Machine Learning based Network Anomaly Detection for IoT environments

Valentina Timčenko*, Slavko Gajin**
* School of Electrical Engineering, Mihailo Pupin Institute, University of Belgrade, Belgrade, Serbia
** School of Electrical Engineering, Belgrade, Serbia
valentina.timcenko@pupin.rs, slavko.gajin@rcub.bg.ac.rs

Abstract— **This paper focuses on the problem of providing security measures, anomaly detection, and prevention to the emerging IoT environment. We have considered several different categories of machine learning classification algorithms with a goal to estimate the proper choice for network anomaly detection in IoT like environments. The special focus is on SVM and the set of bagging and boosting algorithms, the analysis of their performance and further comparison taking for the reference the modern, IoT like, UNSW-NB15 dataset.**

I. INTRODUCTION

The intensified development of network technologies and fast growth of number of different heterogeneous network devices that are interconnected and continuously exchange large data volumes (including Big Data), have created a fertile ground for the development of versatile cyber-attack categories and occurrence of various forms of network traffic anomalies. Internet of Things (IoT) and similar environments are at the momentum extremely interesting for both the research and development community and hacking communities at the other side. Being at least one step ahead from the hackers is a supreme goal of any security concern development team.

For further IoT development, and especially for smart and Industrial IoT applications, there is a need for intelligent processing and analysis of data. Such a challenging environment characteristics is difficult to respond with a use of traditional, stale techniques for traffic anomaly detection and fast developing and adaptable malicious activities.

With a motivation of supporting more intelligent intrusion and anomaly detection in future network and system environments, we have considered a set of machine learning algorithms over one of the newest IoT dataset representatives, UNSW-NB15.

Once applied as a part of any network or even system, IoT technology represent a serious source of different, frequently dynamic data generated in various formats. Besides, the quality of the collected data differs as a result of interconnection of highly heterogeneous devices, existence of environmental and device noise, source trustworthiness degree, as well as on the capabilities of the devices to provide specific level of precision and accuracy during data collection. Such a powerful technology and the need to respond to the raised security concerns has engaged a big research communities with a common task: provisioning of the best security level in given environmental circumstances. As a result, a number of open security questions have emerged, seeking for applicable solution(s). Besides all the benefits and positive characteristics that the wireless medium has brought to IoT, there are same inherited deficiencies which are mostly reflected through the vulnerability to different forms of traffic anomalies and attacks. The intrusion detection and prevention systems represent cornerstone for primary security provisioning, as they are continuously monitoring the network activity, searching for any suspicious, potentially anomalous behavior. New technologies and heterogeneous environments are legitimate targets for malicious activities, thus there is need for machine learning (ML) algorithms that are capable of accurate and fast detection analysis and classification of intruders or traffic anomalies. We have put our efforts to approach this hot topic, understand all the issues and adequately provide a comprehensive evaluation of a range of ML algorithms suitable for environments that fall under the IoT umbrella. The intensive hype related to the introduction of IoT has implicated the noticeable boosting of the ML research area popularity, especially among industrial companies.

For the needs of running a set of experiments we have used Weka environment, widespread software platform in academia and business [1]. The main idea is to analyze the possible role of a set of chosen ML techniques in a way to construct strong defense mechanism for IoT environments based on network dataset/flows and specific identifiers (dataset features).

II. METHODOLOGY

In general, ML techniques can be categorized as supervised, unsupervised or reinforcement learning (hybrid) algorithms, while in more detailed version this categorization is: Deep Learning, Ensemble, Neural Networks, Regularization, Rule System, Regression, Bayesian, Decision Tree, Dimensionality Reduction, Instance Based, Clustering. Despite such diversity, the fact is that there is no a ML technique that would be adequate for every network traffic/dataset case. Our focus is on the application of the supervised ML algorithms that base their functionality on the use of the labeled datasets. We rely on the use of the knowledge gained through labels to build procedures for mapping from input attributes plane to the output plane. We apply and compare the performances of several supervised learning algorithms, including SVM (Supervised Vector Machine) and a range of ensemble classifiers [2]. The ensemble algorithms combine basic classifiers which are trained on different subsets of dataset [3]. We are measuring performances of the chosen algorithms in Weka environment, taking as an IoT representative the UNSW-NB15 dataset [4].

