IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 April 2023, accepted 25 April 2023, date of publication 4 May 2023, date of current version 11 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3272894

RESEARCH ARTICLE

Traffic Classification in IP Networks Through Machine Learning Techniques in Final Systems

JORGE GÓMEZ [1], (Senior Member, IEEE), VELSSY HERNANDEZ RIAÑO [1], AND GUSTAVO RAMIREZ-GONZALEZ [2]

[1]System Department, Universidad de Córdoba, Córdoba 14071, Colombia
[2]Telematics Department, University of Cauca, Popayán 190001, Colombia

Corresponding author: Jorge Gómez (jeliecergomez@correo.unicordoba.edu.co)

This work was supported in part by the SOCRATES Research Group of the Department of Systems Engineering and Telecommunications, and in part by the Universidad de Cordoba under Project FI-05-19.

ABSTRACT Data centers in higher education institutions, as well as those of large corporations, face challenges in terms of traffic flow management. In some cases, due to the limited hardware resources used for this purpose, and in others, despite having enough high-performance equipment, the centers lag behind when the traffic flow grows exponentially due to the memory limitations of the devices, which slows down the network performance. The contribution of this investigation work is the implementation of a classifying elephant and mice system using machine learning techniques for the early detection with the first flow based on the dynamic calculation of the threshold, according to the input parameters of the final system. In the first instance, training algorithms are used to determine the best performance, then the proposed algorithm determines the model with the best prediction, obtained from the supervised learning algorithm trained in off line mode. Finally in the phase of online prediction, the algorithm is capable of predicting with high precision the type of traffic in terms of the input flow, and updates in a dynamic way the threshold to determine whether the traffic is elephant or mice. With this information the network hardware can decide then to route the flows according to their characterization. According to the results, the model that best generates predictions is the decision tree with a 100% confidence level.

INDEX TERMS Packet sniffer, Wireshark, machine learning, traffic classification.

I. INTRODUCTION

The main objective of network management is to preserve network availability and improve performance. Network management is becoming a challenge with the growth in network size, traffic volume and the diversity of Quality of Service (QoS) requirements. There is a wide range of applications with different requirements and constraints on network resources. Often, several flows with other characteristics can be found competing for network resources, and consequently, these resources are not equally used by all flows [1].

In the Internet, a flow is classically defined as the set of packets with the same source and destination IP addresses, the same source and destination port numbers and the same protocol type. It is known that the distribution of flow lengths and sizes on the Internet follows the Pareto principle; most traffic is composed of a relatively small number of flows. These flows are called elephant flows. The remaining flows, which are large in number but carry very little traffic, are called mice flows [2]. The latter carry most traffic in bytes, but most flows are mice. It is common to observe that 95% of the flows are mice, but elephants provide more than 95% of the overall volume. Mice are generally generated by web browsing, while file transfers cause elephants. This behavior is known as the elephants and mice phenomenon and is considered one of the few invariants of Internet traffic [3]. This phenomenon indicates that most flows only transfer a small fraction of data center traffic, i.e., mice flows, while very few transfer a large fraction of data center traffic, i.e., elephant flows [4].

In statistics, this phenomenon is also called "mass-count disparity." "Mass-count disparity" states that, for heavy-tailed distributions, most of the mass in a set of observations is concentrated in a very small subset of the observations [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Yiming Tang.

In order to exploit such property for traffic engineering purposes, it is necessary to identify which flows carry the most bytes. Examples of traffic engineering applications include redirection and load balancing of elephant flows. The interest in this approach to traffic engineering is that by treating a relatively small number of flows differently, a large portion of the total traffic can be affected. For this to be practical, elephant flows should remain for reasonably long periods so that a traffic engineering application does not need to change its policy or state frequently. There is no systematic way to choose the criterion that isolates large-volume flows [4]. The criterion should allow considering any particular flow and deciding whether it should be categorized as an elephant or a mice. Other studies have selected a particular criterion and then examined other problems given this fixed criterion. For example, in [6], an elephant is any flow with a rate greater than 1% of the link utilization; while in [7], an elephant is any flow whose maximum velocity exceeds the mean plus three standard deviations of the aggregate link flow.

For traffic engineering purposes, it is important to design online algorithms to identify elephants and estimate their statistics. Because such algorithms are generally implemented on network devices such as routers or deep packet inspection devices running at very high speeds, limitations in computing power and memory are present in their implementation [8].

One of the main problems faced by the network infrastructure of data centers is traffic congestion. In the first instance, this is largely due to the intrinsic limitations of the network hardware, such as buffer sizes. Another aspect to highlight is the routing of packets, given the same network planning. A major problem is the early classification of traffic (mice or elephant), so the network hardware can make decisions when routing flows depending on the nature of the traffic itself. Based on the latter, it is expected that once the packets arrive at the switching systems, they will be able to identify what type of traffic it is with the first arrival flow and decide in a more optimal way how to route the traffic more efficiently.

In the present study, we propose implementing machine learning models for intelligent prediction of elephant traffic. The objective is to find the model that best fits the predictions, and to detect, based on the first flow, the corresponding type of traffic. With this information, it will be possible to define the flow routing in the network hardware immediately. The paper is organized as follows: introduction, materials and methods, results, conclusions and future work.

II. MATERIALS AND METHODS

A. BACKGROUND

The phenomenon of elephant and mice flows can lead to poor network performance [2]. Limitations in storage and processing resources make it impossible to collect and monitor all network packets. In traffic engineering, what is really important is the volume of flows, not the quantity. Therefore, elephant and mice classification play an important role in this area [9].

