A Holistic Framework for Interpretable and Efficient Darknet Traffic Classification

Abdulkader Hajjouz* and Elena Avksentieva

Faculty of Software Engineering and Computer Systems, ITMO University,
St Petersburg, Russia
*Corresponding author: hajjouz@itmo.ru

Abstract: The classification of encrypted network traffic (e.g., Tor and VPN) remains a difficult and high-stakes problem: many models target benchmark accuracy yet remain opaque and impractical at line rate. We introduce and validate an end-to-end framework that unifies accuracy, interpretability, and efficiency. The pipeline combines a principled hierarchical feature-distillation strategy with a high-throughput CatBoost classifier and is evaluated on CIC-Darknet2020 using a stratified train/test split and 5-fold cross-validation. The system attains 99.99% accuracy on the held-out test set, with cross-fold stability confirming that the result is not a split artifact. SHAP analyses at global and instance levels make the decision process auditable and reveal a consistent mechanism: temporal and topological metadata—such as inter-arrival timing and port structure—are first-order discriminators, surpassing raw byte counts. A computational benchmark shows production-grade throughput of ≈4.52 million classifications per second, indicating suitability for both core and edge deployments. Overall, the work delivers more than an incremental gain: it provides a validated, reproducible blueprint for intelligent security systems that are accurate, transparent, and efficient enough for real-world use.

Keywords: Network Traffic Classification; Cybersecurity; Machine Learning; Interpretability; Darknet

1. Introduction

The proliferation of end-to-end encrypted network traffic, while a boon for user privacy, creates a substantial challenge for network defence and traffic engineering [1]. In this setting, Privacy-Enhancing Technologies (PETs) such as Tor and VPNs are inherently dual-use: they protect legitimate anonymity, but they also provide robust cover for command-and-control channels, data exfiltration and other malicious activities [2]. Because payload inspection is no longer possible at scale, modern traffic classification systems must infer intent from metadata only (e.g., timing, ports and flow statistics), a task where machine learning (ML) and deep learning (DL) have become the dominant approach [3,4].

To enable comparability, the community has converged on CIC-Darknet2020 as a benchmark for dark web and anonymised traffic classification [5]. A wide range of models have been evaluated on this dataset, from CNN–LSTM architectures [6] to stacking ensembles [7], bagging decision trees [8] and adversarially trained random forests [9]. More recent work explores hybrid deep models and image-based representations for dark web traffic [10,11], Transformer-based architectures for encrypted flows [12] and Tor-centric neural frameworks [13]. These methods achieve impressive accuracy, but they also illustrate three persistent limitations: (i) models are often opaque black boxes, (ii) efficiency and resource usage are rarely reported, and (iii) robustness claims typically rely on single static train/test splits without rigorous stability checks. Similar concerns have been raised more broadly for ML-based intrusion detection systems [14–16].

Consequently, the central research problem is no longer simply to maximise accuracy on CIC-Darknet2020, but to design frameworks that are simultaneously accurate, interpretable and efficient enough for deployment. The lack of such holistic evaluations means that many "state-of-the-art" models remain academic prototypes rather than production-ready security components [16]. There is a need for approaches that (i) operate on a clean, well-conditioned

Received: September 27th, 2025 Accepted: December 31th, 2025
DOI: 10.15676/ijeei.2025.17.4.11

feature space, (ii) expose their decision process for audit, and (iii) demonstrate line-rate performance on realistic hardware.

This paper addresses this gap by proposing and validating a holistic framework for darknet traffic classification that combines principled feature distillation, an efficient CatBoost engine and deep, SHAP-based interpretability. Our main contributions are:

- We design a principled two-stage feature distillation pipeline that reduces the original 82 CIC-Darknet2020 features to a 41-dimensional subspace via zero-variance pruning followed by hierarchical, correlation-driven clustering. This subspace is compact, largely de-correlated and tailored for tree ensembles while remaining easy to reproduce.
- We develop a CatBoost-based classifier that, when trained on this distilled feature set, achieves 99.99% accuracy on a held-out CIC-Darknet2020 test set using a stratified 80/20 split. A stratified 5-fold cross-validation on the training data confirms that this performance is stable and not an artefact of a particular partition.
- We apply SHAP-based interpretability at both global and instance levels to make the decision logic auditable. The analysis reveals a consistent and reproducible hierarchy in which temporal and topological metadata (e.g., inter-arrival times, idle durations and source/destination ports) are first-order discriminators, while raw byte and packet counts play a secondary role.
- We provide a deployment-oriented evaluation that quantifies single-sample latency, end-to-end throughput, memory footprint and scaling behaviour. On commodity hardware, the system sustains approximately 4.52 million classifications per second with sub-microsecond latency and a prediction-time memory footprint of only 0.21 MB, demonstrating suitability for both core and edge deployments.

The rest of this paper is organized as follows. Section 2 reviews related work on encrypted and darknet traffic classification, with an emphasis on feature engineering, model design and dataset limitations. Section 3 describes the proposed methodology in detail. Section 4 presents the experimental results and discussion. Section 5 concludes the paper and outlines directions for future work.

