Deep Learning for Encrypted Network Traffic Classification: Exploring New Techniques for Efficient Traffic Management

A Comparative Analysis of Deep Learning Models for Classifying Application Traffic in Encrypted Network Environments

Kieran Brooks

200236702

CS 435 – Dr. Qian Yu

Introduction

Digital networks have become essential in our daily lives, and their management has become crucial in maintaining their security, performance, and reliability. Traffic management techniques are employed to ensure efficient utilization of network resources, enforce network policies, and detect potential threats to network stability and security. However, the increasing use of encryption technologies, such as transport layer security, poses significant challenges to traffic analysis. Encryption conceals more data from network operators, making it difficult for analysts to detect patterns, anomalies, or malicious activities. Additionally, obfuscation techniques, such as protocol obfuscation and traffic flow obfuscation, further complicate traffic analysis by disguising network traffic and hiding patterns in encrypted traffic. Consequently, traffic analysis techniques must evolve to adapt to encrypted networks and maintain their effectiveness in identifying and mitigating threats.

This research aims to explore the feasibility of applying deep learning techniques to traffic payload analysis in highly encrypted network environments to classify application traffic over encrypted channels. As public networks become more secure and encryption is more widely adopted, traditional traffic pattern analysis and payload inspection techniques become increasingly more challenging. The traditional approach of analyzing port numbers, protocol types, and IP addresses is becoming less effective as encryption is used to hide the content of the communication. In response, increasing interest has been in using machine learning and deep learning techniques to analyze encrypted traffic. The primary research question of this investigation is to determine if machine learning or deep learning techniques, such as AdaBoost Decision Trees, 1D-CNN, BD-LSTM, and BD-GRU, can be effectively applied to classify encrypted network traffic based on TLS handshake exchange. Dataset selection and feature engineering will be inspired by the work of Shamsimukhametov et al. (D. Shamsimukhametov et al., 2022), who achieved high accuracy metrics in the literature review. The original dataset will be used, and the recomposed bytes (RB) approach will be recreated and used to engineer the most informative features of the handshake data into a feature vector. This feature vector will be used as input to various classification models. The results will be compared with each other as well as with a control to determine the most effective approach for predicting the payload classification label of the ClientHello + ServerHello feature vector. Overall, this research project aims to contribute to developing new techniques that can improve traffic analysis results in highly encrypted network environments.

Literature Review

In the context of this research paper, the related works encompass various studies and techniques that primarily address the encrypted traffic environment prior to the TSL 1.3 encrypted ClientHello (ECH) amendments. The development of ECH by the TLS Working Group aims to address privacy concerns arising from the exposure of server name indication (SNI) and other sensitive information in TLS handshake packets sent in plaintext protocol payload headers during protocol negotiation. This research paper builds upon these existing works, exploring the effectiveness of various traffic classification techniques and model architectures in examining ECH-protected TLS traffic.

In early relevant work, W. Wang et al. (2017) proposed an end-to-end encrypted traffic classification method using one-dimensional convolutional neural networks (1D-CNN). This method unifies feature extraction, feature selection, and classifier into a single framework, allowing for the automatic learning of nonlinear relationships between raw input and expected output. Representing the first application of classifiers to end-to-end encryption, their method was validated using the public ISCX 2016 traffic dataset (A. Habibi Lashkari et al., 2016), outperforming 11/12 metrics. Using a similar neural network architecture, H. Lim et al. (2019) generated packet-based datasets through network traffic pre-processing and trained five deep learning models using convolutional neural networks (CNN) and residual networks (ResNet) for network traffic classification. The effectiveness of these models was demonstrated through the analysis of network traffic classification performance using the F1 score for CNN and ResNet deep learning models on packet-based datasets.

Expanding on an approach familiar to 2D CNN methods, Lu et al. (2021) proposed the Inception-LSTM method, which converts traffic data into grayscale 2D images and uses an LSTM neural network for feature extraction and classification. Validated on the ISCX 2016 dataset (Lashkari et al. 2016), the Inception-LSTM method achieved over 98% accuracy for regular and VPN encrypted traffic service identification and simplified the feature extraction process.

Cheng et al. (2021) proposed a lightweight, online solution called MATEC, which employs multi-head attention and convolutional networks, significantly reducing parameters and running time. Compared to state-of-the-art models on three datasets, MATEC demonstrates higher accuracy, improved efficiency, and halved training time over other state-of-the-art models. Improving on the performance of MATEC, a notable study by Shamsimukhametov et al. (2022) investigated the effect of ECH on TLS-encrypted traffic classification specifically. The authors collected a diverse dataset, proposed a method for comparing traffic

classification algorithms under different encryption scenarios, and developed an effective novel feature engineering algorithm called Recomposed Bytes (RB). When used as a feature vector input to a Random Forest classifier (RB-RF), they achieved higher classification quality and throughput than state-of-the-art algorithms such as MATEC. The RB algorithm and the novel QoS-aware dataset from Shamsimukhametov et al. are primary contributors to the investigation in the following pages.

