Decision tree-based blending method using deep-learning for network management

Ons Aouedi, Kandaraj Piamrat, Benoît Parrein

Decision tree-based blending method using deep-learning for network management

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 is a key component for network management, Quality-of-Service management, as well as for network security. Therefore, developing machine learning (ML) methods, which can successfully distinguish network applications from each other, is one of the most important tasks. However, among the classification methods applied to network traffic classification so far, there is no one method that outperforms all the others. They all have advantages and inconveniences depending on the application domain. Therefore,this paper proposes an intelligent traffic management model by using deep learning (DL) that incorporates multiple Decision Tree-based models. Our model deploys a blending ensemble learning method to combine tree-based classifiers in order to maximize generalization accuracy. Using two datasets, we show that our proposed ensemble model is efficient for network traffic classification. Furthermore, the proposed approach is also compared against other representative machine-learning and deep-learning models and the results demonstrate that our approach provides better performance compared to others.**

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

I. Introduction

Traffic classification is a subgroup of traffic management strategies that aim 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.) [1]. Identifying different applications from traffic is critical to manage bandwidth resources and to ensure Quality of Service requirements. Consequently, traffic classification helps Internet Services Providers to manage their infrastructures efficiently and hence improve the service offered to their customers. However, traditional techniques such as port-based classification and deep packet inspection are becoming less efficient to handle and classify heterogeneous traffic (i.e., various applications). In this line, ML is opening the ways to develop powerful network traffic classifiers, which achieve an acceptable tradeoff between complexity and accuracy.

ML models can be broadly divided into two categories, which are simple (i.e., individual) machine models and ensemble models. Ensembles are well-established ML techniques that can obtain more accurate prediction results by integrating various base learners [2]. Considering the risk and the difficult task of choosing the suitable model for traffic classification, ensemble learning can be a promising solution as it aims to combine heterogeneous or homogeneous ML models (commonly classifiers) in order to obtain a model that outperforms every one of them and overcome their limitations [3]. Ensemble learning can be generally divided into two groups: (i) homogeneous ensemble (like bagging or boosting) and (ii) heterogeneous ensembles such as blending. The blending takes advantage of different ML models. Through the use of a learning method in the combination stage (i.e., meta-classifier), the blending model is much more powerful than single models. However, little attention has been paid to the application of the blending ensemble to integrate multiple types of classifiers in the network context. Moreover, it is well-known that DL techniques outperform the other ML methods in several fields and have shown success in network traffic classification [1]. In addition, DT-based models are the most suitable learning algorithm for network traffic classification [4] as they can produce solutions in the form of models, which can be easily understood by human experts.

For the aforementioned reasons, in this paper, we focus on the non-linear blending model based on DL as a non-linear meta-classifier using DT-based models as the base classifiers. We propose this ensemble because the non-linear blending performs better than uniform blending (majority voting) and linear blending [5]. Consequently, during the process of blending construction, model selection and model combination are explored. Then, several DT-based classifiers are used as base classifiers including simple DT, Random Forest (RF), AdaBoost, and XGBoost, and then DL is used as a meta-classifier in order to learn the non-linear relationship among the base classifiers. To the best of our knowledge, this investigation of the non-linear blending method is performed in network traffic classification for the first time.

The rest of the paper is organized as follows. Section II discusses related works. Section III presents the proposed models. In Section IV, experimental results and performance of the proposed model are presented. Finally, conclusions and future works are given in Section V.

II. Related Work

This section presents an overview of state-of-the-art approaches that are proposed to solve the traffic classification problem using ML/DL models.

