LIDarknet: Experimenting the Power of Ensemble Learning in the Classification of Network traffic

Nabil MARZOUG
National School of Applied Sciences-Safi
Cadi Ayyad University
Morocco
nabilmarzoug49@gmail.com

Khidhr HALAB
National School of Applied Sciences-Safi
Cadi Ayyad University
Morocco
halabkhidhr@gmail.com

Younes MAMMA
National School of Applied Sciences-Safi
Cadi Ayyad University
Morocco
younesmaama2000@gmail.com

Fadoua KHENNOU
Perception, Robotics, and Intelligent Machines
Computer Science departement
Université de Moncton, Moncton, Canada
fadoua.khennou@umoncton.ca

Othmane EL MESLOUHI
National School of Applied Sciences-Safi
Cadi Ayyad University
Morocco
o.elmeslouhi@uca.ma

***Abstract*—The Darknet is an encrypted corner of the internet, intended for users who wish to remain anonymous and mask their identity. Because of its anonymous qualities, the Darknet has become a go-to platform for illicit activities such as drug trafficking, terrorism, and dark marketplaces. Therefore, it is important to recognize Darknet traffic in order to monitor and detect malicious online activities [1]. This paper investigates the potential effectiveness of machine learning algorithms in identifying attacks using the CICdarknet2020 dataset. The dataset includes two distinct classification targets: traffic label and application label. The objective of our research is to identify optimal classifiers for traffic and application classification by employing ensemble learning methods, aiming to achieve the highest possible results. Through our experimentation, we have found that the best-performing models surpassing all other state-of-the-art machine learning models are LightGBM, achieving a 93.41% f1-score in the Application classification, and Random Forest, achieving a 99.8% f1-score in the traffic classification.**

***Index Terms*—Darknet, Traffic analysis, Ensemble learning methods, Lightgbm, Random forest, ANOVA**

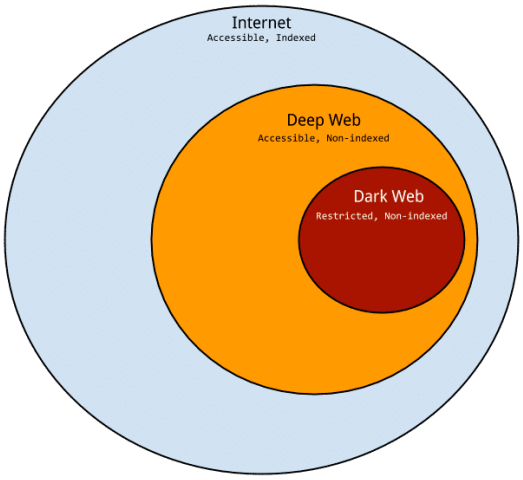

Fig. 1. Layers of the Internet

I. Introduction

Problem statement

In an increasingly interconnected world heavily reliant on the internet, online safety must not be disregarded. While the majority of individuals utilize the internet for good intentions, it can also, unfortunately, serve as a platform for illegal activities, particularly within the darknet. This hidden part of the internet is frequently exploited by individuals or groups seeking to conceal their online activities and identities, thus becoming a breeding ground for various criminal endeavors, including drug trafficking, weapon sales [2], child pornography [3], human trafficking [4], and other egregious violations, we, as data scientists , feel deeply committed to making the internet a safer place using powerful tools. Unlike traditional programming that heavily relies on rule-based algorithms that requires a lot of time and code and fails to cover all possible scenarios and take all edge cases in consideration, machine learning algorithms can learn from historical data and identify patterns that may indicate malicious activities or anomalies that deviate from normal behavior without explicitly being programmed, one more important thing is that machine learning models can easily adapt to new scenarios, In cybersecurity, threats are constantly evolving, with new attack techniques and variations emerging regularly, Machine learning models, unlike traditional programming, don't rely on predefined logic and the creation of rule-based algorithms from scratch whenever a new intrusion technique emerges. Instead, machine learning models have the ability to adjust their parameters and learn to detect patterns in new scenarios, while still being capable of detecting older cyber attacks making them more robust and flexible.

