Volume 19 · Issue 1
January-April 2022
Pages 1-9

CULCYT
CULTURA CIENTÍFICA Y TECNOLÓGICA
ISSN (electrónico) 2007-0411

DOI: 10.20983/culcyt.2022.1.2.2

RESEARCH ARTICLE

Smart architecture for software-defined networking

Arquitectura inteligente para redes definidas por software

Jose Mejia✉[1], *Oliverio Cruz-Mejia*[2], *José Alfredo Acosta-Favela*[1], *Alejandra Mendoza-Carreón*[1], *René Noriega Armendáriz*[1]

[1] Universidad Autónoma de Ciudad Juárez (UACJ)
[2] Universidad Nacional Autónoma de México (UNAM)

ABSTRACT

Software-defined networks (SDN) seek to solve the problems in current network schemes by simplifying their management through their reprogrammability and accessibility to the overall network infrastructure. One aspect to improve in SDN-based schemes is the precise classification of your traffic load, this can improve various aspects such as quality of service, dynamic access control, prioritized random access, among others. This research aims to propose a conceptual architecture of SDN and evaluate different machine learning methods for traffic classification. To this end, SDN architectures are analyzed and different modules are proposed to strengthen their management with the help of low computational cost classifiers. The architecture proposes the following main modules: Capture network traces module, Learning Engine module, and ML-model and Flow classifier. To determine the model to be used in the Learning Engine and Flow classifier modules, different classifiers were evaluated using a database of network traffic, as a result, it was determined that the gradient boosting algorithm is the most suitable to be integrated with the proposed SDN architecture.

KEYWORDS: software-defined networks; machine learning; gradient boosting.

RESUMEN

Las redes definidas por software (SDN) buscan resolver los problemas de los esquemas de red actuales al simplificar su gestión a través de su programabilidad y accesibilidad global a la infraestructura de red. Un aspecto para mejorar en esquemas basados en SDN es la clasificación precisa de su carga de tráfico, esto puede mejorar diversos aspectos como calidad de servicio, control de acceso dinámico, acceso aleatorio priorizado, entre otros. Esta investigación tiene como objetivo proponer una arquitectura conceptual de SDN y evaluar diferentes métodos de aprendizaje automático para la clasificación del tráfico. Con este fin, se analizan las arquitecturas de SDN y se proponen diferentes módulos para robustecer su gestión con ayuda de clasificadores de bajo costo computacional. La arquitectura propone los módulos principales siguientes: módulo de captura de datos de red, módulo de aprendizaje con modelo de Machine Learning y módulo clasificador de flujo. Para determinar el modelo de ML a utilizar en los módulos de aprendizaje y clasificador de flujo, se evaluaron diferentes clasificadores mediante una base de datos, como resultado se determinó que el algoritmo de *gradient boosting* o potenciación del gradiente es el más adecuado para integrarse a la arquitectura SDN propuesta.

PALABRAS CLAVE: redes definidas por software; aprendizaje automático; *gradient boosting*.

Corresponding author: Jose Mejia
Institution: Universidad Autónoma de Ciudad Juárez / Instituto de Ingeniería y Tecnología
Address: Av. Del Charro núm. 450 norte, col. Partido Romero, Ciudad Juárez, Chihuahua, México, C. P. 32310
E-mail: jose.mejia@uacj.mx

Manuscript received: December 8, 2021; accepted: April 1, 2022. Date of publication: April 28, 2022.

I. INTRODUCTION

Software-Defined Networks (SDN) is an approach to address the complexity and challenges that modern computer networks face to manage and scale today's requirements. SDN attempts to solve these issues by providing a control architecture that is centralized and highly scalable. This helps to manage large-scale networks and data centers [1]. SDNs have increased recently because of their flexible management and monitoring capabilities, which has been possible because of their centralized architecture, which can manage a set of switches by modifying flow table entries on demand for better adaptation [2]. The control plane in SDN uses special protocols, such as OpenFlow, which enables the network managers to monitor statistics, decide strategies, and interact with the switches, thus enabling an improvement of the network performance [3].

Because of the importance of network management for an optimum operation, SDN has been applied also to ad hoc networks and in particular to networks consisting of devices known as the Internet of Things (IoT) [4], [5]. These ad hoc networks have recently experienced rapid growth because of their implementation in smart cities, industry 4.0, and home automation [6], [7], [8]. These networks also have increased their services and related applications provided through the network, thus recently research has been carried out on the integration of SDN and IoT [9].