III. SUPERVISED MACHINE LEARNING IN IOT

The fact that IoT represents one of the major sources of information and data flows, has urged for faster development of new, smarter and more powerful techniques for network traffic analysis, keeping efficient, accurate and secure the procedures for their manipulation, storage and exchange. Although mentioned last, security has become an ultimate issue, thus a range of data science areas, including data mining, artificial intelligence (AI), ML, and forensic analysis are increasingly being applied in search for best solutions depending on the specific use case, data characteristics and system needs. Such a diversity of characteristics has again confirmed that there is no unique method that would provide efficient and accurate intrusion detection for every dataset under the analysis. In ML area we are dealing with three approaches: supervised, unsupervised and reinforcement ML. The choice of the category and successively the algorithm to apply depends on data integrity, available feature vector, missing data issue and possibility of building it based on the available, known data, while keeping awareness of the inherent memory and CPU limitations of IoT devices and network resources, their mobility, diversity in communication protocols, and vulnerability to a range of exploits. The aim is to reach flexibility benefits that these algorithms can offer with the minimum of the human intervention. The major goal is to reach the best possible trade-off between the algorithm computational requirements and the classification precision and false alarm generation at the other side.

In order to effectively mitigate and investigate the security threats for IoT environments, it is of major need to explore and enhance the techniques that would efficiently: (1) detect any suspicious activity; (2) detect any traffic anomaly from the "normal profile" behavior; (3) use the benefits of forensic techniques to prevent future attacks/anomalies; (4) mitigate the impact that intrusions/anomalies would have on the proper network/system functioning.

ML algorithms require dataset with a massive number of data instances that are basis for learning phase (Figure 1), which analyzes the input algorithm and applies it to the collected dataset. It explores the provided data, usually through a range of feature vectors. It relies on the provided labels for each instance in order to find internal structures and relationships between them and form a model that is further used for the evaluation phase. As a result, a set of metrics is applied to provide quantitative evaluation of model performance. The idea is to calculate the metrics and choose the best model for further prediction and detection of the intrusions in tested environment/dataset.

Although the research in the area of cyber security is at its peak, adequate datasets are still rarely the available. For a long time the research community has based its efforts in analyzing the 1999 KDDCUP dataset, thus generating a number of scientific studies [5]. This dataset has faced strong criticism, as being outdated, flawed and with stale characteristics. Its refreshed and reorganized version, NLS KDD has prolonged its lifetime by the removal of all the inconsistencies and duplicates from the original dataset and solved unbalanced problem between the training and testing instances, which resulted in providing more accurate false alarm counts. Additionally, the sophisticated organization and arrangement of the testing and training instances has provided higher authenticity of the gained exploration results, giving a possibility of prolongation to its utilization [6].

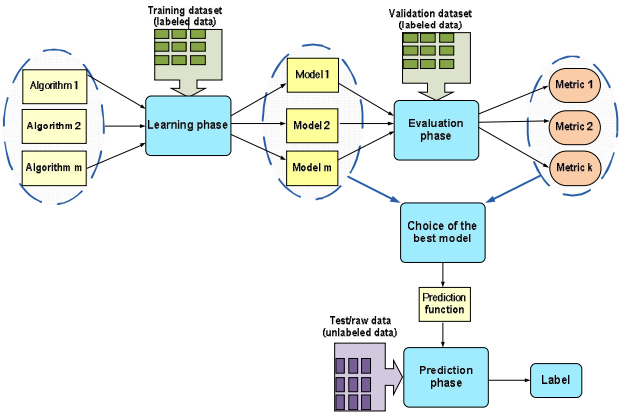

Figure 1. The structure of the supervised machine learning, evaluation and prediction procedures

Nevertheless, the lack of modern attacks footprints has become one of the strongest reasons for switching the attention to rare but existing novel datasets, while many research laboratories have generated their own, proprietary datasets which are sometimes available for further research and contribution of the research community. The first available labeled dataset was generated with honeypot technique by A. Sperotto [7]. Later, Winter has modified the original Sperotto dataset in order to make it more suitable for the use of the SVM algorithm [8]. Recently, a novel and IoT eligible UNSW-NB15 dataset appeared, providing now possibilities for modern IDS investigations [4].

