EVALUATION OF IBM WATSON SERVICES FOR IDS IMPLEMENTATION

Bogdan-Valentin VASILICĂ[1], Florin-Daniel ANTON[2], Anca Daniela IONIȚĂ[3]

In this paper, the free services offered by IBM Watson are evaluated for the purpose of implementing an Intrusion Detection System in the context of offline network traffic analysis. It examines IBM Watson's role in IDS, detailing system architecture, network traffic capture, data processing, and algorithm selection with IBM Watson Studio, using the CIC-IDS2017 dataset. The study assesses AI algorithms like LGBM, XGB, Random Forest, and Extra Trees Classifier for detecting intrusions, concluding with an analysis of these algorithms' performance through precision, recall, and F measure metrics, evaluating the effectiveness of IBM Watson in a cloud-based IDS implementation.

Keywords: IDS, IBM Watson, cybersecurity, machine learning algorithms, network traffic analysis, performance metrics

1. Introduction

The evolution of Intrusion Detection Systems (IDS) has been a focal point in the field of cybersecurity, paralleling the rapid advancements in information technology and the escalating complexity of cyber threats. Initially conceptualized in the early 1980s, IDS have undergone significant transformation, evolving from simple anomaly detection algorithms to sophisticated systems capable of real-time analysis and prevention of intricate attacks.

Initially, IDS were simplistic, signature-based systems designed to match known patterns of malicious activities. However, the dynamic nature of cyber threats quickly outpaced the ability of these systems to provide adequate protection. The emergence of anomaly-based IDS marked a significant advancement, focusing on the detection of unusual patterns of behavior as indicators of potential threats, thereby offering a solution for identifying zero-day attacks [1][2].

The utilization of artificial intelligence (AI) and machine learning (ML) techniques in network traffic analysis has become increasingly popular. Various studies have demonstrated the effectiveness of these techniques in detecting cyber-

[1] PhD student, Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: bogdan.vasilica@upb.ro
[2] Assoc. Prof., Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: florin.anton@upb.ro
[3] Prof., Faculty of Automatic Control and Computer Science, National University of Science and Technology POLITEHNICA Bucharest, Romania, e-mail: anca.ionita@upb.ro

attacks. This leap forward has fundamentally changed the landscape of cybersecurity, making IDS more robust and responsive [3][4].

Moreover, the development of distributed IDS and the concept of Intrusion Detection as a Service (IDaaS) showcase the shift towards scalable, cloud-based solutions, indicative of the broader movement towards virtualization and service-oriented architectures in cybersecurity [5][6].

For example, the IBM Watson service is recognized for its capabilities to process and analyze large data sets in a relatively short time. Compared to other solutions available on the market such as Google AI, Microsoft Azure Machine Learning or AWS Machine Learning, IBM Watson offers a set of robust tools that can be accessed for free, and after exceeding a certain limit, at a reasonable cost. IBM Watson was chosen due to the free facilities it offers and the ability to integrate different machine learning techniques and algorithms, making it suitable for our research objectives. The collaboration between IBM and the broader cybersecurity community, particularly through the application of IBM Watson's cognitive computing capabilities, has further advanced the field of intrusion detection. IBM Watson, with its ability to process and analyze vast amounts of unstructured data at unprecedented speeds, has provided valuable insights into emerging threats, enabling faster and more accurate threat detection and response [7]. This partnership exemplifies the potential of combining industry expertise with cutting-edge technology to enhance security measures and protect against sophisticated cyber-attacks.

The evolution of IDS from basic, rule-based systems to sophisticated AI- and ML-powered solutions illustrates the cybersecurity field's adaptability and innovation in response to an ever-changing threat landscape. This paper evaluates the use of IA and ML cloud services for IDS development, describing the system architecture in Section 2, the data set and algorithms in Section 3, the method if Section 4, and the assessment of the results in Section 5. It is important to mention that in this work we will focus on the offline analysis of network traffic using machine learning techniques that will be implemented within the IBM Watson platform. This technique differs from the implementation of a real-time intrusion detection system, because the IBM Watson service would not be able to manage an extremely large amount of data such as those captured in real time, and for these reasons it was desired to achieve a traffic analysis for offline intrusion detection systems.