2. Related Work

A. Encrypted traffic analysis and PET detection

A substantial body of work studies encrypted traffic analysis from the perspective of malicious traffic detection, QoS management and application identification. Wang et al. survey ML-based approaches for encrypted malicious traffic, highlighting the challenges of evolving protocols and dataset quality [1]. Razooqi and Pekar review VPN traffic analysis and discuss how encryption and tunnelling complicate application-level identification [2]. More recent surveys examine encrypted traffic classification through the lens of ML/DL pipelines and practical deployment concerns, including feature extraction, model selection and evaluation methodology [3,4]. These efforts provide valuable taxonomies and identify open challenges, but they do not propose concrete, deployable frameworks that simultaneously target accuracy, interpretability and efficiency.

B. Deep learning and darknet traffic classification on CIC-Darknet2020

CIC-Darknet2020 has become the de facto benchmark for darknet traffic classification, encompassing Tor, VPN and non-darknet flows [5]. Early work by Sarwar et al. introduced DarkDetect, a CNN–LSTM architecture that operates on flow statistics and reaches high accuracy but remains a black box without interpretability or efficiency analysis [6]. Abu Al-Haija et al. proposed a bagging decision-tree system for IoT-oriented darknet detection [8], while Almomani developed a modified stacking ensemble targeting improved detection rates on darknet versus benign traffic [7]. Rust-Nguyen and Stamp systematically compared several ML and DL models on CIC-Darknet2020, finding that a carefully tuned Random Forest with

feature engineering achieves a near-perfect F1 score of 99.8% [9]. However, they do not provide SHAP-style explanations or detailed performance benchmarks such as throughput and latency.

Beyond these classical architectures, recent work explores more expressive DL models for dark web and encrypted traffic. Mandela et al. proposed a hybrid CNN–LSTM model for dark web traffic classification, treating packet sequences as spatio-temporal signals and reporting accuracy improvements over simpler baselines [10]. Li et al. introduced a 3D-CNN framework that converts dark web flows into multi-channel images and leverages spatial–temporal feature fusion to increase classification precision [11]. Mahboob et al. designed a neural framework that simultaneously performs encrypted traffic classification and application categorisation on Tor traffic, combining CNN and DNN modules [13]. Transformer-based architectures such as TransECA-Net further generalise these ideas to multi-type encrypted traffic, mixing convolutional front-ends with attention mechanisms to improve representation learning and convergence speed [12]. Complementary work studies encrypted traffic data augmentation with dual-stream time-series Transformers to mitigate class imbalance and increase generalisation [17].

While these methods push accuracy with increasingly complex deep models, they generally share three limitations: (i) they are not inherently interpretable, with only coarse feature importance scores (if any) reported; (ii) efficiency in terms of latency, memory and throughput is rarely quantified; and (iii) model robustness is often assessed on a single train/test split of CIC-Darknet2020, despite its known class imbalance and potential for overfitting [9].

C. Feature engineering, interpretability and deployment concerns

A parallel line of research focuses on feature sets and explainability for intrusion detection and encrypted traffic analysis. Sarhan et al. evaluated standard feature sets such as NetFlow and CICFlowMeter across multiple datasets, showing that feature design strongly affects both generalisability and explainability of ML-based NIDS [16]. Lashkari et al. proposed DIDarknet, which uses deep image representations of darknet flows and demonstrates that well-chosen feature encodings can substantially improve detection quality [18]. Other work leverages SHAP to explain ML decisions in security applications, including LGBM-based attack prediction models [19] and cross-dataset analyses of ML-based NIDS [16].

In the specific context of encrypted traffic classification, Li et al. advocate for "trustworthy deep learning" by incorporating adversarial robustness and uncertainty estimation into DL models [20]. Complementary studies explore pre-training strategies and deep architectures tailored to encrypted flows, arguing that representation learning can mitigate the challenges of protocol evolution and obfuscation [21]. However, these works primarily focus on improving deep models themselves and only partially address operational constraints such as line-rate performance and memory footprint.

D. Positioning of this work

Compared to the above literature, the present work makes three differentiating choices. First, instead of designing a heavier deep architecture, we construct a carefully distilled, 41-dimensional feature subspace using zero-variance pruning and hierarchical correlation-based clustering, which leads to a compact, well-conditioned input for tree ensembles. Second, we build on CatBoost, a gradient-boosted decision-tree method with built-in regularisation, and combine it with SHAP to provide both global and instance-level explanations of the decision process. Third, we treat deployment metrics as first-class citizens, reporting comprehensive latency, throughput, memory and scalability results on CIC-Darknet2020. To the best of our knowledge, this is the first work on this dataset that jointly delivers state-of-the-art accuracy, fine-grained interpretability and production-grade efficiency within a single, reproducible framework.

3. Methodology

The methodological framework presented herein constitutes a state-of-the-art, multi-stage pipeline, architected for the robust and high-fidelity classification of anonymized network traffic. We recognize the complexity and potential for artifacts in benchmark datasets so our approach was designed with signal purity, dimensionality reduction and computational efficiency as the foundation. Each stage was designed to distill the raw data into a feature subspace that is maximally informative for a accurate and generalizable model.

A. Dataset and Formal Problem Definition

Our analysis is based on the CIC-Darknet2020 dataset [18], a well known, recent dataset from the Canadian Institute for Cybersecurity. This dataset provides a rich substrate for a supervised multi-class classification task, formally defined as follows: Given a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$, our objective is to learn a mapping function $f: \mathbb{R}^D \rightarrow Y$ that accurately predicts the class label y_i from the feature vector x_i. Here *N*=158,616, *D* is the number of features, and Y is a 4 class space: {Non-Tor, Non-VPN, VPN, Tor}. The class imbalance in this dataset (Table I) was addressed through stratified sampling throughout our validation pipeline to ensure fair model evaluation.