Previous works

The journey to extract as much insights as possible from the CIC-darknet2020 dataset includes many stations each of them focuses on a specific task, here is a table that describe some of them:

Work	Considered Task	Techniques	Obtained Results
Lashkari, et al.	Binary: benign or darknet Multiclass: 8 application types	CNN	Binary: 94% accuracy Multiclass: 86% accuracy
Sarwar, et al.	Multiclass traffic nature: 4 classes Multiclass application type: 8 classes	SMOTE PCA, DT, XGB CNN-LSTM, CNN-GRU	Traffic: 96% F1-score Application: 89% F1-score
Iliadis, et al.	Binary: benign or darknet Multiclass: 4 classes	KNN, MLP, RF, GB	Binary: 98.7% F1-score Multiclass: 89.61% F1-score
Demertzis, et al.	Multiclass: 11 application types	WANN	92.68% accuracy
Futai Zou, et al.	3 hierarchical filtering classifiers: 1st classifier: benign or darknet 2nd classifier: darknet traffic source 3rd classifier: darknet 8 classes application type	LR, RF, MLP, GBDT, LightGBM, XGB, LSTM	Filter 1: 99.42% accuracy Filter 2: 96.85% accuracy Filter 3: 92.46% accuracy

TABLE I
PREVIOUS WORKS

These works have made notable contributions to the field, however, it is essential to critically evaluate certain aspects of their methodologies. One common concern is the lack of focus on feature selection techniques that plays a crucial role in improving the performance and interpretability of machine learning models. Some works haven't pay much attention to this aspect potentially resulting in a suboptimal results, moreover, a significant limitation observed in several studies is the reliance on accuracy as their primary evaluation metric. While accuracy is commonly used, it can be misleading in imbalanced datasets. Imbalanced datasets pose challenges in correctly quantifying model performance on minority classes, which are often the ones of utmost interest. Thus, the use of alternative metrics, such as F1-score would provide a more comprehensive evaluation and better reflect the model's performance on the minority classes.

Proposed solution

Accurately identifying the presence of darknet traffic allows security professionals to focus their efforts on investigating and mitigating potential threats originating from these specific web traffics. Furthermore, by classifying the specific application types utilized within the traffic, such as P2P, audio streaming, chat, file transfer, VOIP, and others, it becomes possible to gain deeper insights about the purposes and goals of the network users [1], [7]–[9]. Some application types [7] may be more exposed to cyber crimes. For instance, P2P networks can facilitate the distribution of pirated content, while video-streaming platforms might be exploited for illegal activities such as child exploitation or Graphic violence content or terrorism-related communication.

By combining the detection of traffic nature and application types, security systems can enhance their capabilities in identifying criminal web activities. This information serves as a valuable resource for cybersecurity professionals, enabling them to prioritize investigations, optimize resources allocation so that they focus more on those suspicious activities, and take measures to protect individuals and organizations from potential threats. Therefore, the approach we adopt is to create two filters, the first one catches web traffics coming from the darknet, and the second one identifies which activity is exploited within the traffics coming from darknet.

Our contribution

For the purpose above, we need two classifier models, the first one will be trained on distinguishing between lightnet and darknet traffics, while the second one will be trained on identifying the application type exploited, and to accomplish that, we need a large dataset that include as much as possible of the specifics (the protocol used to transmit the traffic, destination IP address, source IP address, the length of the traffic in bytes…) of a huge amount of web traffics.

The dataset

Luckily for us, such a dataset exists under the name of CIC-Darknet2020 [7] which is a collection of traffics from diverse origins and different applications types. The CIC-Darknet2020 dataset consists of 141530 sample and 85 features, 6 of them are non-numeric and has two target labels, traffic nature and application type.

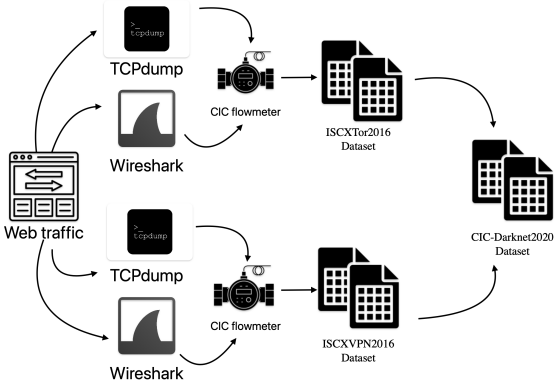

Fig. 2. traffic label distribution

1) **traffic nature label** this label categorize web traffics by their origines Tor,non-Tor,VPN and non-VPN:

Traffic Label	Number of Samples
Non-Tor	69065
NonVPN	23861
VPN	22919
Tor	1179

TABLE II
NUMBER OF SAMPLES PER TRAFFIC TYPE.