A. *WEKA*

WEKA (Waikato Environment for Knowledge Analysis), is an experimental software environment for data analysis which allows the use, analysis and evaluation of the most relevant machine learning techniques. It is Java based, and consisting of a number of open source packages related to different preprocessing, classification, grouping, association, and visualization techniques. It uses its proprietary ARFF (Attribute-Relation File Format) format, along with some adapted formats. Different packages and integrated tools can be implemented for data analysis projects, with a support for user contributions and algorithms development.

B. *UNSW-NB15 dataset*

UNSW-NB15 dataset covers a collection of a large number of normal network traffic and malicious traffic instances [4]. It encompasses realistic normal traffic behavior and combines it with the synthesized up to date attack instances. Malicious traffic is generated for a number of typical, up to date attack types. It is an unbalanced dataset, where the number of attack instances is much lower that of normal traffic instances. It relies on 100 GB of the raw traffic (Tcpdump Pcap files) and packets generated in IXIA PerfectStorm tool environment. Raw data is processed with Argus Bro-IDS tools, generating 49 features with the class label. The total captured traffic consists of 2,540,044 instances, while

parts of dataset were used for generation of two sets for analysis purposes (training and testing) where the instances are organized with 44 features, classified in 7 feature groups: basic (13), content (8), time (9), additional features (12), attack category (1) and label (1). Malicious instances encompass nine attack categories: DoS, Fuzzers, Analysis, Backdoor, Shellcode, Worm, Exploits, Generic and Reconnaissance. UNSW-NB15 is related to actual networking paradigms and trends, tending to provide a good basis for wide-ranging networking IDS analysis.

IV. Machine Learning Algorithms in IDS

Modern technologies have brought better performances, throughputs, extreme connectivity and interoperability. Although these are beneficial as improvements, some unfavorable side effect have come to the surface and brought on the scene a whole new range of issues, thus directly or indirectly targeting the security area. The attack can be defined as activity of an internal or external network element(s) that violate the integrity, confidentiality or availability of available network services and resources. It is of mayor interest to provide an adequate level of network detection, prevention, and reaction to the attacks or network traffic anomalies. Intrusion Detection Systems (IDS) development is all about following the fast advance of (cyber) attacks, in a way to be able to deal with anomalous traffic, as well as with the mushrooming trend of novel attacks appearance. Most of the modern IDSs rely on the use of ML and AI algorithms that are able to efficiently detect/prevent attacks and anomalies, giving adaptability, flexibility and usability in data-process-intensive network environments. The tendency is to follow the needs for higher accuracy, decreased false alarm rates and lower memory and computation consumption, and ML could help in automating the generation of different analytical models for algorithms that would continuously learn based on the available data, find similarities, identify activities that outstand the "normal profile", and find meaningful traffic outliers related to the anomalies detection [9]. ML algorithms can be used in cases where: (1) the preferred result is known - supervised learning; (2) when the data is not known in advance - unsupervised learning; (3) in the case when the learning represents the result of interaction between the defined model and environment - reinforcement learning. The most important difference relies in the way of manipulating the database instances, which highly depends on the existence of the class labels [10]. Supervised algorithms apply the knowledge learned from labels to create the procedures that will be used for mapping the attributes from the input to the output plane [11]. If the dataset is not labeled or if there is certain data/feature inconsistency, the procedure of detection and classification can be significantly complex, seeking for the approaches that are covered by unsupervised and hybrid ML algorithms [12]. The benefits of the continuously evolving models are certain increasingly positive results reflected through automatic generation of the reliable and repeatable decisions thus reducing the need for human interaction. The timeline from the definition of certain ML algorithm to its implementation in real situations, and especially for modern widespread IoT like environments, can be long. Also, some variations of the traffic behavior that should be treated as anomalies are frequently left undetected - "below the radar". ML algorithms are challenged to protect and provide bypass for their weaknesses, as the intruders are becoming extremely adaptive and intelligent, thus can easily manipulate data to compromise learning [13].