B. Data Sanitization and Semantic Normalization

The initial stage involved a rigorous data sanitization and semantic normalization process to guarantee the integrity and utility of the feature space.

Numerical Stability: Non-finite values (an artifact of some flow-rate calculations) were algorithmically identified and imputed with the feature-wise mean. This is a standard practice that keeps the numbers stable without distorting the underlying data distribution.

Semantic IP Encoding: To extract the topological information embedded in IP addresses we used a semantic encoding. Each IPv4 address was converted to its unsigned 32-bit integer representation. This is better than one-hot encoding for tree-based models as it projects the addresses into a continuous space where topological locality is preserved and provides a strong signal for the learning algorithm.

C. Principled Feature Subspace Engineering for Signal Purity

To insulate the model from the 'curse of dimensionality' and the confounding effects of multicollinearity, we engineered a sophisticated, two-phase feature selection architecture. This architecture reduces the feature space from 82 features to a compact and informative subspace of 41 features.

Phase I: Axiomatic Pruning of Zero-Variance Features First we applied an information theory axiom: features with zero variance ($\sigma^2 = 0$) are information-theoretically zero and can't contribute to any discriminative task [22]. We identified and pruned 15 such features and did an initial lossless purging of the feature space.

Phase II: Hierarchical Feature Distillation via Non-Parametric Correlation Clustering To deal with the high multicollinearity in the remaining 67 features, we used a hierarchical distillation technique to get our final feature space. The whole process and its results are shown in Figure 1.

Robust Correlation Assessment: First, we visualized the feature dependencies in this 67-feature space using its Spearman's rank correlation matrix (Figure 1, Left Panel). We computed the Spearman's rank correlation matrix, ρ. We chose this non-parametric method over linear methods like Pearson's to capture the monotonic relationships in network flow statistics which are often non-linear. The coefficient ρ is defined for two ranked variables *rgX* and *rgY* of size *n* as: $\rho = \frac{cov(rgX, rgY)}{\sigma_{rgX}\sigma_{rgY}}$ where $cov(rgX, rgY)$ is the covariance of the rank variables, and $\sigma_{rgX}\sigma_{rgY}$ are their standard deviations.

Cohesive Feature Clustering: This correlation matrix is a similarity measure, we transformed it into a distance matrix D where each element $d_{ij} = 1 - |\rho_{ij}|$. This matrix was

then used as input to a Hierarchical Agglomerative Clustering (HAC) algorithm with Ward's linkage criterion [23]. Ward's method is mathematically optimized to produce maximally-cohesive, minimally-variant feature clusters by minimizing the increase in the total within-cluster sum of squares (ESS) at each fusion step. The objective function to be minimized for merging two clusters, A and B, is given by the change in ESS, $\Delta(A,B)$: $\Delta(A,B) = \frac{n_A n_B}{n_A + n_B} \|m_A - m_B\|^2$ where n_j is the number of points in it. The output of this process is a dendrogram which shows the nested grouping of features (as shown in Figure 1, Central Panel).

Canonical Subspace Distillation: The dendrogram was then partitioned by applying a distance threshold and the entire feature set was grouped into conceptually related clusters. From each cluster, a single, canonical feature was selected and the feature space was reduced from 67 to our final 41. To show the success of this process, the correlation matrix for this new 41-feature space is shown (Figure 1, Right Panel). The visible reduction in the high-correlation blocks compared to the original matrix is the empirical proof of our feature distillation. This distillation is crucial; it ensures each underlying concept is represented only once and we get an orthogonalized and maximally informative 41-dimensional feature vector, perfectly conditioned for robust and high-performance modeling.

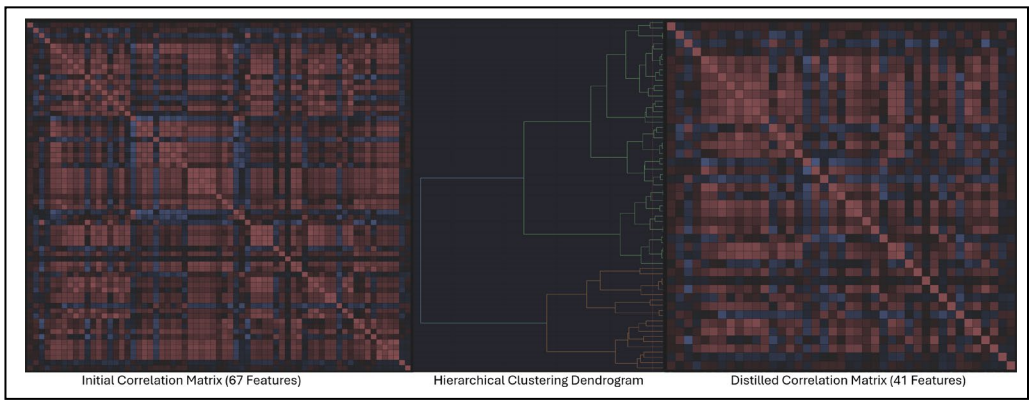

Figure 1. Feature Subspace Engineering process

The figure 1 shows the feature space reduction while preserving signal. (Left Panel) Spearman's rank correlation matrix for the 67 features after pruning zero-variance features. (Central Panel) Dendrogram after applying Ward's linkage clustering. (Right Panel) Correlation matrix for the 41-feature subspace. The reduction in high-correlation blocks compared to the left panel is empirical evidence of our feature distillation.