To address the limitations of conventional methods, Z. Shi et al. (2023) propose a BERT-based byte-level feature convolutional network (BFCN) model. This model captures global traffic features using a packet encoder module with a BERT pre-trained traffic classification model and extracts byte-level local features through a CNN module. The concatenated packet-level and byte-level features better represent encrypted traffic, resulting in state-of-the-art performance on the ISCX-VPN dataset. The BFCN model demonstrates improved encrypted traffic classification by leveraging both byte-level and packet-level features. Autoencoder approaches may represent the newest frontier in traffic classification in highly encrypted environments, including ECH traffic.

In summary, the related works in this research project encompass a diverse range of techniques and studies centered around encrypted traffic classification. As TLS 1.3 and ECH become increasingly prevalent, advances in this field are growing more significant. These studies investigate various traffic classification techniques, model architectures, and strategies to tackle challenges encountered by conventional machine learning methods. This research paper intends to build upon these existing works, enhancing the understanding and efficacy of traffic classification methods in the context of ECH-protected TLS traffic, focusing on the feasibility of deep learning methods and the potential for hybrid approaches that combine feature engineering, machine learning, and deep learning techniques.

Problem Statement

The rapid evolution of encryption technologies, such as TLS 1.3 and the ECH amendment, has made traditional traffic analysis methods increasingly less effective in classifying application traffic over encrypted channels. The concealment of data and obfuscation techniques used in encrypted networks hinder the ability of network operators to effectively detect patterns, anomalies, or malicious activities in traffic. As a result, there is a growing need to develop new traffic analysis techniques that can effectively classify encrypted traffic while maintaining high accuracy and efficiency.

The main problem this research aims to address is identifying and classifying application traffic in highly encrypted network environments such as ECH. The focus will be on determining the effectiveness of machine learning and deep learning techniques, such as AdaBoost Decision Trees (ADT), 1D-CNN, BD-LSTM, and BD-GRU, in classifying encrypted network traffic based on TLS handshake exchange. Additionally, this research will explore the potential benefits of hybrid approaches that combine feature engineering and machine learning techniques to improve traffic classification performance in the context of ECH-protected traffic.

The primary challenges associated with addressing this problem include the following:

- Implementing an effective feature engineering technique that can capture relevant information from TLS handshake exchanges in ECH-protected scenarios.
- Evaluating the performance of various machine learning and deep learning models in classifying encrypted traffic, considering accuracy, efficiency, and generalizability factors.
- Comparing the effectiveness of different classification models and determining the most suitable approach for predicting the payload classification label of the ClientHello + ServerHello feature vector.
- Investigating the potential for hybrid approaches that combine feature engineering, machine learning, and deep learning techniques to enhance the classification performance with respect to ECH encrypted traffic.

By addressing these challenges, this research project aims to contribute to the development of novel traffic analysis techniques capable of maintaining their effectiveness in the context of highly encrypted network environments, ultimately improving network management, resource allocation, and security.

Traffic Management

In the age of rapid technological advancements and growing digital communication, traffic management is critical to digital network administration. Effective traffic management strategies are necessary to ensure optimal network performance, enforce network policies, and maintain robust network security. This involves monitoring, controlling, and optimizing data flow within digital networks to efficiently use resources and prevent congestion. Traffic classification, a fundamental component of traffic management, involves categorizing different types of network traffic based on various attributes such as protocol, application, or source and destination addresses to prioritize them according to importance and network policies. Traffic shaping and prioritization involve controlling the data flow to ensure efficient

resource allocation and prevent congestion. Administrators must continuously observe and analyze network traffic patterns, identify potential issues, detect security threats, and optimize network performance and topology.

Challenges to Traffic Management Posed by Encryption

As cyber threats and privacy concerns rise, the widespread adoption of encryption has become essential for secure communication between users and servers. Encryption technologies help protect sensitive information, such as personal details, financial transactions, and private communications, enabling users to engage in digital activities without fear of data breaches or eavesdropping. Various encryption protocols, such as Transport Layer Security (TLS), Encrypted Server Name Indication (ESNI), and encrypted Domain Name System (DNS) protocols like DNS over HTTPS (DoH) and DNS over TLS (DoT), have been developed to safeguard data transmitted over digital networks.

However, these protocols pose significant challenges for traffic analysis by reducing network data visibility and making it difficult for analysts to detect patterns, anomalies, or malicious activities. TLS 1.3 provides secure communication over the internet, securing applications like web browsing, email, and instant messaging. While it brings several improvements for enhanced security and performance, the increased encryption in TLS 1.3 also hinders traffic analysis, limiting the information available to network operators.

TLS 1.3 introduces encryption enhancements, including Encrypted Client Hello (ECH), that protect sensitive information in the initial handshake process. To successfully deploy ECH, client-facing servers must publish their public keys and metadata for all the domains they serve. This combination of ECH encryption public key and metadata is collectively referred to as an ECH configuration. Clients utilize this configuration to securely encrypt sensitive extensions within their ClientHello message, known as ClientHelloInner. By shielding sensitive information and securely transmitting it to the intended server, ECH minimizes the risk of interception and inspection by unauthorized parties, thereby maintaining the confidentiality of client-server connections and providing robust resistance to potential eavesdropping and malicious actors.

As encryption becomes ubiquitous, traffic analysis techniques must evolve to maintain their effectiveness in identifying and mitigating threats. Novel approaches that can work with limited visibility and maintain the balance between privacy and security are required.