Even though the advantages offered by SDN, network applications still have challenges related to Quality-of-Service (QoS) and customized service level agreements (SLAs) [10], [11]. Allocation of resources to applications to guarantee a certain level of performance, availability, and reliability is one of the main concerns in SDN schemes as well as optimizing the deployment of resources using different mechanisms to meet the QoS and SLA [11]. One way to approach this problem is through the management of network packets or flows through the differentiation of data according to the particular applications or services provided by the ad hoc network. Thus, network traffic classification is an important component of modern SDN [11].

The problems outlined above make it a necessity to develop approaches to automatically classify network data to optimize the deploying of resources using different mechanisms to meet the QoS and SLA. The use of machine learning for classification is key to enabling the improvement of the SDN in the network performance.

In this paper, the performance of several machine learning classification algorithms for use in SDN is reviewed. Also, a conceptual SDN architecture that implements a machine learning classification scheme for network classification is proposed.

In the following paragraphs, we review related work to our investigation. Thus, some relevant algorithms for traffic classification in the context of SDN were reviewed. In [10], it is proposed the integration of SDN and machine learning to classify data traffic based on the applications in a software-defined network platform and also the authors evaluated three different learning models: Support Vector Machines, nearest centroid, and Naive Bayes, methods that were chosen for their simplicity. Network traffic traces and flows features of the captured data, and these attributes are sent to a classifier for prediction were used. Higher accuracy for the Naive Bayes classifier of 96.79% was found.

In [3], the authors propose an SDN that uses in its control plane section a traffic classifier that detects the traffic of the data plane and helps the QoS module to modify flow rules for the switches. This classifier is based on deep learning. The proposed classifier models use long short-time memory layers (LSTM) and the other convolutional layers and LSTM. To find and optimize the neural network hyper-parameters, they used a cross-validated grid search and they found that the LSTM model can classify the network traffic better than the CNN with LSTM because the long-term dependency between packets is handled properly by the LSTM alone.

Additionally, a series of traffic classifiers, in this case, to detect and predict SYN flood, a case of DoS attack, were proposed in [12]. They evaluated machine-learning algorithms based on Decision Trees, Support Vector Machines, and Naive Bayes and perform an analysis of their performance. They also implemented a real-time system implementing a controller that uses the models to classify packets and report that Decision Trees perform best on the analysis of the data, while NB has better performance in a real-time system.

A method was proposed in [11] for improving traffic flows by using machine learning classification. They use 44 attributes of the packet flow as features, which they

reduced further by using Principal Component Analysis. Three different classifier algorithms were evaluated: Nearest Centroid, Naive Bayes, and Vector Machines. It was found that all the classifiers have a good performance, being better by a minimum margin than the Nearest Centroid method.

II. METHODOLOGY

In this section, it is described the methodology used, first a revision of the theory and background is presented.

A. THEORY AND BACKGROUND

In this section, a general review of the architecture of SDN is presented. Also, the considered ML algorithms are described.

SDN ARCHITECTURE

As the modern network increases its complexity, its management also presents challenges and complicities. In recent times, this has been intensified because of the use of manual operations and low-level programming of vendor-specific devices. SDN tries to alleviate the network management and configuration by proposing a centralized approach to network management and more standard programming of the underlying network hardware [2], [13].

The SDN basic architecture consists of an infrastructure or data plane, a control plane, and an application plane, as shown in Figure 1.

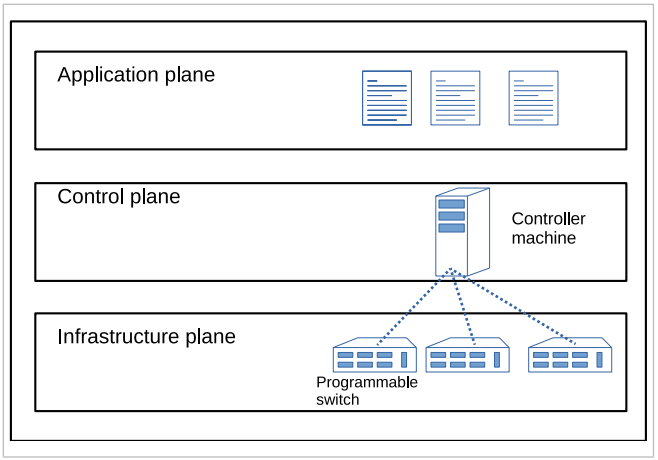

Figure 1. SDN basic architecture.