Using Machine learning methods, many solutions have been proposed for network traffic classification. In this context, Cherif et al. [6] used the symmetric uncertainty feature selection method then XGBoost for traffic classification. Peng et al. [7] tested ten well-known classifiers includes Adaboost, DT, RF, Naive Bayes classifiers. Also, Belavagi et al. [8] have compared several ML models for intrusion detection which are Logistic Regression, Gaussian Naive Bayes, SVM, and RF. The results were indicated obtained that RF outperforms the other classifiers. Qazi et al. [9] presented a mobile application detection framework, called *Atlas*. This framework enables fine-grained application classification using DT 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 based on the amount of label data. In such context, Aouedi et al. [10] 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 classifier and DL can be good classifiers for encrypted traffic characterization. For instance, Draper et al. [11] have used two common ML models, which are DT 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 et al. [12] show that DT performs much better than Genetic Programming and AdaBoost algorithms in classifying VoIP Skype network traffic. In addition, in this field, DL models have been used in order to accurately classify the encrypted network traffic [13] [14].

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 as ensemble models. From the above review, it can be seen that heterogeneous ensemble-learning models' potential in network traffic classification is not fully studied, despite, they achieved a remarkable generalization performance [15]. 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. Consequently, our approach leverages their successful experiences and creates a new model to perform a better classification.

III. Proposition

This paper introduces a new decision tree-based ensemble learning method for network traffic classification. In this section, the architecture of the proposed ensemble model and its principle design are presented. Figure 1 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. After extensive experiments, we decide to deploy a *blending ensemble learning* to improve the performance and generalization capability for network traffic classification.

A. Data pre-processing

Data pre-processing is a data mining technique that is used to transform the data and makes it suitable for another processing use (e.g. classification, clustering). It is a preliminary step that can be done with several techniques among which feature selection. This task was done in our preliminary work [16], 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. After the optimal subset has been selected by RFE, we use these features as input of the proposed ensemble model.

B. Blending ensemble learning

As shown in Figure 1, our model is mainly composed of two levels, which are base-classifiers using DT-based models (level-1) and meta-classifiers using DL (level-2). The principal purpose of level-1 is to construct base classifier models utilizing the training set and to produce the meta-data for the meta-classifier (i.e., DL in our case) using the validation set. Then, the meta-classifier model is used to make the final classification. In other words, we use a hold-out method to divide the training set into a new training set and validation set. Then, the base classifiers models are constructed through the new training set and their predictions on the validation set to generate the meta-data. Next, the meta-classifier uses this meta-data for the training process and to make the final classification on the test set.

In order to construct an ensemble model with both good generalization and classification performance, (as discussed in the literature review), DT, RF, Adaboost, and XGBoost models were 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 [17], 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). Here, we compute the average of the evaluation metric from ten runs. Next, in level-2, we used the DL model to combine the prediction output of the base classifiers (level-1), and explore the non-linear relationship among the base classifiers and in turn, get a non-linear blending model. This step is known as the combination phase where the DL learns from the meta-data generated by the level-1 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 1 explains the steps involved in designing a blending-based ensemble, and Table I explains the notations used in the algorithm.

