Ensemble-based Deep Learning model for network traffic classification

Ons Aouedi, Kandaraj Piamrat, Benoît Parrein

Ensemble-based Deep Learning model for network traffic classification

Ons Aouedi, Kandaraj Piamrat and Benoît Parrein
University of Nantes, LS2N (UMR 6004)
2 Chemin de la Houssinière
BP 92208, 44322 Nantes Cedex 3, France
{firstname.lastname}@ls2n.fr

***Abstract*—Network Traffic Classification enables a number of practical applications ranging from network monitoring to resource management, with security implications as well. Nowadays, traffic classification has become a challenging task in order to distinguish among a variety of applications due to the huge amount of generated traffic. Therefore, developing Machine Learning (ML) models, which can successfully identify network applications, is one of the most important tasks. However, among the ML models applied to network traffic classification so far, no model outperforms all the others. To solve these issues, this paper proposes a novel Deep Learning (DL)-based approach that incorporates multiple Decision Tree based models. This approach employs a non-linear blending ensemble method by combining tree-based classifiers through DL in order to maximize generalization accuracy. This ensemble consists of two levels called base classifiers and meta-classifiers. In the first level, Decision Tree-based models are used as the base classifiers while in the second level, DL is used as a meta-model to combine the outputs of the base classifiers. Using two publicly available datasets, we show that our proposed ensemble is suitable for network traffic classification and outperforms the linear blending (using logistic regression as meta-model) as well as several well-known ML models, which are Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), Multi-Layer Perceptron (MLP), AdaBoost, K-Nearest Neighbors (KNN), LightGBM, Catboost, and XGBoost.**

***Index Terms*—Blending, traffic classification, ensemble learning, machine learning, deep learning, decision tree.**

I. Introduction

Traffic classification is a subgroup of traffic analysis strategies that aims to classify the traffic into predefined categories, such as normal or abnormal traffic, the type of application (e.g., streaming, Web browsing, etc.), or the name of the application (e.g., YouTube, Netflix, Google, etc.) [24]. It helps Internet Services Providers (ISPs) to manage their infrastructures efficiently and ensures Quality of Services requirements. The earliest traffic classifiers were based on the port number used by each application [23], where an analysis of the packet header is applied to identify only the port numbers and their correspondence to the well-known port numbers. This made the port number-based method the simplest and fastest classifier. However, this approach has limitations, for example, some applications may use dynamic port numbers or ports associated with other protocols to hide from network security tools and hence deteriorate the accuracy of port-based classifiers. To avoid total reliance on the semantics of port numbers, Deep Packet Inspection (DPI) tools have appeared [14]. It checks all packet data, which consumes a lot of CPU resources and can cause a scalability problem as well as fails to classify encrypted traffic. To address these issues, Machine Learning (ML) is opening the ways to model, learn, and recognize hidden and complex patterns within the network traffic behavior using training data.

ML-based models can be broadly divided into two categories: simple (i.e., single) and ensemble models. Ensemble models aim to combine heterogeneous or homogeneous models (commonly classifiers) in order to obtain a model that outperforms every one of them and overcome their limitations [34]. They can be generally divided into two groups: homogeneous and heterogeneous ensembles. More specifically, several homogeneous ensemble frameworks such as bagging (e.g., Random Forest) and boosting (e.g., XGBoost) have been proposed and most of them are based on the decision tree (DT) model. On the other hand, a heterogeneous ensemble has been proposed in order to take advantage of the diversity of the models' strengths. The blending and majority voting are heterogeneous models. Blending consists of two levels: base classifier and meta-classifier. Through the use of a meta-classifier, that combines the base classifier, blending is a powerful ensemble technique. However, little attention has been paid to the application of the blending ensemble in network traffic classification. Furthermore, it is well-known that Deep Learning (DL) outperforms the other shallow ML models in several fields such as healthcare, computer vision, and network resource management, and has shown success in network traffic classification [24] [23]. DL is a branch of ML that evolved from Neural Networks (NNs) and has a unique nature and features for solving complex problems. In addition, the DT algorithm is known due to its simplicity and can be easily understood by human experts. It is one of the most suitable learning algorithms for network traffic classification [23] [3].

Based on the aforementioned reasons, in this paper, we focus on blending ensemble using DL and DT-based classifiers. During the process of ensemble construction, model selection and model combination are explored. Then, several DT-based classifiers are used as base classifiers including simple DT, RF, AdaBoost, XGBoost, and then DL is used as a meta-classifier. DL has been used as a meta-classifier in order to learn the non-linear relationship among the base classifiers. Furthermore, as ML models highly depend on the quality of data, we have used the feature selection method to identify the appropriate

features and then discard the irrelevant/redundant ones, with the goal of obtaining a subset of features that best represent the problem and decrease the training time. Finally, we have also used a second dataset including encrypted traffic (contains only time-related features) to evaluate the performance of the proposed model with several scenarios (binary classification and multi-classification). To the best of our knowledge, this investigation of the blending method is performed in network traffic classification for the first time.

A. Motivations:

From a large body of literature in network traffic classification research, we have noticed that:

- ML models are constructed and tested in specific environments (e.g., dataset, features); therefore, it is a risky and difficult task to choose the best model for general traffic classification.
- Each model has its own strength and weakness and they can complement each other to overcome their individual shortcomings.
- DL and tree-based models are widely used for traffic classification tasks and their combination can give a robust model.
- The non-linear blending model can give promising results and outperforms the linear blending model [11].

B. Key Contributions:

To address the above motivations, we propose a blending ensemble using multiple DT-based models as base classifiers and DL as a meta classifier. The main contributions of our research are as follows:

- Performance investigation of seven DT-based models with respect to their algorithms (e.g., gradient boosting or bagging).
- Improved generalization performance by combining decisions from several classifiers using the ensemble method.
- Boosted performance of traffic classification with a blending model of DT-based models and the use of DL model as meta-classifier.
- Performance evaluation of the proposed ensemble using both non-encrypted and encrypted network traffic.
- Comparison of the results against, neural networks, ensemble, and simple models, as well as some state-of-the-art approaches.

The rest of the paper is organized as follows. Section II discusses related works. Section III introduces the essential background in order to understand Section IV, which presents the proposed models. In Section V, experimental results and performance of the proposed model are presented. Discussion and analysis of the results are provided in Section VI. Finally, the conclusion is given in Section VII.

II. Related Work

This section presents an overview of state-of-the-art methods that adopt ensemble learning. Then, it presents the recent achievements of state-of-the-art approaches proposed to solve the traffic classification problem using ML/DL models.

A. Ensemble-models related work

Possebon et al. [26] focused on combining individual meta-learning techniques, including Voting, Stacking, Bagging, and Boosting, in order to find the most robust classification models for the anomaly detection system. They presented a comparative analysis among meta-learning approaches and simple classifiers for network traffic classification tasks.

Kumar et al. [20] proposed an ensemble that combines DT, Naive Bayes, and RF as first-level individual classifiers. In the next level, the classification results are used by XGBoost as a meta-classifier in order to identify normal and attack observations. The experimental results demonstrated that their ensemble model outperforms some of the existing state-of-art techniques.