The Infrastructure or Data plane consists of switching and routing devices. These devices could be programmed employing a forwarding table or flow rules and in the context of SDN these devices act as simple forwarding devices that receive routing paths directly from the controller [14].

The Control plane maintains a global view of the network and controls the infrastructure devices by policy routing decisions. The controller provides the applications in the application plane and a programming interface that hides the specifics of the infrastructure devices [13].

The Application plane consists of SDN applications that help to manage the network devices. Generally, the applications in this plane are driven by events coming from the controller to which the application must react appropriately.

This work explores means to improve the SDN performance using machine learning for classification and the proposal of modules in the control and application planes.

MACHINE LEARNING

Next, several machine learning algorithms that could be used for network traffic classification are described. They were chosen because of their low computational cost and ease to train, as compared, for example, with deep learning classifiers.

Naive Bayes: This classifier is a probabilistic model, which predicts the class with the highest posterior probability. Naive Bayes assumes independence between in-stances. However, this assumption is generally not true for most data sets, but still, the classifier could achieve a decent classification precision. Because of its lower computational cost, the Naive Bayes classifier is considered first when the application requires real-time processing of data. Also, another advantage of this classifier is that it only requires a small amount of data for training [15].

Support Vector Machines: A support vector machine is a classification algorithm that finds the best possible separation between classes. The support vectors define the maximum margin of separation of the hyperplane that separates the classes. When the classes consid-

ered are not linearly separable, kernel functions could be used with the algorithm to solve the more complex problem. The computational cost of this algorithm is relatively lower.

K-Nearest Neighbors: In this classification algorithm, the complete training data is stored because the model does not learn a model but uses explicitly the training samples and then is later used for the prediction. The new instances are classified by choosing the majority class among the K closest examples of the stored data [16].

Decision Tree: These classifiers are composed of nodes, each dividing the space of features of the training data by grouping observations with similar values [17]. To carry out this division, a series of rules called decisions are applied, trying that each sub-region contains the largest possible samples from one of the classes of the data.

Random Forest: In this classification method, bagging is used to combine several decision trees. Hence, the different trees operate on different segments of the data and each tree is trained with different data samples of the training set. Finally, by combining the results of the different trees, errors of a particular tree could be compensated by other trees, increasing the accuracy and generalization of the method [18].

Gradient Boosting: Boosting in the context of machine learning refers to creating a strong classifier out of many weak ones. Thus, Gradient Boosting consists of several classifiers, for these a suitable loss function is used. The error calculated from the loss function using a classifier is the "gradient" of the gradient boosting, which indicates how to build the next classifier [19]. For this work, it is used using an ensemble of decision trees to predict a target label.

In the results section, some of these algorithms are evaluated using a database of network packets.

B. SDN ARCHITECTURE WITH THE CLASSIFICATION OF NETWORK DATA

This section described in detail the proposed architecture. It is outlined the design and described the specific steps for its development, details of the modified planes are also provided.

ARCHITECTURE DESIGN

The main goal of the architecture is to increase the throughput of the network in the face of changes in traffic that occurs in the network's normal operation, and the proposed scheme focused on traffic-based classification in the network. This is accomplished by extending the functionality of the controller.

The proposed architecture is implemented as an application module, consisting of a module for the training model sitting on top of the controller and two modules on the control plane: a capture module and a flow classifier. The architecture is shown in Figure 2 and described next.

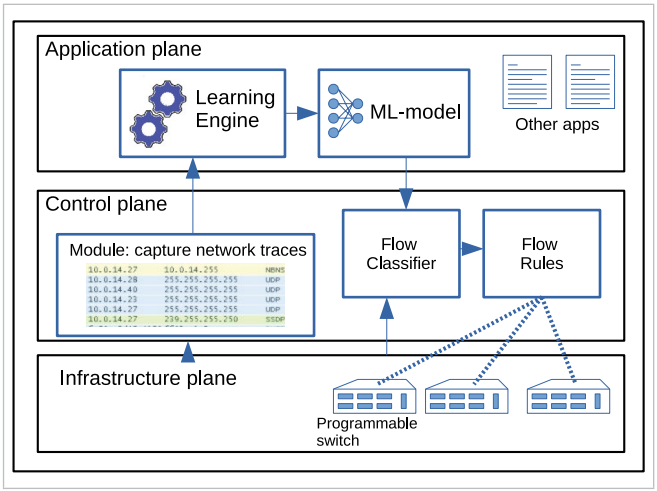

Figure 2. The proposed architecture consists of four modules: Capture, Learning Engine, Machine Learning (ML) model, and Flow classifier.

