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
Université de Moncton, Campus de Moncton
Canada
fadoua.khennou@umoncton.ca

Othmane EL MESLOUHI
SARS Group
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. 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 labels. 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**

I. Introduction

A. 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.

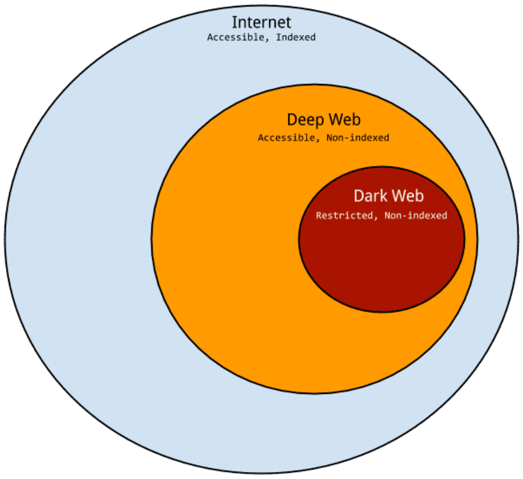


Fig. 1. The Internet Layers

B. Previous work

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 describes some of them:

TABLE I
PREVIOUS WORKS

Work	Considered Task	Techniques	Obtained Results
Lashkari, et al. [8]	Binary: benign or darknet Multiclass: 8 application types	CNN	Binary: 94% accuracy Multiclass: 86% accuracy
Sarwar, et al. [9]	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. [10]	Binary: benign or darknet Multiclass: 4 classes	KNN, MLP, RF, GB	Binary: 98.7% F1-score Multiclass: 89.61% F1-score
Demertizis, et al. [11]	Multiclass: 11 application types	WANN	92.68% accuracy
Futai Zou, et al. [12]	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, Light-GBM, XGB, LSTM	Filter 1: 99.42% accuracy Filter 2: 96.85% accuracy Filter 3: 92.46% accuracy

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. Other works have disregarded this aspect, which could potentially lead to less-than-optimal outcomes. Furthermore, numerous studies have identified a noteworthy constraint: the overemphasis on accuracy as the main assessment measure. Although accuracy is frequently employed, it can be deceptive when dealing with datasets that lack balance. The presence of imbalanced datasets creates difficulties in accurately measuring the performance of models on minority classes, which are frequently the ones of primary concern. Therefore, adopting alternative metrics like the F1-score would offer a more comprehensive assessment and more accurately depict the model's performance regarding the minority classes.

C. 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], [8], [10]. 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.

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.

Fortuitously, we have access to a dataset called CIC-Darknet2020 [7], which serves our purpose. This dataset encompasses a wide range of traffic derived from various sources and spanning different application types. Comprising 141,530 samples and 85 features, including 6 non-numeric attributes, the CIC-Darknet2020 dataset also entails two target labels: traffic nature and application type.

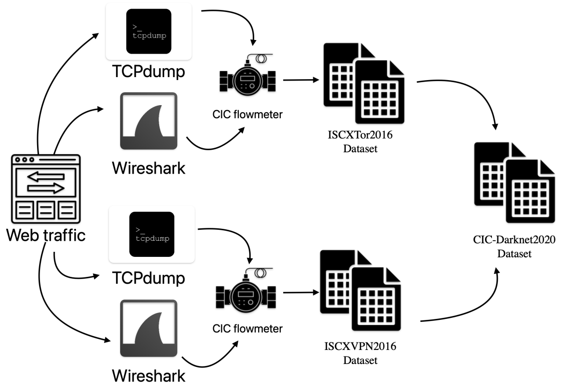


Fig. 2. traffic label distribution

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

TABLE II
NUMBER OF SAMPLES PER TRAFFIC TYPE.

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

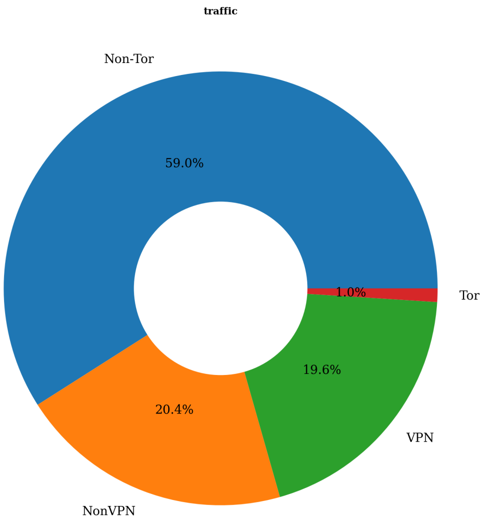


Fig. 3. traffic label distribution

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.