2. System Architecture

In the paper, there will be no apparatus or installation descriptions. In this section we will discuss about the architecture of the system, about the components and the connection between them and the way they communicate. An IDS is a

security system that acts as a layer of protection over the infrastructure, having the role of generating notifications, alerts, and reports about possible security breaches. IDS systems are divided into two main categories, as can be seen in [8], depending on the monitoring activity. The first category of architecture is Host-based IDS (HIDS), which analyses activity within an individual system, such as: an email server, a web server, or an individual workstation. It only acts on individual systems and usually does not have visibility over the entire set of networks or the networks that surround it. The other category is called Network-based IDS(NIDS), as presented in [9]. This type of architecture has the possibility to visualize the activity of the network of which it is a part and cannot observe the activity that takes place within an individual system.

Nowadays both architecture categories can benefit from the use of AI and ML technologies, but in most of the cases these are implemented locally and not as a services, as this paper is exploring.

The architecture of the system has as a central component the services offered by IBM Watson. Watson is specialized for integrating AI services that solve cognitive problems and is currently considered one the most effective systems in this field. Watson simulates the cognitive process of human learning for a machine, thereby allowing machine learning models to be trained in a customized way for data mining. The access to the tool is through the IBM Cloud platform, known as IBM Watson Knowledge Studio.

The architecture of our evaluation system is depicted in Fig. 1 and is composed of the following elements: network traffic capture system, data processing for uploading, IBM Watson Studio, AI algorithms, as well as modules for selecting the best algorithm, and for visualization and analysis of the results.

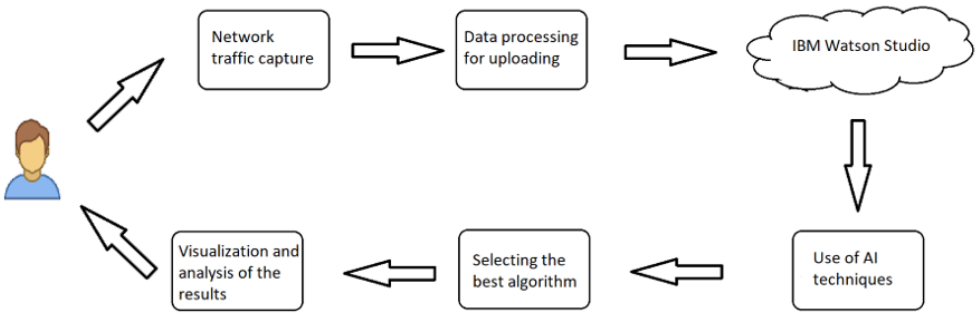


Fig. 1. System architecture in network traffic analysis using the IBM Cloud system

The data set used for this study is CIC-IDS2017, as shown in [10], published by the Canadian Institute for Cybersecurity (CIC). The data collected is based on the work of 25 participants, using the BProfile system, considering different

communication protocols such as HTTP, FTP, HTTPS, SSH and messaging protocols. The data processing stage for uploading consists of converting them from the ".pcap" extension into ".csv" files. In a standard IDS system, the network traffic can be also captured in ".pcap" format, but in our case we focused on processing the data, not on capturing and transferring the data. In this paper, we used the CIC-ISD2017 data set, which collects different network traffic captures, including benign traffic and cyber threats such as DoS, DDoS, PortScan. The dataset gathers a total of 2,827,876 records. The data set was divided into 90% data for training and 10% for testing, totaling 2,544,088 data for training and 283,788 for testing. Considering the constantly changing landscape of cyber attacks, it is important to mention that the CIC-IDS2017 data set represents a basic piece in the detection of cyber attacks despite the fact that it was created in 2017, remaining a valuable tool due to its diversity and complexity. The choice of this data set is based on several considerations that we will illustrate in the following lines. First of all, it includes a varied range of cyber attacks starting from common threats such as DoS or DDoS, ensuring a thorough analysis and an evaluation of detection techniques. In addition, this dataset provides a realistic framework by incorporating real-world network traffic, providing researchers with an environment closely matched to practical scenarios to test and validate their approaches. Another criterion that led to the choice of this data set is its size, being 2.8 million records that facilitate robust training and testing of machine learning models. Although the CIC-IDS2017 dataset was created several years ago, its continued relevance is highlighted by its representation of fundamental attack types and network behavior, making it a fundamental resource for current research in intrusion detection and network security.

In IBM Watson Studio, the data is read, being divided into validation, training and test data. Finally, one selects two algorithm models suitable for solving the problem. Next, Watson performs a first test with one of the automatically selected models, at which point it performs hyperparameter optimization and resource engineering until it obtains four complete output tests. Finally, Watson repeats the tests with the second algorithm and displays the output.

The model was trained within the platform from IBM based on a technique called split. This technique consists of splitting the data set into two different parts, one for model training and one for testing. Thus, the split used was 90% of the dataset for training and the other 10% for testing and validating the proposed model.