The top-notch ML algorithms deal with the needs for the fast real time processing, using variable quantities of data in different formats and from various sources. The increase of the computational power has implicated the possibilities for accurate and reliable (pre)processing of such information. The noticeable increase of the exploitation of the security sensitive applications has stimulated the expansion of the pattern recognition systems among the trending IDSs. All this considered has fortified the efforts towards the use and further improvement of the basic and hybrid classifier categories, mostly SVM, k Nearest Neighbors (kNN), Neural Networks [14]. One of the most popular ensemble classifiers, AdaBoost has gained interest of a group of authors to evaluate its boosting capabilities and issue of efficiently skipping the overfitting [15]. It is recently shown that LogitBoost, which stands for the simple regression function algorithm can be used as the base learner when there is need to fit the logistic models [16]. The datasets with small percentage of attacks can be easily analyzed with the use of the Bagged classifier [17].

No single ML algorithm is suitable for all cases, it depends on the traffic characteristics, whereas every algorithm can be applied with some specific advantages over the others, but it can also be inadequate for circumstances where its advantages cannot be fully used.

A. SVM

Support Vector Machine (SVM) is an supervised ML algorithm that relies on the use of the labeled training instances, searching for the temporal and spatial correlations in available data. SVM places the instances as points in the feature space, thus dividing the space into several parts. The separation margins are used to provide as wide as possible separation of the parts, while newly processed instance is classified and placed in the most suitable area/part. It is suitable for the linearly inseparable data and effectively learns from high dimensional data, eliminating the need for feature reduction (Figure 2).

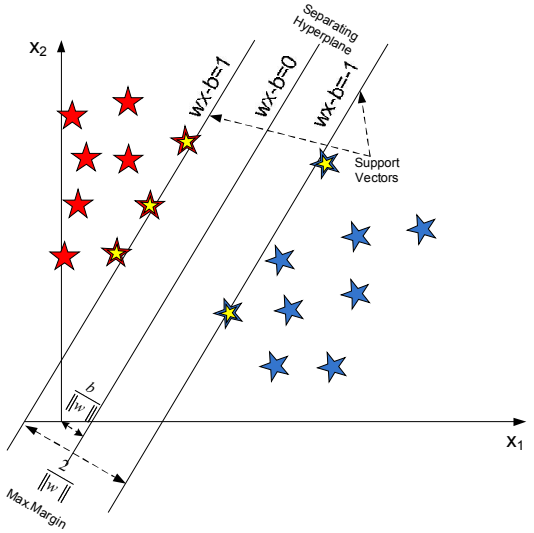

Figure 2. SVM classification

SVM relies on finding a suitable hyperplane that has the largest distance to the closest data point from the dataset, while when finding larger margin it provides lower generalization error. By the means of kernel, SVM maps the instances into a higher dimensional feature space where data becomes easily separable. The SVM classifier is defined by a linear discriminate function $f(x)=\mathbf{w}\cdot\mathbf{x}+b=\sum x_i w_i+b$, where **w** represents the weight vector, **x** is the components x_i vector, and *b* stands for the bias that translates the hyperplane from the origin.

Sequential Minimal Optimization (SMO) stands for a simplified and enhanced SVM version, that has better scaling capabilities applicable to the complex SVM problems [18]. It is easy at implementation and use, as it relies on the application of breaking down a large quadratic programming (QP) optimization SVM function to solve, into a groups of as smallest as possible QP problems. These are further analytically solved, saving time and CPU by avoiding a time-consuming numerical QP optimization as an inner loop.