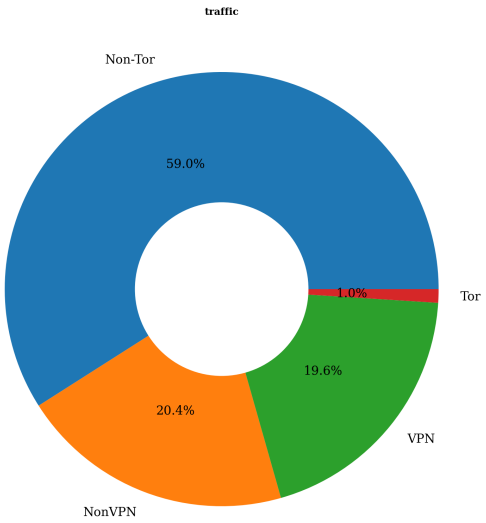

Fig. 3. traffic label distribution

2) **Application type label** This label classify instances by the application used, it includes 8 possible applications Browsing, P2P, Audio-streaming, Chat, File-Transfer, Video-Streaming, Email and VOIP:

Application	Number of Samples
Browsing	32714
P2P	24260
Audio-Streaming	17947
Chat	11473
File-Transfer	11173
Video-Streaming	9748
Email	6143
VOIP	3566

TABLE III
NUMBER OF SAMPLES PER APPLICATION.

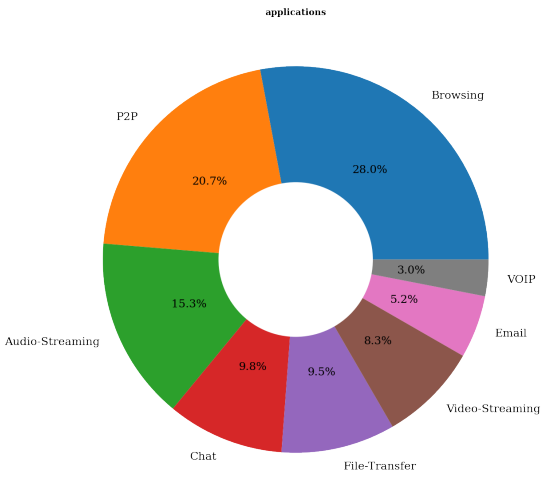

Fig. 4. Application type distribution

we followed two different phases:

(a) **Data-centric phase:** In this phase we focus on transforming our dataset from its raw state into a more consumable and cleaned data that contains the most relevant features with balanced classes.

(b) **Model-centric phase:** In this phase we try to find the best model and the best combination of hyper-parameters that boost the classification performance to the max.

The main contributions of this Thesis can be summarized as follows:

1) 99.8% F1-score by a random forest model for identifying traffic nature.
2) 93.41% F1-score by a lightGBM model for classifying traffics by application type.

Thesis outline

Till the accomplishment of the thesis.

II. METHODOLOGY

In this section, we explain the followed methodology. The central objective of this research is to enhance the current state-of-the-art classification methods for web traffic by exploring the power of ensemble learning methods. Our base models are random forest for traffic nature classification and lightgbm for application type classification.

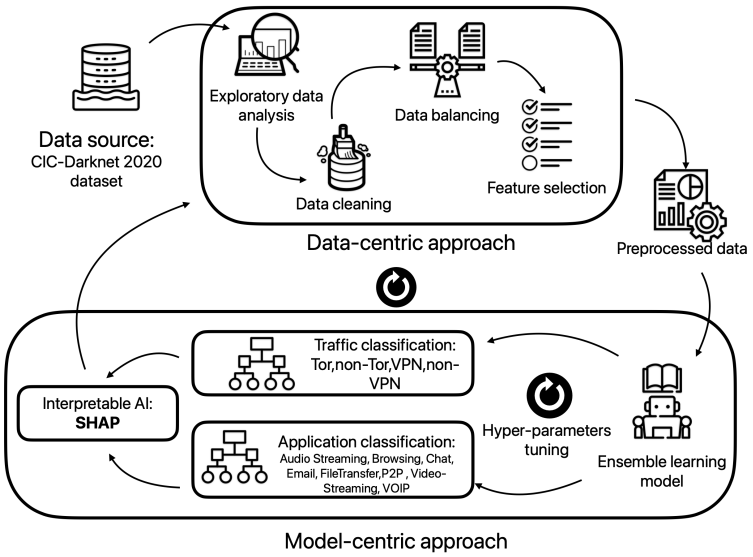

Fig. 5. Methodology

Figure 5 demonstrates the methodology of our research.

A. *Data-centric phase*

1) Exploratory data analysis: After providing data, we try to gain an overview of it and understand the different issues that comes with it to overcome those problems as the quality of the provided data plays a crucial role in enhancing the

performance of our models, here are the issues that the **CIC-Darknet 2020** suffers from:

- Categorical features (IP addresses)
- Noisy data
- Constant features
- Mutually highly correlated features
- Severe classes imbalance

the next steps will ensure to solve these issues.

2) Data cleaning: The CIC-Darknet dataset contain some lectures with missing values, we could use some statistical imputation techniques such as mean/median/mode filling or interpolation techniques or even use a model based filling such as regression model (linear regression , random forest regression ...), However considering the relatitively small number of these cases we decided to drop them as it's not worth it.

