Ensembling to improve infected hosts detection

Paula Venosa[1], Sebastián García[2], Francisco Javier Díaz[1]
[1] LINTI - Facultad de Informática - Universidad Nacional de La Plata
[2] Stratosphere Laboratory, AIC, FEL, Czech Technical University
{pvenosa,jdiaz}@info.unlp.edu.ar
sebastian.garcia@agents.fel.cvut.cz

Abstract. In this paper we describe the main ensemble learning techniques and their application in the cybersecurity threats detection. The state of the art in the use of ensemble learning techniques is presented here as an alternative to the current intrusion detection mechanisms, analyzing their advantages and disadvantages. We propose to incorporate ensemble learning to SLIPS [3], a behavioral-based intrusion detection and prevention system that uses machine learning algorithms to detect malicious behaviors, to obtain better results, taking advantage of the benefits of the SLIPS classifiers and modules. As part of this work we extend ensembling by considering algorithms from different domains (not machine learning domains), as Thread Intelligence. As a first stage of this project, performance tests of ensemble learning algorithms were performed to detect malware from flows evaluating its accuracy. The results of these tests are presented here, as well as the conclusions obtained and the future work.

Keywords: Ensemble learning,cybersecurity,malware,intrusion detection

1 Introduction

Detecting malware and attacks by analyzing network traffic remains a challenge. Although there are several detection mechanisms to accurately separate the malicious behavior from the normal one, it is still extremely difficult to have a detection system that can handle all the situations that arise in the network with low error rate. These techniques include machine learning algorithms, static signatures and experts rules based. In particular, the most used method today is based on the contribution of rules by a large community of analysts, called Threat Intelligence. The most important obstacles for a successful detection are: First, normal traffic is extremely complex, diverse and changing. Second, malicious actions change continuously, adapting, migrating and hiding as normal traffic. Third, the amount of data to analyze is huge, forcing analysts to lose data in favor of speed. Fourth, detection must occur in near real time to be useful. Fifth, there is a large imbalance in the amount of normal traffic compared to the amount of malicious traffic, making very difficult to have good detection results. Sixth, the cost of False Positives errors and False Negative errors is different, further complicating the decision process.

To solve some of these problems, the security learning community proposed the use of ensemble algorithms. These algorithms implement techniques for using, adding and summarizing information about several different detectors in a final decision.

Although there were some good proposals for ensemble learning techniques applied to the security of the network [2], there are two aspects of them that were not fully studied. First, the application of ensemble learning algorithms with Threat Intelligence data (e.g. VirusTotal [3]). Secondly, there are no ensemble learning algorithms that work as a function of time in the detection of the same source hosts.

In the current scenario where it is impossible to stop all threats, the effort should not focus only on protection and prevention, but be directed towards detection and response[4]. In this context, malware continues to represent the main threat and its detection remains one of the main concerns [5].

Given the large amount and exponential growth of malware samples [6] [7] [8] an effective way to detect them is required. The tools that use signature-based methods require maintaining a database to store patterns based on the characteristics of the malware extracted by experts. They present their limitations since a small change in the malware produces a different signature.

Intrusion detection systems have the ability to detect threats in the network but sometimes they are not effective. It is of vital importance that the algorithm used is reliable and can provide high detection accuracy. There are many research works that address this problem using several methods [9].

In order to benefit from multiple different classifiers, and exploit their strengths, we propose the use of ensembling algorithms [10] [11], which combine the results of the individual classifiers into a final result to achieve greater precision and thus a better result. By combining a set of classifiers to decide, the ensemble learning methodology imitates the human nature of seeking various opinions before making a crucial decision, weighing individual opinions and combining them for a better final decision.

2 Ensemble Learning Techniques

Ensemble learning techniques allow combining multiple models, both homogeneous and heterogeneous, with the aim of classifying new instances.

In practice, after constructing a set of classifiers that use some parts of the original dataset, the predictions of the different classifiers are combined to make a final decision. Different schemes are used, either the same algorithm can be trained using different datasets or different algorithms can be trained using the same dataset.