Xiao et al. [35] proposed a new ensemble model using DL. The DL applied to ensemble five classification models, which are KNN, SVMs, DT, RF, and gradient boosting decision trees (GBDTs), to construct an ensemble model and predict better cancer in normal and tumor conditions. The test results indicated that the proposed ensemble increases the prediction accuracy of cancer for all the tested RNA-seq data sets as compared to using a single classifier or the majority voting algorithm.

Fang et al. [16] presented four heterogeneous ensemble-learning techniques, that is, stacking, blending, simple averaging, and weighted averaging, to predict landslide susceptibility in Yanshan County, China. These ensembles combine several classifiers such as convolutional neural network, recurrent neural network, SVM, and logistic regression in order to avoid the problem of model selection. For the stacking and blending methods, the logistic regression model was the meta-classifier for the final prediction. The experimental results indicated that the heterogeneous ensemble learning methods give better performance than the base classifiers. More specifically, the stacking and blending ensembles achieved higher prediction performance than other ensembles because they use meta-classifiers to correct the errors that occur during the learning process of the base classifiers and improve the accuracy. In addition, the blending ensemble-learning achieves the highest accuracy compared to other ensemble learning methods.

Moreover, Song et al. [30] proposed an ensemble approach where five tree-based methods are used as base learners, including Classification and Regression Tree (CART), two bagging methods of RF, Extremely Randomized Tree (Extra-Trees), two boosting methods (GBDT and XGBoost), and Linear Regression is used as a meta-learner. The experimental results indicate that their approach is more effective than the conventional tree-based classifiers and other ensemble learning tree methods. For more details about the ML-based model for traffic classification, please refer to our survey [6] [1].

Remarks: As presented in this section, the blending models have been widely used due to their remarkable generalized performance; however, to the best of our knowledge, their potential in network traffic classification is not fully studied. Consequently, our approach leverages their successful experiences and creates a new model to perform a better classification.

B. Traffic classification related work

Using the ML method, many solutions have been proposed for network traffic classification. In this context, Cherif and Kortebi [13] used the Symmetric uncertainty feature selection method followed by XGBoost for traffic classification. Peng et al. [25] tested ten well-known classifiers including Adaboost, DT (j48), RF, and Naive Bayes classifiers. Also, Belavagi and Muniyal [7] have compared several ML models for intrusion detection which are Logistic Regression, Gaussian Naive Bayes, SVM, and RF. The results indicate that RF outperforms the other classifiers. Qazi et al. [27] presented a mobile application detection framework called Atlas. This framework enables fine-grained application classification using DT (C5.0) as a classifier.

DL models have advanced considerably and are being widely adopted in several domains. Several studies show that it completely outperforms traditional methods in most areas. Thus, researchers have tried to apply DL for traffic classification and it has been used as semi-supervised and supervised learning [4] [33] based on the amount of label data. Wang [33] used deep neural networks and a deep AutoEncoder to identify protocols and they achieve excellent precision and recall rates. In addition, Aouedi et al. [4] have used DL as semi-supervised learning for network traffic classification with the help of dropout and denoising code hyper-parameters in order to improve the classification performance.

Moreover, the increasing demand for user privacy and data encryption has tremendously raised the amount of encrypted traffic on today's Internet. Consequently, encrypted traffic classification has become a challenge in modern networks. In this context, DT-based classifiers and DL can be good classifiers for encrypted traffic characterization. For instance, Draper-Gil et al. [15] have used two common machine learning algorithms, which are DT (C4.5) and KNN (K-Nearest Neighbor), to distinguish between VPN and non-VPN network traffic. The results show that DT has achieved better results. Also, Alshammari and Zincir-Heywood [2] show that DT (C5.0) performs much better than Genetic Programming and AdaBoost algorithms in classifying VoIP Skype network traffic. In this field, DL models have been used in order to accurately classify an encrypted network traffic [32] [22]. Wang et al. [32] developed encrypted data classification framework called DataNet, which is embedded in the SDN home gateway. This classification was achieved through the use of several DL techniques using encrypted traffic. The DL models used in their work are multilayer perceptron (MLP), stacked autoencoder (SAE), and convolutional neural networks (CNN).

Remarks: The current literature shows that it is promising to classify network applications using ML-based approaches. Additionally, strategies such as bagging and boosting are widely used to build ensemble models. However, our literature review shows that heterogeneous ensemble learning is still rarely used in traffic classification. Moreover, although the efficiency of DT-based classifiers and DL for network traffic classification, to the best of our knowledge no paper exploits them together in one model.

III. Background

This section details the theoretical background of the basic concepts, which are useful to understand our proposition.

A. Blending Ensemble

Blending ensemble is very close to stacking, which is originally introduced in the Netflix competition [31]. But, unlike stacking, blending uses only a holdout (validation) set from the train set to make predictions. Consequently, it is simpler and requires less computation for the training task. Moreover, it works better than stacking ensemble [16]. Blending ensemble consists of two levels, which are base-classifiers used in level-1 and meta-classifier used in level-2. The base classifiers are used to provide base predictions as new features. Then the meta-classifier is trained on these new features to give the final decision. Thus, the blending can collectively estimate the errors of all base classifiers through basic learning steps and use a meta-classifier to reduce the prediction residuals. All this made the blending leads to greater awareness and familiarity in a dataset [11]. A general overview of the training and testing process is shown in Figure 1 and Figure 2. For more details, the blending method consists of two levels and several steps:

1) Use a hold-out method to divide the training set into a new training set and validation set.
2) Train the base classifiers through the new training dataset, and the prediction of the base classifiers on the validation set forms the meta-training dataset (level-1).
3) Join the prediction on the validation set of the base classifiers to form the meta-training dataset. In other word, the prediction of the base classifiers on the validation set is concatenated to compose the meta-training set.
4) Train the meta-classifier using the meta-training dataset (level-2).
5) Use the meta-classifier to make the final prediction through the intermediate test produced by the base classifiers (Figure 2).

B. Tree-Based Single Model and Ensemble Models

1) Decision tree: is a simple algorithm that can be used for both regression and classification tasks. It is a tree-like structure where each node represents a test and the branches correspond to the partitioning of the variable above or below a splitting value for a predictor. In this work, we use the CART algorithm to build single and ensemble tree models.

2) Tree-Based Ensemble Learning Models: DT has unfavorable performance that we cannot use as a final model (low bias because it overfits the training data). One way to improve the accuracy without losing the benefits of DT is the combination of several DT models' predictions. Consequently, two well-known DT-based ensemble techniques have been proposed, namely Bagging [9] and Boosting [17].