3) Data transformation: Alongside transforming categorical features to numeric data using ordinal encoding, the features that has IP adresses need to be transformed,to achieve that, we first convert them to a 32-bit packed binary representation, every 8 bits represent a number of the IP address as they usually range from 0 to 255, after that, we convert the packed binary IP address into its numeric form, by following this process consistently for all IP addresses, we ensure that the pattern of the addresses is preserved.

4) Data splitting: To ensure that all the sets (such as training, validation, and testing sets) preserve a representation of the original distribution of the classes, we opt to use a stratified splitting which will reduce the bias from a specific split while giving the model an idea about how the web traffics are distributed in real life.

5) Data balancing: To address the problem of the severe classes imbalance, and to diversify the synthetic generated samples, we used 3 different oversampling techniques: SMOTE, ADASYN and Borderline-SMOTE, however the balancing didn't introduce any significant improvement in the obtained results due to the other precautions we took such as stratified splitting and the usage of F1-score.

6) Feature selection: The CIC-Darknet2020 dataset consists of 85 features. To select the best features, we first eliminate 'Flow ID' and 'Timestamp' columns, then we remove constant features as they bring no relevant information, the analysis shows there are 15 invariant feature so we are left with 68, as mentioned earlier many of them are mutually highly correlated, in order to keep the best ones we first define all features that their correlation is above a threshold of 0.8 then to choose which column to drop , we use Random Forest feature importance, the one assigned to the lowest importance by the classifier will be dropped. Moving forward we costumize feature selection for the treated task(traffic nature or application type), for that we applied different methods including filter methods such as ANOVA, information gain and CHi-square test, wrapper methods such as recursive feature elimination and embedded methods such as random forest feature importance.

B. Model-centric phase

This section present the models that forms the core of our research methodology; Random forest for traffic nature classification and lightGBM for application type classification.

1) Evaluationn metrics: To evaluate our models, we need a metric that takes in consideration the classes imbalances, as mentioned above, accuracy can be very misleading in such a case, let's consider this confusion matrix:

Actual / Predicted	Positive	Negative
Positive	14 (TP)	130 (FN)
Negative	6 (FP)	850 (TN)

TABLE IV
CONFUSION MATRIX

note that the formula of the accuracy is:

$$\text{Accuracy} = \frac{\text{True Predictions}}{\text{data size}}$$

In our case, that will be:

$$\frac{850 + 14}{14 + 6 + 850 + 130} \approx 0.86$$

This evaluation means that our model is right 86% of the times, which can give us an impression that our model is effective and doing well while we can see that in the positive class , from 144 predictions , only 14 were right, the model has a poor performance on the positive class but since it represent the minority, it has a slight effect on the accuracy.
F1-score is a widely used evaluation metric for assessing the performance of classification models, particularly in scenarios involving unbalanced datasets [10]. which is our case, making accuracy an inadequate choice for evaluation. Consequently, the F1 score emerges as a more suitable metric due to its ability to consider both precision and recall, thereby computing their harmonic mean. With a range from 0 to 1, a higher F1 score signifies superior model performance in effectively balancing precision and recall to achieve accurate classification results.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

where:

- **Precision**:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

- **Recall**:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

If we evaluate the F1 score for the previous case, we will obtain: F1-score ≈ 0.17 which is a more accurate result.

*2) **Random forest**:* Random Forest is an ensemble method that combines the results of multiple decision trees, denoted as $T_1, T_2, \ldots, T_n$, where n is the number of trees [11]. Each decision tree T_i is constructed by recursively partitioning the training data into subsets based on different features. At each split, a feature is selected based on a randomly chosen subset of features. This randomness ensures diversity among the trees.

To make predictions using the Random Forest, the algorithm employs a voting mechanism for classification tasks and averaging for regression tasks. For classification, the predicted class $\hat{y}$ is determined by majority voting among the trees:

$$\hat{y} = \arg\max_{c} \sum_{i=1}^{n} \mathbb{I}(T_i(\mathbf{x}) = c) \tag{1}$$

where $\mathbf{x}$ is the input instance, c is a class label, and $\mathbb{I}$ is the indicator function.

Random Forest also provides a measure of feature importance. The importance score I_f of a feature f is calculated as the average decrease in impurity (e.g., Gini impurity or entropy) caused by that feature across all the trees:

$$I_f = \frac{1}{n} \sum_{i=1}^{n} \text{impurity}(T_i) - \text{impurity}(T_i|f) \tag{2}$$

where $\text{impurity}(T_i)$ is the impurity of tree T_i, and $\text{impurity}(T_i|f)$ is the impurity of tree T_i after splitting on feature f.

