves an accuracy of more than 99% in detecting known intrusions, but failed to detect novel intrusions with the accuracy rates of around just 60%. The use of bagging, boosting and stacking is unable to significantly improve the accuracy. Stacking is the only method that was able to reduce the false positive rate by a relatively high amount; unfortunately, this method has the longest execution time which is a serious disadvantage in the intrusion detection field. Of the four single classifiers used, J48 outperformed the three other methods by achieving the highest accuracy rates and the lowest false positive rate, with a relatively fast execution time. To improve the ability to detect new intrusions, we propose to develop an anomaly detection module and integrate both systems to produce a hybrid intrusion detection system.

References

1. Gudadhe, M., Prasad, P., Wankhade, K.: A new data mining based network intrusion detection model. In: International Conference on Computer & Communication Technology (ICCCT 2010), pp. 731–735 (2010)
2. Schapire, R.A.: The Boosting Approach to Machine Learning An Overview. In: Nonlinear Estimation and Classification. Springer (2003)
3. Lee, K.C., Cho, H.: Performance of Ensemble Classifier for Location Prediction Task: Emphasis on Markov Blanket Perspective. International Journal of u- and e- Service, Science and Technology 3(3) (September 2010)
4. Polikar, R.: Ensemble Based Systems in Decision Making. IEEE Circuits and Systems Magazine 6(3) (2006)

5. Dietterich, T.G.: Machine learning research: Four current directions. AI Magazine 18(4), 97–136 (1997)
6. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
7. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics 26(5), 1651–1686 (1998)
8. Graczyk, M., Lasota, T., Trawiński, B., Trawiński, K.: Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal. In: Nguyen, N.T., Le, M.T., Świątek, J. (eds.) ACIIDS 2010, Part II. LNCS, vol. 5991, pp. 340–350. Springer, Heidelberg (2010)
9. Freund, Y., Schapire, R.E.: A Decision-Theoritic Generalization of on-line Learning and an Application to Boosting (1995)
10. Zhou, Z.-H.: Ensemble Learning. In: Encyclopedia of Biometrics, vol. 1, pp. 270–273. Springer, Berlin (2009) ISBN: 978-0-387-73002-8
11. DARPA Intrusion Detection Data Sets, http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html
12. KDD Cup 1999 Intrusion Data Sets, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
13. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.: A Detailed Analysis of the KDD CUP 99 Data Set. In: Second IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA (2009)
14. Dong, L., Yuan, Y., Cai, Y.: Using Bagging Classifiers to Predict Protein Domain Structural Class. Journal of Biomolecular Structure & Dynamics 24(3) (2006) ISSN 0739-1102
15. Dong, Y.S., Han, K.S.: A comparison of several ensemble methods for text categorization. In: The 2004 IEEE International Conference on Service Computing (SCC 2004), pp. 419–422. IEEE Computer Society, Washington DC (2004) ISBN:0-7695-2225-4
16. Panda, M., Patra, M.R.: Ensemble of Classifiers for Detecting Network Intrusion. In: International Conference on Advances in Computing, Communication and Control (ICAC3 2009), pp. 510–515 (2009)
17. Garcia-Teodoro, P., Diaz-Verdejo, J., Macia-Fernandez, G., Vazquez, E.: Anomaly-based network intrusion detection: Techniques, systems and challenges. Computer & Security 28(1-2), 18–28 (2009)
18. Davis, J.J., Clark, A.J.: Data preprocessing for anomaly based network intrusion detection: A review. Computer & Security 30(6-7), 353–375 (2011)
19. Whitman, M.E., Mattord, H.J.: Principles of Information Security, 4th edn. Course Technology (2011) ISBN: 1111138214