3. Data set and AI algorithms

In this section we will detail information about the data set such as: its component, processing methods and AI algorithm that was used.

As mentioned before, the data report was designed based on the activity of 25 participants, using the BProfile system, considering different communication protocols. The dataset is structured in 8 distinct files with the ".pcap" extension, and they reproduce to some extent similar behavior to the real world. Given the extension of these files, namely: ".pcap", the files were converted into ".csv" files in order to be able to be analysed. Within the 8 files, different IP addresses are mentioned for the source and for the destination, the ports, as well as the protocols used and the type of attack.

In the Table 1 one can see the data set that contains information about different attacks during five days of the work week. Due to the limited resource plan, only two of these files were scanned fully, and only one file was partially scanned.

Table 1

The CIC-IDS2017 data set divided by days.

File name	The day of capture	Attacks found
Monday-WorkingHours.pcap_ISCX.csv	Monday	Benign (Normal Traffic)
Tuesday-WorkingHours.pcap_ISCX.csv	Tuesday	Benign, FTP-Patator, SSH-Patator
Wednesday-WorkingHours.pcap_ISCX.csv	Wednesday	Benign, DoSGoldenEye, DoSHulk, DoSSlowhttptest, DoSslowloris, Heartbleed
Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv	Thursday - Morning	Benign, Web Attack – Brute Force, Web Attack – Sql Injection, Web Attack – XSS
Thursday-WorkingHours-Afternoon-Infiltration.pcap_ISCX.csv	Thursday - Afternoon	Benign, Infiltration
Friday-WorkingHours-Morning.pcap_ISCX.csv	Friday - Morning	Benign, Bot
Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv	Friday - Afternoon	Benign, PortScan
Friday-WorkingHours-Afternoon-DDoS.pcap_ISCX.csv	Friday - Afternoon	Benign, DDoS

In this research, we used the free services offered by IBM Watson to implement and test different machine learning algorithms. Due to the limited resource plan, we only processed specific parts of the data set, such as the Friday morning data, which contains benign traffic and bots, and the Friday afternoon data, which contains benign traffic and DDos. Monday's data, which only contains benign traffic, was also analyzed for validation. This aspect has been managed to maximize the efficient use of available resources without compromising the objectivity and relevance of the research. We chose the free option because it

offered a cost-effective solution that met the budget constraints of our research project. In addition, the free tier of the IBM Watson service provided enough functionality to perform the experiments and analyzes required for the study. While full access to paid features would have been beneficial, the free option allowed us to use the capabilities of IBM Watson within the limits of available resources. In the future, there is the possibility to consider another analysis, of another data set with the full capabilities of IBM Watson.

At the end of the process, we should obtain an intrusion detection model, capable of detecting any malicious activity. Taking these aspects into account, to build a robust IDS, traffic captures must be aggregated to form a single unitary data set to be used by the detection model. The distribution of the classes within the dataset is detailed in Table 2.

Table 2

The classes present in the dataset.

Class labels	Number of records
BENIGN	2271320
MALIGN	556556
Bot	1956
DdoS	128025
DoS GoldenEye	10293
DoS Hulk	230124
Dos Slowhttptest	5499
DoS Slowloris	5796
FTP-Patator	7935
HeartBleed	11
Infiltration	36
PortScan	158804
SSH-Patator	5897
Web Attack – BruteForce	1507
Web Attack – SqlInjection	21
Web Attack - XSS	652

In machine learning, a common step is researching and building algorithms that can learn and make predictions about data, as presented in [11]. Based on this, it is possible to train a model so that it can differentiate between which requests are (malicious) attacks and which are common (benign) requests. These models perform the classification through machine learning algorithms that aim to differentiate which cases are of one type or another, depending on the training these models go through. The dataset utilized for both training and evaluating the model comprises a blend of standard traffic patterns for training purposes and anomalous traffic patterns for testing. Model training was performed on the IBM Watson Studio platform.

Within the IBM platform, the algorithms that were used to analyze csv files are: LGBM, XGB, Random Forest and Extra Trees Classifier. These algorithms were chosen by the IBM Watson utility as the most suitable for the data sets provided. The selection of algorithms was based on their performance in previous studies and their ability to adapt to large and complex datasets. These algorithms classify objects based on prior knowledge, obtained through a training base containing labeled objects, this process is called the training or learning phase. Object recognition occurs by similarity and what separates one class from another is the difference between the objects. In this sense, the more different the objects of one reference class are compared to another, the easier they will be classified by the algorithm.

