APS 360 Project Proposal

Andre Brian Danny

January 30, 2026

Student number : 1009793788

Abstract

This project develops a network traffic analyzer to detect Darknet activities (Tor/VPN) using flow metadata from the CIC-Darknet2020 dataset. We contrast a standard Random Forest classifier with a 1D-Convolutional Neural Network (1D-CNN) designed for tabular signal processing. The objective is to determine if deep learning architectures can better capture latent correlations in encrypted traffic compared to traditional feature-engineered models.

1 Introduction

1.1 Goal:

The objective of this project is to develop a Deep Learning model capable of classifying network traffic as "Benign" (e.g., Browsing, Streaming) or "Darknet" (Tor, VPN) using only time-based statistical flow features, without decrypting packet payloads.

1.2 Motivation:

As internet traffic increasingly shifts toward end-to-end encryption (HTTPS, TLS 1.3), traditional Deep Packet Inspection (DPI) methods are becoming obsolete. Network administrators and cybersecurity analysts face a critical blind spot: they cannot distinguish between legitimate privacy-seeking users and malicious actors using Darknet tunnels for command-and-control or exfiltration.

1.3 Why Deep Learning:

While traditional machine learning models like Random Forest perform well on summary statistics, they often struggle to capture complex, non-linear dependencies in high-dimensional feature spaces specifically the 100+ hashed IP features we generate to avoid overfitting. A 1D-Convolutional Neural Network (CNN) is a reasonable approach because it can learn to extract latent "fingerprints" from this dense feature vector, effectively embedding the high-dimensional input into a robust representation for classification.

2 Illustration:

2.1 Architecture Illustration

Figure 1 visualizes the data flow through the proposed 1D-CNN. The architecture processes the input traffic as a sequential signal, transforming the raw feature vector into a high-level abstract representation for classification.

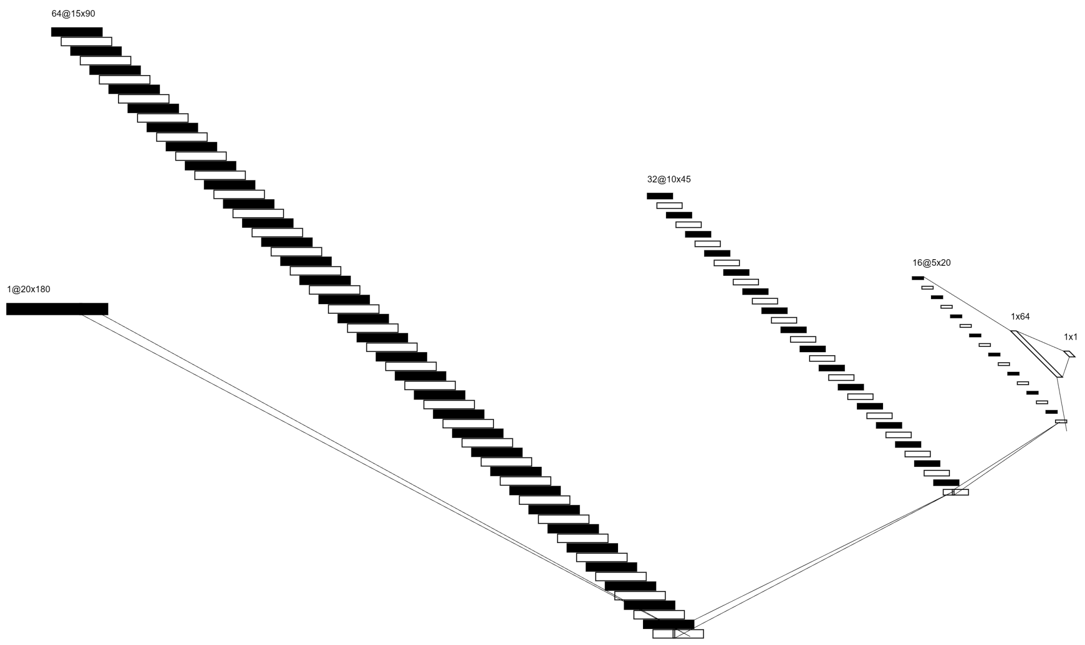

Figure 1: Proposed 1D-CNN Data Flow.

2.2 Visual Breakdown:

- **Input Stage (Leftmost Block):** The model ingests a single channel input vector of dimensions 1×180, representing the hashed IP and flow statistics sequence.
- **Feature Extraction (Stacked Bars):** The subsequent three stages represent the convolutional blocks.
  - **Decreasing Length (Horizontal):** As the signal propagates to the right, the horizontal length of the bars halves ($180 \rightarrow 90 \rightarrow 45 \rightarrow 20$). This visualizes the **MaxPooling** operations, which downsample the signal to retain only the most salient features.
  - **Increasing Depth (Vertical):** Simultaneously, the number of stacked bars increases ($1 \rightarrow 64 \rightarrow 32 \rightarrow 16$). This illustrates the **Convolutional Filters** expanding the channel depth to learn hierarchical patterns.
- **Classification Head (Rightmost Components):** The final convolutional output is collapsed into a single vertical vector (1×64), representing the **Flatten and Dense** layer. This vector converges to a single point (1×1), representing the **Sigmoid Output** which provides the final probability of the traffic flow being 'Darknet'.