TABLE III
NUMBER OF SAMPLES PER APPLICATION.

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

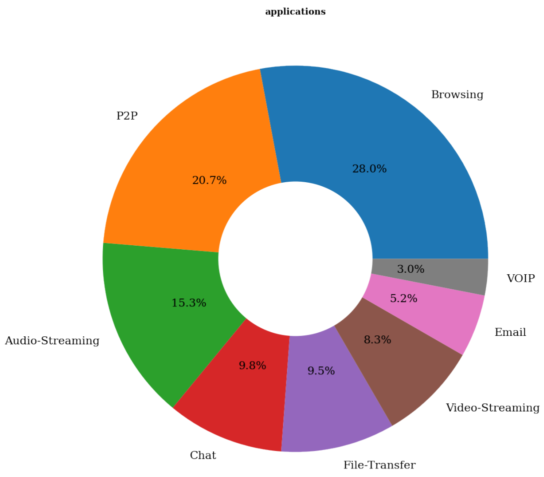


Fig. 4. Application type distribution

2) 93.41% F1-score by a lightGBM model for classifying traffics by application type.

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. Figure 5 demonstrates the methodology of our research.

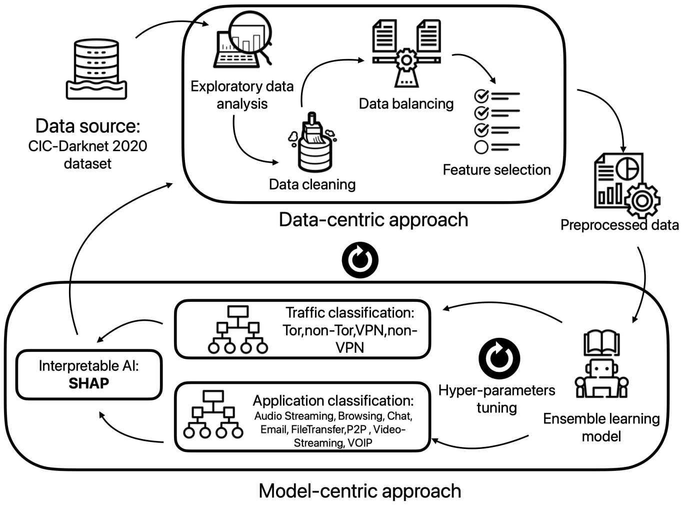


Fig. 5. Methodology

A. *Data-centric phase*

1) Exploratory data analysis: Upon furnishing the data, our initial step involves obtaining a comprehensive overview and comprehending the diverse challenges it entails. This understanding is essential for devising solutions to surmount these obstacles, given that the data's quality significantly influences the effectiveness of our models. The ensuing list outlines the predicaments afflicting the **CIC-Darknet 2020** dataset:

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

The subsequent stages will be dedicated to resolving the following concerns.

2) Data cleaning: Within the CIC-Darknet dataset, there are instances of lectures featuring incomplete values. To address this, various statistical imputation methods could be applied, such as mean, median, or mode filling, along with interpolation techniques. Alternatively, a model-based approach like regression models (such as linear regression or random forest regression) could be employed. However, due to the relatively limited occurrence of such cases, the decision was made to omit them, as the effort involved in addressing them wouldn't yield significant benefits.

3) Data transformation: In addition to converting categorical features into numerical data using ordinal encoding, the features containing IP addresses require a transformation. To achieve this, the initial step involves converting the IP addresses into a compact 32-bit binary representation, with each set of 8 bits representing a segment of the IP address, typically ranging from 0 to 255. Subsequently, the compact binary representation is further converted into its corresponding numerical format. This process is consistently applied to all IP addresses, ensuring the preservation of their inherent patterns.

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 encompasses 85 features. In the process of selecting the most relevant features, we start by excluding the 'Flow ID' and 'Timestamp' columns. Subsequently, constant features, which offer no meaningful information, are removed. Following this analysis, it becomes evident that there are 15 features that remain invariant, resulting in a final count of 68 features.

As mentioned earlier, a significant number of these features display strong mutual correlations. To retain the most informative ones, we initiate by identifying features with correlations exceeding a threshold of 0.8. Subsequently, we leverage Random Forest feature importance to make informed decisions about feature elimination. The feature deemed least important by the classifier is discarded.