Subsequently we shall detail the algorithms used in our study, depending on their characteristics.

a. **Random Forest** is a formation of classification trees, as illustrated in [12]. This type of algorithm generates a multitude of regression trees. Each tree is constructed from a different bootstrap sample of the genuine data, using a tree classification algorithm. These decision trees use graphs for decision modelling, each node in the graph being represented by a question, and the branches are the answers to those questions. By composing various such graphs, random forests are born. This method, in addition to the high degree of accuracy it shows, is quite difficult to interpret and for this fact it is also considered a black box type method, as illustrated in [13]. In addition to this fact, it is quite popular due to the high accuracy, but also to the relatively low implementation costs. Object classification is achieved by combining one or more algorithms.
b. **Light Gradient Boosting Machine** is based on the Gradient Boosting Decision Tree algorithm. The LGBM classifier is a gradient boosting framework that uses the leaf-to-leaf tree-based learning algorithm (horizontal branching). The main difference between LightGBM and other tree-based algorithms is that the tree grows vertically. It is an algorithm that has started to gain more and more popularity, as it is considered fast, with relatively low memory costs and with support for processing on GPUs. This algorithm includes more than 100 parameters that can be adjusted for processing. However, when using IBM Watson, the parameters were adjusted and adapted to our test predictions automatically.
c. **eXtreme Gradient Boosting** is a machine learning algorithm that operates on the principles of decision trees within a gradient boosting framework. For predictions involving non-structural data like images or text, artificial neural networks are typically the superior option compared to other algorithms. Conversely, for handling structured data of relatively small dimensions, tree-based algorithms are often preferred for their effectiveness.

d. Extremely Randomized Trees Classifier is a type of ensemble learning technique that fuses the results of multiple decorrelated decision trees summed into a "forest" to output its classification result, as presented in [14]. As a basic principle, it is extremely similar to random forests (Random Forest), and the only difference is the construction method of the decision trees. Each tree, within this type of algorithm, is made up of the original sampled training data. Then, within each node, each tree is randomly given a sample of k features, and each tree must choose the best method to partition the data based on some mathematical relationships. This random sampling of features leads to the creation of several decorrelated decision trees.

4. Evaluation Method

Choosing the best algorithm is an essential step in any AI program, as there are a multitude of options; considering this, in-depth knowledge of the various strengths and weaknesses of different applications is essential. This section discusses the metrics used to evaluate the performances of the algorithms and the proposed model, as well as the results obtained through these evaluations. We consider metrics like precision, recall and F measure, with the equations given as follows.

A diagram of the proposed IDS model is illustrated in Fig. 2. While developing this application, we created a model based on the steps necessary to implement a robust IDS system. So, we collected the data from the network traffic, after which the data processing took place, including the elimination of duplicate data or those containing incomplete or incorrect elements. Immediately after, the processed data set was divided into 3 large components: training set, validation set, and test set. The training set is used for the learning stage of the algorithm, the validation set in the final analysis stage, and the test set goes towards prediction.

The confusion matrix summarizes the classification performance of a classifier against some test data. The result can be of one or more types of classes, as can be seen in [15]. In much simpler terms, this matrix is a two-dimensional table, in which the actual and predicted values are noted. Besides these, each term of the matrix contains the values for: true negative (TN), true positive (TP), false negative (FN), false positive (FP), as is illustrated in the Table 3.