3 Background & Related Work

Traffic analysis has evolved significantly from port-based classification to statistical analysis and, more recently, deep learning approaches. This section reviews five key studies that establish the foundation for classifying encrypted Darknet traffic.

3.1 Statistical and Time-Based Approaches:

Early research focused on identifying encrypted tunnels using manual feature engineering. [DGLMG16] demonstrated that virtual private network (VPN) traffic exhibits distinct time-related signatures compared to non-VPN traffic. By analyzing flow duration and idle times, they successfully distinguished VPN tunneling using standard machine learning algorithms like C4.5 and k-Nearest Neighbors. Building on this, [LDGDG17] focused specifically on The Onion Router (Tor) network, establishing that time-based features such as inter-arrival time (IAT) could identify Tor circuits even when the application layer was encrypted.

3.2 The CIC-Darknet2020 Baseline:

Most relevant to this project is the work by [LKR20], who introduced the [FL20] dataset. They proposed a two-layered approach using Random Forests to first classify traffic as VPN/Tor and then identify the specific application (e.g., Skype, Facebook). Their work validated that statistical network behaviors (jitter, packet size variance) are sufficient to detect Darknet activities with over 90% accuracy, serving as the benchmark for our current study.

3.3 Shift to Deep Learning:

Recent literature suggests that Deep Learning can outperform these traditional tree-based methods by automating feature extraction. [WZW+17] pioneered the use of 1D-Convolutional Neural Networks (1D-CNNs) for encrypted traffic classification. They treated traffic flows as sequential data, demonstrating that CNNs could learn local correlation patterns directly from raw packet headers and payloads. Similarly, [RL19] explored multi-task deep learning frameworks, showing that deep neural networks are particularly robust against traffic obfuscation techniques where manual feature engineering often fails.

Our project synthesizes these approaches by applying the 1D-CNN methodology of Wang et al. to the statistical feature set established by Lashkari et al., aiming to capture latent behavioral fingerprints in Darknet traffic.

4 Data Processing

4.1 Dataset Source

We will utilize the [FL20] dataset, generated by the Canadian Institute for Cybersecurity. This dataset contains captured network traffic from over 20 representative applications (e.g., Skype, Tor, Facebook) routed through VPN and Tor gateways. The raw data provides 85 statistical flow features (such as Flow Duration, Flow Bytes/s, and Inter-Arrival Times) for over 141,000 samples. This allows the model to process novel IP addresses in the test set without retraining, mapping them to existing "buckets" and preserving the statistical structure of the network traffic. enditemize

4.2 Data Cleaning and Label Sanitization

The raw CSV dataset contains inconsistent label naming conventions (e.g., 'AUDIO-STREAMING' vs. 'Audio-Streaming') which will be sanitized to unify duplicate categories. We will also handle missing or infinite values artifacts of flow calculation errors by dropping rows containing `NaN` or `Infinity` in critical fields such as `Flow Bytes/s`. Finally, we will aggregate the specific application labels into two binary super-classes: **Benign** (Non-Tor/Non-VPN) and **Darknet** (Tor/VPN) to formulate a binary classification problem.

4.3 Feature Transformation

To prepare the tabular data for the 1D-CNN architecture, we will apply the following transformations:

- **IP Encoding:** The Source and Destination IP addresses exhibit high cardinality. To utilize them without overfitting, we will apply **Feature Hashing** to project the IP addresses into a fixed-size dense vector.
- **Normalization:** We will apply `StandardScaler` to all numerical flow statistics (e.g., Flow Duration, Packet Counts) to normalize them to zero mean and unit variance. To prevent data leakage, this scaler will be fit exclusively on the training split.

5 Architecture

5.1 Proposed Architecture Design

We propose a **1D-Convolutional Neural Network (1D-CNN)** architecture specifically adapted for tabular signal processing. The design is motivated by the need to capture latent correlations between the hashed IP vectors and temporal flow statistics.