The elephant flow has a high bandwidth demand, and the mice flow needs a low delay. When forwarded along the same path, the two types of flows may have conflicts [10]. Elephant flows could be discarded by early mice flooding, and if elephant flows occupy the buffer in a switching device, mice flows arriving at that device will be delayed. Elephant flows can saturate buffers, leading to queuing delays affecting latency-sensitive flows or mice flows. Therefore, after identifying elephant flows, conflicts must be avoided by making different routing strategies for the two types of flows [11].

Due to the need to identify the different requirements of the services provided in the networks to have an accurate knowledge of the network behavior, machine learning (ML) has been used to extract knowledge from the data through methods to classify the network traffic [12].

Traditionally, the main solutions to identify elephants include counting, hashing and sampling-based methods [9]. In the count-based method, a limited number of counters are used to find frequent elements in a data stream. In the hash-based method, one- or two-dimensional counters are used to construct a hash table to estimate the frequencies of different items. And finally, in the sampling-based method, the frequency of items is estimated by periodic sampling in the data stream. Although these three types of methods can achieve high accuracy with low complexity, they identify elephant flows only after a large volume of traffic passes and cannot provide an early prediction. Traffic identification performance can be further improved by predicting elephants and mice for later detection. However, most existing methods for elephant detection are designed for offline classification, and it is difficult to employ them for online traffic identification, which requires early identification [9].

There are two main aspects to the problem of Internet traffic flow characterization [13]: (i) how to collect flows efficiently and (ii) how to accurately infer general traffic behavior from the collected data. Approaches for flow characterization are based on: (i) statistical sampling of packets [6], [14] or (ii) inferring traffic characteristics based primarily on flows that carry a large number of packets or bytes and are long-lived in nature, i.e., elephants, while ignoring flows that carry a very small number of packets or bytes and are short-lived in nature, i.e., mice [6], or (iii) using estimation algorithms on data structures with losses, e.g., bloom filters and hash tables [15], [16] to recover missing information. However, even in sampled traffic, separating elephants and mice is a cumbersome task [14], as there are no standard approaches to draw the dividing line between the two. This scheme eliminates the limitation to infrastructure, and its bandwidth consumption is also acceptable. Unfortunately, existing sampling-based methods still sample too many useless packets, resulting in high sampling overhead and long detection time [17].

Current elephant flow detection methods can be classified into five categories [4]: (1) sampling [6], (2) individual statistics extraction [18], (3) host-based detection [19], (4) switch-trigger, and (5) collection of all packets [20].

State-of-the-art forwarding in data centers uses Equal Cost Multipath (ECMP) to statically split flows among several equal-length paths for load balancing [21]. This static forwarding method does not take flow size into account. Under this mechanism, it could route multiple elephants flows on the same path and eventually cause overflows in the switching buffers. It could result in the degradation of the overall network bandwidth utilization [4].

To improve system performance and satisfy user requirements as much as possible, elephant and mice flows should be efficiently detected and scheduled [17]. Several elephant flow detection schemes have been proposed; [4], [18], [19], [22]. However, these schemes introduce a large sensing overhead or require specialized installation, generating costly software and hardware investment. Specifically, host-based flow detection requires the unification of the operating systems of all hosts for flow monitoring [19]. Extraction-based detection schemes must periodically collect information on all switches' flows, generating large bandwidth and time costs [4], [22]. To reduce the bandwidth consumption for flow information collection, switch activation detection sets the threshold on specialized switches beforehand [19]. It is difficult to determine the appropriate thresholds, and the static threshold may cause inaccurate detection results due to dynamic network environments.

B. RELATED WORKS

This section presents studies on optimizing network performance by identifying and managing elephant and mice flows. In [23], a new routing strategy is presented based on classifying flows into mice and elephant according to their size to minimize the flow completion time. They use the machine learning technique called association rules to generate the forwarding rules and route each flow according to its class, i.e., mice or elephant. Experimental results show successful identification in 80% of the flows. Moreover, their class-based routing outperforms basic routing strategies in flow completion time, throughput, and packet loss by almost 47%, 41%, and 23%, respectively. In [20], minimizing network congestion and load imbalance in SDN networks is the main focus. They implement a routing algorithm that performs elephant flow identification and routing. A flow is identified as an elephant only if the number of bytes in the TCP buffer exceeds a predefined threshold. All elephant flows are split into multiple sub-flows. To improve load balancing and link utilization, each subflow is routed through a different path according to the link utilization. In [19], the Mahout traffic management system is presented to minimize the workload of the network controller by identifying and routing flows. Mahout detects elephant flows at the end host through a correction in the operating system. If the amount of data in the TCP buffers exceeds a threshold, then it is an elephant flow. When an elephant flow is detected, the network controller is notified using an in-band signalling mechanism. The controller calculates forwarding rules based on link utilization; the link with the lowest utilization is selected. Mice flows are routed using ECMP. In [24], the problem of flow classification in software-defined data centers is addressed. They propose a fast switch-level elephant flow detection method, called FlowSeer, which uses data flow mining. The developed method collects statistics from each flow's first packets to train flow classification models. These statistics include the IP address and the maximum and minimum packet size. FlowSeer allows the switch to identify elephant flows. Finally, elephant flows are routed through less congested routes to improve throughput. In [25], they propose an online flow size prediction to improve network routing using several machine learning techniques, including neural networks, Gaussian process regression, and online Bayesian moment matching. The prediction is based on certain information collected from the first packets, including source IP, destination IP, source port, destination port, protocol and the size of the first three packets. After predicting the flow size, elephant flows are routed through the least congested routes. In [17], an efficient sampling and classification approach (ESCA) is proposed to detect elephant flows with low bandwidth consumption in two phases. In the first phase, an algorithm to estimate the arrival interval of elephant flows is proposed, which reduces the sampling time and improves the sampling efficiency. In the second phase, ESCA uses a new classification algorithm to determine whether samples belong to elephant flows.