- ***Bagging*** called bootstrap aggregating is one of the earliest ensemble method [9]. It is a parallel ensemble method that aims to decrease the variance (i.e., overfitting). The Bagging process consists of three steps (i) sub-sampling the training set randomly to obtain the sub-training sets, (ii) using these

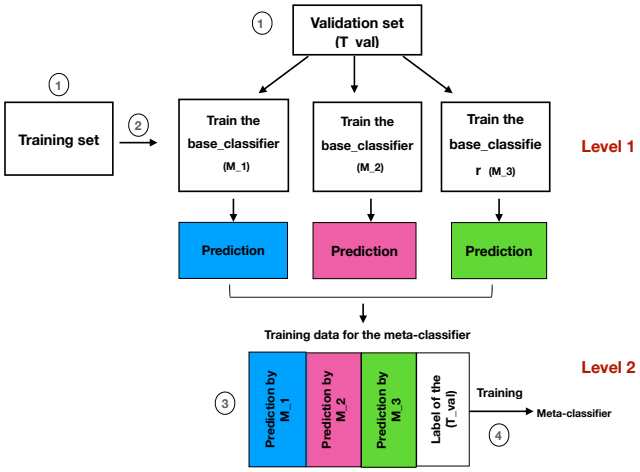

Fig. 1: Training process of the blending ensemble

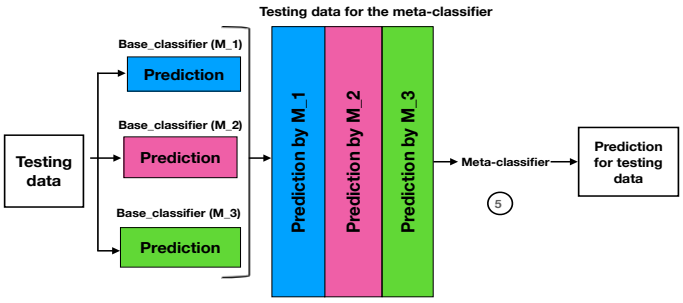

Fig. 2: Test process of the blending ensemble

sub-training sets to train several weak models independently, and (iii) combining the outcome of these models by voting technique. RF model is one of the most popular bagging-based models developed by Breiman [10]. It is an ensemble of independent DT in which the base trees train not just on a randomly chosen subset of the data, but also on a randomly chosen subset of input features. Then, the outputs of the base classifiers are combined by the majority voting technique. The main advantage of the RF algorithm, compared to the gradient boosting algorithms, is that it requires fewer hyper-parameter tuning [8] [5].

- ***Boosting*** is a sequential ensemble method used to improve the performance of the DT model [17]. It adds the models in a sequential manner and each base model reduces the error of previous base models. AdaBoost is the first boosting algorithm developed by Freund and Schapire [17]. It does not randomly select training samples like RF but focuses on the samples that do not have accurate predictions (misclassified samples). In other words, after training the base model, AdaBoost increases the weight of the misclassified samples. Therefore, AdaBoost obtains different training sets by focusing on the instances that are misclassified by the previously trained classifiers.

XGBoost is one of the most efficient implementations of gradient boosted decision trees and it is developed by Chen and Guestrin in 2014 [12]. It has been selected as one of the best ML algorithms used in Kaggle competitions due to its advantages such as easy parallelism and use as well as high prediction accuracy. This algorithm learns from the error of prior trees to improve the accuracy in subsequent iterations. However, instead of increasing the instance weights at every iteration as AdaBoost does, XGBoost tries to fit the new model according to residual errors made by the previous model.

IV. Proposition

This paper introduces a blending ensemble learning method for network traffic classification. In this section, the architecture of the proposed ensemble model and its design principles are presented. Figure 3 summarizes the methodology of this model. For the training data, *data pre-processing* is performed. It includes data normalization using Min-Max as well as the feature selection process. To automatically obtain an optimal ensemble classifier, *model hyper-parameters tuning* of each base classifier has been conducted. After extensive experiments, we decide to deploy a *blending ensemble learning* to improve the performance and the generalization capability for network traffic classification.

A. Data pre-processing

The learning algorithms require the representation of the data via a feature (attribute) set where the incoming traffic has different types of features, including numeric and categorical values. Therefore, it is important to pre-process this traffic in order to build the proposed model. Data pre-processing is a data mining technique that is used to transform the data and make it suitable for other processing use (e.g. classification, clustering). It is a preliminary step that can be done with several techniques among which are data cleaning, and feature selection.

One of the crucial steps with our model is to find the optimum number of features that can provide the best classification performance in a real dataset. To do so, we have used feature selection methods to identify the best features set in the offline run. Features selection methods try to pick a subset of features that are relevant to the target concept and have become an indispensable component of the ML process.

This task has been done in our preliminary work [5], in which we have compared Recursive Feature Elimination (RFE) against Information Gain Attribute Evaluation (IG). RFE is a wrapper method that recursively evaluates alternative sets by running some induction algorithms. Starting from all the feature sets, the method recursively removes the less relevant features. The algorithm used here is the classification and regression tree (CART). As the optimal subset has been selected by RFE we will use these features as input for the proposed ensemble.

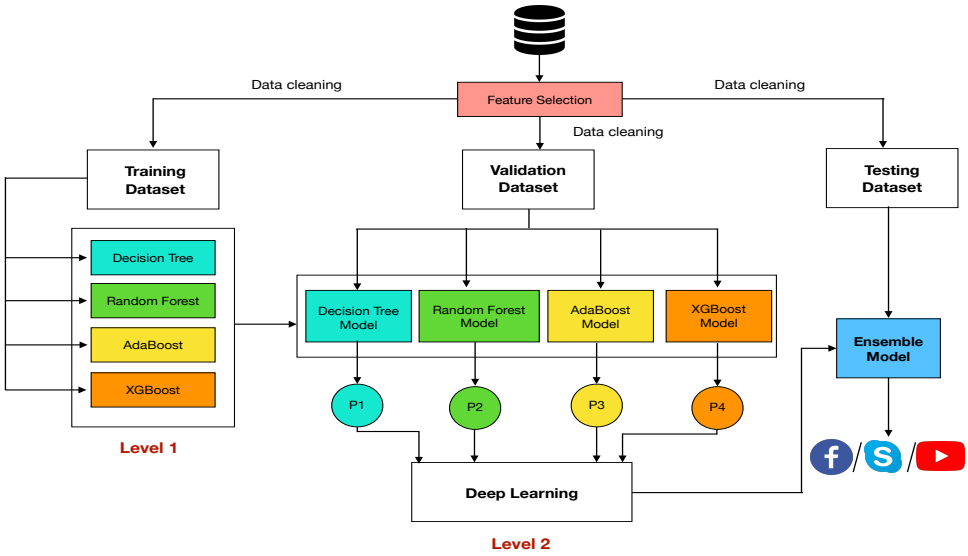

Fig. 3: Flowchart for the proposed traffic classification model

B. Models hyper-parameters tuning

The combination of several classifiers is not enough, we should also find the right configuration of the models for the base and meta classifiers by tuning its hyper-parameters, i.e. parameters that are not directly set during the training phase because they should not be learned from the training set [37]. The main steps of hyper-parameters tuning are summarized in Algorithm 1. In particular, to find the optimal hyper-parameters, we have trained our model using different combinations of hyper-parameters. Then, we evaluated the performances of the different models on the validation set. This process has been repeated until we achieved a satisfactory performance.