This distillation process is critical; it ensures that each underlying predictive concept is represented precisely once, yielding an orthogonalized and maximally-informative feature vector of 41 dimensions. These features, which form the basis for our classification model, are detailed and categorized in Table 1.

Table 1. The final 41-dimensional feature subspace

Feature Name
Src Port, Dst Port, Protocol, Timestamp, Flow Duration, Total Fwd Packet, Total Bwd packets, Total Length of Fwd Packet, Total Length of Bwd Packet, Fwd Packet Length Max, Fwd Packet Length Mean, Fwd Packet Length Std, Bwd Packet Length Min, Bwd Packet Length Mean, Bwd Packet Length Std, Flow Bytes/s, Flow Packets/s, Flow IAT Mean, Flow IAT Min, Fwd Iat Total, Fwd IAT Min, Bwd IAT Total, Bwd IAT Min, Fwd PSH Flags, Fwd Header Length, Bwd Packets/s, Packet Length Mean, Packet Length Std, FIN Flag Count, SYN Flag Count, RST Flag Count, ACK Flag Count, Down/Up Ratio, Bwd Packet/Bulk Avg, Subflow Fwd Packets, Subflow Fwd Bytes, Fwd Act Data Pkts, Fwd Seg Size Min, Idle Mean, Source IP (int), and Destination IP (int)

D. CatBoost Classification Engine: A Choice for Robustness and Performance

The heart of our classification pipeline is the CatBoost Classifier, a Gradient Boosting on Decision Trees (GBDT) implementation. We chose this because of its architectural superiority, especially its ordered boosting and symmetric trees which provide built-in overfitting protection [24].

The algorithm builds an ensemble of decision trees $F(\mathbf{x})$ iteratively. At each step m, a new tree hm is built to approximate the negative gradient of a predefined loss function $L(y, \hat{y})$ with respect to the predictions of the previous ensemble, $F_{m-1}(x)$. The model is updated as: $F_m(x) = F_{m-1}(x) + \alpha h_m(x)$ where α is the learning rate. The new tree is trained on pseudo-residuals r_{im} for each observation i: $r_{im} = -\left[\frac{\partial L(y_i, \hat{y})}{\partial \hat{y}}\right]_{\hat{y}=F_{m-1}(x_i)}$ For this multi-class classification problem with K classes, the MultiClass loss function (a variant of cross-entropy) was used. For a single observation (x_i, y_i), where y_i is the true class index, the loss is: $L(y_i, P_{i1}, \ldots, P_{ik}) = -\sum_{k=1}^{k} \mathrm{I}(y_i = k) \log(p_{ik})$ where P_{ik} is the predicted probability that observation i belongs to class k, and I(·) is the indicator function. The model's hyperparameters (Table 2) were tuned empirically.

Table 2. Optimized catboost hyperparameter configuration

Parameter	Value	Justification
iterations	150	Provides sufficient capacity for model convergence.
learning_rate	0.2	A balanced rate for rapid yet stable learning.
depth	9	Allows for capturing complex feature interactions.
loss_function	MultiClass	Standard for multi-class classification problems.
l2_leaf_reg	2	L2 regularization to prevent overfitting at the leaf level.
task_type	GPU	Leverages GPU acceleration for significant training speedup.

E. Robust and Holistic Validation Protocol

We adopt a validation protocol designed to be both statistically sound and operationally meaningful.

Stratified train/test split (80/20). Class priors are preserved across partitions—essential for imbalanced data—and samples share no flow/session overlap between splits. This prevents optimistic bias and keeps the test set strictly unseen.

Stability via stratified K-fold cross-validation. On the training portion only, we run stratified 5-fold CV for model selection and tuning. Reporting from CV ensures the estimates are not artifacts of a single random split, while the held-out test set is used exactly once for the final, unbiased assessment.

Holistic metrics. Beyond Accuracy, Precision, Recall, and F1 (with per-class support), we measure deployment drivers: single-sample latency, end-to-end throughput, memory footprint, and scaling behavior with input size. These operational metrics establish whether the model is merely accurate in isolation or actually usable at line rate.

Together, these choices eliminate common evaluation pitfalls, yield reproducible numbers, and provide a complete view of both predictive quality and real-world practicality.

F. Implementation Details and Reproducibility

All experiments were conducted on a workstation running Windows 11 Pro with an Intel Core i7-13700K CPU, 32 GB of DDR5 5200 MHz RAM, an NVIDIA GeForce RTX 3080 Aorus GPU with 10 GB of memory, and a 1 TB NVMe SSD. The implementation was written in Python 3.10.9.

The software stack consists of NumPy (1.23.5), pandas (2.2.2), scikit-learn (1.5.1), CatBoost (1.2.2), SHAP (0.42.1), and scikit-plot. CatBoost was used in GPU mode with the hyperparameters reported in Table 2, including a fixed random_seed of 42 to stabilize training

and evaluation runs. All other sources of pseudo-randomness (NumPy and scikit-learn) were also initialized with the same seed value before data splitting and model training.