In [26], unsupervised and semi-supervised machine learning approaches classify flows in real time. They propose a Gaussian mixture model combined with an initialization algorithm to develop a general-purpose method to aid network site-based classification regarding data transfers, flow rates, and durations. The results show that, despite variable flows at each site, the proposed algorithm can cluster elephant and mice flows with an accuracy rate of 90%.

In [23], an elephant flow classifier is proposed using association rules implemented in switches for traffic routing in SDN networks. Its main objective is to reduce file transfer time. The architecture of the proposed model, called LUNA, consists of three modules: (i) monitoring module, which extracts traffic information generated by user requests, such as user IP, request date and size in bytes of the response, (ii) association rule generator module, performs flow classification using the K-means algorithm and then relates users to the flow classification using association rules, (iii) routing rule generator module, where flow routing is performed according to the rules obtained in the previous stage. The results show more than 80% accuracy in identifying flows according to their classification as elephants or mice.

In [27], a heavy traffic prediction system for SDN networks is proposed by implementing a simple decision tree algorithm in the switches that constitute the data plane to prevent hardware overload due to its limitations in performing information processing and storage. The proposed model is composed of two stages: the first stage is performed in the control plane and is responsible for extracting information

from the flows to be used by training algorithms offline using decision trees, and the second stage performs online inference for the classification of incoming traffic by labeling it with a flag as heavy flow or not. The results indicate an accuracy of 85% when the first five packets of a flow have been received and an accuracy of 98% when the first 20 packets of a flow have been received.

In [7], previous studies of Internet flow characterization with different classification schemes are shown. A flow classification can be found according to size: small size flows are called mice flows, and large size flows are called elephant flows; by duration: where longer flows are called turtle, and shorter flows are called dragonfly; by speed: heavy flows are called cheetah and light flows are called snails; and by burstiness: bursty flows are called porcupine and non-bursty flows are called stingray, they are also called alpha and beta traffic. As a general rule, a classification was adopted according to which a flow in a trace is considered an elephant if its size is larger than the average size plus three times the standard deviation of the size. A flow in a trace is viewed as a turtle if its duration is longer than the average duration plus three times the standard deviation of the duration. Similarly, a flow in a trace is a cheetah if its rate is greater than the average rate plus three times the standard deviation of the rate. Finally, a flow is labeled as a porcupine if its burst is greater than the average burst plus three times the standard deviation of the burst.

In [31], a solution is proposed for offloading network functions backed by hardware programmable tables, using elephant flow detection by processing the first packet of a flow as the offloading criterion. The detection is achieved by training machine learning algorithms, obtaining higher performance results versus a scenario using the packet sampling model.

Finally, in [32], an elephant flow detection method for SDN networks is proposed, adopting two different classification components in the control and data planes. The first component is based on a count-min algorithm placed at the switch side to filter the mice flows between candidates and non-candidates. Subsequently, flows that are identified as non-candidates enter the second detection component located on the controller side, to be filtered by a very fast decision tree (VFDT) algorithm that will use parameters delivered by the switch, such as IP addresses and port numbers to identify elephant flows and forward them along a scheduled route. Simulated results demonstrate higher detection accuracy and efficiency than other existing methods.

C. SELECTION OF ELEPHANT TRAFFIC CLASSIFICATION TECHNIQUE IN IP NETWORKS ON END SYSTEMS

1) TRAFFIC CAPTURE IN IP NETWORKS

The host-based detection method [19] was employed for elephant traffic classification. In addition, the count-based technique was used. This method uses a limited number of counters to find frequent elements in a data stream, according to [9]. Considering this technique, a threshold is defined to do elephant traffic counting. To calculate the traffic threshold, we use Chebyshev's theorem. This function takes as reference the mean plus three standard deviations, i.e., whatever is above this value will be considered elephant traffic [7]. See equation 1.

$$Threshold = \bar{f} + 3\left(\sqrt{\frac{\sum_{i=1}^{n}(f_i - \bar{f})^2}{n-1}}\right)$$

EQUATION. 1. Equation to calculate the threshold.

According to the literature, traffic classification can be done by parameters such as size, speed and time estimation. In this study, we chose only the size parameter to make predictions due to preprocessing and data processing costs. The intention is to detect with the first flow and determine the type of traffic. While the other parameters are an option, they slow down the process because they would require more time, making them less efficient when making real-time predictions.

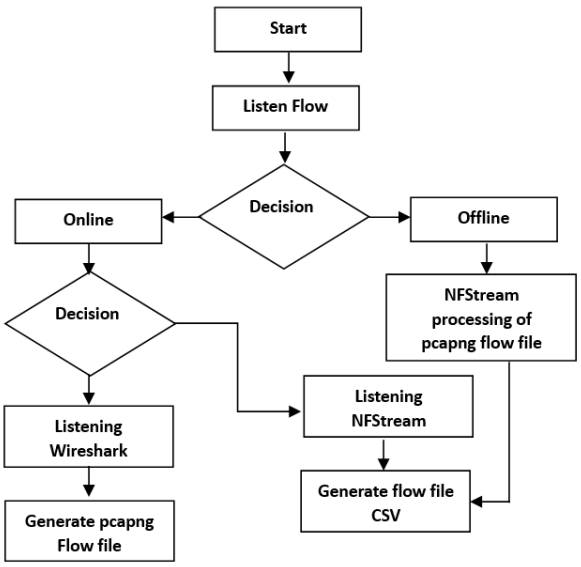

FIGURE 1. Flowchart for IP traffic capture.