Ensemble learning models combine the decisions of multiple models to improve overall performance. Minimize noise errors, bias and variance. Among the main techniques are some very simple as Majority Voting, Average and Average with Weight[12]. **Majority voting:** In this technique, multiple models are used to make predictions for each tuple. The predictions of each model are considered as a separate vote. The prediction that is obtained from most models is used as the final prediction. **Average:** This technique takes an average of predictions of all models and is used to make the final prediction. **Average with weight:** In this technique all models are assigned different weights that define the importance of each model for prediction.
And these are added others such as bagging, boosting, random forest and stacking [13]. **Bagging** (also known as boostrap aggregating): each classifier is trained with a random subset with replacement of the original training set. With each sample a model is constructed. The results of these models are combined using average or majority voting. .**Bootstrap or boosting:** It is an iterative technique that adjusts the weight of an observation based on the last classification. If an observation was incorrectly classified, try increasing the weight of this observation and vice versa. The first algorithm is trained in the entire data set and subsequent algorithms are constructed by adjusting the residuals of the first algorithm, thus giving greater weight to those observations that the previous model predicted poorly. It is based on the creation of a series of weak algorithms, each of which may not be good for the entire data set, but is good for a part of the data set. Therefore, each model actually increases the performance of the set. In bagging, the samples used are independent, so the algorithms can be run in parallel. Boosting works in a sequential way. **Random forest**: It is a supervised machine learning algorithm based on ensemble learning. It is used for regression and classification. The idea is to build multiple decision trees and

add them to give an exact result. Decision tree is a deterministic algorithm that tend to overfit because it build the best possible tree for given data and fail to generalize when unknown data is provided. Each tree is constructed with a different random subset of our data. Random forest is more accurate than a simple decision tree because it minimizes the overfiting. In the classification problems majority voting is used to combine the predictions while in the regression, the predictions are made taking the average of the tree predictions.

Stacked generalization [13] is a method that uses a different way of combining multiple models introducing the meta learner concept. Stacking is the generalization of other teaching methods. The process is: 1)Split the training into a sample for training and a sample for testing. 2)A set of base learners are trained with the training sample.3)The models are tested using the test sample. 4)The predictions are used as input and the current response as output to train the highest level learner.

3 Ensemble Learning Applied to Network Security

Ensemble learning techniques represent an improvement over machine learning techniques, we seek to combine them to obtain better classifiers taking advantage of the potential of each one.

The IDSs are today with large volumes of data and with the difficulty of detecting new attacks that arise day by day. In this regard, there are several investigations that propose that they are nourished by machine learning, taking the advantages that this entails: 1) The ability of machine learning to generalize to detect new types of intrusions, 2) Attack signatures can be automatically extracted from tagged traffic data, 3) Ability to adapt to new attacks. At the same time, since in an attack they differ: intrinsic features (general information), traffic features (connection features) and content features (package info), there are proposals around implementing models with armed dataset samples from the different sets of features and different classifiers, which are then combined with ensemble learning techniques[14][15].

[14] presents the state of the art of the teaching methods used in modern IDSs describing different works that use basic Machine Learning techniques as Ensemble Learning techniques and tests are carried out using the KDD'99 dataset in most of

them. They conclude that ensemble learning presents an improvement to basic machine learning techniques since they improve detection given the possibility of using parallel architectures, such as GPGPU. In[14]a similar analysis is made from previous work, emphasizing that given the importance of the selection of features, it should be studied in this regard. It is also important to evaluate which base classifiers to use and how it should be combined in a way to design architectures that make multiple classifiers collaborate with each other instead of competing. This problem is addressed in a more specific way in [15], where ensemble learning is applied for intrusion detection to detect DoS, R2L, U2R and Probing attacks in FTP service traffic. The work performs the tests based on the UCI KDD and Neural Networks dataset with different feature sets: intrinsic features, content features and traffic features using MPLM and Majority, Bayesian Average, Belief, as ensemble learning algorithms. In this work the metrics to compare the results were:% error, average cost, % false alarms. And it was concluded that ensemble learning reduces the percentage of error but also reduces the capabilities of generalization.