- **Input Layer:** The model accepts the processed feature vector as a 1D signal channel. This treats the 186 tabular features as a sequential input, allowing the network to learn spatial relationships between the hashed IP buckets.
- **Convolutional Feature Extractors:** We will employ a stack of `Conv1d` layers.
  - **Goal:** To apply sliding filters across the input vector, detecting non-linear interactions between the IP address "fingerprints" (from feature hashing) and the associated traffic volume metrics.
  - **Normalization:** Given the high variance in the raw dataset (e.g., `Flow Duration` in milliseconds vs. `Packet Counts` as integers), we will incorporate `BatchNorm1d` layers after convolutions. This is critical to prevent covariate shift and ensure stable gradient propagation across diverse feature scales.
- **Dimensionality Reduction:** Pooling layers (e.g., `MaxPool1d`) will be inserted between convolutional blocks to downsample the feature map, reducing the model's sensitivity to small shifts in the feature hashing space and lowering computational cost.
- **Classification Head:** The extracted feature maps will be flattened and passed to a Multi-Layer Perceptron (MLP) block. This block will utilize **Dropout** for regularization to prevent overfitting on the specific noise profile of the training data. The final layer will be a single neuron with a Sigmoid activation to output the probability of the flow being "Darknet".

5.2 Training Strategy

The problem is formulated as a binary classification task. Therefore, we will optimize the model using **Binary Cross-Entropy (BCE) Loss**. The weights will be updated using an adaptive gradient optimizer (such as Adam or RMSProp), which is well-suited for non-stationary objectives often found in network traffic data.

6 Baseline Model

To validate the effectiveness of the deep learning approach, we will compare its performance against a **Random Forest Classifier**, which serves as the industry standard for metadata-based network traffic analysis.

Implementation Details: We will implement the baseline using the `scikit-learn` library with the following configuration:

- **Ensemble Size:** The model will utilize `n_estimators=100` decision trees. This ensemble size provides a robust estimate by averaging predictions across 100 independent trees, reducing the variance associated with individual decision trees.
- **Splitting Criterion:** We will use Gini Impurity to measure the quality of splits.
- **Input:** The same 186-feature vector used for the CNN will be flattened and fed into the Random Forest.

Justification: Random Forests are highly effective at handling tabular data and are robust to outliers in flow statistics. However, they treat features independently at each split node. If our hypothesis is correct that latent relationships exist between the hashed IP buckets and temporal flow metrics the 1D-CNN should outperform this baseline by learning these coupled representations via convolution.

7 Ethical Considerations

7.1 Privacy and Data Minimization:

A primary ethical advantage of our approach is the exclusive reliance on *flow metadata* (e.g., inter-arrival times, packet sizes) rather than Deep Packet Inspection (DPI). By discarding payload contents and hashing IP addresses, our model adheres to privacy-preserving principles, ensuring that the specific content of user communications (messages, passwords) remains inaccessible even during the classification process.

7.2 Dual-Use Risk:

This technology falls under the category of ”Dual-Use” research. While our objective is to assist network administrators in detecting malicious tunnels (e.g., ransomware exfiltration), the same methodology could be weaponized by authoritarian regimes. Such actors could deploy this model to identify and block citizens using Tor or VPNs to evade censorship, effectively undermining the anonymity these tools are designed to protect.

7.3 Dataset Bias and Limitations:

The [FL20] dataset was generated in a controlled environment, which introduces inherent selection bias. The model may overfit to the specific VPN protocols and Tor circuits used during data generation. Consequently, there is a risk of *False Positives* when deployed in the wild, where legitimate, novel privacy tools might be misclassified as malicious Darknet traffic, potentially leading to unjustified service denial for innocent users.

8 Links

8.1 google Colab notebook:

https://colab.research.google.com/drive/1nL9Q4zheGxyk3z4--xaZe6uA4iY3bVsP?usp=sharing

8.2 GitHub Repository:

https://github.com/redbottleneck/CNN-Darknet-Traffic-Classifier

References

[DGLMG16] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. Characterization of encrypted and vpn traffic using time-related features. In *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP)*, pages 407–414, 2016.

[FL20] Peter Friedrich and Arash Habibi Lashkari. CIC-Darknet2020 Internet Traffic. Kaggle Dataset, 2020. Available at https://www.kaggle.com/datasets/peterfriedrich1/cicdarknet2020-internet-traffic/data.

[LDGDG17] Arash Habibi Lashkari, Gerard Draper-Gil, M Davoudi, and Ali A Ghorbani. Characterization of tor traffic using time based issues. In *Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP)*, pages 174–181, 2017.

[LKR20] Arash Habibi Lashkari, Gurdip Kaur, and Abir Rahali. Didarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning. In *10th International Conference on Communication and Network Security (ICCNS)*, 2020.

[RL19] Shahbaz Rezaei and Xin Liu. Deep learning for encrypted traffic classification: An overview. *IEEE Communications Magazine*, 57(5):76–81, 2019.

[WZW$^+$17] Wei Wang, Ming Zhu, Xuewen Wang, Cohen McQuade, and Mehdi Mirakhorli. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 43–48. IEEE, 2017.