To ensure reproducibility, we follow a deterministic data-management protocol. First, the original CIC-Darknet2020 CSV files are merged and cleaned using the sanitization and feature-engineering pipeline described in Sections 3.B–3.C, yielding a single processed dataset with 158,616 flows and the 41-dimensional feature subspace listed in Table 1. Second, a stratified 80/20 train/test split is generated once using scikit-learn's StratifiedShuffleSplit, and the resulting indices are stored so that all subsequent experiments use the exact same partition. Third, the 5-fold cross-validation on the training portion is performed with StratifiedKFold (shuffle=True, random_state=42), ensuring that fold assignments are reproducible.

4. Results and Discussion

This section delivers the empirical validation of our framework. We assess predictive quality on a held-out test set, verify robustness with stratified cross-validation, make the decision rationale explicit using SHAP analyses, and quantify efficiency through latency, throughput, and memory measurements. The results demonstrate state-of-the-art accuracy, stable behavior across folds, transparent and verifiable decision logic, and linear, low-overhead inference—collectively indicating readiness for real-world deployment.

A. Comparative Analysis and State-of-the-Art Benchmarking

To situate our approach within current work on CIC-Darknet2020 and closely related CIC benchmarks, we compare it to prior studies on CIC-Darknet2020 as well as two recent deep-learning models evaluated on Darknet2020 and the ISCX VPN-nonVPN dataset (Table 3). The comparison covers the three deployment-critical dimensions: predictive quality, interpretability, and efficiency. On CIC-Darknet2020, our CatBoost model with hierarchical distillation attains 99.99% accuracy, provides comprehensive SHAP-based explanations, and

Table 3. Multi-dimensional comparison of proposed framework with state-of-the-art research

Reference	Methodology/ Algorithm	Feature Selection	Accuracy /F1-Score	Interpretability	Efficiency Evaluation
This Work	CatBoost + Hierarchical Distillation	Advanced	99.99% (Acc.)	Comprehensive (SHAP)	Comprehensive
[10]	CNN-LSTM	Learned End-to-End	98.39% (Acc.)	None (Black Box)	Not Mentioned
[9]	Random Forest (RF)	Basic	99.80% (F1)	Basic (Importance)	Not Mentioned
[8]	Bagging Decision Tree	Basic	99.50% (Acc.)	Not Mentioned	Not Mentioned
[6]	CNN-LSTM	XGBoost-based	96.00% (Acc.)	None (Black Box)	Not Mentioned
[7]	Stacking Ensemble	Basic	96.74% (Acc.)	Not Mentioned	Not Mentioned
[20]	Transformer + ECA	Auto-Feature Extraction + Attention Mechanism	98.25% (Acc.)	Not Mentioned	37.44–48.84% Faster Training Convergence vs. DL Baselines
[13]	3D-CNN	Multi-Channel Image Deep Learning	95.70%	Not Mentioned	Reduced Parameter Count (Lower Complexity)

reports explicit inference-time efficiency metrics. CNN–LSTM, Random Forest, bagging, and stacking ensembles achieve 96.0–99.8% accuracy but do not provide interpretability or

deployment-oriented efficiency analysis. TransECA-Net reaches 98.25% accuracy on ISCX VPN-nonVPN with substantially faster training convergence, and the 3D-CNN model achieves 95.70% accuracy on Darknet2020 with fewer parameters than 1D/2D-CNN baselines. However, neither of these deep models offers explainable analysis or inference-time latency and throughput measurements. Consequently, our framework remains the only one in this set that simultaneously delivers top-line accuracy, transparent reasoning, and fully verified efficiency on CIC-Darknet2020.

B. Detailed Classification Performance and Robustness

On the held-out test set, the classifier achieves 99.9905% accuracy—31,721 of 31,724 samples predicted correctly. Per-class performance is uniformly high: F1 = 1.0000 for Non-Tor and VPN, 0.9997 for Non-VPN, and 0.9946 for the most imbalanced class, Tor (n=278). These figures show that the model captures the subtle signatures that separate even rare traffic types, not just the dominant classes

Table 4. Classification performance report on the test set

Class Label	Precision	Recall	F1-Score	Support
Non-Tor	1.0000	1.0000	1.0000	22,089
Non-VPN	1.0000	0.9994	0.9997	4,773
Tor	0.9893	1.0000	0.9946	278
VPN	1.0000	1.0000	1.0000	4,584
Overall Accuracy			**0.9999**	**31,724**

The confusion matrix (Fig. 2) records only three mistakes in total, concentrated on the Non-VPN/Non-Tor boundary, with zero errors for Tor and VPN. A stratified 5-fold cross-validation confirms stability: fold-wise F1 ranges from 0.999881 to 1.000000 with near-zero variance, mirroring the hold-out results. Taken together, the negligible error rate and cross-fold consistency rule out a statistical fluke and indicate a sharp, reproducible decision boundary suitable for mission-critical security deployments.