MPLM proposes to analyze the different phases or facets of a problem and build on this perspective of the problem. Perspectives are represented by a set of dataset features. Models are obtained from the different perspectives that are then combined with ensemble learning techniques. A problem to solve in this framework is the criteria and the implementation of the selection of features for each model to build the perspectives. MPLM can be applied to the detection of various attacks such as DoS, R2L, U2R and Probing [17] and for the detection of botnet activity [18], where the perspectives represent the different stages of their life cycle. In [18] a new model for applying learning in multifaceted problems is presented, focusing on the selection of features to be included in each perspective (network-based perspectives, host-based perspectives, DNS-based perspectives), one of the aspects that is raised as an aspect to be solved. In addition, the criteria of what features make sense to include in each model, taking into account that the inclusion of strongly correlated features may not contribute to the classification process.

There are also proposals as [19] based on modified Stacking to detect network intrusions (Probe, DoS, UR2 and R2L) where models are generated using samples from the random selection of dataset features, then select the best models according to a defined criteria (accuracy, information gain, recall mean and true positive rate) and combine them with stacking as the ensemble learning technique. And other more innovative works such as [20] that proposes to apply ensemble learning clustering to detect botnets. They generate partitions and use link based algorithm to combine them (in that step is where the ensemble learning takes place) and apply machine learning to perform the classification in each cluster of the final partition.

No proposals described include Thread Intelligence in the classification process.

4 Ensemble Learning to Improve Detection of Infected Machines

SLIPS [1] is a behavioral-based intrusion detection and prevention system that uses machine learning algorithms to detect malicious behaviors. In addition to the different classifiers that this tool has today (MLDetection and Portscan detector) other modules such as Thread Intelligence are in development to incorporate more information to the detection, and thus improve the accuracy of the tool from its use.

The proposal presented in this article and which is currently being worked on consists of the incorporation of ensemble learning algorithms that allow combining the information obtained from the different classifiers in order to improve the results in the detection of infected IPs, taking into account the anomalies that the classifiers are able to detect, in each of the stages, from the analysis of the flows that represent the connections of the hosts of the network on which SLIPS[1] operates.

The contribution consists in improving the detection process by implementing ensemble learning to take decisions based on different data provided by SLIPS[1], considering that to determine if an IP is infected in a time window, it has: 1)For a flow, different predictions, one for each classifier. 2)A set of flows associated with the given IP (the source IP of those flows), with its corresponding prediction. 3)A set of malicious behavior alerts associated with the given IP (that have this IP as source IP). 4)Information from different Thread Intelligence sources that indicate destinations of the analyzed flows that are malicious (with some confidence percentage).

In 1 we propose to apply ensemble learning to decide whether the flow is classified as malware or normal. In 2 to determine if the IP is infected or not, our proposal consists in applying ensemble learning to the set of flows and their predictions. Regarding the information provided by 3, the proposal is to take into account the percentage of alerts reported for that time interval. While, with respect to 4, it is also planned to implement ensemble learning to the information provided from the different IT sources, and evaluate whether to incorporate this information into each flow or associate it with the source IP directly, and take it into account in the decision.

5 First tests -Ensemble learning applied to network security

As we mentioned in previous section, within the framework of our proposal, a possible application of ensemble learning is to combine the results of multiple classifiers for a flow, where each one predicts whether it corresponds to malware or normal traffic. Then in SLIPS[1] we have n predictions for a flow, where n is equal to the number of classifiers in operation. As described in section 4 of this article, the advantages that can be obtained from applying ensemble learning to intrusion detection depends on the attack to be detected, the machine learning techniques to be combined and the features to be taken into account in the classification of the flows.

Thus, as a first stage of this project, performance tests of the teaching algorithms were performed to detect malware from flows evaluating its accuracy compared to a set of classic Machine Learning (ML) algorithms. To carry out the tests, the Stratosphere dataset [21] was used. It is a mixed dataset with tags corresponding to normal traffic and malware traffic, from a botnet known as Rbot. The following ML algorithms were tested: Logistic Regression(LR), Naive Bayes(NB), Random Forest(RF), Kneighbords(KN) and Decision Tree(DT). And the ensemble learning techniques used: voting hard (majority voting), voting soft (Using the sum of the predicted probabilities), voting with weight, boosting and bagging [22]. Tests with the Scikit-learn library [23] were implemented and the accuracy obtained from applying the cross_val_score function to the model was used as metric.