Progressing, we tailor the feature selection process to the specific task at hand, whether it's predicting traffic nature or application type. For this purpose, we employ diverse methodologies, including filter methods such as ANOVA, information gain, and CHi-square tests. Additionally, we utilize wrapper methods like recursive feature elimination and embedded techniques 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) Evaluation metrics: In the process of evaluating our models, we require a metric that accommodates the presence of class imbalances. As elucidated earlier, accuracy can be deceptive in such contexts. To delve further, let's examine the following confusion matrix:

TABLE IV
CONFUSION MATRIX

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

We note that the formula of the accuracy is as follows:

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

In our case, the formula becomes:

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

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 [14]. 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 [15]. 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 [16].

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*

By solely implementing the data-centric phase steps, encompassing data cleaning, balancing, and selecting optimal features for the traffic nature classification task, we achieve an impressive F1-score of 99.8%. This outcome underscores the crucial significance of this phase.

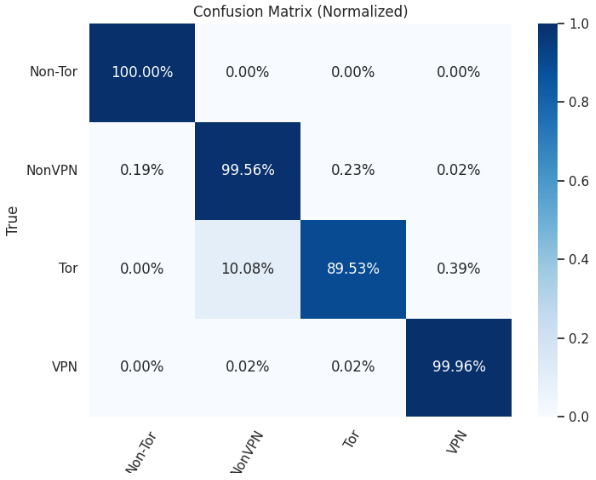


Fig. 6. Confusion Matrix for traffic nature

1) Stratified cross validation : The accompanying standard deviation offers insights into the model's performance consistency across varying fold sizes. Notably, with an increase in the number of folds, there is a slight enhancement in the F1-score. The peak performance is observed during 100-fold cross-validation, while the standard deviation remains consistently low. Table V shows the results.

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

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%

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.

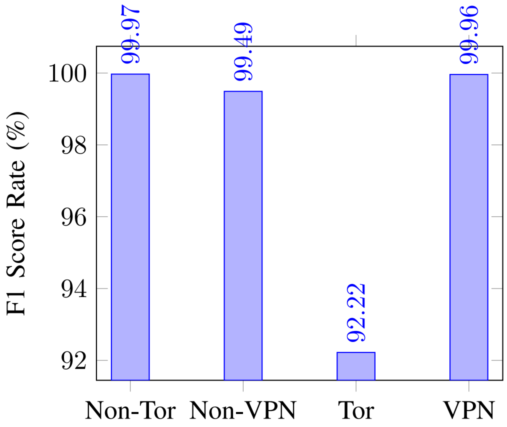


Fig. 7. F1 Score by class using Random Forest

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, our focus shifts to the most challenging aspect of the project: discerning the specific application (Audio-Streaming, Browsing, Chat, Email, File-Transfer, P2P, Video-Streaming, VOIP) utilized within each instance of web traffic. To accomplish this, we tailor the data to our objective by channeling it through the pipeline established during the data-centric stage. This pipeline encompasses data balancing for the application type label and the selection of an optimal feature set through the utilization of our base model. Notably, in our case, the traffic nature column is also treated as a feature.

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.

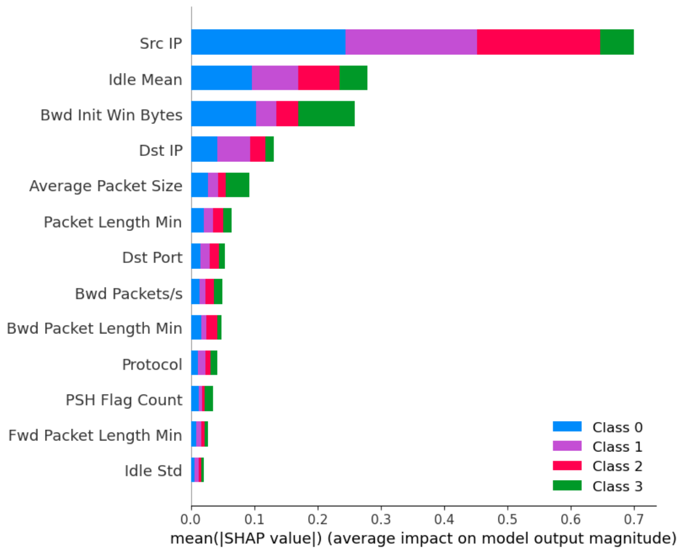


Fig. 8. most influencing features for traffic nature task

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:

C. Confusion Matrix:

To ensure there's no overfitting, we assess the tuned model's performance using unseen test set data, yielding an F1-score of 93.43%. The subsequent figure illustrates the confusion matrix for the test set.