CAPTURE NETWORK TRACES MODULE

This module is connected to the underlying SDN infrastructure to acquire data from the network. Then new data is sent to the learned Engine to adapt the model to the new environment or infrastructure changes.

LEARNING ENGINE MODULE AND ML-MODEL

The module is connected to the capture module, it processes the data from the capture module and uses machine learning algorithms to process the data and train a model. The specific implementation depends on the machine learning model, for example, for GB the module minimizes its loss function with the available data to adapt the model to the incoming packets. This module

is executed only when the model needs to adapt to new data or changing conditions.

FLOW CLASSIFIER

The objective of this module is to classify the flow of information. It uses a trained ML model obtained from the Learning Engine. Once the flow has been classified, it is expected that traffic performance increases throughout the network.

CLASSIFICATION OF NETWORK DATA

For data testing of the Learning Engine module and ML model, a machine learning model must be selected. The selected model must be easy to implement and with a computational complexity that permits real-time processing of the data. Thus, for the selection of the classification model, the following models are tested: Naive Bayes, Support Vector Machines, k-nearest neighbors, Random Forest, Decision Tree, and Gradient Boosting.

C. EXPERIMENTAL SETUP

The experimental setup is aimed at assessing machine learning algorithms for integration with the *flow classification module* of the proposed architecture. To this end, the performance of several classification methods was evaluated. From the available evaluation approaches that can be considered as metrics of performance, the following were selected: precision, recall, and F1 score.

The experiments involved the use of a data set to test the proposed scheme. Thus, we used traffic flow records from a database [20]. The data was acquired from a network of the Universidad del Cauca, Popayán, Colombia.

Data packet captures were acquired at different hours of the day between April and June 2019. The database consists of a comma-separated values file with each record containing 50 features, more details of the database are in [20].

For this study, the packet types specified in Table 1, extracted from the database, were used and the number of records to 1495.

Even though the classes are unbalanced, no actions were taken to correct it to observe the adaptability of the classification algorithms. Note that this database was already used to evaluate ML algorithms in [20], however, in this work the target classes are different from those used in that work.

TABLE 1
CLASSES OF THE DATABASE

TYPE	#CASES
'Network'	270
'Web'	465
'SoftwareUpdate'	22
'Unspecified'	355
'RPC'	28
'System'	41
'Cloud'	40
'Email'	274
Total	1495

The ML models to evaluate were those presented in Section A. As stated before, those were chosen because of their low computational cost and ease of training. The machine learning models were trained on subsets of the database and evaluated on the complementary subset of the data. To this end, K-fold cross-validation was used, this method also allows us to detect possible over-fitting. This technique is frequently used in machine learning to compare, validate, and select models, because it provides a good estimate of the generalization achieved by the algorithms, and results in performance estimates generally have a lower bias than other methods. For this experiment, the data set was split into ten sections or folds, that is, a value for K equal to 10 was used, thus becoming 10-fold cross-validation. During the test, each fold was used as a testing set at some point in the experiment.

III. RESULTS AND DISCUSSIONS

Here we present the results obtained from evaluating the ML models on the dataset. The accuracy of each method is shown in Figure 3 using a box plot that condenses the obtained results.

From Figure 3, Random Forest (RF) obtained the best accuracy overall, followed by Gradient Boosting (GB) and Decision Trees (DT), while Gaussian Naive Bayes (GNB) has the worst accuracy. This last classifier GNB is one with the lowest computational costs, so a future work could be to adapt the distribution of the classifier to that of the data from network packets.

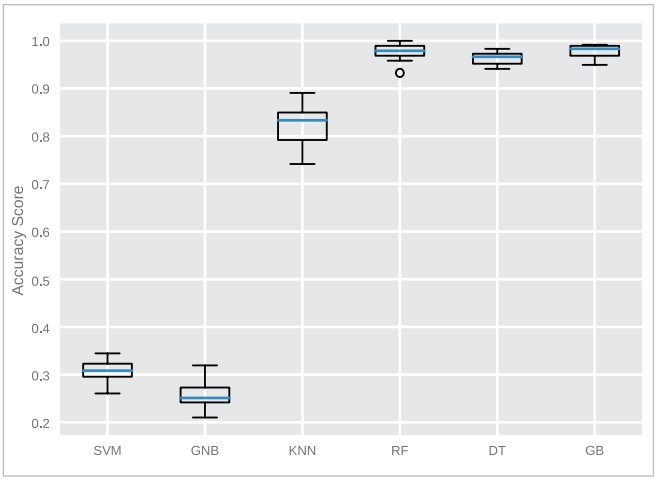

Figure 3. Box plot of the accuracy achieved by each method. The models presented in the horizontal axis are: Support Vector Machine (SMV), Gaussian Naive Bayes (GNV), K-Nearest Neighbors (KNN), Random Forest (RF), Decision Trees (DT), and Gradient Boosting (GB).