Algorithm 1: Hyper-parameters and parameters tuning

while *Accuracy on validation set not satisfactory* **do**
  Choose a set of hyper-parameters;
  Given the chosen hyper-parameters train the model and optimize its parameters using the training set;
  Evaluate the model performance on the validation set;
end

C. Blending ensemble model

The traffic classification framework proposed in this paper is presented in Figure 3. It is mainly composed of two levels, which are base-classifiers using DT models (level-1) and meta-classifiers using DL (level-2). The principal purpose of level-1 is to construct base classifiers models through the training set and to produce the meta-data using the validation set. Then, the meta-classifier uses the meta-data (i.e., DL in our case) for the training task and makes the final classification on the test set. More specifically, we use a hold-out method to divide the training set into a new training set and validation set. Then, the base classifiers are trained through the new training set and their predictions on the validation set are used as the meta-data for the meta-classifier. Next, the meta-classifier uses the meta-data for the training process and makes the final classification using the test set. In order to construct an ensemble model with both good generalization and classification performance, as discussed in the literature review, DT-based models are our choice as base classifiers (level-1). Since, the base classifiers need to be accurate, diverse, and complementary as possible in order to provide highly discriminative meta-data for classification [29], a comparative analysis has been done on seven DT-based models with respect to the algorithm on which it is based (e.g. gradient boosting or bagging). Next, in level-2, we use the DL model as a meta-classifier that combine the prediction output of the base classifiers (level-1), and explore the non-linear relationship among them and hence getting a non-linear blending model. This step is known as the combination phase where the DL learns from the meta-data generated by the base classifiers to find one and final decision. As DL acts as a combination method and its inputs are the prediction of the base-classifiers, known as meta-data, we use a simple Neural Network architecture (i.e., three hidden layers.)

Algorithm 2 describes the steps involved in designing blending-based ensemble design, and Table I describes the notations used in Algorithm 2.

V. Experimental Study and Results Analysis

In this section, we evaluate the performance of the proposed ensemble model by performing extensive experiments using two different datasets. Then, the obtained results are analyzed and discussed.

A. Evaluation metrics

To determine the quality of features after using feature selection methods and to evaluate the classification quality,

Algorithm 2: Learning step of the proposed model

Input: Training Dataset $DT = \{x_i, y_i\}$ $(i = 1, 2, ..., n$, $x_i \in X$, $y_i \in Y)$;
Output: An Ensemble Algorithm H;
Use Hold-out validation to divide the training set into a new training (D_train) set and validation set (D_val);
Step 1: Train the base classifiers used in the level 1
for j=1 to 4 **do in parallel**
    Train M_j from D_train;
end for
Step 2: Construct new dataset of prediction
D'=$\{P, Y\}$, where P=$M_1(D\_val)$,..., $M_4(D\_val)$;
Step 3: Train the meta-classifier MC
Train MC based on meta-data D' (level 2);
return $H(X) = MC(M_1(X), \ldots, M_4(X))$

TABLE I: List of notations used in ensemble learning model.

List of notations	Meaning
X	Set of features
Y	Class label set
M	Base classifier
H	Ensemble classifier
n	Number of training samples
D_train	training set of the base classifier (level-1)
P	Base classifier prediction using D_val
D'	meta-data used to train the meta-classifier (level-2)

several performance metrics have been used; time-related metrics (classification and training time), accuracy, F1-score, precision, and recall [22]. To calculate these metrics, there are four important terms: *TP*: True Positive, *FP*: False Positive, *TN*: True Negative, *FN*: False Negative.

Accuracy is the proportion of correct classification (TP and TN) from the overall number of cases.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

F1-score is the harmonic mean of precision and recall. If its value is high and closer to accuracy, the performance of classification is better.

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2}$$

where:

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

There are different evaluation models such as *k-fold cross-validation* and *train/validation/test split*. In this research, we used the train/validation/test split because the validation set is an essential part to build our blending ensemble. For reliable evaluation, the reported results are averaged from ten runs of each model.

B. Experiment setup

The purpose of this experiment is to select relevant features from the initial dataset to constitute a reduced dataset. Then, the performance of several supervised machine learning algorithms was analyzed and followed by a blending ensemble used for traffic classification that combines those algorithms. Python 3.7 is used as a programming language and $Scikit-learn$ 0.23 the conventional model and $Keras$ 2.4 framework for the Deep learning meta-classifier. $XGBoost$, $LightGBM$, and $Catboost$ libraries were used for those models. All experiments were run using four core Intel® Core™ i7-6700 CPU@3.40GHz processor, and 32.00 GB of RAM.

1) Dataset description: We evaluate the different classifiers on a real-world traffic dataset[1]. It is a public dataset and has been presented in a research project and collected in a network section from Universidad Del Cauca, Popayán, Colombia [28]. It has been constructed by performing packet captures at different hours, during the morning and afternoon over six days in 2017. We chose this dataset because it can be useful to find many traffic behaviors as it is a real dataset and rich enough in diversity and quantity. It consists of 87 features, 3,577,296 instances, and 78 classes (Facebook, Google, YouTube, Yahoo, Dropbox, etc.). In this experiment, we have separated the dataset into 80% for training, 10% for validation, and 10% for testing.

It is important to note that this dataset consists of flow and packet-based features (attributes) for network traffic classification tasks (e.g., flow duration, packet length, port number, etc.). The statistical classification is able to identify a protocol without the need to examine the payload carried by the packet [18]. A flow is a set of network packets with the same source/destination IP addresses, source/destination port numbers, and protocol [1].

2) Hyper-parameters selection: The selection of hyper-parameters (also called parameter tuning) is a crucial step in the construction of ML/DL-based models and the final classification results. These hyper-parameters were set to reach the right trade-off between latency and accuracy. All the hyper-parameters descriptions and settings in our experiments are presented in Table II, which lists the hyper-parameters description and their values for each model. It is important to note that the unmentioned hyper-parameters are set to the default values.

C. Ensemble-based deep learning classifier

In order to find the optimized model, we have conducted feature selection and base classifier selection as explained below.

1) Feature Selection: Using RFE, we have derived a method to identify the best 15 features out of 87 features in our preliminary work [5]. The 15 selected features are listed hereafter : *DestinationIP, sourceIP, sourcePort, destinationPort, FlowIATMax, FwdIATTotal, Timestamp, FlowDuration, InitWinBytesBackward, InitWinBytesForward, FwdPacketLengthMax, BwdPacketLengthMax, SubflowFwdBytes, BwdPacketLengthMean, FwdPacketLengthStd.*

[1]https://www.kaggle.com/jsrojas/ip-network-traffic-flows-labeled-with-87-apps

TABLE II: Hyper-parameters values of the different classifiers

Model	Hyper-parameters	Values
Decision Tree	max_depth	40
	$min\_samples\_split$	40
Random Forest	max_depth	50
	$n\_estimators$	50
AdaBoost	max_depth	35
	$n\_estimators$	50
	$learning\_rate$	0.4
XGBoost	max_depth	35
	$learning\_rate$	0.2
	$n\_estimators$	100
CatBoost	max_depth	8
	$n\_estimators$	1000
LightGBM	max_depth	35
	$boosting\_type$	goss
	$learning\_rate$	0.2
	num_leaves	1000
	top_rate	0.6
	$other\_rate$	0.4
	$n\_estimators$	250
Meta-classifier (DL)	$hidden\_layer$	3
	$activation\_function$	ReLu
	$Learning\_rate$	0.001
	$Optimizer$	Nadam

2) Base classifiers selection: To start, we first randomly partitioned the data into a ratio of 80% for training, 10% for validation, and 10% for testing the different models. Table III shows the accuracy of different DT-based models. It can be seen that RF has the best accuracy as a Bagging method with 85.28% accuracy whereas the accuracy of Extra-Trees is 84.87%. Also, it is more efficient in terms of training and classification time compared to Extra-Trees. Coming to the boosting models, where the AdaBoost and XGBoost outperform the LightGMB and CatBoost in terms of accuracy. They also outperform LightGBM in terms of classification time and CatBoost in terms of training time.