The following flowchart, shown in Figure 1, describes the traffic capture process. The traffic capture process starts, and then it is decided whether the flow captures will be in offline or online mode. Next, if the online decision is made, it is again determined whether the capture will be performed with the Wireshark sniffer or NFStream. In the eventual case of selecting the stream with Wireshark, these will be saved in pcapng format. For capture with NFStream, NFStream automatically saves the file in CSV format. Table 1 shows the characteristics of the stream stored in the file.

On the other hand, if chosen offline, NFStream extracts the flows from the pcapng file and converts them into CSV

TABLE 1. CSV file stream characteristics, source (NFStream, 2022).

Name of the identifier
src_ip
src_mac
src_oui
src_port
dst_ip
dst_mac
dst_oui
dst_port
protocol
ip_version
bidirectional_duration_ms
bidirectional_packets
bidirectional_bytes
src2dst_first_seen_ms
src2dst_last_seen_ms
src2dst_duration_ms
src2dst_packets
src2dst_bytes
dst2src_first_seen_ms
dst2src_last_seen_ms
dst2src_duration_ms
dst2src_bytes

format, with the same characteristics noted in Table 1. The traffic capture was done offline on an end system, capturing frames using Wireshark at the data center of the University of Cordoba. The capture process was done during peak hours, i.e., in high traffic demand. In around five hours, the frame was stored in pcapng format, with a size of approximately 1.9 Gibabytes.

2) PREPROCESSING AND IDENTIFICATION OF ELEPHANT TRAFFIC AND MICE TRAFFIC

Once the CSV file containing the five-hour capture flow is obtained, it is preprocessed for training, as shown in Figure 2.

This process is executed in a final system in offline mode. The first step is to read the file containing the flow in CSV format and then identify the type of traffic based on the counting technique. For this, the condition must meet the parameters such as ip_origin, ip_destination, port_origin and port_destination. This value is compared with all elements that meet that characteristic. There is a threshold that allows defining an upper limit for elephant traffic. Once the count is compared with the threshold, if it is above the threshold, it will be considered an elephant flow and stored in a CSV file with an additional field called target_traffic. For this case, the value will be 1. If the counter is lower than the threshold, the target_traffic field is marked with 0, which indicates that it is mice traffic and is stored in the file.

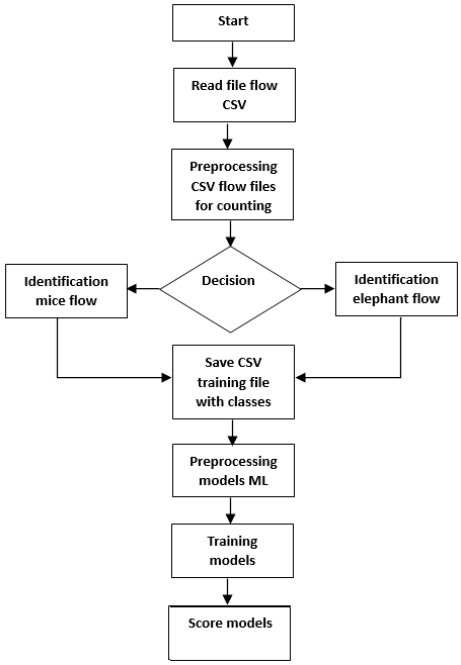

FIGURE 2. Preprocessing of IP traffic for identification.

These are the fields taken into account for traffic identification.

writer.writerow({'src_ip': src_ip,'dst_ip':dst_ip, 'src_port':src_port,'dst_port':dst_port,'bidirectional_first_seen_ms':bidirectional_first_seen_ms,'bidirectional_last_seen_ms':bidirectional_last_seen_ms,'bidirectional_bytes': bidirectional_bytes,'target_traffic': target_traffic,'size_traffic':size_traffic}).

3) TRAINING OF MACHINE LEARNING MODELS FOR TRAFFIC PREDICTION AND CLASSIFICATION IN IP NETWORKS

After the flowchart shown in Figure 2, training the models followed using these machine learning models:

- Logistic Regression
- Support Vector Machine (SVM)
- Random Forest Classifier
- Linear Discriminant Analysis (LDA)
- K-Neighbors Classifier (KNN)
- Naive Bayes Gaussian (Gaussian NB)
- Decision Tree Classifier

These seven supervised learning models were taken for the training and prediction process to determine the model that best generates predictions.

The Pandas Jupyter-lab environment with the Scikit-learn machine learning libraries [28], [29] was used to perform the training. The parameters taken into account for the training are described below:

src_ip, dst_ip, src_port, dst_port, bidirectional_first_seen_ms, bidirectional_last_seen_ms, bidirectional_bytes, target_traffic, size_traffic. 44581 instances were processed for 80% training data and 20% for testing, out of a total of 55726. Null values were excluded, as well as five independent

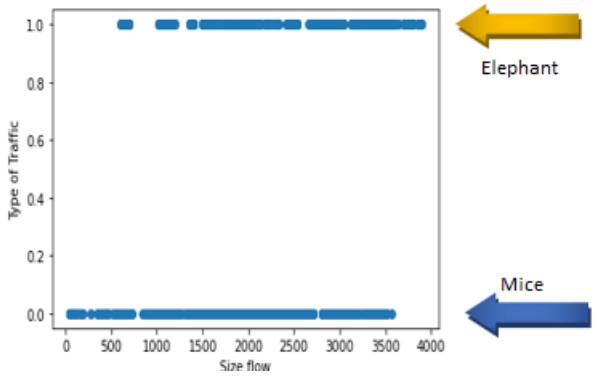

FIGURE 3. Network traffic identification.