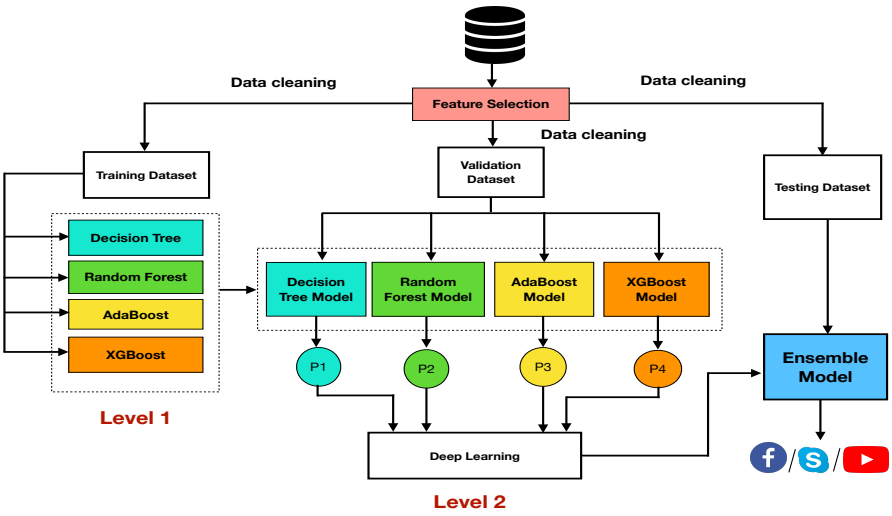

Fig. 1: Flowchart for the proposed traffic classification model

Algorithm 1: 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)
D_val	validation dataset
P	Base classifier prediction using D_val
D'	meta-data used to train the meta-classifier (level-2)

IV. Experimental Study and Results Analysis

In this section, we evaluate the performance of the proposed ensemble model by performing extensive experiments. The obtained results are analyzed and discussed as well as we show on the used datasets that our proposed model improves the classification performance (e.g., accuracy and generalization).

A. Experiment setup

Here, we compute the average of the evaluation metric from ten runs. Also, there are different evaluation models 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.

The code is made available online [1].

- **Dataset description**

We evaluate the different classifiers on a real-world traffic dataset. This dataset was presented in a research project and collected in a network section from Universidad Del Cauca, Popayán, Colombia [18]. It was 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.

B. Ensemble-based deep learning classifier

In order to find the optimal model, we conduct base classifiers 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 [16]. The 15 selected features are listed hereafter : *DestinationIP, sourceIP, sourcePort, destinationPort, FlowIATMax, FwdIATTotal, Timestamp, FlowDuration, InitWinBytesBackward, InitWinBytesForward, FwdPacketLengthMax, BwdPacketLengthMax, SubflowFwdBytes, BwdPacketLengthMean, FwdPacketLengthStd.*

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 methods. It can be seen that RF has the best accuracy as a bagging method

[1] https://github.com/aouedions11/Decision-tree-based-blending-method-using-deep-learning-for-network-management.git

TABLE II: 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	89.78	89.74	89.78	89.95
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	**89.77**	**89.70**	**89.76**	**89.93**

with 85.28% accuracy whereas the accuracy of Extra-Tree 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 LightGBM and CatBoost. The accuracy of these models is 88.51% and 88.70%, respectively (they 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

Based on these results, 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, and XGBoost and AdaBoost as boosting models can help to avoid the problem of underfitting (bias). We have chosen XGboost and Adaboost as they perform better than the other boosting models (i.e., LightGBM, CatBoost) and their boosting process is different. 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 guarantee the diversity of level-1 models.

3) Results of the proposition: In this subsection, different results are presented along with their analysis and discussion.

- **Classification performance**

The classification performance of the proposed ensemble using training and test set is presented in Table II. Note that the DL (meta-classifier) hyper-parameters is as follows, number of hidden layer=3, activation function=ReLu, learning rate=0.001, and optimizer=Adam.

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 (Table II and Figure 7), where the training and test accuracy are **89.78%** and **89.77%**. This demonstrates that our ensemble does not have an overfitting problem. This may be attributed to the generalization ability of the blending ensemble. Moreover, its accuracy is **1.07%**, **1.26%**, **4.49%**, **7.53%** 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 2 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). Also, we evaluated the impact of this ratio in terms of several evaluation metrics. It can be seen that the blending method maintained a better prediction performance than the base classifiers in all cases in terms of different evaluation metrics. 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., less training set), and hence the performance of the blending method decreases. This is because increasing the hold-out validation set (i.e., the training data for the meta-classifier) means decreasing the training set of the base classifiers. Therefore, the performance of the blending method depends on the quality of the base classifiers and that is why choosing the appropriate models for the ensemble is a crucial step.