TLS 1.3 ECH Overview

TLS 1.3 generally and ECH specifically attempts to block interception and analysis of this client and server fingerprinting data exchanged during the TLS handshake process. To successfully deploy ECH, client-facing servers must publish their public keys and metadata for all the domains they serve. This combination of ECH encryption, public key, and metadata is collectively called an ECH configuration. Clients utilize this configuration to securely encrypt sensitive extensions within their ClientHello message, known as ClientHelloInner.

When initiating a TLS session with a backend server, the client first constructs the ClientHelloInner message containing the sensitive information. This message is then encrypted using the ECH configuration's public key. Subsequently, the client creates a new ClientHello message called ClientHelloOuter, which contains innocuous values for sensitive extensions and an "encrypted_client_hello" extension. This extension includes the encrypted ClientHelloInner and specifies the ECH configuration utilized for encryption. Finally, the client sends the ClientHelloOuter to the server.

Upon receiving the ClientHelloOuter, the client-facing server takes one of three possible actions, depending on its ECH support and ability to decrypt the "encrypted_client_hello" extension. If the server does not support ECH or cannot decrypt the extension, it will either proceed with the handshake as usual or terminate the handshake using ClientHelloOuter. Conversely, if the server supports ECH and successfully decrypts the extension, it forwards the ClientHelloInner to the backend server, which terminates the connection.

A fundamental objective of TLS 1.3 with Encrypted ClientHello (ECH) is to ensure that connections to servers within the same anonymity set are indistinguishable without affecting any existing security properties of TLS 1.3. ECH achieves this by implementing an inner and outer handshake, allowing clients to encrypt sensitive information within the payload bytes of the ECH TLS 1.3 extension. It is crucial to understand that ECH is only an extension of TLS 1.3, serving as an additional layer of protection for the encrypted sensitive information typically subject to traffic analysis by both good and bad actors. The client decides which extensions it will encode into the ClientHelloInner, which may be duplicated in the ClientHelloOuter. This approach protects the Server Name Indication (SNI) and the Application Layer Protocol Negotiation list (ALPN).

By shielding sensitive information and securely transmitting it to the intended server, ECH minimizes the risk of interception and inspection by unauthorized parties. This results in maintaining the confidentiality of client-server connections and providing robust resistance to potential eavesdropping and malicious actors.

Methods

This section of the research project presents a comprehensive overview of the experimental design, encompassing dataset selection, feature extraction, feature engineering, model architecture and training, evaluation metrics, and performance measurement. The primary research question of this investigation is driven by a desire to delve deeper into the work of Shamsimukhametov et al. (2022), who achieved the highest accuracy metrics in the literature review. The experimental design is inspired by their approach, with a critical addition of supervised neural network models and a Boosted Decision Tree classifier to attempt to improve the original results. The implementation comprises several stages, including data collection and preparation, replicating the RB algorithm, feature selection and engineering, training randomization, supervised learning or fitting of the output classifier models for the models under study, and evaluating the models against unseen data.

For this project, the original dataset from the Shamsimukhametov paper was chosen, which includes 3547 flows divided into 12 service classes. The flows consist of live video traffic, buffered audio, video traffic from popular services like YouTube, Netflix, and Spotify, and web flows from 100 popular websites. The data was collected from various devices with different operating systems in different networks in 3 Russian cities from 2020 through 2021. The flows were cut to the first 200 packets to create a lightweight dataset suitable for testing classifiers in the ECH scenario based on analyzing statistics and TLS key exchange. The dataset's acquisition, pre-processing, cleaning, and organization will ensure high-quality input for subsequent analysis and modeling to replicate and improve upon the results presented in the original paper.

The feature selection process in this research project aims to identify the most informative features from the raw packet bytes of the ClientHello and ServerHello handshake data. An algorithm will be developed to recreate the RB approach used in the Shamsimukhametov paper. This technique serves as a comprehensive feature engineering and selection mechanism. The RB algorithm rearranges the contents of the raw packet bytes from ClientHello and ServerHello, transforming them into ordered arrays of integer values representing their data contents. The algorithm for the TLS handshakes involves the decomposition of the ClientHello and ServerHello messages into lists of fields, extension types, lengths, and data

(Shamsimukhametov et al., 2022). Payload parameters are assigned fixed positions and lengths to match data columns across training samples from different handshakes and improve classification quality. Achieving dimensionality reduction and composing a fixed-length feature vector of concatenated ClientHello and ServerHello byte strings.

In Shamsimukhametov et al., this feature vector was then fit to a Random Forest classifier and used to predict against unseen data from the same dataset. This process was recreated, and the results were recorded as a control. To attempt to extend the effectiveness of the RB-RF approach, five different model architectures and algorithms were compared, including the AdaBoost Decision Trees (ADT), Stacked 1D Convolutional Neural Network (1D-CNN), a Bidirectional Long Short-Term Memory (BD-LSTM) network, and a Bidirectional Gated Recurrent Unit Network(BD-GRU). Additionally, a stacked ensemble approach combining these models will be trained and tested to facilitate a comprehensive comparison of individual model architecture performance and determine the most effective approach for predicting the payload classification label of the ClientHello + ServerHello messages.