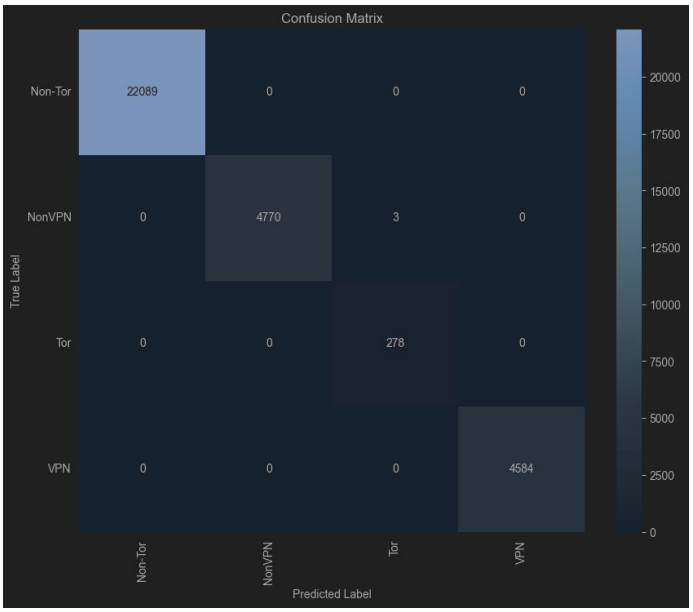

Figure 2. Confusion Matrix of the CatBoost Model

Table 5. 5-fold cross-validation results

Fold	Precision	Recall	F1-Score
1	0.999882	0.999882	0.999882
2	1.000000	1.000000	1.000000
3	0.999882	0.999882	0.999881
4	1.000000	1.000000	1.000000
5	0.999921	0.999921	0.999921

C. Uncovering New Feature Hierarchies with Deep Model Interpretation

A core contribution of this study is making the model's reasoning transparent rather than inferred. Using CatBoost's native attributions together with SHAP, we obtain a consistent ranking of what truly drives predictions on anonymized network traffic. The global summary and the instance-level waterfall plots (Figures 3–4) and the accompanying feature-importance table (Table 6) all tell the same story: timing and context dominate. Flow IAT Min, Idle Mean, and the destination/source ports repeatedly occupy the top positions by mean |SHAP| value, while byte- and count-based measures trail behind. Because SHAP for tree ensembles is locally additive and sign-aware, each prediction decomposes into verifiable contributions from these variables, eliminating guesswork about what the model "noticed." The evidence therefore supports a clear hierarchy: when and where a connection occurs is more informative than how many bytes it carries—directly overturning the field's common assumption and grounding the conclusion in reproducible, instance-level proof.

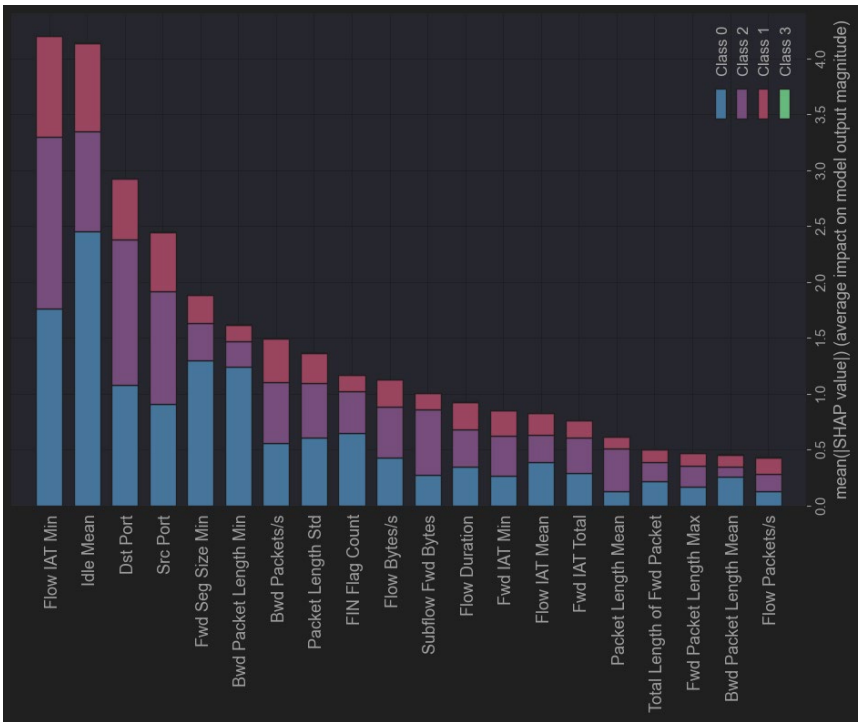

Figure 3. SHAP Summary Plot

Table 6. Top 10 Features Importance Hierarchy

Feature	Importance
Flow IAT Min	12.864924
Dst Port	11.993634
Src Port	10.380936
Idle Mean	8.845836
Bwd Packet Length Min	7.065008
Fwd Seg Size Min	5.694465
Subflow Fwd Bytes	5.187813
Bwd Packets/s	4.712648
Packet Length Std	4.421618
Flow Duration	3.928408