- **Impact of the base classifiers**

To find the base classifiers that can impact the performance of the proposed ensemble model, some experiments have been conducted. We tried to evaluate the performance of the proposed model with different combinations of base classifiers. Figure 3 presents the accuracy of different combinations of three base classifiers and the one with all the base classifiers

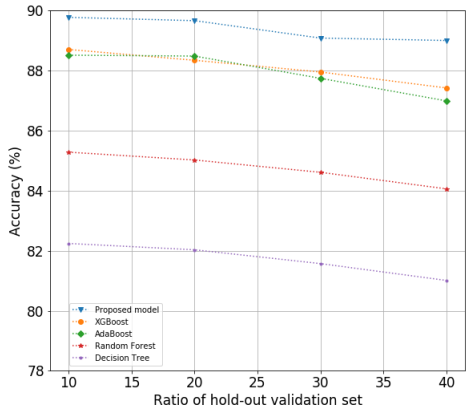

Fig. 2: Effect of hold-out validation set ratio

(proposed model). It is important to note here that with all the cases, we used DL as a meta-classifier.

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 when we drop out one of the boosting models (AdaBoost and XGBoost) the accuracy of our ensemble decreases significantly. 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.

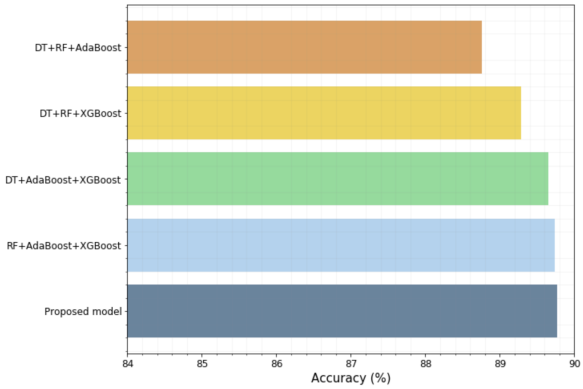

Fig. 3: Results of base classifier experiments.

- **Comparative analysis over various ML algorithms**

In order to further validate the effectiveness of the proposed ensemble, which demonstrated the best performance against the base classifiers, we compared this framework with some representative ML algorithms. The optimal hyper-parameters of these algorithms have been used. These classifiers 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, DT, 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, the same learning rate, and the activation function as our meta-classifiers), as well as CatBoost and LightBGM because they are recent models for the classification task.

Figure 4 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 **4.9%**, **5.2%**, **11,98%**, **14.29%**, **16.11%**, **42.75%** 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 to yield far superior results compared to several ML models. Consequently, we can conclude that our model is a good model and can better differentiate the applications.

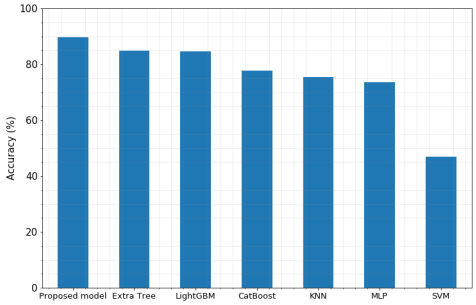

Fig. 4: Comprehensive comparison of well-known classifiers

- **Time cost of the proposed model**

We now analyze the computational efficiency of the proposed model against other ML models. This comparison has been done in terms of training and classification time. All experiments were run using four core Intel® Core™ i7-6700 CPU@3.40GHz processor, and 32.00 GB of RAM. 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 more the training time of the proposed model than RF and DT.

As shown in Figure 5 and Figure 6, we have compared classification time (CT) per sample and the training time (TT) taken by respective models. In the case of AdaBoost, TT is **7921s** and CT is **149.38µs**. Whereas, for XGBoost, TT is 52×10^3s and CT is **550.89µs**. Finally, for the blending model, TT is 53×10^3s and CT is **581.6µs**. Moreover, as shown in Figure 7, that the meta-classifier (i.e., DL) converges quickly and its accuracy starts to be stable after 20 epochs, indicating that the training process of the DL is time-efficient. Thus, although we used a large amount of data, DL (level-2 of the blending) requires just a few epochs to converge. This is because the new features used by the DL are the output of the base classifiers. Consequently, the features deployed by the meta-classifier are relevant features and can help it to converge fast and very well.