Model Evaluation Metrics

Several metrics are employed to evaluate the performance of each model under study, including overall accuracy, model training time, precision, recall, and F1 score. The overall accuracy was computed by dividing the number of correctly classified instances by the total number of instances in the test set. This metric measures the model's ability to classify instances correctly and is widely used in classification tasks. However, when the class distribution is imbalanced, or the cost of misclassifying one class differs, additional metrics such as precision, recall, and F1 score are employed.

$$\text{precision}_i = \frac{TP_i}{TP_i + FP_i}.$$

$$\text{recall}_i = \frac{TP_i}{TP_i + FN_i}$$

$$\text{f-score}_i = \frac{2}{\text{recall}_i^{-1} + \text{precision}_i^{-1}}.$$

$$\text{accuracy} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + FN_i}$$

Precision measures the proportion of correctly classified instances out of the total number of instances predicted as positive by the model. It represents the model's ability to avoid false positives and is computed as the ratio of true positives to the sum of true positives and false positives. Conversely, recall measures the

proportion of correctly classified instances out of the total number of positive instances. It represents the model's ability to identify all positive instances and is computed as the ratio of true positives to the sum of true positives and false negatives. The F1 score is the harmonic mean of precision and recall, providing a single measure that balances both metrics.

In addition to these metrics, model training time was also measured to evaluate the computational efficiency of each model. The training time was recorded in seconds and represents the time the model took to fit the training data and generate and converge a set of parameter weights for the classification task over a fixed number of epochs for each neural network model. The ADT and Random Forest classifiers are not neural networks, so comparing time scores across architectures is problematic. However, the ML times are so vastly different that they warrant a data point for comparison. The reader is advised to understand that comparing ML to DL is not comparing the same type of underlying complexity, as DL approaches are vastly more computationally complex.

To visualize each model's training and validation performance, plots were generated that show the validation accuracy and loss for each training epoch over a fixed length of 50 epochs. These plots provide insights into the model's training progress and can be used to identify issues such as overfitting or underfitting. Overfitting occurs when the model performs well on the training data but poorly on the test data. Conversely, underfitting occurs when the model performs poorly on both the training and test data.

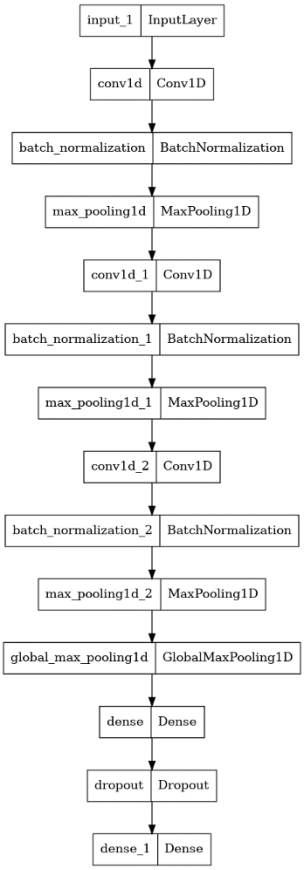
Classification Models Under Study

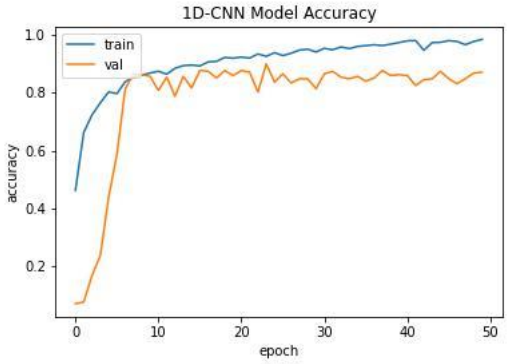
ADT

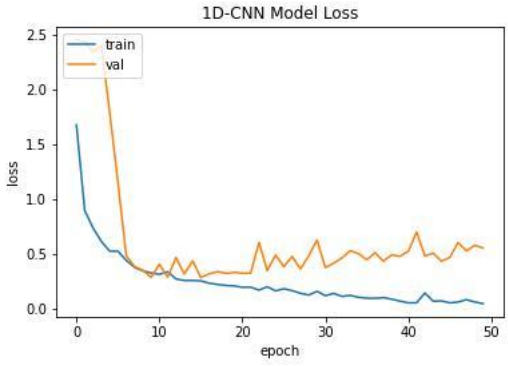
The AdaBoost algorithm is an ensemble-based machine-learning approach that combines multiple weak classifiers to create a strong classifier. In this implementation, the weak classifier is the decision tree algorithm. Multiple decision trees are trained on a subset of the data and iteratively weighted to minimize the classification error rate of the ensemble (Freund, 1997). The AdaBoost algorithm is known for its ability to handle complex data and achieve high accuracy in classification tasks. ADT is distinct from the other models under study as it is not a deep learning algorithm and does not employ artificial neural networks during training. It has been included to provide some comparison beyond the domain of neural network.

1D-CNN

One-dimensional convolutional neural networks are a type of convolutional neural network architecture designed explicitly for processing one-dimensional sequences of data (Y. Kim, 2014), such as those obtained from the RB process. The architecture comprises several convolutional layers that extract local features from the input data and a fully connected layer that aggregates the features and performs the final classification. Using convolutional layers allows the network to learn hierarchical representations of the input data, which can lead to improved performance in classification tasks.