In WEKA, SMO and LibSVM C-SVM are different algorithms, and stand to be specific variations of the original SVM [18]. Precisely, for the needs of training a support vector classifier, SMO implements sequential minimal optimization algorithm while LibSVM C-SVM is a wrapper class for SVM algorithm implementation.

B. *Ensemble Classifiers*

The ensemble classifiers stand for a specific type of predictive algorithms which combine effects of various basic classification models in order to obtain the maximum benefits from the advantages related to each of the used classifiers and finally better predictive performance that any of them. These are usually trained on different subsets of the dataset in order to jointly produce output results with the reduced false alarms and increased accuracy. The advantages of the ensemble classifiers also rely on the possibility of applying them over the updated or new data, while there is no need for retraining the whole ensemble. The diversity of the basic classifier outputs, reflected through a variety of the resulting ensembles, provides a possibility to capture the differences of the analyzed data, and to match more precisely their specific characteristics in a way to make instance classification highly accurate and increase the overall predictive performance. Figure 3 depicts the procedure of the ML ensemble learning and classification principles. Training dataset is represented through several smaller datasets, which represent particles of the main preprocessed dataset. These are applied for the training routine for different base learners with an aim to learn model and obtain results for each of them.

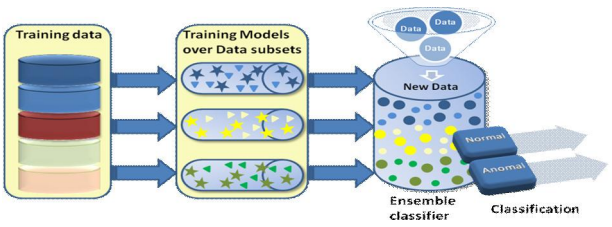

Figure 3. Ensemble classification architecture

The final output is defined by the combination of the individually obtained results. Ensemble algorithms mostly rely on the application of the decision trees (DT) as a base learner [19]. DT algorithms rely on recursively generating classification tree, where the values of the decision nodes depend on the corresponding feature(s) and its/their classification on the leaves of the tree. There are two node types connected by branches. Each branch defines a condition that is tested by each dataset instance. The DT classifies the instances starting from the tree root node, up until the final node that transmits the instance to one of defined classes. The result will be classified according to the class that the end node of the branch belongs to.

Supervised ensemble classifiers are defined as either bagging or boosting algorithms.

Boosting algorithms involve the sequential and incremental generation of an ensemble by applying different ML models for training each new and reweighted versions of the instances from the dataset. As the results, the generated weighted majority vote of the sequence of classifiers is taken and further analyzed. The goal is to emphasize the training instances that were misclassified with previously applied ML models. This concept combines a number of weak ML classifiers in order to enhance the classification performance [15].

Different variations of boosting algorithms appeared with a focus on specific ML issues, mostly related to the dataset (data format, size, number and type of the features), traffic and connectivity characteristics. AdaBoost (Adaptive Boosting) algorithm gives an advantage to the weak classifier by building it iteratively on others depending on the performances achieved by previously applied weak classifier. This approach provides a complex, but much stronger learner. MultiBoosting is an extension to AdaBoost technique for forming decision committees. It represents a combination of the boosting and wagging mechanisms, which provides a possibility of parallel execution and harnesses well the AdaBoost high bias characteristics with superior variance reduction brought by wagging [20]. Yet, boosting lacks good mechanisms to deal with the noisy data and frequently faces the overfitting problem. In that case, it is recommendable to apply LADTree (Logistic Alternating Decision Tree), as the representative of the LogitBoost algorithm which wisely uses its regression learner for providing better generalization and decreasing the percentage of the training errors [21]. It solves the issues of hardly separable classes in binary classification.

Conversely, bagged tree (Bootstrap AGGregation) defines set of ML models, each with an equally-weighted prediction. It is shown that bagged trees deal efficiently with noisy data, while increasing accuracy and reducing variance. Bootstrap aggregation draws random samples with replacement from the initial training set, as an alternative to relying on the same training set which will be applied to every ensemble classifier. Weka platform supports REPTree (Reduces Error Pruning) which is a regression tree logic bagging algorithm. It is an algorithm based on variance error minimization and information gain computing procedures, which generates multiple trees in altered iteration. After spawning the trees, the best one is chosen for further analysis [22].