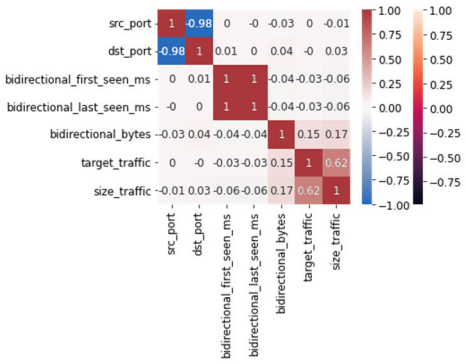

FIGURE 4. Correlation matrix for traffic analysis.

variables and one dependent variable (target_traffic). The dependent variable will identify the type of traffic (0 for mice, 1 for elephant). Two independent variables (src_ip and dst_ip) were removed, given that due to the nature of IP addresses, they are not understandable for the model and are not significant for training and predictions.

On the other hand, the threshold parameter is updated periodically, which allows having reliable predictions and can be used for decision-making by the data center switching systems.

The contribution of this investigation work is the implementation of a classifying elephant and mice system using machine learning techniques for the early detection with the first flow based on the dynamic calculation of the threshold, according to the input parameters of the final system. In the first instance, training algorithms are used to determine the best performance, then the proposed algorithm determines the model with the best prediction, obtained from the supervised learning algorithm trained in off line mode. Finally in the phase of online prediction, the algorithm is capable of predicting with high precision the type of traffic in terms of the input flow, and updates in a dynamic way the threshold to determine whether the traffic is elephant or mice. It is important to say that the use of the threshold to determine the traffic in data network centers is still under investigation [33]. So, the proposal realized in this work is an approximation for the detection and classification of traffic, based on the first input flow.

III. ANALYSIS AND RESULTS

After training the machine learning models, it is observed that 94.89% corresponds to mice traffic and 5.11% to elephant traffic. In other words, the traffic analyzed in the University of Cordoba data centre has an average behavior similar to any network that transfers and receives packets over the Internet. This means that although only 5.11% of the network traffic is elephant traffic, it manages to consume more of the corporate network bandwidth, as shown in Figure 3.

A. CORRELATION MATRIX

A correlation matrix is a tool used to compare the correlation coefficients between different characteristics in a data set. This tool allows us to visualize how much or little correlation exists between different variables. The correlation matrix allows us to identify the variables with high degrees of correlation. In addition, they allow us to reduce the number of features we can have in a dataset [30]. Once the correlation matrix is processed, we can identify that the variable target_traffic has a direct correlation with size_traffic with a value of 0.62 strong positive, i.e., there is a dependency between these two variables, which implies that depending on the packet size, we can determine the type of traffic, which can be mice or elephant. On the other hand, a negative correlation is observed in the variables src_port and dst_port with a value of -0.98 strong, as shown in Figure 4. Considering these values described above, they can be used to create machine learning models, which will be described later.

B. PERFORMANCE COMPARISON OF THE IMPLEMENTED ALGORITHMS

Once the independent and dependent variables were defined, the data were preprocessed and normalized. For this purpose, the K-fold cross-validation procedure was used, which is a standard method for estimating the performance of a machine learning algorithm on a set of data. The scores obtained from each of the supervised learning models used to predict the type of traffic are shown below. As seen in Table 2 and Figures 5 to 11, according to the five scores for each model, the one with the best prediction performance is the decision tree classifier with a score of 100%, followed by the random forest algorithm with a score above 99%. Only in the first score, it has a value close to 98%. On the other hand, with very similar scores, we find 98% and 97% of the Naive Bayes and K-Neighbors Gaussian classification algorithms. Similarly, we found that SVM, linear discriminant analysis and logistic regression have a very similar score of 96%. In other words, practically all seven implemented models have a prediction score of 97%, which makes them very efficient for predicting traffic classification in an IP network.

C. CONFUSION MATRIX

The best-performing model-based confusion matrix was the decision tree classification algorithm, which scored 100% for

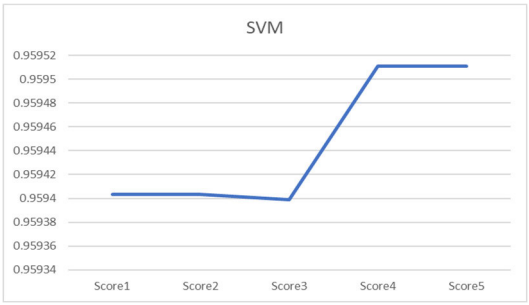

FIGURE 5. Score SVM.

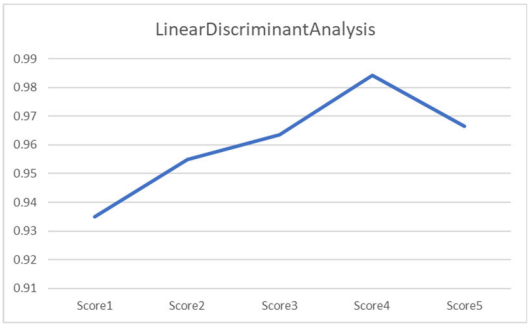

FIGURE 6. Score linear discriminant analysis.