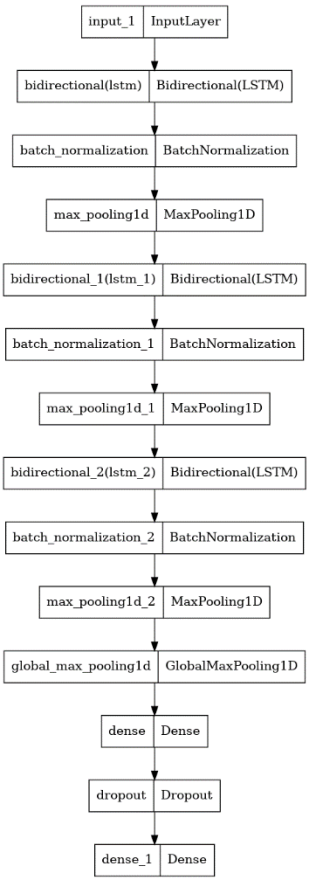


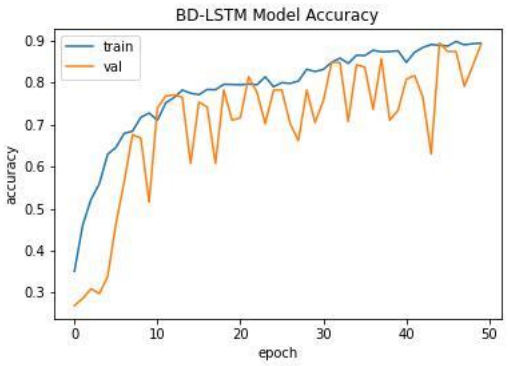


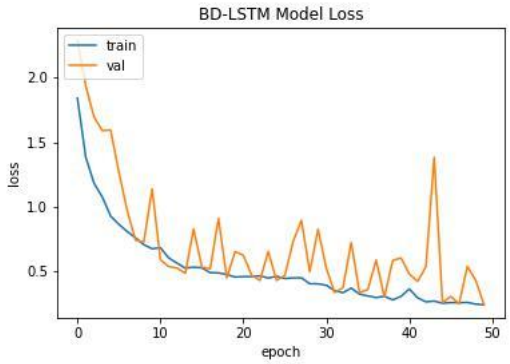


BD-LSTM

Bi-directional long short-term memory is a recurrent neural network that employs forward and backward LSTM to capture long-term dependencies in sequential data. The memory cell of an LSTM is a transitional state where new information can be added, old information can be removed, and intermediate states can be maintained (S. Hochreiter and J. Schmidhuber, 1997). This allows the network to selectively remember or forget information over long sequences and makes it particularly useful for tasks that require modeling of temporal dependencies. The use of bidirectional LSTMs allows the network to consider past and future information when making predictions, which can be particularly useful in time series data.

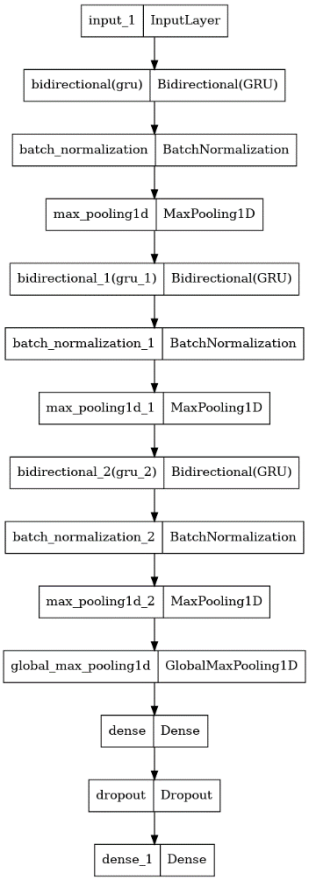


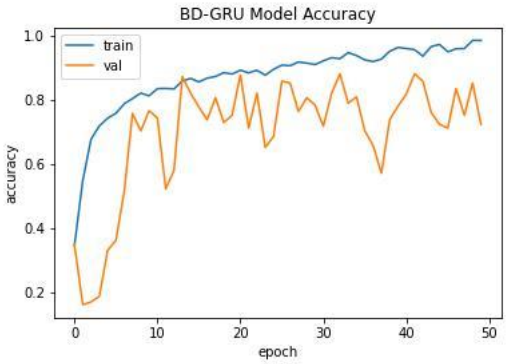


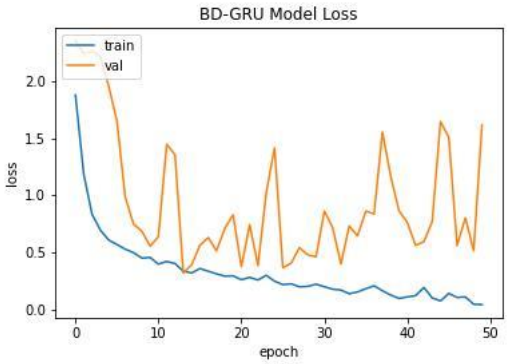


BD-GRU

The Bidirectional Gated Recurrent Unit Network is a variation of the bidirectional RNN architecture, similar to the BD-LSTM, that uses Gated Recurrent Units (GRUs) instead of LSTMs. GRUs are a type of RNN cell that was introduced as a more straightforward and faster alternative to LSTMs while maintaining some of their performance characteristics (J. Chung et al., 2014). Like the LSTM, the GRU has a set of learnable gates that control the flow of information through the cell. However, the GRU has a simpler architecture with fewer parameters, making it easier and faster to train. The BD-GRU architecture in this research project uses two GRU layers, each processing the input data in a different direction (forward and backward) to capture past and future information. The output of the first layer is passed as input to the second layer, allowing the model to learn more complex features and dependencies between the input features.