Figures 4 to 7, present class prediction error reports from the best four classifiers: K-Nearest Neighbors (KNN), Random Forest (RF), Decision Trees (DT), and Gradient Boosting (GB). For this report, the full database was used, 80% for training and 20% for validation.

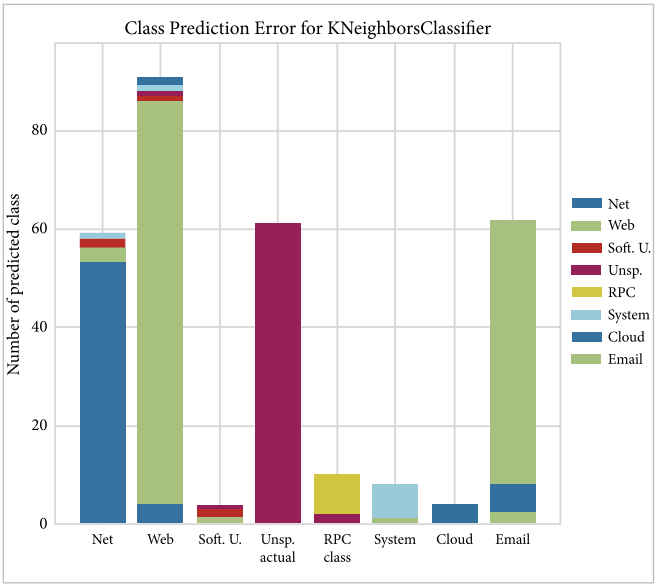

Figure 4. Class Prediction error for K-Neighbors.

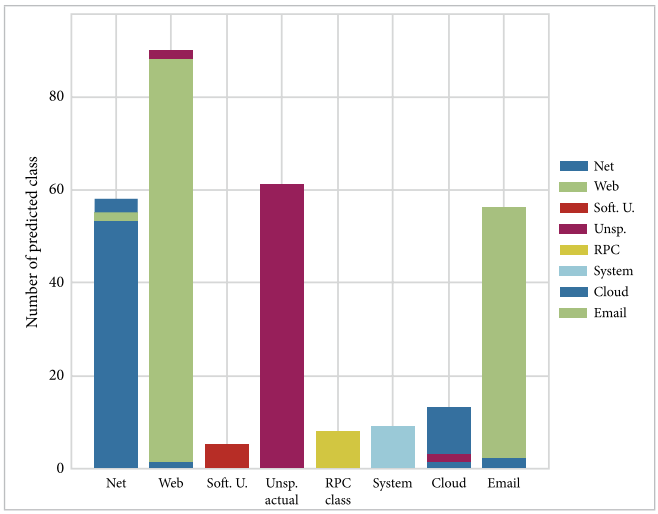

Figure 5. Class Prediction error for Random Forest.

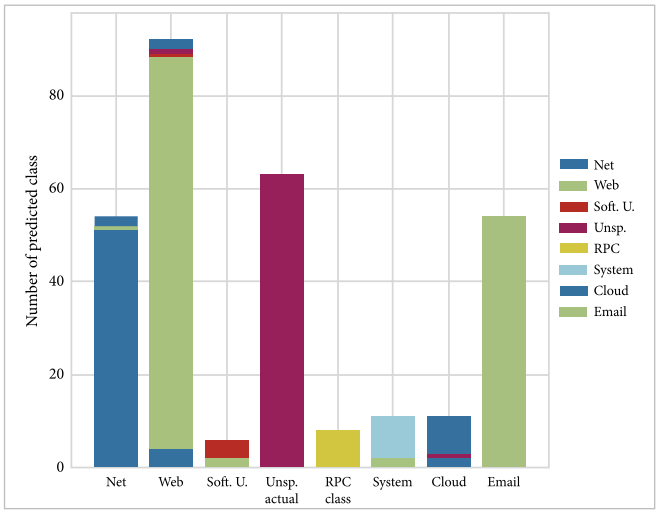

Figure 6. Class Prediction error for Decision Trees.