TABLE III: Comparison of different methods

Methods	Learning Algorithms	Accuracy (%)	Training time (s)	Test time (s)
Single classifier	Decision Tree	82.24	110.21	0.24
Bagging	Random Forest	85.28	193.67	9.63
	Extra Tree	84.87	205.874	150.544
Boosting	AdaBoost	88.51	7921.74	53.44
	XGBoost	88.70	52423.84	197.07
	CatBoost	77.79	231064.01	15.54
	LightGBM	84.57	6292.55	987.50

To improve the generalization performance of Tree-based models and based on the above results, we will use DT, RF, XGBoost, and AdaBoost as base classifiers (i.e., level-1 of the blending). DT is a simple and classical model, RF is a bagging model that improves the classification performance and builds different versions of the training set by sampling with replacement. XGBoost and AdaBoost as boosting models can help to avoid the problem of underfitting (bias). We have chosen XGboost and Ababoost as they perform better than the other boosting models (i.e., LightGBM, CatBoost) and their boosting process is different as explained in Section III. Although all the base models in this study are based on DT, they work in a different way (e.g., their training process does not use the same training set) and hence guarantee the diversity of level-1 of the ensemble model.

3) Proposed Ensemble classifier: The classification performance of the proposed ensemble using training and test set is presented in Table IV. It can be seen that using the training set AdaBoost gives the best results, followed by RF and XGBoost, our model, and DT. The accuracy achieved by the base classifiers which are DT, RF, AdaBoost, and XGBoost on the training set is 87.56%, 94.68%, 97.13%, and 92.59%, respectively. However, the classification performance of these models using the test set is different where their accuracies are 82.24%, 85.28%, 88.51%, and 88.70%, respectively. It is important to note here that ML models need to perform very well on the test set (i.e., unseen data during the training). In contrast to the base classifiers, the difference between classification results using the training and test set of our ensemble model is almost negligible compared to the base classifiers (Table IV), where the training and test accuracy are **91.57%** and **91.51%**. This demonstrates that this ensemble does not have a problem with overfitting. This may be attributed to the generalization ability of the blending ensemble. Moreover, its accuracy is **2.81%**, **3%**, **6.23%**, **9.27%** better than XGBoost, AdaBoost, RF, and DT, respectively. Similarly, in terms of the other measures, the proposed model also outperforms its base classifiers.

- **Impact of the hold-out validation set**

Figure 4 shows the impact of the hold-out validation set on the performance of the blending method as well as the base classifiers. The hold-out method is used to divide the training set into a new training set and a validation set. Here, the ratio of the validation set varied between 10% and 40% with a step size of 10% (we stopped because the performance started to decrease). It can be seen that the blending method maintained a better prediction performance than the base classifiers. Moreover, when the ratio of the hold-out validation set is 10%, the blending method can obtain the best results. In addition, we can notice that the performance of the base classifiers decreases with more validation sets (i.e., fewer training sets), and hence the performance of the blending method decreases. It is important to note here that we have used the same test with all the ratios of the hold-out validation set as well as for the rest of the paper.

This may be attributed to the fact that increasing the hold-out validation set (i.e., the training data for the meta-classifier) means decreasing the training set of the base classifiers. Consequently, we can notice that the performance of the blending method depends on the quality of the base classifiers. That is why choosing the appropriate models for the ensemble is a crucial step.

- **Impact of the base classifiers**

To find the base classifier that can optimize the performance of the proposed ensemble model, some experiments have been conducted (Figure 5). To do so, we try to evaluate the performance of the proposed model with different combinations of base classifiers. Figure 5 shows the accuracy of different combinations of three base classifiers and the one with all the base classifiers (proposed model). It is important to note here that with all the cases we used DL as a meta-classifier.

TABLE IV: Statistical measures of the base classifiers and blending methods using Training and Test sets.

Dataset	Model	Accuracy	Precision	Recall	F1-score
Training set	DT	87.56	87.55	87.56	87.36
	RF	94.68	94.81	94.68	94.59
	AdaBoost	**97.13**	**97.18**	**97.13**	**97.11**
	XGBoost	92.59	92.71	92.59	92.43
	Proposed model	91.57	91.81	91.57	91.70
Test set	DT	82.24	82.06	82.24	81.99
	RF	85.28	85.57	85.28	84.71
	AdaBoost	88.51	88.72	88.51	88.16
	XGBoost	88.70	88.72	88.70	88.47
	Proposed model	**91.51**	**91.59**	**91.51**	**91.64**

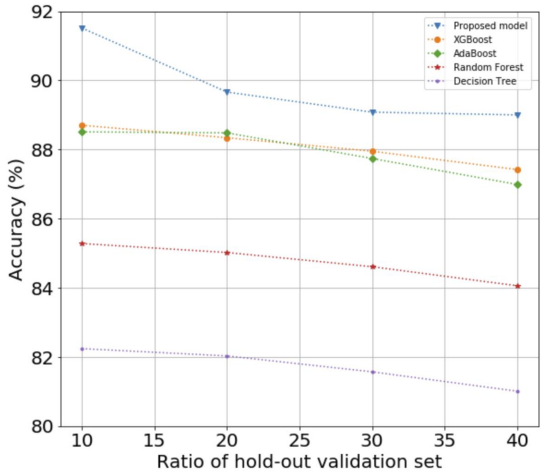

Fig. 4: Effect of hold-out validation set ratio.

The proposed model that integrates all the base classifiers has a better result than the three base classifiers. This means that no model negatively impacts the performance of the proposed ensemble. Moreover, as shown in this figure, dropping out one of the boosting models (AdaBoost and XGBoost) the accuracy of our ensemble decreases. Specifically, XGBoost plays the most critical role in achieving good performance. The accuracy of the blending decreases notably without XGBoost, which means that this model is an important component of our ensemble-learning method.

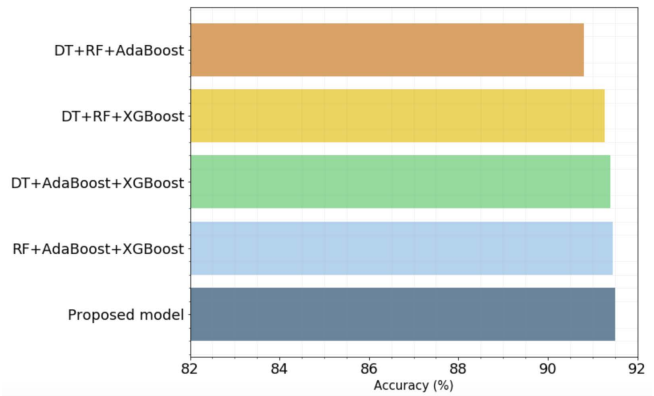

Fig. 5: Results of base classifier experiments.

- **Comparative analysis over various ML algorithms**

In order to further validate the efficiency of the proposed ensemble, which demonstrated the best performance against the base classifiers, we compared this framework with other ML models. The optimal hyper-parameters of these models have been used. These models include (i) **simple classifiers** such as SVM and KNN, (ii) **ensemble models** such as Extra-Trees, LightBGM, and CatBoost, (iii) **neural network classifier**, which is Multi-layer Perceptron (MLP) classifier. We select these classifiers as our baselines because Extra-Trees, Decision Tree, and KNN are easy to train, SVM is widely used and proved to be useful in several applications [19], and MLP is a neural network model (we have used the optimal architecture, which is two hidden layers, learning rate=0.001, and ReLu as an activation function), as well as CatBoost and LightBGM because they are recent models for the classification task.