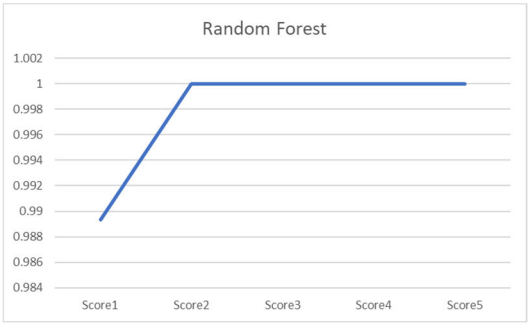

FIGURE 7. Score random forest.

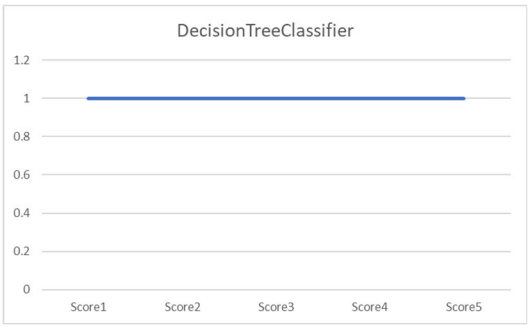

FIGURE 8. Score decision tree classifier.

predictions. Then according to Figures 12 and 13, we can deduce the following:

The algorithm can predict the type of traffic depending on its size, i.e., the larger the flow size, the higher the probability

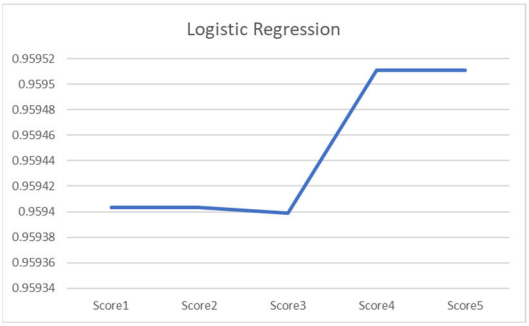

FIGURE 9. Score logistic regression.

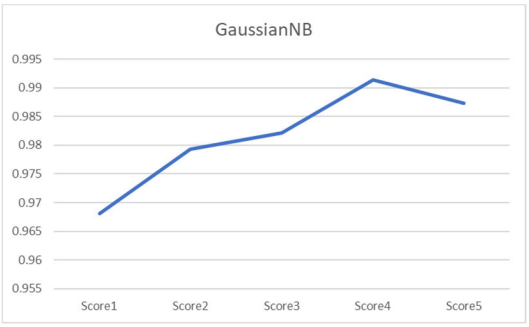

FIGURE 10. Score gaussian NB.

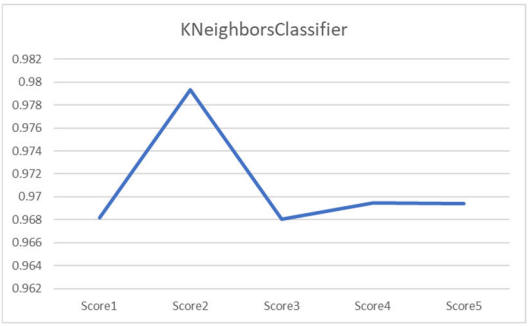

FIGURE 11. KNeighbors classifier.

of classifying it as elephant traffic. Similarly, if the size is smaller, it will be identified as mice traffic. It is important to note that, by default, most of the traffic in the network is usually mice traffic. However, for obvious reasons, elephant traffic, despite having smaller flow sizes, tends to consume bandwidth in telecommunications networks. In terms of the efficiency of the classification algorithm, it can be stated that it has a value of 0.0 error at the time of classification because it has no false positives or false negatives. It has a high accuracy of 100%. This allows generating a correct prediction depending on the associated variables. In addition, the algorithm shows high sensitivity, around 100%, allowing it to discriminate the negatives and positives of the trained instances correctly. In other words, when predicting, the model is highly accurate and sensitive, and can clearly identify what type of traffic it is (mice or elephant).

TABLE 2. Accuracy of the implemented models.

Model	Score 1	Score 2	Score 3	Score 4	Score 5	Score Mean
SVM	0.9594033	0.9594033	0.9593988	0.9595109	0.9595109	0.9594455
Logistic Regression	0.9594033	0.9594033	0.9593988	0.9595109	0.9595109	0.9594455
Random Forest	0.9893461	1	1	1	1	0.9978692
GaussianNB	0.9681507	0.9793652	0.9821668	0.9913638	0.9873261	0.9816745
LinearDiscriminantAnalysis	0.9350678	0.9550297	0.9635486	0.9842978	0.9665769	0.9609042
KNeighborsClassifier	0.9681507	0.9793652	0.9680349	0.9694930	0.9693808	0.9708849
DecisionTree Classifier	1	1	1	1	1	1

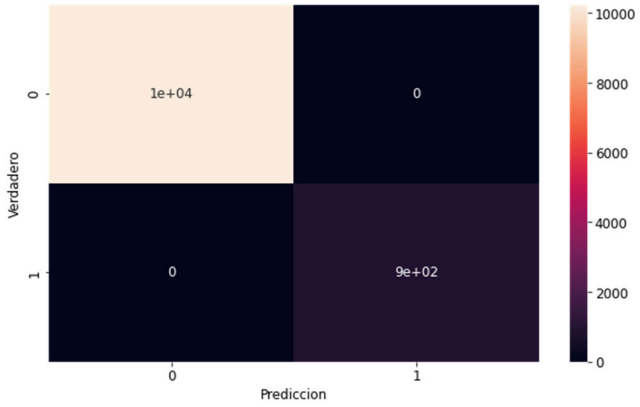

FIGURE 12. Confusion matrix based in a decision tree classification model.

```
Confusion matrix

 [[10241     0]
 [    0   904]]

True Positives(TP) =  10241

True Negatives(TN) =  904

False Positives(FP) =  0

False Negatives(FN) =  0
```

FIGURE 13. Confusion matrix based on a discriminative decision tree classification model.