One of the most applied ensemble ML algorithms, Random Forest (RF), is a specific form of the bagged tree

algorithm that can be used for both classification and regression [23]. In RF, algorithm builds the „forest“ that is represented by multiple DTs which are mostly trained by the “bagging” technique, thus usually providing more accurate and stable prediction. This algorithm introduces additional randomness to the model, during the process of generating the trees. As a valuable advantage, RF searches for the best features from a randomly generated subset of features, contrary to the commonly implemented search for the best feature while splitting a node. As a result, a wider diversity is created thus bringing significant improvements to the model. The RF trees can be even more randomly created by applying the random thresholds for every selected feature. Conversely, the standard DTs perform the search for the best possible thresholds.

Ensemble classifiers have shown some remarkable results when applied to the IoT dataset environments [24, 25]. In this paper we will put the efforts to prove that efficiency by estimating some of the Weka ensemble representatives on the UNSW-NB15 dataset.

C. IDS Evaluation Method

The performance metrics for classification problems in ML relate to the steps necessary to make after the model application and retrieval of the results. This step is responsible for obtaining the information on model efficiency and is based on the calculations for a set of metric. The proper choice of performance metrics will have an influence on the evaluation and comparison of ML algorithms. Majority of performance metrics are based on correct interpretation of the confusion matrix. For every existing class of instances, normal or malicious (anomalous), confusion matrix will provide information on the number of the correctly and incorrectly detected instances. The four basic metrics are: true positives (TP), that corresponds to the cases when both the actual and predicted class of the data are "true"; true negatives (TN), that are the events when both the actual and predicted classes are "false"; false positives (FP) represents the cases when the actual class belongs to the "false" category, while the predicted is defined as "true". Actually, the model has incorrectly made a prediction, and the predicted class was a positive one - number of incorrect positive predictions on a testing instance set; and the false negatives (FN) which stand for the cases when the actual class is "true" while the predicted class is "false". It is "false" as the model has made incorrect prediction and negative as the predicted class a negative one. Based on these four elementary values, we can further calculate a number of measures which correspond to the metrics that aid in providing valid ML comparative results. Accuracy represents a number of correctly predicted instances over a total number of fulfilled predictions. Although it is one of the most explored performance measure, accuracy's major deficiency is that it is inadequate for the case when there is a highly unbalanced dataset, where the target class in the dataset is the actually the majority class.

Alternatively, it is recommendable to rely on the results gathered through precision and recall metrics. Precision (relation 1) represents a ratio of a number of correctly predicted instances and all the predictions made by ML model. It measures the quality of predictions based on the predictors claims on what is positive, regardless of what it misses.

$$precision = PPV = \frac{TP}{TP + FP} \tag{1}$$

On the other hand, recall (sensitivity) is of paramount importance for highly imbalanced datasets. It is a metric that examines all the instances that were classified as from the positive class and makes comparison to a number of all the instances that should have been classified as from the positive class. In other words, it is the ratio of a number of instances that were correctly recalled to a number of all correctly classified events, while precision stands for a mixture of the correct and wrong recalls and could be understood as the ratio of a number of events that can be correctly recalled to a number all events that can be recalled by an used ML algorithm. Specificity is an opposite measure from the recall (relations 2 and 3):

$$recall = TPR = \frac{TP}{TP + FN} \tag{2}$$

$$specificity = TNR = \frac{TN}{TN + FP} \tag{3}$$

F-score covers a trade-off between precision and recall. It stands for the combined metric calculated as the Harmonic mean value of recall and precision (relation 4):

$$F - score = 2 * \frac{(recall * precision)}{recall + precision} \tag{4}$$

F-score metric has mostly an intuitive meaning in the classification procedure, which shows how precise the chosen classifier is in a sense of a number of instances that it correctly classifies, but also shows its robustness if it does not miss to correctly classify a significant number of instances. ROC Area Under Curve (AUC) is another combined metric that has high appreciation in the researchers communities. It evaluates the ratio between the sensitivity and specificity, and provided through the "Sensitivity vs. (1 − Specificity)" plot. Every classification instance of a confusion matrix stands for one point in ROC area. Actually it gives valuable information on classifier performance variation while its discrimination threshold is altered.