Figure 6 shows the accuracy of the proposed ensemble and the various ML models. In this case, a clear hierarchy of models emerges from best to worst. We can see from the results that the proposed model performs well when compared with the different ML algorithms. Its accuracy is **6.64%**, **6.94%**, **13.72%**, **16.03%**, **17.85%**, **44.75%** better than Extra-Trees, LightBGM, CatBoost, MLP, KNN, and SVM, respectively. This may be attributed to the fact that the combination of DT-based models and DL helps the proposed ensemble to yield far superior results compared to several ML models. From this, we can conclude that our model is a good model and can better differentiate among applications.

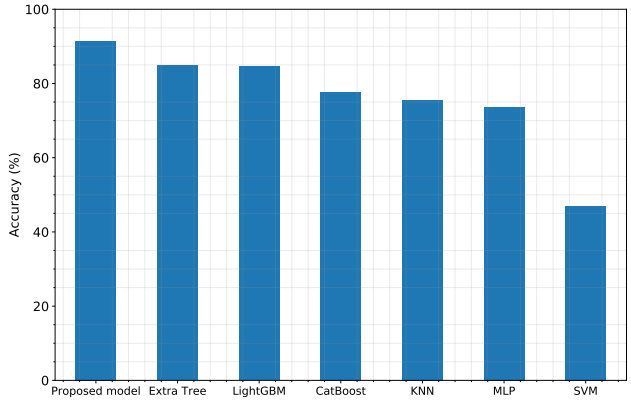

Fig. 6: Comprehensive comparison of well-known classifiers

TABLE V: Linear blending vs our model for application classification

Class Name	Linear Blending			Our model		
	Precision (%)	Recall (%)	F1-score (%)	Precision (%)	Recall (%)	F1-score (%)
Google	85.3	**93.43**	89.18	**95.11**	86.93	**90.83**
HTTP	93.65	93.26	93.45	**94.45**	**94.83**	**94.64**
Amazon	**98.52**	93.11	95.74	94.94	**98.84**	**96.85**
Microsoft	**90.24**	81.93	85.88	89.82	**93.35**	**91.55**
Skype	91.49	80.36	85.57	**92.55**	**92.15**	**92.35**
Facebook	**97.89**	91.26	94.46	94.27	**98.52**	**96.35**
Dropbox	**98.61**	92.62	95.52	96.84	**98.1**	**97.47**
Yahoo	**90.87**	77.46	83.63	86.79	**93.59**	**90.06**
Twitter	**94.24**	73.86	82.81	84.81	**94.81**	**89.53**
Apple	93.80	82.45	87.76	**94.11**	**95.65**	**94.87**
Whatsapp	**94.57**	74.2	83.15	94.5	**94.7**	**94.6**
Instagram	**94.76**	79.74	86.6	93.24	**97.78**	**95.46**
Wikipedia	**99.32**	71.71	83.29	90.41	**98.01**	**94.06**
Netflix	**98.99**	69.01	81.33	82.91	**97.76**	**89.72**
Spotify	**96.3**	80	87.39	88.63	**96.69**	**92.49**
TeamViewer	**100**	87.3	93.22	96.82	**100**	**98.38**
Telegram	0	0	0	**100**	**100**	**100**

- **Comparative analysis over linear blending**

In this section, we compare the proposed model with the linear blending model where the logistic regression model is used as a meta-classifier instead of a DL model. In other words, here we have used the same base classifiers and we only changed the meta-classifier in order to study the impact of DL on our ensemble. Figure 7 shows that our model outperforms the linear blending on all metrics. Then, Table V presents the achieved performance of both linear and non-linear blending for the application identification task. It can be seen that the F1-score (the harmonic mean of precision and recall) of our model outperforms that of linear blending for all the applications. Also, we can notice that linear blending can not detect the Telegram application whereas our model achieved a 100% F1-score. This is maybe attributed to the non-linearity of the DL as a meta-classifier in our model.

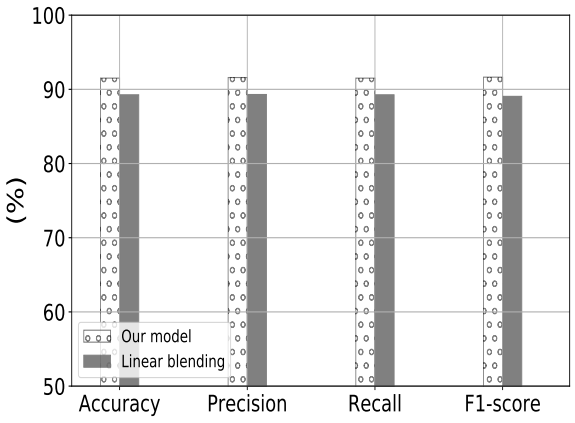

Fig. 7: Our model vs linear blending

- **Computational costs of the proposed model**

Also, we have analyzed the computational efficiency of the proposed model against other ML models. This comparison has been done in terms of training and classification time. One of the advantages of this model is that the base classifiers: DT, RF, AdaBoost, and XGBoost can be trained in parallel. Consequently, since boosting models take a long time to train (Table III), thus they can impact the training time of the proposed model more than RF and DT.

As shown in Table VI, we have compared the classification time (CT) per sample and the training time (TT) taken by respective models. As explained in Section IV, our model consists of two main phases (level 1, and level 2 process) and it trains data more than the base classifier (training set+validation set). Thus, this can explain its high training time. In addition, there is a slight difference with XGBoost. However, the training time can be proceeded offline and thus does not impact the real-time utilization of the classification process.

TABLE VI: Training and classification time comparison

Metric	XGBoost	AdaBoost	Linear blending	Our model
TT (s)	52423	7921	52523	53100
CT_per_sample (µs)	550.89	149.38	552.67	582.7

D. Experiments on the second dataset (VPN-nonVPN dataset)

To validate the effectiveness of the proposed ensemble, we also implemented experiments based on another public dataset, which includes encrypted data (VPN and nonVPN data). This is one of the most popular encrypted traffic classification datasets. It contains only time-related features. For more details on the captured traffic and the traffic generation process, refer to [15].

To evaluate the performance of the proposed ensemble, we used two scenarios as shown in Table VIII. Scenario A, **(Sc_A)**, is a binary classification to indicate whether the traffic flow is VPN or not. Both scenario B, **(Sc_B)**, and scenario C, **(Sc_C)**, are 7-classification tasks. Scenario B is to distinguish between seven non-VPN traffic services like audio, browsing, etc. Scenario C is similar to Scenario B, while its target labels are seven traffic services of the VPN version Scenario D,

TABLE VII: Linear blending vs our model using VPN-nonVPN dataset

Class Name	Linear Blending			Our model		
	Precision (%)	Recall (%)	F1-score (%)	Precision (%)	Recall (%)	F1-score (%)
Browsing	92.73	**98.08**	95.33	**98.91**	97.15	**98.02**
Chat	95.24	90.91	93.02	**97.50**	**95.12**	**96.29**
FT	**95.45**	87.50	91.30	85.71	**98.82**	**91.80**
Mail	**100**	**100**	**100**	92.30	**100**	96.00
P2P	93.75	**100**	96.77	**100**	95.91	**97.91**
Streaming	**100**	75.00	85.71	87.17	**97.14**	**91.89**
VOIP	**100**	98.48	99.24	99.62	**100**	**99.81**
VPN-Browsing	91.67	89.19	90.41	**97.39**	**97.39**	**97.39**
VPN-Chat	83.33	86.96	85.11	**90.32**	**97.39**	**93.72**
VPN-FT	90.70	90.70	90.70	**96.27**	**99.04**	**97.64**
VPN-Mail	**100**	83.33	90.91	**100**	**93.33**	**96.55**
VPN-P2P	80.95	**94.44**	**87.18**	**95.32**	80.31	87.17
VPN-Streaming	**100**	**100**	**100**	97.82	**100**	98.90
VPN-VOIP	97.73	**100**	98.85	**100**	99.54	**99.77**

(Sc_D), mixes all fourteen applications to perform the 14-classification task. To find the most relevant features in the two scenarios, we have used RFE also as a feature selection set in order to find the optimal subset. Then, using those features, we compared the performance of our model against several ML-based classifiers as well as against the linear-blending model in the two scenarios.