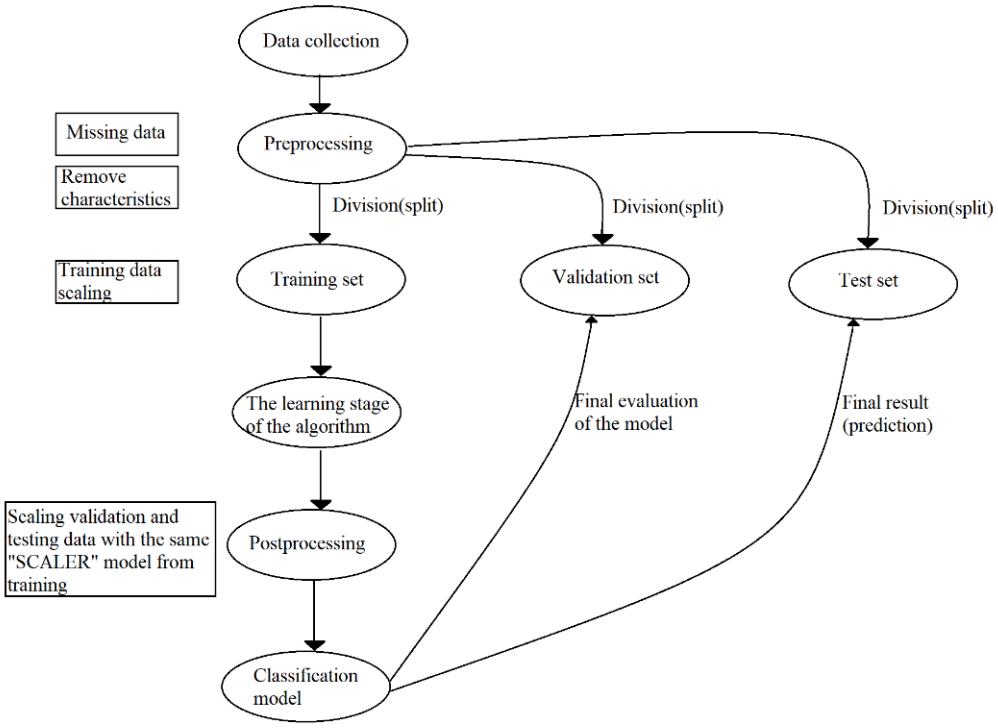


Fig. 2 The proposed IDS model

Table 3

Confusion matrix

	Forecasted negative	Forecasted positive
Real negative	True negative (TN)	False positive (FP)
Real positive	False negative (FN)	True positive (TP)

In the following lines we shall explain what each term in the confusion matrix means. Let us start with the true positive value, or TP. This represents the case where both the actual class and the predicted class have the value 1. True negative, or TN, occurs when both classes present the value 0. The false positive, or FP case, occurs when the actual class has the value 0 and the predicted 1 The last case, false negative, occurs when the actual class has the value 1 and the predicted class has the value 0.

Another important metric to consider is accuracy. This can be defined as the number of correct positive solutions, i.e., TP values, divided by the sum of TP and classifier predicted results - FP. The calculation relation can be visualized in the equation 1.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

The recall value, or sensitivity, as named in some publications, is given as the number of correct positive values, i.e. TP, divided by the sum of the correct positive values and the data that are positive, FN. The mathematical relation is given in equation 2.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

Another important metric is the F1 measure. In simple terms, it is the harmonic mean between recall and precision. This metric can take values between 0 and 1. It is essential because it shows how robust a classifier is, that is, how correctly the cases were identified. If the recall value is small and the precision is high, the model shows high accuracy. In other words, the higher the value of this metric, the more robust the model. The expression of this metric is:

$$F1 = \frac{1}{\frac{1}{\frac{TP}{TP + FP}} + \frac{1}{\frac{TP}{TP + FN}}} \tag{3}$$

5. Results and Discussion

This section describes the experiments performed and their results. For the first experiment conducted, using the Friday morning dataset, containing both benign traffic and malicious bot activity, the results obtained can be seen in the confusion matrix from Fig. 3.

Observed	Predicted		
	BENIGN	Bot	Percent correct
BENIGN	194	1	99.5%
Bot	4	18216	100.0%
Percent correct	98.0%	100.0%	100.0%

Less correct — More correct

Fig. 3 Confusion matrix for the first experiment

The matrix is made based on the LGBM algorithm and is also considered by the analysis application to be the best variant, being on the first place in the ranking. From it, one can extract the following information: TN = 194, FP = 1, FN = 4, TP = 18216.

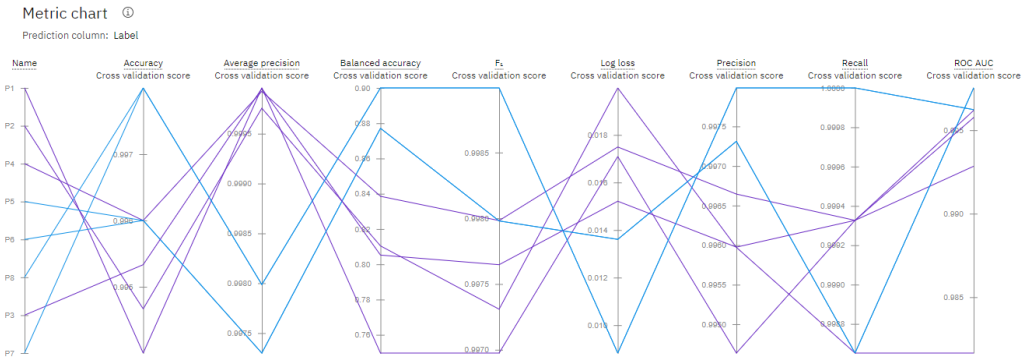


Fig. 4 Metric chart for the 8 Pipelines, where the blue chart corresponds to LGBM and the purple one to XGB