As explained in Section III, 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.

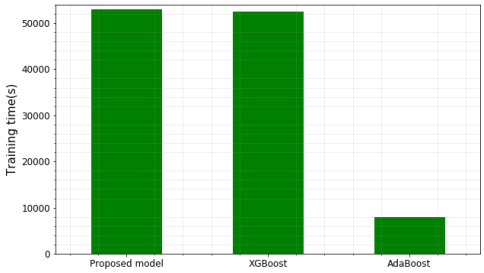

Fig. 5: Training time comparison

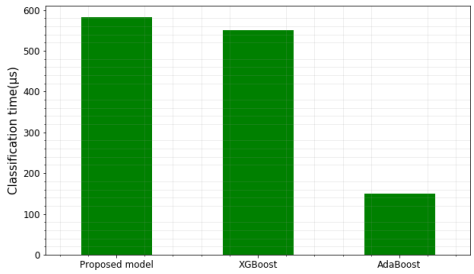

Fig. 6: Classification time per sample

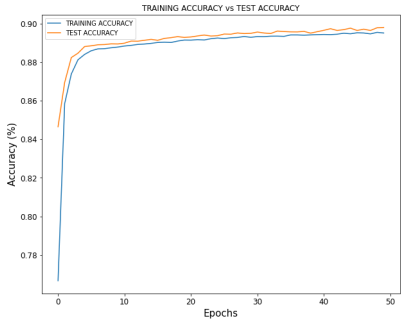

Fig. 7: The training accuracy vs validation accuracy for the proposed ensemble

C. Experiments on other datasets

Recently, as many solutions have been proposed for encrypted traffic classification, we also implemented experiments based on another dataset, called *VPN-nonVPN dataset* in order to validate the effectiveness of the proposed ensemble on encrypted traffic. The VPN-nonVPN dataset consists of encrypted traffic characterized only by time-related features [11].

- **The proposed ensemble with VPN-nonVPN dataset**

To evaluate the performance of the proposed ensemble, four scenarios of different classification tasks were proposed in experiments (for more details on the captured traffic and the traffic generation process, refer to [11]). 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 **(Sc_D)** mixes all the fourteen types mentioned in scenario B and scenario C to perform the 14-classification task. Similar to the first dataset, we have used the RFE as a feature selection method for each scenario.

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

Model	Sc_A	Sc_B	Sc_C	Sc_D
DT	87.85	89.85	84.52	78.83
KNN	82.87	86.39	80.79	72.71
SVM	61.32	69.49	58.41	44.19
MLP	69.33	78.40	75.80	54.63
Extra-Tree	90.42	92.89	87.77	84.12
RF	90.69	93.24	88.78	84.19
CatBoost	91.65	94.61	88.26	85.20
LightGBM	92.26	94.87	89.59	86.90
AdaBoost	90.66	93.67	89.23	83.64
XGBoost	92.52	94.71	89.45	86.70
Proposed model	**96.16**	**95.76**	**92.86**	**94.02**

It can be seen from Table IV that our ensemble achieved the best results with the VPN-nonVPN dataset in all scenarios. This can explain the performance of our model using only time-related features. Specifically, for example, in Scenario A, the accuracy of model is (**8.31%**, **13.29%**, **34.84%**, **26.83%**, **5.74%**, **5.47%**, **4.51%**, **3.9%**, **5.5%**, **3.64%**) better than DT, KNN, SVM, MLP, Extra-Tree, RF, CatBoost, LightGBM, AdaBoost, XGBoost, respectively. On the other hand, in scenario D, the accuracy of our model is (**15.19%**, **21.31%**, **49.83%**, **39.39%**, **9.9%**, **9.83%**, **8.82%**, **7.12%**, **10.38%**, **7.32%**) better than DT, KNN, SVM, MLP, Extra-Tree, RF, CatBoost, LightGBM, AdaBoost, and XGBoost models, respectively. These results illustrate the high performance of our model with both binary and especially for multi-classification scenarios.