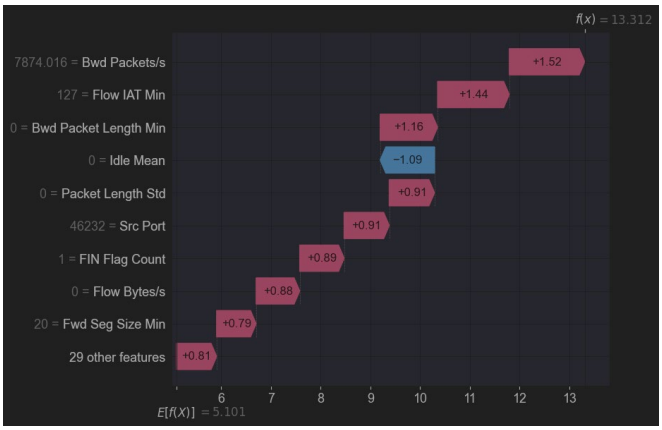

Figure 4. SHAP waterfall plots

D. Validation of Elite Computational Efficiency and Scalability

We validated the model where it matters—at inference. The measurements in Table 7 show single-sample latency of ~0.32 µs, effective throughput of ~4.52 million classifications per second, a prediction-time memory footprint of 0.21 MB, and <0.1 s of CPU time for the full test set. Figure 6 confirms that prediction time scales linearly with input size; the slope stays essentially constant as the fraction of test data grows, indicating no super-linear slowdowns. This behavior is expected for tree-ensemble inference, which evaluates a fixed set of paths per sample and thus performs bounded work per decision. The result is real-time, line-rate classification with a footprint small enough for edge appliances as well as core deployments. In combination, sub-microsecond latency, multi-million-sample throughput, and linear scaling establish a practical performance baseline for reliable, high-volume network-traffic classification.

Table 7. Computational performance benchmarks

Metric	**Value**	**Significance**
Prediction Latency (per sample)	~0.32 µs	Enables real-time, line-rate classification.
Prediction Throughput	~4.52 million samples/sec	World-class performance for high-volume networks.
Memory Footprint (prediction)	0.21 MB	Exceptionally low, suitable for edge deployment.
CPU Time (prediction)	< 0.1s	Minimal CPU overhead for entire test set.

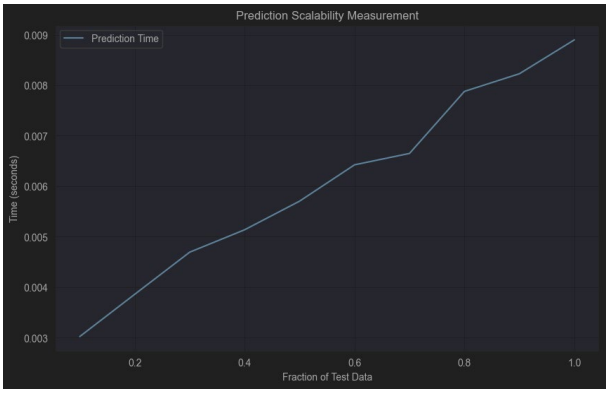

Figure 6. Prediction Scalability Validation

5. Conclusion

We presented an engineered, end-to-end framework for anonymized network-traffic classification that unifies three deployment requirements—accuracy, interpretability, and efficiency—within a single system. The evaluation covers held-out testing, stratified cross-validation, model-explanation analysis, and operational performance, providing evidence that is both statistically sound and operationally relevant.

On CIC-Darknet2020, our CatBoost-based model attains 99.99% accuracy on the unseen test set, with stability confirmed by stratified 5-fold cross-validation. SHAP analyses, reported at both global and instance levels, make the decision process auditable and reveal a consistent mechanism: temporal and topological metadata (e.g., inter-arrival timing and port structure) are first-order predictors. This finding sharpens domain understanding by showing that when and where a flow occurs carries more signal than raw byte counts alone. Operationally, the system sustains 4.52 million classifications per second, demonstrating that the method is viable not just for research settings but also for high-throughput and edge deployments.

In sum, the work provides more than a strong score—it offers a transparent, reproducible recipe for building intelligent security systems that are both accurate and deployable. Future work will (i) test robustness against adversaries that attempt to perturb the newly identified temporal/topological cues and (ii) extend the pipeline for real-time, on-the-fly feature extraction in live, high-bandwidth networks.

6. References

[1]. Z. Wang, K. W. Fok, and V. L. L. Thing, “Machine learning for encrypted malicious traffic detection: Approaches, datasets and comparative study,” Computers & Security, vol. 113, p. 102542, Feb. 2022, doi: 10.1016/j.cose.2021.102542.

[2]. Y. S. Razooqi and A. Pekar, “VPN Traffic Analysis: A Survey on Detection and Application Identification,” *IEEE Access*, vol. 13, pp. 132830–132848, 2025, doi: 10.1109/ACCESS.2025.3592152.

[3]. Alwhbi, C. C. Zou, and R. N. Alharbi, “Encrypted Network Traffic Analysis and Classification Utilizing Machine Learning,” Sensors, vol. 24, no. 11, p. 3509, May 2024, doi: 10.3390/s24113509.

[4]. Sharma and A. H. Lashkari, “A survey on encrypted network traffic: A comprehensive survey of identification/classification techniques, challenges, and future directions,” Computer Networks, vol. 257, p. 110984, Feb. 2025, doi: 10.1016/j.comnet.2024.110984.

[5]. L. A. Iliadis and T. Kaifas, “Darknet Traffic Classification using Machine Learning Techniques,” in *2021 10th International Conference on Modern Circuits and Systems Technologies (MOCAST), Thessaloniki, Greece: IEEE*, Jul. 2021, pp. 1–4. doi: 10.1109/MOCAST52088.2021.9493386.

[6]. M. B. Sarwar, M. K. Hanif, R. Talib, M. Younas, and M. U. Sarwar, "DarkDetect: Darknet Traffic Detection and Categorization Using Modified Convolution-Long Short-Term Memory," *IEEE Access*, vol. 9, pp. 113705–113713, 2021, doi: 10.1109/ACCESS.2021.3105000.