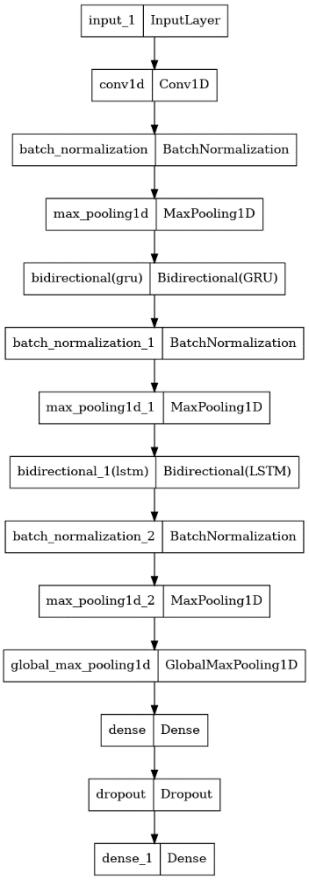


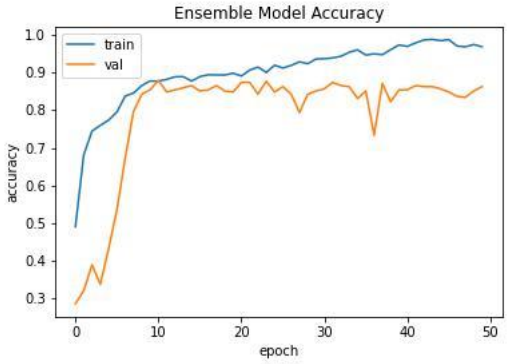


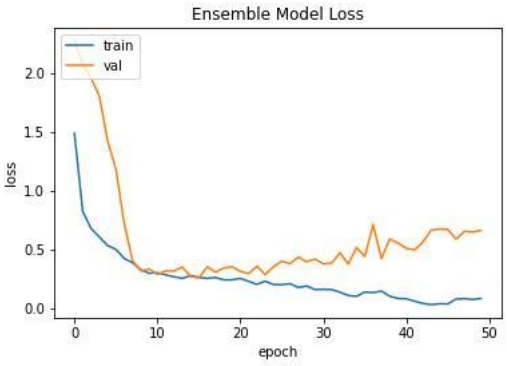


Stacked Ensemble

In addition to the individual models previously discussed, an ensemble model was created by stacking the 1D-CNN, BD-GRU, and BD-LSTM models. This ensemble model aimed to provide perspective on the effectiveness of combined approaches in encrypted traffic classification tasks. The ensemble model leverages the strengths of each model by integrating their predictions, potentially leading to improved overall performance. Combining the outputs of the 1D-CNN, which excels in capturing local patterns, the BD-GRU, adept at capturing long-range dependencies, and the BD-LSTM, proficient in modeling temporal dependencies, the ensemble model aims to capitalize on the complementary properties of these individual models. This approach was added to the study to evaluate the potential benefits of combining different deep-learning models and to explore whether a synergistic effect could be achieved regarding classification performance.







Generic Dense Classifier

All neural network models under study will have a generic dense classifier stack as their output layers. This classifier is a simple but effective approach for stacked neural network classification tasks. It involves passing the output vector from the neural network under study through two fully connected dense layers and a softmax activation function. The dense layers learn weights for each input feature and generate a new feature space passed through the softmax function. The softmax function converts the output of the dense layer into a probability distribution over the possible classes. The class with the highest probability is then assigned the predicted label (Y. LeCun et al., 2015).

Random Forest

Finally, the Random Forest classifier from the original paper will also be evaluated as a baseline model. This machine learning classifier operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes of the individual trees (L. Breiman, 2001). The Random Forest algorithm can handle high-dimensional data and is robust to noisy features. The recomposed bytes will be input to the Random Forest classifier, and its performance will be compared against that of the ADT, 1D-CNN, BD-LSTM, and DB-GRU networks.

Results

This study developed and compared models for classifying network traffic based on TLS handshake payload data to identify the most accurate and efficient model. The baseline was an RF ML model, and subsequent models were evaluated based on training time and overall accuracy. The models were trained on a dataset of network traffic TLS handshake payloads, and their accuracy was evaluated on an unseen test dataset using recall, precision, and F1 metrics. The study provides insights into the effectiveness of different models for traffic classification and can inform the development of future traffic management strategies.

The baseline RF model performed marginally worse than the original paper though it still provided the baseline required for comparison with the other models. An exact explanation as to why the performance of RB-RF was not identical to the original paper will need further research and confirmation from the original authors, who were not available at the time of writing.