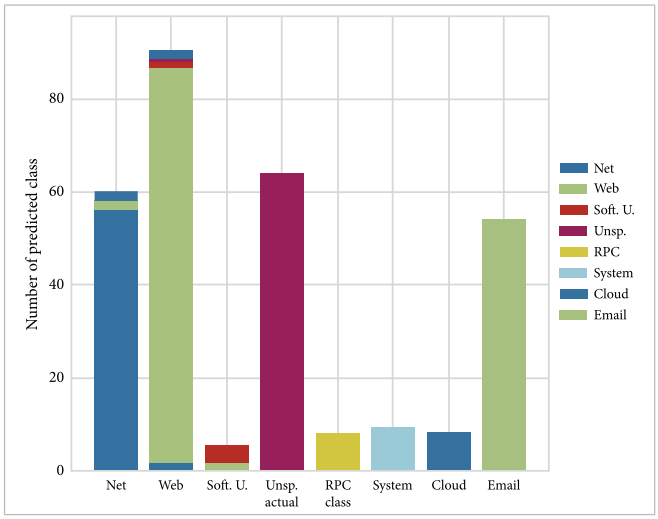

Figure 7. Class Prediction error for Gradient Boosting.

Also, Figures 8 to 11, present the summary of the precision, recall, and f1 scores.

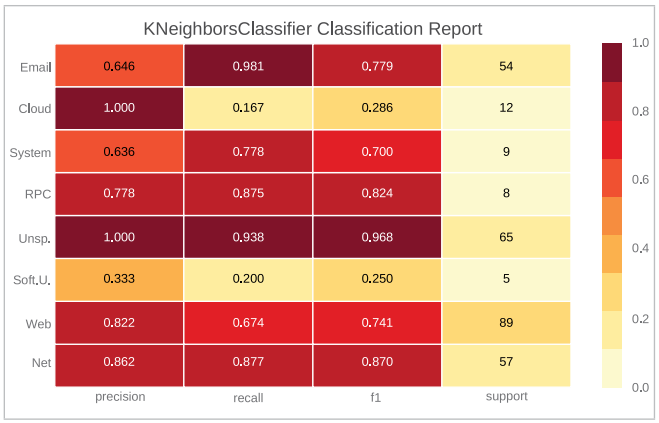

Figure 8. Classification report for K-Neighbors.

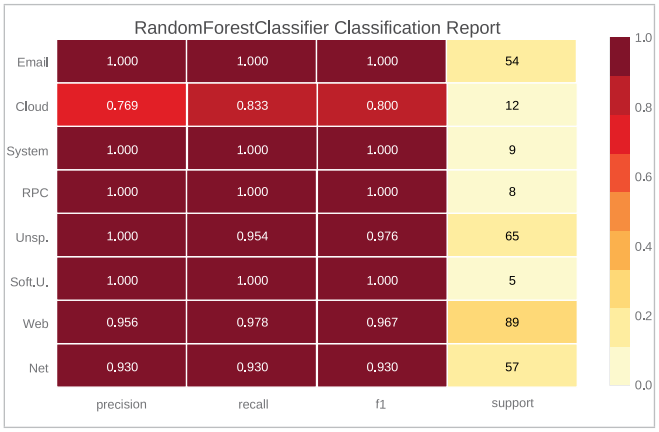

Figure 9. Classification report for Random Forest.

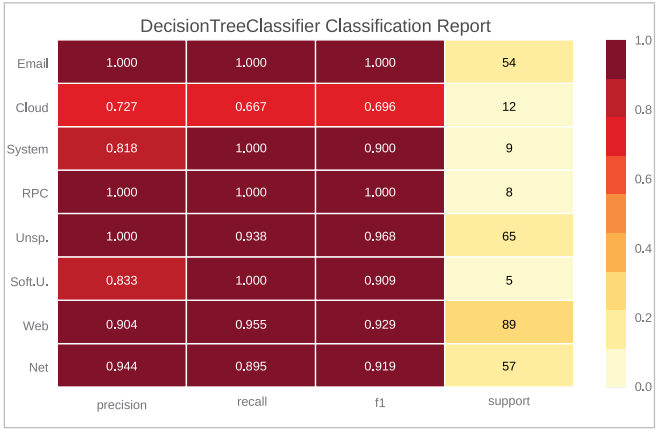

Figure 10. Classification report for Decision Trees.