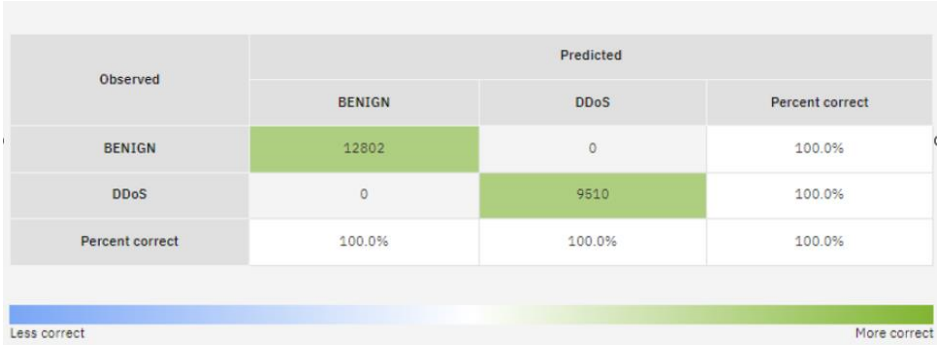
Pipeline leaderboard

	Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Average precision Cross Validation	F1 Cross Validation	Precision Cross Validation	Recall Cross Validation	Enhancements	Build time
★	1	Pipeline 8	LGBM Classi...	0.998	0.998	0.999	0.998	1.000	HPO-1 FE HPO-2	00:00:58
	2	Pipeline 7	LGBM Classi...	0.998	0.998	0.999	0.998	1.000	HPO-1 FE	00:01:20
	3	Pipeline 4	XGB Classifier	0.996	1.000	0.998	0.997	0.999	HPO-1 FE HPO-2	00:01:59
	4	Pipeline 5	LGBM Classi...	0.996	0.997	0.998	0.997	0.999	None	00:00:21
	5	Pipeline 6	LGBM Classi...	0.996	0.997	0.998	0.997	0.999	HPO-1	00:00:28
	6	Pipeline 3	XGB Classifier	0.995	1.000	0.996	0.996	0.999	HPO-1 FE	00:01:43
	7	Pipeline 2	XGB Classifier	0.995	1.000	0.997	0.996	0.999	HPO-1	00:01:36
	8	Pipeline 1	XGB Classifier	0.994	1.000	0.997	0.995	0.999	None	00:00:24

Fig. 5 Ranking of Pipelines by characteristics

As shown in Fig. 4, which displays the metric chart for the 8 pipelines with the blue chart representing LGBM and the purple one representing XGB, the algorithm used by Watson (LGBM) achieved significantly better precision metrics compared to XGB. Fig. 5 ranks the pipelines by their characteristics and illustrates that while XGB excelled in accuracy, LGBM performed better on the recall metric, with only marginal differences compared to XGB.

The second experiment used the Friday data set, more specifically the afternoon ones, containing benign traffic but also DdoS-type threats. The confusion matrix of this experiment is illustrated in Fig. 6.



Observed	Predicted		
	BENIGN	DDoS	Percent correct
BENIGN	12802	0	100.0%
DDoS	0	9510	100.0%
Percent correct	100.0%	100.0%	100.0%

Fig. 6 Confusion matrix for the second experiment

The matrix is made based on the Extra Trees Classifier algorithm. From this, one can extract the following information: TN = 12804, FP = 0, FN = 0, TP = 9510. Also, the result is on the first place in the ranking.

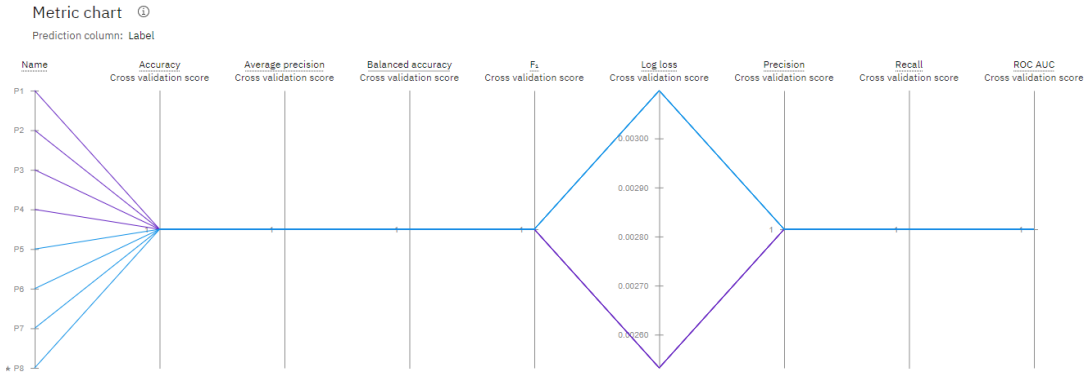


Fig. 7 Metric chart for the 8 Pipelines, where the blue chart corresponds to Random Forest and the purple one to Extra Trees