Model	Precision	Recall	F1	Accuracy	Training Time
RF	0.895	0.917	0.900	0.880	**1**
ADT	**0.921**	**0.921**	**0.918**	**0.899**	3.6
1D-CNN	0.903	0.908	0.905	0.880	41.5
BD-LSTM	0.892	0.864	0.857	0.865	137.9
BD-GRU	0.891	0.876	0.874	0.860	139.5
Ensemble	0.895	0.897	0.893	0.894	86

Overall, neural network approaches performed worse than the RF baseline across all categories, with the ensemble approach fairing well in accuracy but poorly in other categories. Of the neural networks under study, 1D-CNN performed best in precision, recall, F1, and training time. The ensemble method performs second best regarding precision, recall, f1, and training time. Neural networks were the poorest regarding training time as they are highly complex and require specialized computing resources to calculate the model parameters (Appendix 1). Overall, a significant result is that computational complexity does not equate to model performance.

The value of ML approaches vs. DL is especially apparent when the results of the ADT model are considered. ADT outperformed the Random Forest baseline in all categories other than training time, where it trailed by a negligible amount in terms of absolute time spent to fit the model to the training data.

Discussion

This study highlights the importance of carefully selecting and optimizing the model architecture and parameters for traffic classification based on TLS handshake payload data. Among the neural network models investigated, the 1D-CNN outperformed others regarding precision, recall, F1, and training time, suggesting that it is well-suited for encrypted traffic classification tasks. This finding aligns with the growing body of literature indicating that simpler models can often achieve comparable or superior performance to more complex models, especially when the task is not highly complex or dynamic (C. Zhang et al., 2017).

The ADT model emerged as the top-performing model, outperforming the RF baseline and all neural network models in terms of accuracy, precision, recall, and F1, except for training time, where it trailed the RF model by a negligible amount. This result demonstrates the importance of considering simpler models, such as decision trees and ensemble methods, alongside or complimenting more complex deep learning models, particularly in constrained resource operating environments. Furthermore, the study underscores that

computational complexity alone does not guarantee better performance, as evidenced by the simpler ADT model outperforming more complex neural network models.

Future research could explore the potential benefits of hybrid approaches combining feature engineering and machine learning techniques to improve traffic classification performance on ECH-protected traffic. Integrating various feature extraction techniques, such as PCA or autoencoders, with classification models could develop more robust and efficient classifiers. A more significant body of training data gathered as close as possible to study time is an ongoing challenge in this space, particularly when considering the emerging ubiquity of the ECH extension of the TSL 1.3 protocol. Further, as more extensions become adapted for ECH inclusion by application layer protocols and less information can be composed into a feature vector, more novel approaches or techniques will need to be developed.

Conclusion

In conclusion, the findings of this study have implications for the development of future traffic management strategies. By identifying the most accurate and efficient models and feature engineering methods for traffic classification, network administrators can develop the means to effectively monitor and control network traffic, prevent congestion, and ensure optimal network performance in highly encrypted network environments. This study demonstrates the potential of machine learning approaches in traffic classification. The ADT model is promising as a more straightforward yet highly effective alternative to more complex neural network models. Future research in this field could focus on exploring alternative model architectures and optimization techniques for traffic classification based on TLS handshake payload data, investigating the feasibility of implementing these models in production network environments and assessing their scalability and adaptability to evolving network conditions using real-time traffic data.

References

D. Shamsimukhametov, A. Kurapov, M. Liubogoshchev and E. Khorov, "Is Encrypted ClientHello a Challenge for Traffic Classification?," in IEEE Access, vol. 10, pp. 77883-77897, 2022, doi: 10.1109/ACCESS.2022.3191431.

W. Wang, M. Zhu, J. Wang, X. Zeng and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China, 2017, pp. 43-48, doi: 10.1109/ISI.2017.8004872.

Habibi Lashkari, G. Draper Gil, M. Mamun, and A. A. Ghorbani, "Characterization of Encrypted and VPN Traffic Using Time-Related Features," in Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP), Italy, Feb. 2016, pp. 407-414, doi: 10.5220/0005740704070414.

H. Lim, J.-B. Kim, J.-S. Heo, K. Kim, Y.-G. Hong and Y.-H. Han, "Packet-based Network Traffic Classification Using Deep Learning," 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2019, pp. 46-51, doi: 10.1109/ICAIIC.2019.8669045.

Lu, N. Luktarhan, C. Ding and W. Zhang, "ICLSTM: Encrypted traffic service identification based on inception-LSTM neural network," Symmetry, vol. 13, no. 6, p. 1080, Jun. 2021, doi: 10.3390/sym13061080.

J. Cheng, Y. Wu, Y. E, J. You, T. Li, H. Li, and J. Ge, "MATEC: A Lightweight Neural Network for Online Encrypted Traffic Classification," Comput. Netw., vol. 199, no. C, Nov. 2021, Art. no. 108472, doi: 10.1016/j.comnet.2021.108472.

Z. Shi, N. Luktarhan, Y. Song and G. Tian, "BFCN: A Novel Classification Method of Encrypted Traffic Based on BERT and CNN," Electronics, vol. 12, no. 3, p. 516, 2023. doi: 10.3390/electronics12030516.

Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," Journal of Computer and System Sciences, vol. 55, no. 1, pp. 119-139, 1997, doi: 10.1006/jcss.1997.1504.