IV. CONCLUSION AND FUTURE WORK

Detecting and classifying the type of traffic in telecommunication networks is a very important task since it allows for better management and planning. In traditional IP networks, flow detection and classification techniques are usually done in end systems because the network hardware architecture is usually proprietary, making it very difficult to access this equipment from the network layer. Faced with this situation, what is proposed in most research studies is to implement the classification in end systems, which usually become tools to understand the type of flow that is occurring in the network and make decisions.

This research was implemented in an end system, which allowed listening to traffic and then preprocessing and classifying it. With the implementation of different supervised learning models, it was possible to classify traffic with an accuracy level above 96% for all models. However, the best-performing algorithm was the decision tree classifier, with a confidence level of 100%. The performance study of this algorithm showed that it has a high level of accuracy and sensitivity, which means that it can accurately predict the type of traffic (mice or elephant).

Our implementation detects with the first flow the type of traffic it corresponds to (elephant or mice), using the prediction models described in the article. Although the other parameters are an option, they slow down the process and are therefore less efficient when making real-time predictions, as they would require more time.

Although the results allow classifying the type of traffic with high accuracy, these algorithms can be implemented in Software Defined Networks (SDN) because these types of networks have a controller that allows generating the store and forward instructions in the intercommunication equipment (switches and routers). In other words, these algorithms would provide the SDN controller with information to decide which device to send the data flow to, depending on its classification. This would make the network much more efficient, as well as more flexible and adaptable.

For future work, we propose the implementation of the early prediction model of elephant traffic flow in an SDN controller.

ACKNOWLEDGMENT

The authors would like to thanks the Systems Office of Universidad de Córdoba for providing the data that allowed them to analyze this study. They would also like to thank engineer Samir Rubio for his cooperation. Finally, they would like to thank the Office of the Vice-Rector of Research of Universidad de Córdoba for financing the resources to carry out this research.

REFERENCES

[1] M. Afaq, S. U. Rehman, and W.-C. Song, "Visualization of elephant flows and QoS provisioning in SDN-based networks," in *Proc. 17th Asia–Pacific Netw. Oper. Manag. Symp. Manag. (APNOMS)*, Aug. 2015, pp. 444–447.

[2] P. Jurkiewicz, "Boundaries of flow table usage reduction algorithms based on elephant flow detection," in *Proc. IFIP Netw. Conf. (IFIP Netw.)*, Jun. 2021, pp. 1–9, doi: 10.23919/IFIPNetworking52078.2021.9472832.

[3] S. Floyd, "Simulation is crucial," *IEEE Spectr.*, vol. 38, no. 1, p. 76, Jan. 2001.

[4] C.-Y. Lin, C. Chen, J.-W. Chang, and Y. H. Chu, "Elephant flow detection in datacenters using OpenFlow-based hierarchical statistics pulling," in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, pp. 2264–2269.