Pipeline leaderboard

	Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Accuracy (Optimized) Holdout	Average precision Cross Validation	F1 Cross Validation	Precision Cross Validation	Recall Cross Validation	Enhancements	Build time
★	1	Pipeline 1	Extra Trees Classifier	1.000	1.000	1.000	1.000	1.000	1.000	None	00:00:23
	2	Pipeline 2	Extra Trees Classifier	1.000	1.000	1.000	1.000	1.000	1.000	HPO-1	00:00:31
	3	Pipeline 3	Extra Trees Classifier	1.000	1.000	1.000	1.000	1.000	1.000	HPO-1 FE	00:00:50
	4	Pipeline 4	Extra Trees Classifier	1.000	1.000	1.000	1.000	1.000	1.000	HPO-1 FE HPO-2	00:00:42
	5	Pipeline 5	Random Forest Clas...	1.000	1.000	1.000	1.000	1.000	1.000	None	00:00:24
	6	Pipeline 6	Random Forest Clas...	1.000	1.000	1.000	1.000	1.000	1.000	HPO-1	00:00:35
	7	Pipeline 7	Random Forest Clas...	1.000	1.000	1.000	1.000	1.000	1.000	HPO-1 FE	00:00:54
	8	Pipeline 8	Random Forest Clas...	1.000	1.000	1.000	1.000	1.000	1.000	HPO-1 FE HPO-2	00:00:48

Fig.8 Ranking of Pipelines by characteristics

Fig. 7 presents the metric chart for the 8 pipelines, where the blue chart represents the Random Forest algorithm and the purple one corresponds to Extra Trees. As depicted, the performance metrics vary between the two algorithms. Following this, Fig. 8 ranks these pipelines by their characteristics, providing a clearer overview of each algorithm's performance across different metrics. The ranking and associated data might suggest that, depending on the metric of interest, one algorithm may have advantages over the other. Regarding the ranking of the Pipelines, all the characteristics obtained the value 1. This event can be explained in two ways. Either the data was processed optimally, resulting in some outstanding data, or the data was insufficient, and training and validation was not properly processed.

If we were to make a comparison between the two matrices obtained, in the case of the first experiment the number of threats was much higher than that of ordinary traffic, while in the case of the second experiment ordinary traffic was remarkably higher compared to the malignant one. So, the malicious traffic in smaller amount was easier to be detected.

The third file that was analysed was the traffic captured on Wednesday, which contains, in addition to benign traffic, numerous threats such as: DoSGoldenEye, DoSHulk, DoSSlowhttptest, DoSslowloris, Heartbleed. Due to the large size of this file, it was stopped from running by IBM's utility because we exceeded the computing capacity allocated by the Watson Machine Learning service plan. The resulting confusion matrix can be seen in Fig. 9.

Observed	Predicted						
	BENIGN	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	Heartbleed	Percent correct
BENIGN	41691	3	11	0	3	0	100.0%
DoS GoldenEye	0	1025	1	1	0	0	99.8%
DoS Hulk	8	0	17273	0	1	0	99.9%
DoS Slowhttptest	2	0	0	520	1	0	99.4%
DoS slowloris	2	1	0	2	533	0	99.1%
Heartbleed	1	0	0	0	0	1	50.0%
Percent correct	100.0%	99.6%	99.9%	99.4%	99.1%	100.0%	99.9%

Less correct More correct

Fig.9 Confusion matrix for the Wednesday dataset, partially done

Considering that the experiment has not been completed, it can be noted how the percentage of correctness gradually decreases, for the Heartbleed threat only two values are identified with a percentage of correctness of 50%.

Thus, we presented an intrusion detection system using IBM Watson as a form of training and evaluation for a machine learning model. This model was developed based on the CIC-IDS2017 dataset and was evaluated against other algorithms known in the literature such as Random Forest, LGBM, XGB, Extra Trees Classifier. Considering the limited financial plan by those from IBM, we had the opportunity to analyze only two of the files presented. To maximize security, all retrieved files should be concatenated to defend against other types of attacks in addition to those presented. In this sense, we have gone through all the stages for setting up an IDS system. After cleaning and loading the data, the algorithm learning step followed. After this moment we had the opportunity to see the ranking of the algorithms and the calculated metrics. There were times when the LGBM algorithm performed better on the accuracy side compared to XGB, which ranked better when it comes to the F1 parameter, providing the robustness.