V. EXPERIMENTS AND RESULTS

The experimental evaluation of the chosen ML algorithms over the UNSW-NB15 dataset is carried on with Weka (v.3.6), in Windows environment, on 3GHz Intel(R)Core(TM)and 8GB RAM. We have used algorithms from SVM category: SMO and LibSVM C-SVM classifier, and a range of ensemble classifiers: LADTree, REPTree, MultiBoost and RF.

TABLE I.
UNSW-NB15 CLASSIFICATION METRICS RESULTS

Algorithm	Measures			
	Precision	Recall	ROC	Time (s)
LADTree	0.96/0.97	0.99/0.92	0.99/0.99	525
Random Forest	0.98/0.99	0.99/0.97	0.99/0.99	211
REPTree	0.95/0.99	0.99/0.88	0.99/0.99	37
MultiBoost	0.89/0.99	0.99/0.76	0.92/0.92	38
SMO	0.91/0.99	0.99/0.79	0.89/0.89	17716
LibSMV	0.70/0.99	1/0.1	0.55/0.55	51512

Classification learner was set to apply the five-fold cross validation over the 175,341 imported and preprocessed dataset instances. We have run a number of simulations, whereas the obtained results are used for calculating precision, recall, ROC values and time needed for the experiment. The metrics are provided in the form of the obtained values for each class from the dataset - Normal/Malicious (Table I). The very first thing that catches the sight are the time values, as there is a remarkable difference between the two groups of algorithms.

SVM algorithms are far slower in classification, whereas LibSVM C-SVM has shown extremely low performances when compared to all others. It is far the slowest, and has the lowest performance. Base classifiers have their strength and can learn much better from balanced data sets, but when imbalance ratio is high, the ensemble classifiers are classifying the instances with superior performances, wining the time category as well. Among them, RF is far the best ensemble classifier, providing highest values for precision, recall and gaining almost perfect ROC (~1). The obtained results show superiority of the RF tree. As it belongs to the bagged tree category it relies on the bootstrap sampling technique that decreases the variance and enhances the accuracy. Bagged tree algorithms are proficient in learning from extremely imbalanced datasets where the number of malicious/anomaly representatives is low. RF handles much better the over fitting issues and obtains high ROC, while keeping acceptable timeline. Still, when there is lack of time and need for fast examination, a smart solution to apply would be the REPTree. Although being precise ensemble classifier, LADTree has high delay, while boosting representative, MultiBoost, has shown the lowest performances results among the examined set of ensemble classifiers, winning the battle in time category.

I. CONCLUSION

This paper addresses the comparison of several frequently used ML classifiers from the group of SVM like classifiers, namely SMO and C-SMV algorithm, and a range of ensemble algorithms on the other side, namely LADTree, REPTree, RF and MultiBoost. The analysis is based on a range of testing procedures in Weka, with a goal to estimate a set of selected performance metrics and make classifier comparison. As the analyzed UNSW-NB15 dataset belongs to a unbalanced dataset category, for the proper examination of the classifiers we have assumed the need for calculating the precision, recall, ROC and necessary time for classification.

The obtained results have indicated the strong classification capabilities of the RF algorithm, whereas in the timeline more constraining cases the REPTree algorithm stands as the alternative recommendation. Both SVM like algorithms have shown lack of fast enough performances, although SMO can be recommendable in the cases when the time delays are not of the major importance. Although fast in classification, MultiBoost is, for this environmental characteristics, the weakest algorithm from the ensemble group.