The model appears to encounter the greatest confusion with instances belonging to the email, VOIP, chat, and Video-Streaming classes. Notably, approximately 13.52% of the

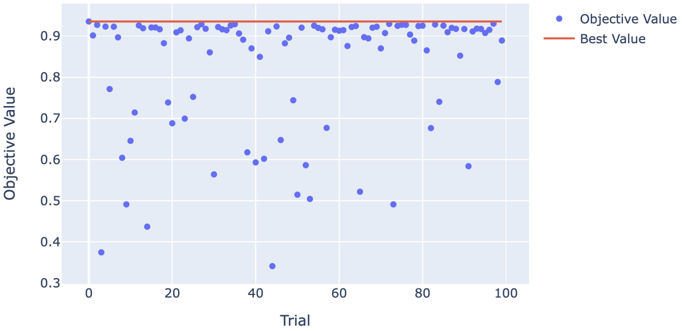
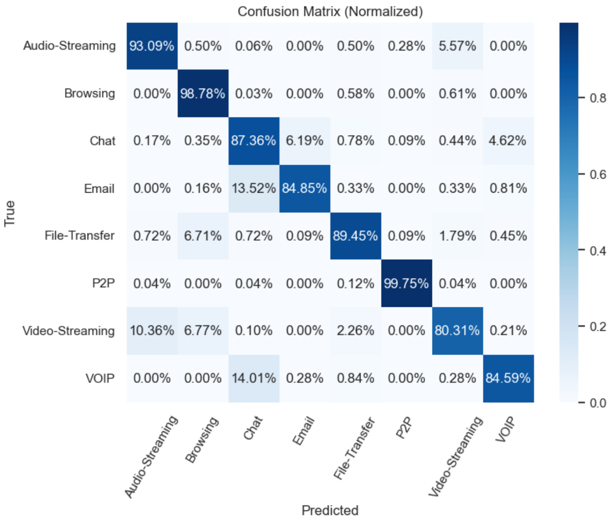


Fig. 9. History plot optimization



	Audio-Streaming	Browsing	Chat	Email	File-Transfer	P2P	Video-Streaming	VOIP
Audio-Streaming	93.09%	0.50%	0.06%	0.00%	0.50%	0.28%	5.57%	0.00%
Browsing	0.00%	98.78%	0.03%	0.00%	0.58%	0.00%	0.61%	0.00%
Chat	0.17%	0.35%	87.36%	6.19%	0.78%	0.09%	0.44%	4.62%
Email	0.00%	0.16%	13.52%	84.85%	0.33%	0.00%	0.33%	0.81%
File-Transfer	0.72%	6.71%	0.72%	0.09%	89.45%	0.09%	1.79%	0.45%
P2P	0.04%	0.00%	0.04%	0.00%	0.12%	99.75%	0.04%	0.00%
Video-Streaming	10.36%	6.77%	0.10%	0.00%	2.26%	0.00%	80.31%	0.21%
VOIP	0.00%	0.00%	14.01%	0.28%	0.84%	0.00%	0.28%	84.59%

Fig. 10. Test Set Confusion Matrix

email samples and 14% of the VOIP data points are misclassified as chat instances. Similarly, around 10.36% of Video-Streaming instances are inaccurately 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.

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.

Figure 12 depicts the collective influence of features on the model's decision-making process. This is determined by calculating the mean of absolute SHAP values for all features across all classes. Notably, ”Idle Max” and ”traffic nature” exhibit the most substantial impact on model predictions, closely followed by ”Dst Port” and ”Src IP.”