V. Conclusion and Future Research

In this paper, ensemble learning based on deep learning and four DT-based models has been proposed in order to provide efficient solutions for network management. The proposed ensemble consists of pre-processing tasks and classification tasks. 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 deep learning is applied to improve overall classification accuracy. By using deep learning as a meta-classifier, the non-linear relationships among the base classifiers are learned automatically, thus enabling the ensemble method to achieve a better classification. Using two datasets, 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).

For future work, we will investigate further the performance of our model in another context (e.g., intrusion detection) and compare its performance with other ensemble models such as majority voting and linear blending.

References

[1] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, "Towards the deployment of machine learning solutions in network traffic classification: A systematic survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1988–2014, 2018.

[2] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and systems magazine*, vol. 6, no. 3, pp. 21–45, 2006.

[3] Y. Li and Y. Pan, "A novel ensemble deep learning model for stock prediction based on stock prices and news," *arXiv preprint arXiv:2007.12620*, 2020.

[4] P. Amaral, J. Dinis, P. Pinto, L. Bernardo, J. Tavares, and H. S. Mamede, "Machine learning in software defined networks: Data collection and traffic classification," in *Proceedings of the 24th International Conference on Network Protocols (ICNP)*, Singapore, November 2016, pp. 1–5.

[5] C.-H. Chen, K. Tanaka, M. Kotera, and K. Funatsu, "Comparison and improvement of the predictability and interpretability with ensemble learning models in qspr applications," *Journal of Cheminformatics*, vol. 12, pp. 1–16, 2020.

[6] I. L. Cherif and A. Kortebi, "On using extreme gradient boosting (xgboost) machine learning algorithm for home network traffic classification," in *2019 Wireless Days (WD)*. IEEE, 2019, pp. 1–6.

[7] L. Peng, B. Yang, Y. Chen, and Z. Chen, "Effectiveness of statistical features for early stage internet traffic identification," *International Journal of Parallel Programming*, vol. 44, no. 1, pp. 181–197, 2016.

[8] M. C. Belavagi and B. Muniyal, "Performance evaluation of supervised machine learning algorithms for intrusion detection," *Procedia Computer Science*, vol. 89, pp. 117–123, 2016.

[9] Z. A. Qazi, J. Lee, T. Jin, G. Bellala, M. Arndt, and G. Noubir, "Application-awareness in sdn," in *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, 2013, pp. 487–488.

[10] O. Aouedi, K. Piamrat, and D. Bagadthey, "A semi-supervised stacked autoencoder approach for network traffic classification," in *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. IEEE, 2020, pp. 1–6.

[11] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. 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)*, 2016, pp. 407–414.

[12] R. Alshammari and A. N. Zincir-Heywood, "Identification of voip encrypted traffic using a machine learning approach," *Journal of King Saud University-Computer and Information Sciences*, vol. 27, no. 1, pp. 77–92, 2015.

[13] P. Wang, F. Ye, X. Chen, and Y. Qian, "Datanet: Deep learning based encrypted network traffic classification in SDN home gateway," *IEEE Access*, vol. 6, pp. 55 380–55 391, 2018.

[14] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[15] Y. Xiao, J. Wu, Z. Lin, and X. Zhao, "A deep learning-based multi-model ensemble method for cancer prediction," *Computer methods and programs in biomedicine*, vol. 153, pp. 1–9, 2018.

[16] O. Aouedi, K. Piamrat, and B. 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.

[17] M. P. Sesmero, A. I. Ledezma, and A. Sanchis, "Generating ensembles of heterogeneous classifiers using stacked generalization," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 1, pp. 21–34, 2015.

[18] J. S. Rojas, Á. R. Gallón, and J. C. 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*, Melbourne, Australia, July 2018, pp. 543–557.

[19] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," *Computer Networks*, p. 107840, 2021.