TABLE VIII: Scenario description.

Scenario	Description
Sc_A	VPN and Non-VPN traffic identification 2-class
Sc_D	All traffic classification 14-class

- **Comparative analysis over various ML algorithms**

It can be seen from Table IX that our ensemble achieves the best results with the VPN-nonVPN dataset on the two scenarios. This demonstrates that our model can achieve high performance using only time-related features. Specifically, for example, in scenario A, the accuracy of our model is **9.69%**, **14.67%**, **36.22%**, **28.21%**, **7.12%**, **6.85%**, **5.89%**, **5.28%**, **6.88%**, **5.02%** better than DT, KNN, SVM, MLP, Extra-Tree, RF, CatBoost, LightGBM, AdaBoost, XGBoost, respectively. Also, in scenario D, the accuracy of our model is **18.02**, **24.14**, **52.66**, **42.22**, **12.73**, **12.66**, **11.65**, **9.95**, **13.21**, **10.15**, better than DT, KNN, SVM, MLP, Extra-Tree, RF, CatBoost, LightGBM, AdaBoost, XGBoost, respectively. These results illustrate the high performance of our model with both binary and multi-classification scenarios.

- **Comparative analysis over linear blending**

Similar to the first dataset, we compared the proposed model with the linear blending model using the VPN-nonVPN dataset. Figure 8 shows that our model outperforms the linear blending on all metrics, similar to the first dataset. Also, to better evaluate the performance of our model, the fourteen classification tasks of the VPN-nonVPN dataset are reported in Table VII. From Table VII, it can be seen that our model, achieves the F1-score up to 91% in mostly all categories. Also, our model has a better F1-score than linear blending in 11 out of 14 cases. Consequently, we can conclude that our model is a promising model and can better differentiate the applications.

- **Performance against state-of-the-art models**

TABLE IX: The classification accuracy (%) of baseline and ensemble methods on VPN-nonVPN Dataset.

Model	Sc_A	Sc_D
DT	87.85	78.83
KNN	82.87	72.71
SVM	61.32	44.19
MLP	69.33	54.63
Extra-Tree	90.42	84.12
RF	90.69	84.19
CatBoost	91.65	85.20
LightGBM	92.26	86.90
AdaBoost	90.66	83.64
XGBoost	92.52	86.70
Linear blending	95.48	93.88
Proposed model	**97.54**	**96.85**

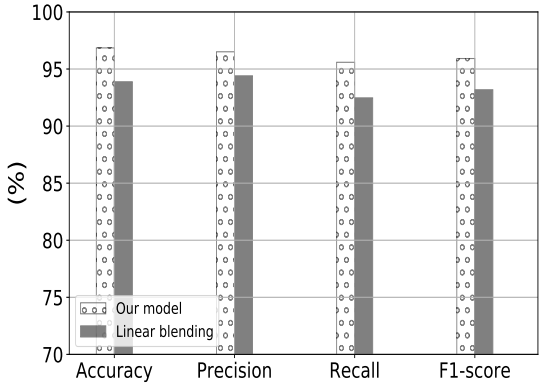

Fig. 8: Our model vs linear blending

Finally, we apply our ensemble on the VPN-nonVPN dataset with some state-of-art approaches including [15], [36], and [21] to validate the performance of our ensemble model. [15] is the original dataset paper, where the authors used a decision tree, i.e., C4.5 and KNN classifiers in order to characterize network traffic. However, since the decision tree performed a little better we use its result for the comparison

task. In contrast, the authors in [36] and [21] proposed DL-based ensemble approaches. More specifically, Yao et al. [36] used the Attention Based Long Short-Term Memory (LSTM) model in order to improve the classification performance. Then, Lin et al. [21] combined Convolutional neural networks and LSTM models to extract the spatio-temporal features and realize the traffic classification.

From the simulation results in Table X, we find that the blending scheme (linear/non-linear) can achieve competitive accuracy with the state-of-art methods. In particular, we can see that our model outperforms the proposed approach in [15] for both scenarios (Sc_A and Sc_D), however, it is not the case with [36] and [21] for the Sc_A (binary classification). For Sc_D (multi-classification) we can see that the blending scheme and especially our model (non-linear blending) achieves much better accuracy performances. This demonstrates that with a more complicated classification task (multi-classification) our model performs well. This is because the use of a meta-classifier corrects the errors that occur during the learning process of the base classifiers.

TABLE X: The classification accuracy (%) of baseline and ensemble methods on VPN-nonVPN Dataset.

Schemes	Sc_A	Sc_D
DT [15]	89.7	81.77
Attention-LSTM [36]	**99.7**	91.2
CNN-LSTM [21]	**99.7**	91.7
Linear blending (ours)	95.48	**93.88**
Non-linear blending (ours)	97.51	**96.85**

VI. Discussion

In this study, a novel classifier model is proposed to explore the potential of ensemble learning and improve the performance of DT-based models. The results demonstrate that by taking advantage of several classifiers, our ensemble model is shown to be accurate and efficient for traffic classification tasks. Specifically, the use of a meta-classifier corrects the errors that occur during the learning process of the base classifiers. Moreover, it prevents overfitting and reduces bias simultaneously to some extent (Table IV). The presented results also confirm that deep learning discovers the nonlinear relationships with very little engineering by hand and increases the learning ability of the whole ensemble model. In addition, as the performance of the classifier depends on the quality of features used during the training process, we used the feature selection method in order to reduce the complexity and improve the performance of the proposed model. However, the shortcoming of the proposed method is that (i) it takes more training time than the base classifiers, and (ii) it is difficult to decide which classifiers should be used in level-1.

VII. Conclusion

In this paper, ensemble learning based on DL and four DT-based models has been proposed consisting of pre-processing tasks and classification tasks in the proposed ensemble. For the base classifiers selection, a comparative analysis has been conducted based on the complexity and the accuracy of the classifiers. Next, an ensemble model that incorporates several DT-based models and DL is applied to improve overall classification accuracy. By using DL as a meta-classifier, the relationships among the base classifiers are learned automatically, thus enabling the ensemble method to achieve better classification. The simulation results show that the proposed ensemble model outperforms other traditional machine learning models (DT, SVM, KNN) and ensemble learning models (e.g., RF, MLP) as well as the linear blending model (logistic regression as meta-classifier). Moreover, we have studied the impact of the base classifiers and the hold-out validation set ratio on the performance of the whole model. In addition, the performance of the proposed model is evaluated using two datasets with different use cases and scenarios.