*3) **Lightgbm**:* LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is designed to be efficient and provides excellent performance on large-scale datasets. LightGBM builds an ensemble of decision trees, where each tree is trained to correct the mistakes of the previous trees. Given a training dataset $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i$ represents the input features and y_i is the corresponding target variable, LightGBM aims to learn a prediction function $F(\mathbf{x})$ that minimizes a differentiable loss function $L(y, F(\mathbf{x}))$. The prediction function $F(\mathbf{x})$ is modeled as the sum of M individual trees:

$$F(\mathbf{x}) = \sum_{m=1}^{M} f_m(\mathbf{x}) \tag{3}$$

where $f_m(\mathbf{x})$ is the prediction of the m-th tree.

To train the individual trees, LightGBM uses a gradient-based optimization approach. It minimizes the loss function by iteratively adding trees to the ensemble. At each iteration, a new tree is constructed to fit the negative gradient of the loss function with respect to the current ensemble predictions:

$$\text{residual}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{\text{current}}(\mathbf{x}_i)} \tag{4}$$

where $F_{\text{current}}(\mathbf{x}_i)$ represents the current ensemble prediction for the i-th instance.

LightGBM employs a technique called "leaf-wise" tree growth, which aims to grow the tree by splitting the leaf with the highest gain. The gain is calculated as the improvement in the loss function after the split, taking into account the samples assigned to each leaf. This approach leads to a more efficient and effective tree construction process.

Additionally, LightGBM includes regularization techniques such as shrinkage (learning rate) and feature sub-sampling to prevent overfitting and enhance generalization performance.

4) Interpreting the results: In this section we try to unravel the 'black box' state of the models and answer the question:how the influences of the features adds up to make a prediction? For that purpose,we will use SHAP which is a model-agnostic technique that can be used for all tasks whether supervised or unsupervised [12].

III. Results and Discussion

This section presents the major findings of this research. First, we bring to light the experiment results of the two classifications, then we interprate them using SHAP.

A. *Traffic nature classification*

With only the steps of the data-centric phase that include data cleaning, balancing and selecting the best features for the traffic nature classification task, we manage to obtain 99.8%F1-score which emphasis the importance of this phase.

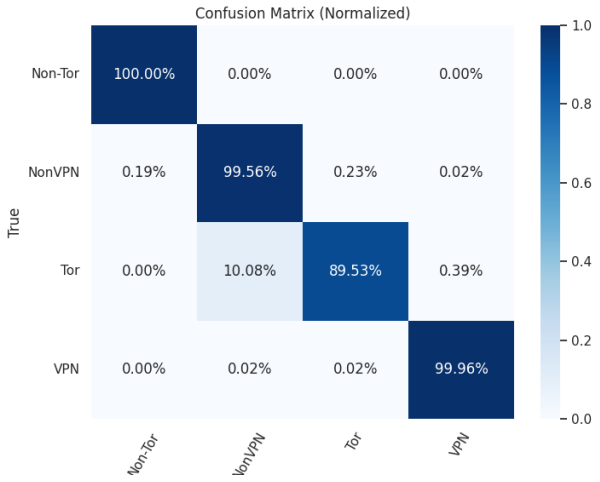

Fig. 6. Confusion Matrix for traffic nature

1) Stratified cross validation : The accompanying standard deviation provides insights into the consistency of the model's performance across different fold sizes. Notably, as the number of folds increased, the F1-score slightly improves, reaching its peak at 100-fold cross-validation with standard deviation remaining low . Table V shows the results.

Number of Folds	F1-score	Standard Deviation
5	99.85%	± 0.02%
10	99.86%	± 0.03%
20	99.86%	± 0.05%
50	99.87%	± 0.06%
100	99.87%	± 0.07%

TABLE V
F1-SCORES AND STANDARD DEVIATIONS FOR DIFFERENT NUMBERS OF FOLDS FOR TRAFFIC NATURE TASK

2) Exploring F1-Scores Across Different Classes: Figure 11 shows the F1-score obtained by each class, as shown below the model lowest performance is with instances originating from the Tor class.