[5] M. E. Crovella, "Performance evaluation with heavy tailed distributions," in *Proc. Int. Conf. Modelling Techn. Tools Comput. Perform. Eval.*, vol. 2221, 2001, pp. 1–10.
[6] C. Estan and G. Varghese, "New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice," *ACM Trans. Comput. Syst.*, vol. 21, no. 3, pp. 270–313, Aug. 2003.
[7] K.-C. Lan and J. Heidemann, "A measurement study of correlations of internet flow characteristics," *Comput. Netw.*, vol. 50, no. 1, pp. 46–62, Jan. 2006.
[8] Y. Chabchoub, C. Fricker, F. Guillemin, and P. Robert, "On the statistical characterization of flows in internet traffic with application to sampling," *Comput. Commun.*, vol. 33, no. 1, pp. 103–112, Jan. 2010.
[9] Y. Shao, B. Yang, J. Jang, Y. Xue, and J. Li, "Emilie: Enhance the power of traffic identification," in *Proc. ICNC*, Feb. 2014, pp. 31–35.
[10] S. Hegde, S. G. Koolagudi, and S. Bhattacharya, "Scalable and fair forwarding of elephant and mice traffic in software defined networks," *Comput. Netw.*, vol. 92, pp. 330–340, Dec. 2015.
[11] C. Wang, G. Zhang, H. Chen, and H. Xu, "An ACO-based elephant and mice flow scheduling system in SDN," in *Proc. IEEE 2nd Int. Conf. Big Data Anal. (ICBDA)*, Mar. 2017, pp. 859–863.
[12] M. Al-Saadi, A. Khan, V. Kelefouras, D. J. Walker, and B. Al-Saadi, "Unsupervised machine learning-based elephant and mice flow identification," in *Intelligent Computing*. Cham, Switzerland: Springer, 2021, pp. 357–370.
[13] S. Kundu, S. Pal, K. Basu, and S. Das, "Fast classification and estimation of internet traffic flows," in *Proc. PAM*, Apr. 2007, pp. 155–164.
[14] T. Mori, M. Uchida, R. Kawahara, J. Pan, and S. Goto, "Identifying elephant flows through periodically sampled packets," in *Proc. IMC*, Taormina, Italy, Oct. 2004, pp. 115–120.
[15] A. Kumar, M. Sung, J. Xu, and J. Wang, "Data streaming algorithms for efficient and accurate estimation of flow size distribution," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 177–188, 2004.
[16] A. Kumar, L. Li, and J. Wang, "Space-code Bloom filter for efficient traffic flow measurement," in *Proc. ACM SIGCOMM Conf. Internet Meas. (IMC)*, 2003, pp. 1762–1773.
[17] F. Tang, H. Zhang, L. T. Yang, and L. Chen, "Elephant flow detection and load-balanced routing with efficient sampling and classification," *IEEE Trans. Cloud Comput.*, vol. 9, no. 3, pp. 1022–1036, Jul. 2021.
[18] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, "Hedera: Dynamic flow scheduling for data center networks," in *Proc. 7th USENIX Conf. Netw. Syst. Design Implement.*, 2010, p. 19.
[19] A. R. Curtis, W. Kim, and P. Yalagandula, "Mahout: Low-overhead datacenter traffic management using end-host-based elephant detection," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 1629–1637.
[20] W. Liu, W. Qu, Z. Liu, K. Li, and J. Gong, "Identifying elephant flows using a reversible MultiLayer hashed counting Bloom filter," in *Proc. IEEE 14th Int. Conf. High Perform. Comput. Commun. IEEE 9th Int. Conf. Embedded Softw. Syst.*, Jun. 2012, pp. 246–253.
[21] D. Sanvito, "CEDRO: An in-switch elephant flows rescheduling scheme for data-centers," in *Proc. IEEE NetSoft*, Jul. 2020, pp. 368–376.
[22] Z. Su, T. Wang, Y. Xia, and M. Hamdi, "CheetahFlow: Towards low latency software-defined network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 3076–3081.
[23] H. Yahyaoui, S. Aidi, and M. F. Zhani, "On using flow classification to optimize traffic routing in SDN networks," in *Proc. IEEE 17th Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2020, pp. 1–6, doi: 10.1109/CCNC46108.2020.9045216.
[24] S.-C. Chao, K. C.-J. Lin, and M.-S. Chen, "Flow classification for software-defined data centers using stream mining," *IEEE Trans. Services Comput.*, vol. 12, no. 1, pp. 105–116, Jan./Feb. 2016, doi: 10.1109/TSC.2016.2597846.
[25] P. Poupart, Z. Chen, P. Jaini, F. Fung, H. Susanto, Y. Geng, L. Chen, K. Chen, and H. Jin, "Online flow size prediction for improved network routing," in *Proc. IEEE 24th Int. Conf. Netw. Protocols (ICNP)*, Nov. 2016, pp. 1–6.
[26] A. Chhabra and M. Kiran, "Classifying elephant and mice flows in high-speed scientific networks," in *Proc. INDIS*, 2017, pp. 1–8.
[27] X. Zhang, L. Cui, F. P. Tso, and W. Jia, "PHeavy: Predicting heavy flows in the programmable data plane," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 4, pp. 4353–4364, Dec. 2021.
[28] F. Pedregosa, S. Varoquaux, A. Gramfort, V. Michel, and B. Thirion, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Dec. 2011.
[29] L. Buitinck, "API design for machine learning software: Experiences from the scikit-learn project," in *Proc. ECML PKDD Workshop Lang. Data Mining Mach. Learn.*, 2013, pp. 108–122.
[30] C. Wang, J. Du, and X. Fan, "High-dimensional correlation matrix estimation for general continuous data with Bagging technique," *Mach. Learn.*, vol. 111, pp. 2905–2927, Mar. 2022.
[31] R. Durner and W. Kellerer, "Network function offloading through classification of elephant flows," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 807–820, Jun. 2020.
[32] M. Hamdan, B. Mohammed, U. Humayun, A. Abdelaziz, S. Khan, M. A. Ali, M. Imran, and M. N. Marsono, "Flow-aware elephant flow detection for software-defined networks," *IEEE Access*, vol. 8, pp. 72585–72597, 2020.
[33] H. Dong, A. Munir, H. Tout, and Y. Ganjali, "Next-generation data center network enabled by machine learning: Review, challenges, and opportunities," *IEEE Access*, vol. 9, pp. 136459–136475, 2021.

JORGE GÓMEZ (Senior Member, IEEE) received the degree in systems engineering from Fundación Universitaria San Martín, Colombia, in 2006, the master's degree in telematics engineering from Universidad del Cauca, and the Ph.D. degree in ICT from the University of Granada, Spain, in 2018. He is currently a Professor and a Researcher with the Systems Department, Universidad de Córdoba. He has participated in national and international projects in Colombia. He has published several research papers in recognized journals. His research interests include the Internet of Things, context-aware systems, and SDN. He has served as a guest editor for several special issues at many journals.

VELSSY HERNANDEZ RIAÑO received the master's degree in teleinformatics. She is currently a Professor with the Systems Engineering Program, Universidad de Córdoba, Colombia, and a member of the SOCRATES Research Group. She is also a Systems Engineer.

GUSTAVO RAMIREZ-GONZALEZ received the B.S. degree in electronic and telecommunications engineering and the M.S. degree in telematics engineering from the University of Cauca, Colombia, in 2001, and the Ph.D. degree in telematics engineering from Universidad Carlos III de Madrid, Spain, in 2010. He is currently a Professor and a Researcher with the Department of Telematics, University of Cauca. He has participated in national and international projects in Colombia and Spain. He has published several research papers in reputed journals and served as a guest editor for several special issues at many journals. His research interests include image processing, secure communication, machine learning, and the IoT.