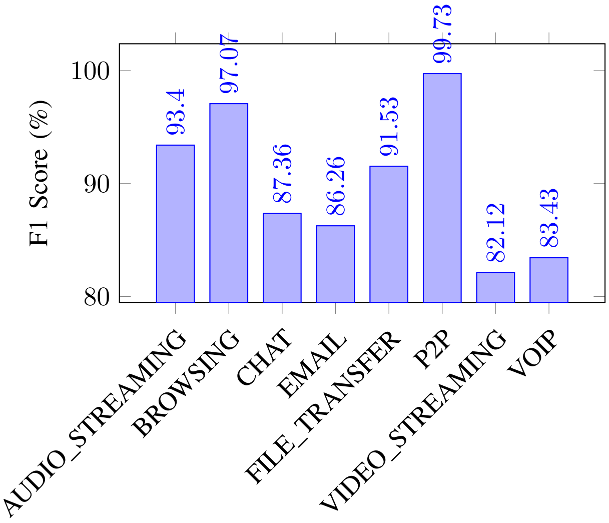


Fig. 11. F1 Score by Class for the LightGBM model

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

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%

IV. Conclusion and Perspective

In this work, ensemble learning models were introduced for the classification of network traffic, with Random Forest utilized for traffic nature classification and LightGBM for application type classification. The initial phase of the approach emphasized data preparation, including data cleaning, splitting, label encoding, and feature selection, to ensure high-quality data for model building. The next step involved selecting optimal ensemble learning models for the classification

TABLE VII
COMPARING THE RESULTS

Work	**Considered Task**	**Techniques**	**Obtained Results**
Stamp, et al. (previous state of the art) [13]	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 proposed approach	Multiclass traffic nature: 4 classes Multiclass application type: 8 classes	RF, lightGBM, XGBoost	Traffic: 99.8% F1-score Application: 93.4% F1-score

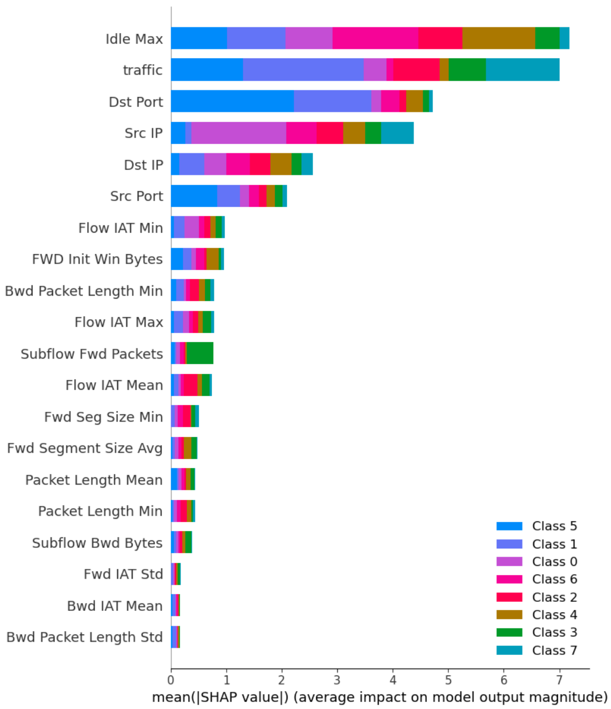


Fig. 12. most influencing features for application task

tasks and further improving their performance through hyperparameter tuning, with a particular focus on application type classification. The results were subsequently interpreted using SHAP (SHapley Additive exPlanations) techniques.

The following insights can be inferred from the research conducted 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, the consideration of removing perplexing features is advisable, bearing in mind that these features collectively shape predictions rather than acting in isolation. While removing a misleading feature may impact other valuable features, creating a trade-off, the decision to remove a feature should depend on its specific influence within the ensemble models. In essence, a feature's apparent negative impact on predictions may contribute positively to overall model performance and feature behavior. To make informed choices about feature exclusion, a comprehensive analysis of feature interactions, particularly within decision trees, is essential. This analysis should involve measuring the effects of including or excluding a feature across various combinations. If minimal interaction is detected, and a feature primarily affects incorrect predictions rather than accurate ones, it may be a candidate for removal, thus optimizing the model's performance while preserving valuable feature contributions.

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, Sarah 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] Muhammad Bilal Sarwar, Muhammad Kashif Hanif, Ramzan Talib, Muhammad Younas, and Muham- mad Umer Sarwar. *Darkdetect: Darknet traffic detection and categorization using modified convolution- long short-term memory*. In IEEE Access, 9:113705–113713, 2021.

[10] 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.

[11] Konstantinos Demertzis, Konstantinos Tsiknas, Dimitrios Takezis, Charalabos Skianis, and Lazaros Iliadis. *Darknet traffic big-data analysis and network management for real-time automating of the malicious intent detection process by a weight agnostic neural networks framework*. https://arxiv.org/abs/2102.08411, 2021.

[12] Yuzong Hu, Futai Zou, Linsen Li, and Ping Yi. *Traffic Classification of User Behaviors in Tor, I2P, ZeroNet, Freenet*. In 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom, pages 418–424, 2020.

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

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

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

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