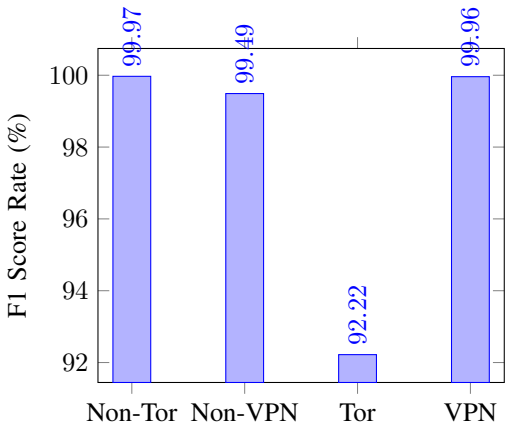

Fig. 7. F1 Score by class using Random Forest

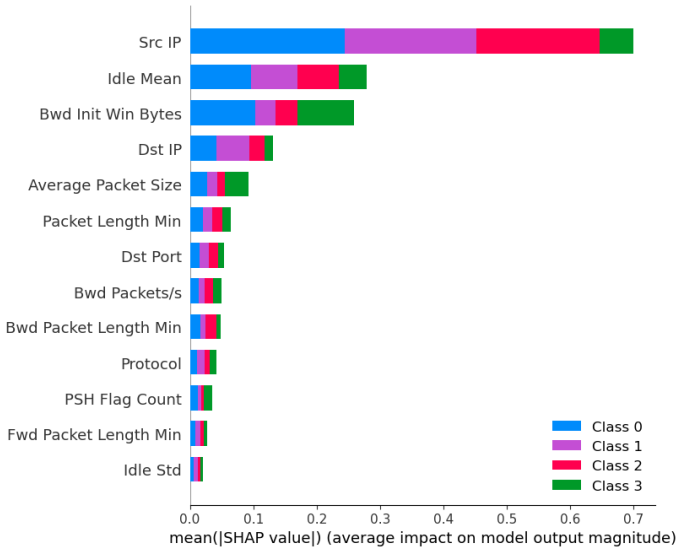

Fig. 8. most influencing features for traffic nature task

In Figure 8, we can see that the most influencing feature on the classification of web traffics by nature is *Src IP*.

B. *Application type classification*

In this section, we will focus on the most challenging part of the work which is identifying which application (Audio-Streaming, Browsing, Chat, Email, File-Transfer, P2P, Video-Streaming, VOIP) is used within each web traffic, for that purpose, we will customise the data for our goal by throwing the data through the pipeline defined on the data-centric stage which involve balancing the data for the application type label then selecting the best set of features that gives the best results using our base model, it is worth mentioning that in our case, the traffic nature column was considered as a feature as well.

On the initial trials of training and tuning, lightGBM seems to achieve better results in comparison to other models, which is why we decide to continue with it.

The next step is to find the best combination of hyper-parameters that boost the results to the max, for that we use different hyper-parameters tuning techniques including random search, grid search, bayesian optimization, genetic algorithms, but after many essays, we found out that optuna provides state of the art optimization algorithms like Tree-structured Parzen Estimator (TPE) that are efficiently implemented, it comes with many benefits, one of the most interesting features of Optuna is its ability to store trials on a database, this means that you can pause a search trial and resume it at a later time or even on a different machine, this flexibility is especially useful in scenarios where you have limited computational resources or need to interrupt the optimization process, it also allows to resume a previous study with a new optimization algorithm which can be so important, for our case, we first used TPE to find an initial well doing model, after that we switched to using the NSGAII (Non-dominated Sorting Genetic Algorithm II), that way, instead of optimizing a population of randomly initialized models, it will work on a population of "good" models which will optimize the time to find the best performing model as well as increasing the chances to find the most efficient one, the metric that we optimize is the F1-score.

After few hundreds of exploration trials of the search spaces of the hyper-parameters, the results seem to converge to 93.4% F1-score:

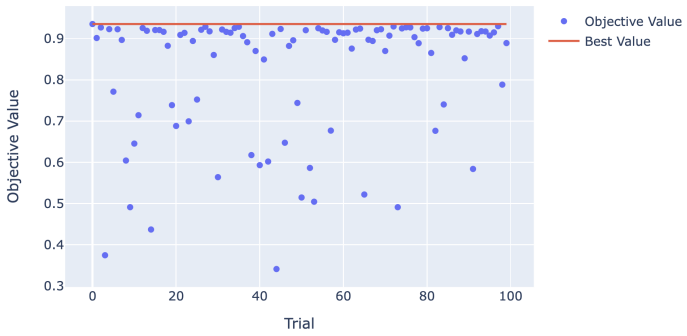

Fig. 9. optimization history plot

C. *Confusion Matrix:*

Now to make sure we don't have an over-fitting, we feed the tuned model with the unseen test set data, we achieve a result of 93.43% F1-score. The next figure shows the confusion matrix of the test set.