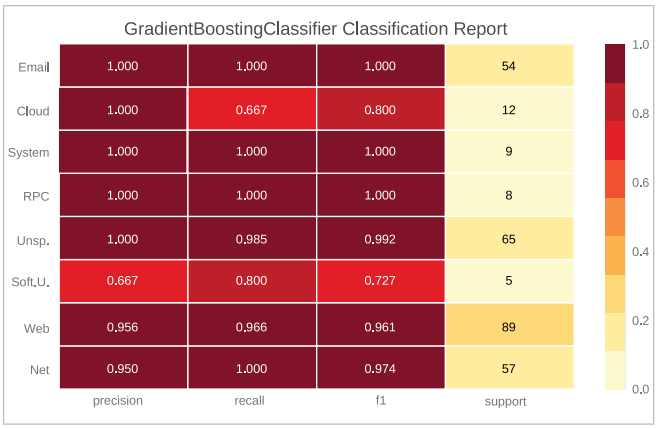

Figure 11. Classification report for Gradient Boosting.

Gradient Boosting Decision Trees outperforms the rest of the classifiers in most categories. However, with a low margin concerning gradient boosting, for class 'web' the best precision is achieved by the random forest classifier and for class 'net' decision tree is best. It is remarkable how gradient boosting Decision Trees achieve a good classification even in categories with low support. Finally, Table 2 shows a global summary of the classification results.

Finally, in this work, it was decided to use classifiers that are faster to train and easier to integrate with the classifier module, unlike other works, such as [3], where more complex architectures such as LSTM are proposed. It was also decided not to calculate features through complex transformations such as the principal component analysis, which is proposed in [11]. Furthermore, unlike [10], the proposed architecture integrates with the relevant layers of an SDN infrastructure rather than as a separate module.

IV. CONCLUSIONS

In this work, a new SND architecture that includes a classifier module is presented. The proposed architecture consists of a modification in the application and control planes of the SND architecture, to accommodate a classification stage.

TABLE 2
Global Summary

Type	Precision				Recall				F1			
	K-N	RF	DT	GB	K-N	RF	DT	GB	K-N	RF	DT	GB
Email	0.871	0.964	**1**	**1**	**1**	**1**	**1**	**1**	0.931	0.982	**1**	**1**
Cloud	**1**	0.833	0.727	**1**	0.333	**0.833**	0.667	0.667	0.5	**0.833**	0.696	0.8
System	0.875	**1**	0.818	**1**	0.778	**1**	**1**	**1**	0.824	**1**	0.9	**1**
RPC	0.8	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0.889	**1**	**1**	**1**
Unsp.	**1**	**1**	**1**	**1**	0.938	0.938	0.954	**0.985**	0.968	0.968	0.976	**0.992**
Soft. U.	0.5	0.833	0.714	**1**	0.4	**1**	**1**	0.8	0.444	**0.909**	0.833	0.889
Web	0.901	**0.967**	0.913	0.946	0.921	**0.978**	0.944	**0.978**	0.911	**0.972**	0.928	0.961
Net	0.898	0.93	**0.944**	0.933	0.93	0.93	0.895	**0.982**	0.914	0.93	0.919	**0.957**

Several classification algorithms in a database of network packets were tested. Algorithms were selected by their relatively low computational cost as compared with other algorithms such as those based on deep learning. Random Forest (RF), Decision Trees (DT), and Gradient Boosting (GB) were found to be suitable algorithms, each achieving more than 0.9 of accuracy. However, overall results show gradient boosting algorithm is the most adequate for the classification module in the SDN architecture proposed.

REFERENCES

[1] N. Noorani y S. A. H. Seno, "SDN- and fog computing-based switchable routing using path stability estimation for vehicular ad hoc networks", *Peer-to-Peer Netw. Appl.*, vol. 13, pp. 948-964, may. 2020, doi: 10.1007/s12083-019-00859-4.

[2] A. R. Mohammed, S. A. Mohammed y S. Shirmohammadi, "Machine Learning and Deep Learning Based Traffic Classification and Prediction in Software Defined Networking", *2019 IEEE International Symposium on Measurements & Networking (M&N)*, 2019, pp. 1-6, doi: 10.1109/IWMN.2019.8805044.

[3] J. Mejia, L. Avelar-Sosa, B. Mederos, E. Santiago y J. D. Díaz, "Prediction of time series using an analysis filter bank of LSTM units", *Comput Ind Eng*, vol. 157, 2021, doi: 10.1016/j.cie.2021.107371.

[4] K. Sood, S. Yu y Y. Xiang, "Software-Defined Wireless Networking Opportunities and Challenges for Internet-of-Things: A Review", en *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 453-463, ag. 2016, doi: 10.1109/JIOT.2015.2480421.