Table 1. Tests without applying ensemble learning

Algorithm	Accuracy
Logistic Regression	0.9945719941 (+/- 0.0064141127)
Random Forest	0.9999557803 (+/- 0.0000413645)
naive Bayes	0.9899729966 (+/- 0.0124655918)
Kneighbords	0.9997567974 (+/- 0.0002254790)
Decision Tree	0.9999336723 (+/- 0.0000644568)

Table 2. Tests with voting techniques including LR, Naive Bayes and RF:

Algorithm	Accuracy
Ensemble [Majority Voting]	0.9973136314 (+/- 0.0028659845)
Ensembling [Voting with probabilities sum]	0.9975347114 (+/- 0.0029525249)
Voting con peso (LR=1, RF=3 y NB=1)	0.999934 (+/-0.000041)

Algorithm	Accuracy
Voting con peso (LR=1, RF=3 y NB=2)	0.999912 (+/-0.000075)
Voting con peso (LR=2, RF=3 y NB=1)	0.999889 (+/-0.000092)

The tests with both voting techniques give the same results. Both improve the LR and the NB but not the RF. This happens because when combining RF that gives good results with two other algorithms that are worse, and then majority decides badly.

Voting with weight does not present improvements for RF although it does show improvements compared to the other voting techniques tested. Those that best classify are that give RF the greatest weight and the least weight to the other two algorithms.

Table 3. Tests by changing Random Forest to KNeighbords:

Algorithm	Accuracy
Voting con peso (LR=1, KN=2 y NB=1)	0.999757 (+/-0.000254)
Voting con peso (LR=1, KN=3 y NB=1)	0.999757 (+/-0.000254)
Voting con peso (LR=2, KN=3 y NB=1)	0.999746 (+/-0.000254)

Table 4. Tests changing Random Forest to Decision Tree:

Algorithm	Accuracy
Voting con peso (LR=1, DT=3 y NB=1)	0.999934 (+/- 0.000064)
Voting con peso (LR=1, DT=3 y NB=2)	0.999912 (+/- 0.000075)
Voting con peso (LR=2, DT=3 y NB=1)	0.999934 (+/- 0.000064)

Instead of using voting combining LR, RF and NB, it was tried to combine LR and NB with KN and with DT, and improvements could also be seen when applying teaching regarding not applying it, although they were not significant but they were significant. there were them for all the algorithms involved.

Table 5. Tests using a seed value = 8

Algorithm	Accuracy
Logistic Regression (con bagging)	0.9941959620 (+/- 0.0173140208)
Random Forest (con bagging)	0.9999778908 (+/- 0.0001326553)
Naive Bayes(con bagging)	0.9916532294 (+/- 0.0272774708)
Decision Tree (con bagging)	0.9999778908 (+/- 0.0001326553)
Kneighbords (con bagging)	0.9998010158 (+/- 0.0003396534)

The precedent table shows that improvements are obtained in all tested algorithms.

Table 6. Tests of boosting (Adaboost) tests:

Algorithm	Accuracy
Logistic Regression (con boosting)	0.9855731305 (+/- 0.0170767141)

Algorithm	Accuracy
Random Forest (con boosting)	0.9999557803 (+/- 0.0000541578)
Naive Bayes(con boosting)	0.7788494492 (+/- 0.2931913628)
Decision Tree (con boosting)	0.9999336711 (+/- 0.0001326557)

Adaboost improves for DT is the same for RF and does not improve either NB or LR.

6 Conclusions and future work

From the previous section it is concluded that the teaching improves the results against the algorithms that do not apply it, and that the improvement is significant when the base algorithms are weak. If any of the base algorithms has a high percentage of successes, heavy voting, giving more weight to this algorithm, is the better. This technique is appropriate to apply to the problem of deciding the prediction for a flow, given a set of predictions resulting from the different SLIPS[1] classifiers.

Of the works that describe the use of ensemble learning applied to cybersecurity, none incorporates the valuable information provided by Thread Intelligence sources, which is part of our proposal.

On the other hand, to carry out the proposal described here, as we have described in section 4, the most appropriate techniques to apply depend on the problem given, so in order to find the best options for the implementation of the proposal, specific tests must be carried out with the data that SLIPS[1] provides. These tests, the design of the solution and its implementation are the keys to the future work of this project.