6. Conclusions

In this work, the CICIDS-2017 data set was analyzed with the help of the free service provided by IBM Watson. Three data sets were chosen and loaded, and with the help of IBM Watson, the efficiency of traffic analysis techniques and the comparison of various automatic learning algorithms were evaluated. The research did not evaluate a full intrusion detection system (IDS), but rather analyzed the performance of offline traffic classification techniques. The efficiency of these techniques has been demonstrated against historical data, but a comparative analysis with other current traffic analysis techniques would provide a more complete perspective. Our results highlight the potential of these techniques in the context of intrusion detection, but we recommend further research to validate these findings in an operational environment. The application of AI algorithms like LGBM, XGB, Random Forest, and Extra Trees Classifier has provided significant insights into intrusion detection. The evaluation of these algorithms using metrics such as precision, recall, and F measure underlines their effectiveness in identifying and mitigating cyber threats. The results from this study underscore the necessity of continual advancements in cybersecurity technologies to combat evolving cyber threats.

There are also some aspects that should be considered: such a system can be successfully implemented to protect systems/applications hosted in cloud in the same location as IBM Watson, in order to reduce the traffic in internet. IBM Watson services can also be used for a remote IDS implementation, but, in this case, one must accept that the detection may not be executed in real time; alternatively, one may implement a real time detection, but for a reduced set of data. In this case a local data preprocessing should be implemented in order to select only a part from the network traffic. Future work will focus on integrating more diverse datasets and

exploring the potential of emerging AI techniques to further bolster the efficiency of IDS.

REFERENCES

[1]. A. Khraisat, et al., "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity, vol. 2, no. 1, 2019, pp. 1-22.

[2]. B. A. Tama, M. Comuzzi, and K.-H. Rhee, "TSE-IDS: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," IEEE Access, vol. 7, 2019, pp. 94497-94507.

[3]. S. Hajj, et al., "Anomaly-based intrusion detection systems: The requirements, methods, measurements, and datasets," Transactions on Emerging Telecommunications Technologies, vol. 32, no. 4, 2021, e4240.

[4]. E. Alhajjar, P. Maxwell, and N. Bastian, "Adversarial machine learning in network intrusion detection systems," Expert Systems with Applications, vol. 186, 2021, 115782.

[5]. V. Chang, et al., "A survey on intrusion detection systems for fog and cloud computing," Future Internet, vol. 14, no. 3, 2022, 89.

[6]. M. Herman, et al., Nist cloud computing forensic science challenges. Gaithersburg, MD, USA: US Department of Commerce, National Institute of Standards and Technology, 2020.

[7]. E. Opara, H. Wimmer, and C. M. Rebman, "Auto-ML cyber security data analysis using Google, Azure and IBM Cloud Platforms," in 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET), IEEE, 2022.

[8]. I. Dutt, S. Borah, and I. K. Maitra, "Immune system based intrusion detection system (IS-IDS): A proposed model," IEEE Access, vol. 8, 2020, pp. 34929-34941.

[9]. Z. Ahmad, et al., "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," Transactions on Emerging Telecommunications Technologies, vol. 32, no. 1, 2021, e4150.

[10]. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization”, in 4th International Conference on Information, Canadian Institute for Cybersecurity, CICIDS‘17: Intrusion Detection Evaluation Dataset (CICIDS2017).

[11]. B. Mahesh, "Machine learning algorithms-a review," International Journal of Science and Research (IJSR), vol. 9, no. 1, 2020, pp. 381-386.

[12]. E. Y. Boateng, J. Otoo, and D. A. Abaye, "Basic tenets of classification algorithms K-nearest-neighbor, support vector machine, random forest and neural network: a review," Journal of Data Analysis and Information Processing, vol. 8, no. 4, 2020, pp. 341-357.

[13]. V. Buhrmester, D. Münch, and M. Arens, "Analysis of explainers of black box deep neural networks for computer vision: A survey," Machine Learning and Knowledge Extraction, vol. 3, no. 4, 2021, pp. 966-989.

[14]. M. R. C. Acosta, et al., "Extremely randomized trees-based scheme for stealthy cyber-attack detection in smart grid networks," IEEE Access, vol. 8, 2020, pp. 19921-19933.

[15]. A. Shafique, et al., "Detecting the security level of various cryptosystems using machine learning models," IEEE Access, vol. 9, 2020, pp. 9383-9393.