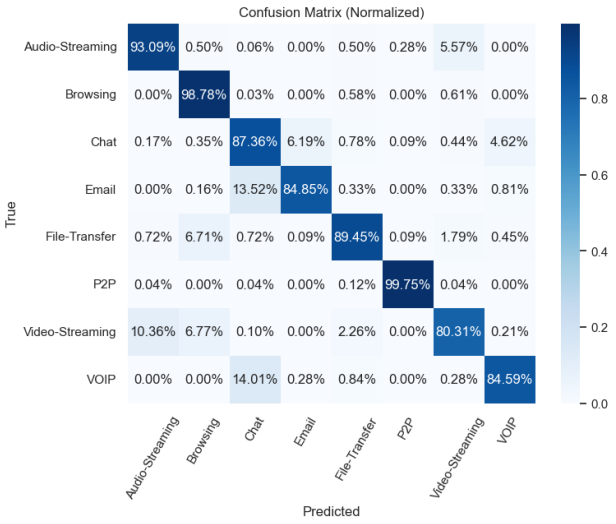

Fig. 10. Confusion matrix of the test set

It seems that the most confusing instances for the model are coming from the email, VOIP, chat and Video- Streaming classes, 13.52% of the email samples and 14% of the VOIP data points were classified as chat instances while 10.36% of the Video-Streaming instances were classified as Audio-Streaming samples.

1) Exploring F1-Scores Across Different Classes: Figure 11 shows the F1-score by class. The model achieved a high F1-score of 99.73% for the P2P class, indicating accurate predictions for the majority of P2P instances. However, the Video-Streaming class had the lowest F1-score of 82.12%, suggesting some difficulty in distinguishing it from the Audio-Streaming class. This confusion between the two classes is understandable due to their similarity as streaming applications.

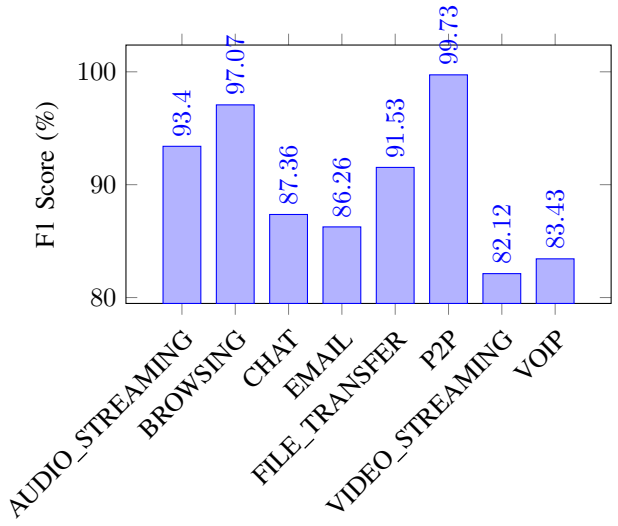

Fig. 11. F1 Score by Class for the LightGBM model

2) Stratified Cross Validation: To ensure that we are not facing a bias introduced by a single train-validation split, we will check the stratified cross validation F1-score on different numbers of folds and the standard deviation of results on each fold to ensure that the results are similar. The lower the standard deviation, the better. Table VI shows the results

Number of Folds	F1-score	Standard Deviation
5	93.07%	± 0.14%
10	93.24%	± 0.17%
20	93.38%	± 0.19%
50	93.39%	± 0.49%
100	93.41%	± 0.72%

TABLE VI
F1-SCORES AND STANDARD DEVIATIONS FOR DIFFERENT NUMBERS OF FOLDS FOR APPLICATION TYPE TASK

Figure 12 shows the overall most influencing features on the model decision making by calculating the mean of absolute values of SHAP values of all the features across all the classes, Idle Max and traffic nature seems to have the biggest impact on the model predictions followed by Dst Port and Src IP.

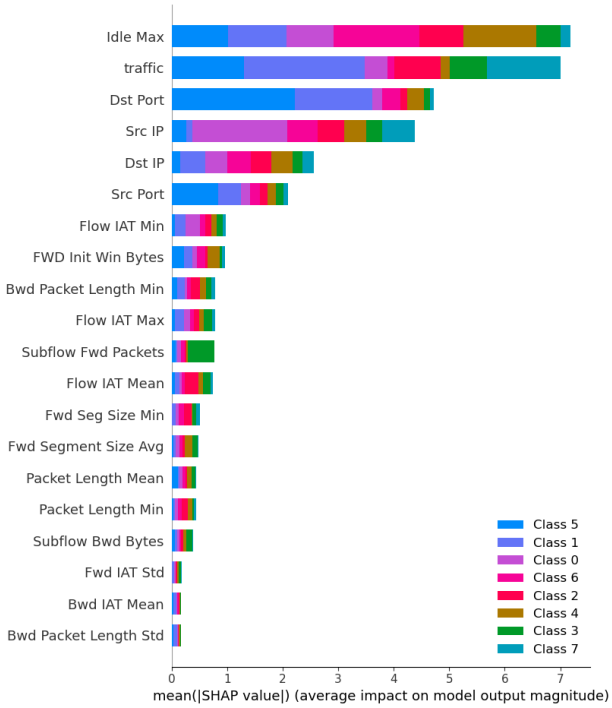

Fig. 12. most influencing features for application task

IV. Conclusion and Perspective

Work	Considered Task	Techniques	Obtained Results
Stamp, et al. (previous state of the art)	Multiclass traffic nature: 4 classes Multiclass application type: 8 classes	AC-GAN, CNN, RF, SVM	Traffic: 99.8% F1-score Application: 92.2% F1-score
Our work	Multiclass traffic nature: 4 classes Multiclass application type: 8 classes	RF, lightGBM, XGBoost	Traffic: 99.8% F1-score Application: 93.4% F1-score