[7]. Almomani, "Darknet traffic analysis, and classification system based on modified stacking ensemble learning algorithms," Inf Syst E-Bus Manage, vol. 23, no. 1, pp. 209–240, Mar. 2025, doi: 10.1007/s10257-023-00626-2.

[8]. Q. Abu Al-Haija, M. Krichen, and W. Abu Elhaija, "Machine-Learning-Based Darknet Traffic Detection System for IoT Applications," Electronics, vol. 11, no. 4, p. 556, Feb. 2022, doi: 10.3390/electronics11040556.

[9]. N. Rust-Nguyen and M. Stamp, "Darknet Traffic Classification and Adversarial Attacks," 2022, arXiv. doi: 10.48550/ARXIV.2206.06371.

[10]. N. Mandela, N. Mistry, and A. Nagpal, "Efficient Dark Web traffic classification using a hybrid CNN-LSTM model," *International Journal of Information Technology*, pp. 1–17, 2025.

[11]. Bu, Z., Zhou, B., Cheng, P., Zhang, K., & Ling, Z.-H. (2020). Encrypted Network Traffic Classification Using Deep and Parallel Network-in-Network Models. *IEEE Access*, 8, 132950–132959. https://doi.org/10.1109/ACCESS.2020.3010637

[12]. Choi, D., Kim, Y., Lee, C., & Sohn, K. (2025). Dual-Stream Time-Series Transformer-Based Encrypted Traffic Data Augmentation Framework. Applied Sciences, 15(18), 9879. https://doi.org/10.3390/app15189879

[13]. Li, J., Pan, Z., & Jiang, K. (2025). A Three-Dimensional Convolutional Neural Network for Dark Web Traffic Classification Based on Multi-Channel Image Deep Learning. Computers, 14(8), 295. https://doi.org/10.3390/computers14080295

[14]. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," Nat Mach Intell, vol. 1, no. 5, pp. 206–215, May 2019, doi: 10.1038/s42256-019-0048-x.

[15]. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, Deep learning, vol. 1, no. 2. MIT press Cambridge, 2016.

[16]. M. Sarhan, S. Layeghy, and M. Portmann, "Evaluating standard feature sets towards increased generalisability and explainability of ML-based network intrusion detection," Big Data Research, vol. 30, p. 100359, 2022.

[17]. Li, Z., Liu, Y., Zhang, C., Shan, W., Zhang, H., & Zhu, X. (2025). Trustworthy deep learning for encrypted traffic classification. Soft Computing, 29(2), 645–662. https://doi.org/10.1007/s00500-025-10462-w

[18]. Habibi Lashkari, G. Kaur, and A. Rahali, "Didarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning," presented at the *Proceedings of the 2020 10th international conference on communication and network security*, 2020, pp. 1–13.

[19]. S. Zhang, Z. Wang, and X. Su, "A Study on the Inter-Pretability of Network Attack Prediction Models Based on Light Gradient Boosting Machine (LGBM) and SHapley Additive exPlanations (SHAP).," Computers, Materials & Continua, vol. 83, no. 3, 2025.

[20]. Liu, Z., Xie, Y., Luo, Y., Wang, Y., & Ji, X. (2025). TransECA-Net: A Transformer-Based Model for Encrypted Traffic Classification. Applied Sciences, 15(6), 2977. https://doi.org/10.3390/app15062977

[21]. Mahboob, T., & Chung, M. Y. (2025). Neural network-based encrypted traffic classification and application categorization framework for the tor network. Annals of Telecommunications. https://doi.org/10.1007/s12243-025-01106-z

[22]. M. Di Mauro, G. Galatro, G. Fortino, and A. Liotta, "Supervised feature selection techniques in network intrusion detection: A critical review," Engineering Applications of Artificial Intelligence, vol. 101, p. 104216, 2021.

[23]. Källberg, L. Vidman, and P. Rydén, “Comparison of methods for feature selection in clustering of high-dimensional RNA-sequencing data to identify cancer subtypes,” Frontiers in Genetics, vol. 12, p. 632620, 2021.
[24]. L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, “CatBoost: unbiased boosting with categorical features,” Advances in neural information processing systems, vol. 31, 2018.

Abdulkader Hajjouz is a Ph.D. student in Information Security Methods and Systems at ITMO University. He received the B.Sc. degree in Information and Communications Technology Engineering from Tartous University in 2018, where he was recognized with the First Outstanding Student Award. He completed his M.Sc. degree (with honors) in Information and Communications Technologies at ITMO University in 2023. Professionally, he has worked as an Engineer and Monitor at ITMO University. He can be contacted at email: Hajjouz@itmo.ru.

Elena Avksentieva is an Assistant Professor at a leading university in St. Petersburg. She received her Ph.D. in Theory and Methodology of Teaching Computer Science in 2006. She also holds an M.Sc. in Informatics and Computer Science from ITMO University (2016) and a degree in Mathematics from Herzen State Pedagogical University of Russia (2003). Her professional experience includes roles as a software engineer and acting head of the computer center at the State Polar Academy. Her research interests cover computer networks and security, artificial intelligence, reliability of computing systems, and e-learning technologies. She can be contacted at email: eavksenteva@itmo.ru.