References

[1] Mahmoud Abbasi, Amin Shahraki, and Amir Taherkordi. Deep learning for network traffic monitoring and analysis (ntma): A survey. *Computer Communications*, 2021.

[2] Riyad Alshammari and A Nur Zincir-Heywood. Identification of voip encrypted traffic using a machine learning approach. *Journal of King Saud University-Computer and Information Sciences*, 27(1):77–92, 2015.

[3] Pedro Amaral, Joao Dinis, Paulo Pinto, Luis Bernardo, Joao Tavares, and Henrique S Mamede. Machine learning in software defined networks: Data collection and traffic classification. In *Proceedings of the 24th International Conference on Network Protocols (ICNP)*, pages 1–5, Singapore, November 2016.

[4] Ons Aouedi, Kandaraj Piamrat, and Dhruvjyoti Bagadthey. A semi-supervised stacked autoencoder approach for network traffic classification. In *Proceedings of the 28th International Conference on Network Protocols (ICNP) HDR-Nets Workshop*, Madrid, Spain, October 2020.

[5] Ons Aouedi, Kandaraj Piamrat, and Benoît Parrein. Performance evaluation of feature selection and tree-based algorithms for traffic classification. In *2021 IEEE International Conference on Communications (ICC) DDINS Workshop*, Montreal, Canada, June 2021.

[6] Ons Aouedi, Kandaraj Piamrat, and Benoît Parrein. Intelligent traffic management in next-generation networks. *Future internet*, 14(2):44, 2022.

[7] Manjula C Belavagi and Balachandra Muniyal. Performance evaluation of supervised machine learning algorithms for intrusion detection. *Procedia Computer Science*, 89(2016):117–123, 2016.

[8] Candice Bentéjac, Anna Csörgő, and Gonzalo Martínez-Muñoz. A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, pages 1–31, 2020.

[9] Leo Breiman. Bagging predictors. *Machine learning*, 24 (2):123–140, 1996.

[10] Leo Breiman. Random forests. *Machine learning*, 45(1): 5–32, 2001.

[11] Chia-Hsiu Chen, Kenichi Tanaka, Masaaki Kotera, and Kimito Funatsu. Comparison and improvement of the predictability and interpretability with ensemble learning models in qspr applications. *Journal of Cheminformatics*, 12:1–16, 2020.

[12] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[13] Iyad Lahsen Cherif and Abdesselem Kortebi. On using extreme gradient boosting (xgboost) machine learning algorithm for home network traffic classification. In *2019 Wireless Days (WD)*, pages 1–6. IEEE, 2019.

[14] Alberto Dainotti, Antonio Pescape, and Kimberly C Claffy. Issues and future directions in traffic classification. *IEEE network*, 26(1):35–40, 2012.

[15] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. Characterization of encrypted and vpn traffic using time-related. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, pages 407–414, 2016.

[16] Zhice Fang, Yi Wang, Ling Peng, and Haoyuan Hong. A comparative study of heterogeneous ensemble-learning techniques for landslide susceptibility mapping. *International Journal of Geographical Information Science*, pages 1–27, 2020.

[17] Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[18] Joao V Gomes, Pedro RM Inácio, Manuela Pereira, Mário M Freire, and Paulo P Monteiro. Detection and classification of peer-to-peer traffic: A survey. *ACM Computing Surveys (CSUR)*, 45(3):1–40, 2013.

[19] Ilhan Firat Kilincer, Fatih Ertam, and Abdulkadir Sengur. Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Computer Networks*, page 107840, 2021.

[20] Prabhat Kumar, Govind P Gupta, and Rakesh Tripathi. An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for iomt networks. *Computer Communications*, 2020.

[21] Kunda Lin, Xiaolong Xu, and Honghao Gao. Tscrnn: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of iiot. *Computer Networks*, 190:107974, 2021.

[22] Mohammad Lotfollahi, Mahdi Jafari Siavoshani, Ramin Shirali Hossein Zade, and Mohammdsadegh Saberian. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Computing*, 24 (3):1999–2012, 2020.

[23] Thuy TT Nguyen and Grenville Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE communications surveys & tutorials*, 10 (4):56–76, 2008.

[24] Fannia Pacheco, Ernesto Exposito, Mathieu Gineste, Cedric Baudoin, and Jose Aguilar. Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. *IEEE Communications Surveys & Tutorials*, 21(2):1988–2014, 2018.

[25] Lizhi Peng, Bo Yang, Yuehui Chen, and Zhenxiang Chen. Effectiveness of statistical features for early stage internet traffic identification. *International Journal of Parallel Programming*, 44(1):181–197, 2016.

[26] Isadora P Possebon, Anderson S Silva, Lisandro Z Granville, Alberto Schaeffer-Filho, and Angelos Marnerides. Improved network traffic classification using ensemble learning. In *2019 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2019.

[27] Zafar Ayyub Qazi, Jeongkeun Lee, Tao Jin, Gowtham Bellala, Manfred Arndt, and Guevara Noubir. Application-awareness in SDN. In *Proceedings of the ACM SIGCOMM conference on SIGCOMM*, pages 487–488, Hong Kong, China, August 2013.

[28] Juan Sebastián Rojas, Álvaro Rendón Gallón, and Juan Carlos Corrales. Personalized service degradation policies on OTT applications based on the consumption behavior of users. In *Proceedings of the International Conference on Computational Science and Its Applications*, pages 543–557, Melbourne, Australia, July 2018.

[29] M Paz Sesmero, Agapito I Ledezma, and Araceli Sanchis. Generating ensembles of heterogeneous classifiers using stacked generalization. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(1): 21–34, 2015.

[30] Jiahui Song, Yi Wang, Zhice Fang, Ling Peng, and Haoyuan Hong. Potential of ensemble learning to improve tree-based classifiers for landslide susceptibility mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:4642–4662, 2020.

[31] Andreas Töscher, Michael Jahrer, and Robert M Bell. The bigchaos solution to the netflix grand prize. *Netflix prize documentation*, pages 1–52, 2009.

[32] Pan Wang, Feng Ye, Xuejiao Chen, and Yi Qian. Datanet: Deep learning based encrypted network traffic classification in sdn home gateway. *IEEE Access*, 6:55380–55391, 2018.

[33] Zhanyi Wang. The applications of deep learning on traffic identification. *BlackHat USA*, 24(11):1–10, 2015.

[34] Michał Woźniak, Manuel Graña, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3–17, 2014.

[35] Yawen Xiao, Jun Wu, Zongli Lin, and Xiaodong Zhao. A deep learning-based multi-model ensemble method for cancer prediction. *Computer methods and programs in biomedicine*, 153:1–9, 2018.

[36] Haipeng Yao, Chong Liu, Peiying Zhang, Sheng Wu, Chunxiao Jiang, and Shui Yu. Identification of encrypted traffic through attention mechanism based long short term memory. *IEEE Transactions on Big Data*, 2019.

[37] Alessio Zappone, Marco Di Renzo, and Mérouane Debbah. Wireless networks design in the era of deep learning: Model-based, ai-based, or both? *IEEE Transactions on Communications*, 67:7331–7376, 2019.