[5] S. H. Rastegar, A. Abbasfar y V. Shah-Mansouri, "Rule Caching in SDN-Enabled Base Stations Supporting Massive IoT Devices With Bursty Traffic", en *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8917-8931, sept. 2020, doi: 10.1109/JIOT.2020.3000393.

[6] B. K. Mukherjee, S. I. Pappu, M. J. Islam y U. K. Acharjee, "An SDN based distributed IoT network with NFV implementation for smart cities", en *Cyber Security and Computer Science*, T. Bhuiyan, M. M. Rahman y M. A. Eli, Eds., Springer, Cham, 2020, doi: 10.1007/978-3-030-52856-0_43.

[7] Y. -W. Ma, Y. -C. Chen y J. -L. Chen, "SDN-enabled network virtualization for industry 4.0 based on IoTs and cloud computing", *2017 19th International Conference on Advanced Communication Technology (ICACT)*, 2017, pp. 199-202, doi: 10.23919/ICACT.2017.7890083.

[8] L. Silva, P. Pedreiras, P. Fonseca y L. Almeida, "On the adequacy of SDN and TSN for Industry 4.0", *2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC)*, 2019, pp. 43-51, doi: 10.1109/ISORC.2019.00017.

[9] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang y K. -K. R. Choo, "An Energy-Efficient SDN Controller Architecture for IoT Networks With Blockchain-Based Security", en *IEEE Transactions on Services Computing*, vol. 13, no. 4, pp. 625-638, 1 jul.-ag. 2020, doi: 10.1109/TSC.2020.2966970.

[10] M. M. Raikar, S. M. Meena, M. M. Mulla, N. S. Shetti y M. Karanandi, "Data Traffic Classification in Software Defined Networks (SDN) using supervised-learning", *Procedia Comput. Sci.*, vol. 171, pp. 2750-2759, 2020, doi: 10.1016/j.procs.2020.04.299.

[11] M. Amiri, H. Al Osman y S. Shirmohammadi, "Game-Aware and SDN-Assisted Bandwidth Allocation for Data Center Networks", *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2018, pp. 86-91, doi: 10.1109/MIPR.2018.00023.

[12] S. Gangadhar y J. P. G. Sterbenz, "Machine learning aided traffic tolerance to improve resilience for software defined networks", *2017 9th International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2017, pp. 1-7, doi: 10.1109/RNDM.2017.8093035.

[13] P. Goransson, C. Black y T. Culver, *Software defined networks: a comprehensive approach*, Morgan Kaufmann, 2016.

[14] N. McKeown *et al.*, "Openflow: enabling innovation in campus networks", *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69-74, 2008, doi: 10.1145/1355734.1355746.

[15] R. Mosquera, O. D. Castrillón y L. Parra, "Máquinas de Soporte Vectorial, Clasificador Naïve Bayes y Algoritmos Genéticos para la Predicción de Riesgos Psicosociales en Docentes de Colegios Públicos Colombianos", *Inf. Tecnol.*, vol. 29, no. 6, pp. 153-162, 2018, doi: 10.4067/S0718-07642018000600153.

[16] P. Horton y K. Nakai, "Better Prediction of Protein Cellular Localization Sites with the it k Nearest Neighbors Classifier", en *Proc Int Conf Intell Syst Mol Biol*, vol. 5, pp. 147-152, jun. 1997.

[17] S. R. Safavian y D. Landgrebe, "A survey of decision tree classifier methodology", en *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 3, pp. 660-674, may.-jun. 1991, doi: 10.1109/21.97458.

[18] V. F. Rodriguez-Galiano, B. Ghimire, J. Rogan, M. Chica-Olmo y J. P. Rigol-Sanchez, "An assessment of the effectiveness of a random forest classifier for land-cover classification", *ISPRS J. Photogramm. Remote Sens.*, vol. 67, pp. 93-104, 2012, doi: 10.1016/j.isprsjprs.2011.11.002.

[19] G. Biau, B. Cadre y L. Rouvière, "Accelerated gradient boosting", *Mach Learn*, vol. 108, no. 6, pp. 971-992, 2019, doi: 10.1007/s10994-019-05787-1.

[20] J. S. Rojas, A. Pekar, Á. Rendón y J. C. Corrales, "Smart User Consumption Profiling: Incremental Learning-Based OTT Service Degradation", en *IEEE Access*, vol. 8, pp. 207426-207442, 2020, doi: 10.1109/ACCESS.2020.3037971.