TABLE VII
Previous Works

We presented ensemble learning models that can classify network traffic. Random forest for traffic nature classification and Lightgbm for Application type classification. In the first part, the focus was on the data centric phase in which we applied the necessary operations to have cleaned and good data for our models including data cleaning, data splitting, label encoding,feature selection...

In the next stage, we search for the best ensemble learning models for both classifications. After finding the right models we apply hyper-parameter tuning techniques to boost the performance of our models for the application type. Then we Interpret the results using SHAP.

Following conclusions can be drawn from research work in this paper:

- Data-centric phase is a crucial part in the training of every model.
- Ensemble learning methods are powerful in the classification tasks and can make more accurate predictions.
- Random forest was able to reach 99.8% F1-score in the traffic nature classification
- Lightgbm was able to reach 93.4% F1-score in the application type classification outperforming the state-of-the-art studies on CIC-Darknet2020 [13].

In future work, We may consider removing one or two of the misleading features, but we should keep in mind that those features are not influencing the predictions independently but they are working together to make the prediction so removing a misleading feature may have a "bad" impact on other "good" features. To find the best set of features to remove, we must carefully consider the interactions between the entire features and study how a feature impact an other one especially in a decision tree, which is the base model of our ensemble models, where the next feature to consider depends on the value of the current feature. To summarize, a feature may have a direct "bad" impact on the predictions but it can have an overall "good" impact indirectly on the model and the other features, we still can measure the interaction values by comparing the absence and presence of a feature in all possible sets of features, if the interaction is close to nothing and the feature is positively impacting the wrong predictions more than the right predictions we may have the right to remove that feature.

References

[1] Mahmoud Alimoradi, Mahdieh Zabihimayvan, Arman Daliri, Ryan Sledzik, and Reza Sadeghi, "Deep Neural Classification of Darknet Traffic," *Artificial Intelligence Research and Development*, pp. 105–114, 2022, IOS Press.

[2] Gwern Branwen, *DNM Archive*, Accessed on April 28, 2023.

[3] Afsana Anjum, Dr Kaur, Sunanda Kondapalli, Mohammed Ashafaq Hussain, Ahmed Unissa Begum, Samar Mansoor Hassen, Adam Boush, Mawahib Sharafeldin, Atheer Omar S Benjeed, Dr Osman Abdalraheem, and others, "A Mysterious and Darkside of The Darknet: A Qualitative Study," *Webology*, vol. 18, no. 4, 2021.

[4] Joan Reid and Bryanna Fox, "Human trafficking and the darknet: Technology, innovation, and evolving criminal justice strategies," *Science Informed Policing*, pp. 77–96, 2020, Springer.

[5] Encyclopedia, *Machine Learning in Cybersecurity*, *https://encyclopedia.pub/entry/25675*, Accessed on April 28, 2023.

[6] CrowdStrike, *Machine Learning in Cybersecurity*, *https://www.crowdstrike.com/cybersecurity-101/machine-learning-cybersecurity/*, Accessed on April 28, 2023.

[7] University of New Brunswick, *CIC Darknet2020 Dataset*, *https://www.unb.ca/cic/datasets/darknet2020.html*, 2020, Accessed on April 28, 2023.

[8] Arash Habibi Lashkari, Gurdip Kaur, Abir Rahali, *Didarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning*, in *2020 the 10th International Conference on Communication and Network Security*, pp. 1–13, 2020.

[9] Lazaros Alexios Iliadis, Theodoros Kaifas, *Darknet traffic classification using machine learning techniques*, in *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1–4, 2021, IEEE.

[10] Stephen Allwright, *What is a Good F1 Score*, *https://stephenallwright.com/good-f1-score/*, Accessed on April 28, 2023.

[11] IBM, *Random Forest - Overview*, *https://www.ibm.com/topics/random-forest* Accessed on April 28, 2023.

[12] Towards Data Science, *Introduction to SHAP Values and Their Application in Machine Learning*, Accessed on April 28, 2023.

[13] Rust-Nguyen, N., Sharma, S., & Stamp, M. *Darknet Traffic Classification and Adversarial Attacks Using Machine Learning*. Computers & Security, 103098, 2023. Elsevier.