Y. Kim, "Convolutional Neural Networks for Sentence Classification," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, Oct. 2014, pp. 1746-1751, doi: 10.3115/v1/D14-1181.

S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

J. Chung, C. Gulcehre, K. Cho and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," in ArXiv, Dec. 2014. [Online]. Available: https://arxiv.org/abs/1412.3555

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. doi: 10.1038/nature14539

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32. doi: 10.1023/A:1010933404324.

Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in Proceedings of the 5th International Conference on Learning Representations (ICLR), 2017, arXiv:1611.03530.

K. Brooks, "TLS Traffic Analysis," GitHub. [Online]. Available: https://github.com/kierankyllo/tls_traffic_analysis.

Wireless Networks Lab, "Novel QoS-Aware TLS Dataset for Encrypted Traffic Classification," Institute for Information Transmission Problems, Moscow, Russia. [Online]. Available: http://wireless.iitp.ru/qos-tls-dataset-of-enc-traffic/.

Appendices

Appendix 1 – Results Data

Random Forest	Precision	Recall	F1
Apple Music	0.8545	1	0.9216
Web	0.6452	0.9231	0.7595
Kinopoisk	1	1	1
Facebook Live	1	1	1
YouTube Live	0.7955	1	0.8861
Netflix	0.9744	0.8444	0.9048
Prime Video	0.9437	0.9853	0.964
Sound Cloud	0.8923	0.9831	0.9355
Spotify	0.8571	0.7826	0.8182
Vimeo	0.9602	0.7598	0.8484
Yandex Music	0.8462	0.7253	0.7811
YouTube	0.9688	1	0.9841
μ	**0.8948**	**0.9170**	**0.9003**
Accuracy			**0.8797**
Training Time			1

ADT	Precision	Recall	F1
Apple Music	0.9038	1	0.9495
Web	0.8197	0.7692	0.7937
Kinopoisk	1	1	1
Facebook Live	1	1	1
YouTube Live	0.8125	0.9905	0.8927
Netflix	0.9111	0.9111	0.9111
Prime Video	0.9706	0.9706	0.9706
Sound Cloud	0.9344	0.9661	0.95
Spotify	1	0.7391	0.85
Vimeo	0.9321	0.811	0.8674
Yandex Music	0.7864	0.8901	0.8351
YouTube	0.9841	1	0.992
μ	**0.9212**	**0.9206**	**0.9177**
Accuracy			**0.8992**
Training Time			3.6

1D-CNN	Precision	Recall	F1
Apple Music	0.8571	0.8571	0.8571
Web	0.8889	0.8	0.8421
Kinopoisk	1	1	1
Facebook Live	1	1	1
YouTube Live	0.7955	0.8974	0.8434
Netflix	0.8947	0.9444	0.9189
Prime Video	0.9375	1	0.9677
Sound Cloud	0.9545	0.9545	0.9545
Spotify	0.8333	0.8333	0.8333
Vimeo	0.8315	0.8043	0.8177
Yandex Music	0.8462	0.8049	0.825
YouTube	1	1	1
μ	**0.9033**	**0.9080**	**0.9050**
Accuracy			**0.8797**
Training Time			41.5

BD-LSTM	Precision	Recall	F1
Apple Music	0.6452	0.9524	0.7692
Web	0.6829	0.9333	0.7887
Kinopoisk	1	1	1
Facebook Live	1	1	1
YouTube Live	0.8298	1	0.907
Netflix	0.8947	0.9444	0.9189
Prime Video	0.9375	1	0.9677
Sound Cloud	0.9412	0.7273	0.8205
Spotify	1	0.333	0.5
Vimeo	0.9722	0.7609	0.8537
Yandex Music	0.7949	0.7561	0.775
YouTube	1	0.9643	0.9818
μ	**0.8915**	**0.8643**	**0.8569**
Accuracy			**0.8653**
Training Time			137.9

BD-GRU	Precision	Recall	F1
Apple Music	0.6786	0.9048	0.7755
Web	0.8214	0.7667	0.7931
Kinopoisk	1	1	1
Facebook Live	1	1	1
YouTube Live	0.8333	0.7692	0.8
Netflix	0.9444	0.9444	0.9444
Prime Video	0.9375	1	0.9677
Sound Cloud	0.8462	1	0.9167
Spotify	1	0.5	0.6667
Vimeo	0.8452	0.7717	0.8068
Yandex Music	0.814	0.8537	0.8333
YouTube	0.9655	1	0.9825
μ	**0.8905**	**0.8759**	**0.8739**
Accuracy			**0.8596**
Training Time			139.5

Ensemble	Precision	Recall	F1
Apple Music	0.9048	0.9048	0.9048
Web	0.7576	0.8333	0.7937
Kinopoisk	1	1	1
Facebook Live	1	1	1
YouTube Live	0.8261	0.9744	0.8941
Netflix	0.9444	0.9444	0.9444
Prime Video	0.9375	1	0.9677
Sound Cloud	0.875	0.9545	0.913
Spotify	0.75	0.5	0.6
Vimeo	0.9487	0.8043	0.8706
Yandex Music	0.8333	0.8537	0.8434
YouTube	0.9655	1	0.9825
μ	**0.8952**	**0.8975**	**0.8929**
Accuracy			**0.894**
Training Time			86