ACKNOWLEDGMENT

The work presented in this paper has partially been funded by the Ministry of Education, Science and Technological Development of Republic of Serbia: TR-32025 and TR-32037.

REFERENCES

[1] E. Frank et al, "Data Mining: Practical Machine Learning Tools and Techniques," *The WEKA Workbench*, M. Kaufmann, 2016.

[2] J. Brownlee, "Master Machine Learning Algorithms: Discover How They Work and Implement Them From Scratch," 2017.

[3] B. W. Yap et al., "An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets," *DaEng2013*, Springer Singapore, 2013, pp. 13-22.

[4] N. Moustafa, J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Security J.: A Global Perspective*, 25(1-3), 2016, pp.18-31.

[5] S. J. Stolfo, "KDD cup 1999 dataset", UCI KDD repository, http://kdd. ics. uci. edu. 1999.

[6] M. Tavallaee et al, "Nsl-kdd dataset," http://www. unb. ca/research/iscx/dataset/iscx-NSL-KDD-dataset. html, 2012.

[7] A. Sperotto et al., "A labeled data set for flow-based intrusion detection," *Int. Workshop on IP Operations and Management*, Springer Berlin Heidelberg, 2009, pp. 39-50.

[8] P. Winter et al, "Inductive intrusion detection in flow-based network data using one-class support vector machines," *NTMS*, 2011, pp. 1-5.

[9] S. Robin, V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," *IEEE Symposium on Security and Privacy (SP)*, 2010.

[10] R. Hu et al, "Sweetening the dataset: Using active learning to label unlabelled datasets," 2008.

[11] T. Mitchell, "Machine learning," McGraw Hill Boston, 1997.

[12] A. Basant, N. Mittal, "Hybrid approach for detection of anomaly network traffic using data mining techniques," Procedia Technology 6, 2012, pp. 996-1003.

[13] B. Battista et al, "Pattern recognition systems under attack: Design issues and research challenges," *Int. J. of Pattern Recognition and Artificial Intelligence* vol. 28, n. 7, 2014, pp.1460002.

[14] M. Barreno et al, "The security of machine learning," *Machine Learning*, vol. 81, n. 2, 2010, pp.121-148.

[15] B. Kégl, "The return of AdaBoost. MH: multi-class Hamming trees," arXiv preprint ar-Xiv:1312.6086, 2013.

[16] J. Friedman et al, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors) ," *The annals of statistics*, vol. 28, n. 2, 2000, pp. 337-407.

[17] Š. Gustav et al, "Learning to detect network intrusion from a few labeled events and background traffic," *Autonomous Infrastructure, Management Security*, Springer, 2015, pp. 73-86.

[18] F. Rong-En et al, "Working Set Selection Using Second Order Information for Training Support Vector Machines, *J. of Machine Learning Research*, vol. 6, 2005, pp. 1889–1918.

[19] L. Rokach, O. Maimon, "Data mining with decision trees: theory and applications," (2nd ed.) (Series in Machine Perception and Artifical Intelligence), World scientific, 2015.

[20] G. I. Webb, "Multiboosting: A technique for combining boosting and wagging," Machine learning, vol. 40, no.2, 2000, pp. 159-196.

[21] G. Holmes et al, "Multiclass Alternating Decision Trees," *European Conference on Machine Learning*, Springer, Berlin, Heidelberg, 2002, pp. 161-172.

[22] C. Lakshmi Devasena, "Comparative Analysis of Random Forest, REP Tree and J48 Classifiers for Credit Risk Prediction,", *Int. J. of Computer Applications (0975-8887), ICCCMIT-2014*, 2014.

[23] L. Breiman, "Random Forests," *Machine Learning*, 2001, vol. 45, no. 1, pp. 5-32.

[24] V.Timčenko, and S. Gajin, "Ensemble classifiers for supervised anomaly based network intrusion detection," *ICCP2017*, 2017, pp. 13-19.

[25] I. Franc et al., "Detecting malicious anomalies in IoT: Ensemble learners and incomplete datasets," BISEC2016, 2016, pp. 44-49.