References

1. https://www.stratosphereips.org/technology. Stratosphere IPS Project site.
2. Vanerio, J., & Casas, P. (2017, August). Ensemble-learning Approaches for Network Security and Anomaly Detection. Workshop on Big Data Analytics and Machine Learning for Data Communication Networks (pp. 1-6). ACM.
3. https://www.virustotal.com/. File and suspicious urls analysis site to detect malware.
4. Kasey Panetta. (2017). 5 Trends in Cybersecurity for 2017 and 2018. Diciembre de 2018, Smarter with Gartner website: https://www.gartner.com/smarterwithgartner/5-trends-in-cybersecurity-for-2017-and-2018/

5. ENISA. (2017). ENISA Threat Landscape Report 2017 15 Top Cyber-Threats and Trends. December 2018, de ENISA Sitio web: https://www.enisa.europa.eu/publications/enisa-threat-landscape-report-2017
6. PANDA. (2018). 2017 in Figures: The Exponential Growth of Malware. December 2018, de PANDA Website: https://www.pandasecurity.com/mediacenter/malware/2017-figures/
7. AV-Test. (2018). Malware. Diciembre de 2018, de AV-Test The Independ IT-Security Institute Sitio web: https://www.av-test.org/en/statistics/malware/
8. McAfee Labs. (2017). McAfee Labs Threat Report. December de 2018, de McAfee website: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-dec-2017.pdf
9. R. Z. A. Mohd, M. F. Zuhairi, A. Z. A. Shadil and Hassan Dao, "Anomaly-based NIDS: A review of machine learning methods on malware detection," 2016 International Conference on Information and Communication Technology, Kuala Lumpur, 2016, pp. 266-270.
10. Zhi-Hua Zhou. (2012). Ensemble Methods: Foundations and Algorithms (Chapman & Hall/Crc Machine Learnig & Pattern Recognition) 1st Edition. US: Chapman and Hall/CRC.
11. Cha Zhang Yunqian Ma. (2012). Ensemble Machine Learning: Methods and Applications 2012th Edition. United States: Springer.
12. Lior Rokach (2009). Ensemble-based classifiers. Published online: 19 November 2009. United States: Springer.
13. Martin Sewell (2011). Ensemble learning.United Kindom: UCL Research Note (UCL Department of Computer Science).
14. Gianluigi Folino, Pietro Sabatino (2016). Review Ensemble based collaborative and distributed intrusion detection systems: A survey. Journal of Network and Computer Applications. Elsevier Journal.
15. Luca Didaci, Giorgio Giacinto and Fabio Roli (2002). Ensemble Learning for Intrusion Detection in Computer Networks. ACM Journal.
16. Chih-Fong Tsai, Yu-Feng Hsu, Chia-Ying Lin, Wei-Yang Lin (2009). Intrusion detection by machine learning: A review. United States: Expert Systems with Applications 36 (2009) 11994–12000.
17. Miller, Sean & Busby-Earle, Curtis. (2018). Multi-Perspective Machine Learning (MPML) A Machine Learning Model for Multi-faceted Learning problems 10.1109/CSCI.2017.60.
18. Miller, Sean & Busby-Earle, Curtis. (2018). Multi-Perspective Machine Learning A Classifier Ensemble method for intrusion detection.10.1145//CSCI.2017.
19. Necati DEMİR ∗ , Gökhan DALKILIÇ (2018). Modified stacking ensemble approach to detect network intrusionNecati DEMİR ∗, Gökhan DALKILIÇ. Turkish Journal of Electrical Engineering & Computer Sciences 26: 418 – 433
20. Mai, Long & Kun Noh, Dong. (2017). Cluster Ensemble with Link-Based Approach for Botnet Detection. Journal of Network and Systems Management. 10.1007/s10922-017-9436-x.
21. https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-51/capture20110818.pcap.netflow.labeled. Stratosphere Datasets download site.
22. Aishwayra Singh (2018)A Comprehensive Guide to Ensemble Learning (with Python codes). Published online in Analytics Vidhya: June 18, 2018.
23. https://scikit-learn.org/stable/. Scikit-learn python library website.