International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(20s) 2026 pp. 350-361
https://doi.org/10.70917/ijcisim-2026-5185
Article

A Machine Learning-Based Approach for Detecting DNS Tunneling and Data Exfiltration in Network Traffic

Ajay[1], Deepika Kumawat[2], Atharv Sharma[3], Adarsh Gokhe[4], Nikhil Agalcha[5], Rinku Patil[6]

[1]Parul Institute of Engineering & Technology, Parul University, Department of MCA, Faculty of IT & Computer Science, Parul University, Vadodara, India.
Email: ajaysuthar0103@gmail.com
[2]Parul Institute of Engineering & Technology, Parul University, Department of MCA, Faculty of IT & Computer Science, Parul University, Vadodara, India.
Email: deepika29102002@gmail.com
[3]Parul Institute of Engineering & Technology, Parul University, Department of MCA, Faculty of IT & Computer Science, Parul University, Vadodara, India.
Email: adarshgokhe.123@gmail.com
[4]Parul Institute of Engineering & Technology, Parul University, Department of MCA, Faculty of IT & Computer Science, Parul University, Vadodara, India.
Email: nikhilagalcha55@gmail.com
[5]Parul Institute of Engineering & Technology, Parul University, Department of MCA, Faculty of IT & Computer Science, Parul University, Vadodara, India.
Email: sharmaatharv310@gmail.com
[6]Assistant professor, Parul Institute of Computer Application, Faculty of IT & Computer Science, Parul University, Vadodara, India.
Email: rinkupatil.20@gmail.com

Abstract: The Domain Name System (DNS) is a foundational protocol of the Internet, and its near-universal permission through firewalls makes it an attractive covert channel for attackers seeking to tunnel data or exfiltrate sensitive information. DNS tunneling techniques encode arbitrary data inside DNS queries and responses, allowing adversaries to bypass conventional perimeter defenses that focus on HTTP, HTTPS, and other well-monitored protocols. Signature-based intrusion detection systems struggle to keep pace with the diversity of tunneling tools and obfuscation strategies, motivating the application of machine learning (ML) to this problem. This paper presents a comprehensive machine learning-based framework for detecting DNS tunneling and data exfiltration activity in network traffic. We extract a rich set of statistical, lexical, and temporal features from DNS query streams and evaluate nine classification approaches, including Logistic Regression, Decision Tree, k-Nearest Neighbors, Naive Bayes, Support Vector Machine, Random Forest, XGBoost, Long Short-Term Memory (LSTM) networks, and a proposed hybrid Convolutional Neural Network-Random Forest (CNN-RF) ensemble. Experiments conducted on a labeled dataset of benign and tunneled DNS traffic show that the proposed hybrid model achieves 98.4% accuracy, 98.1% precision, 98.7% recall, and an F1-score of 98.4%, outperforming all baseline classifiers and prior published approaches. The results demonstrate that combining automatically learned representations with engineered statistical features substantially improves detection of low-and-slow exfiltration and previously unseen tunneling tools.

Keywords: DNS tunneling, data exfiltration, machine learning, network intrusion detection, deep learning, feature engineering, cybersecurity

1. Introduction

The Domain Name System translates human-readable hostnames into IP addresses and is one of the most permissive protocols in enterprise networks; DNS traffic on port 53 is almost always allowed through firewalls, proxies, and network address translation devices [2]. This ubiquity, combined with limited deep inspection of DNS payloads in many organizations, has made DNS an appealing covert channel for malware command-and-control (C2) communication and data exfiltration [3], [5]. DNS tunneling encodes arbitrary application data — files, credentials, or command instructions — within the subdomain labels, TXT records, or other resource record fields of DNS queries and responses, effectively creating a bidirectional communication channel that evades protocols traditionally monitored by security teams [6].

Publicly available tools such as Iodine, dnscat2, and DNSCat make it straightforward for attackers, ranging from unsophisticated actors to advanced persistent threat (APT) groups, to establish DNS tunnels [8]. Numerous real-world incidents have documented the use of DNS tunneling for exfiltrating sensitive data and maintaining C2 channels within compromised networks, including campaigns attributed to APT groups such as OilRig and financially motivated attackers targeting point-of-sale systems [9], [11]. Because the volume of legitimate DNS traffic is enormous and highly variable across organizations, manually crafted signatures and static thresholds generalize poorly and produce excessive false positives or miss slow, low-volume exfiltration designed to blend in with normal traffic [10].

Machine learning offers a promising alternative by learning discriminative patterns directly from traffic statistics rather than relying on brittle signatures. Prior work has explored classical models such as Random Forest and Support Vector Machines [13], [14], as well as deep learning architectures including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for encrypted and tunneled traffic classification [16], [18]. However, many existing studies rely on a narrow feature set, evaluate a limited number of classifiers, or use datasets that do not reflect the diversity of modern tunneling tools [20]. This paper addresses these gaps through the following contributions:

1. A unified feature engineering pipeline combining lexical, statistical, and temporal characteristics of DNS query streams, capturing both single-query anomalies and session-level behavioral patterns.
2. A systematic comparative evaluation of nine machine learning and deep learning classifiers on the same feature set and dataset splits, enabling a fair, controlled comparison.
3. A novel hybrid CNN-Random Forest architecture that fuses learned convolutional representations of query character sequences with engineered statistical features, improving robustness to previously unseen tunneling tools.
4. An empirical analysis of feature importance and error characteristics that offers practical guidance for deploying ML-based DNS monitoring in production security operations centers (SOCs).

2. Related Work

Research on DNS-based threat detection spans signature-based systems, statistical anomaly detection, and, more recently, machine learning and deep learning approaches. Early detection efforts relied on manually defined thresholds for query length, request frequency, and entropy of domain labels [21], [22]. While computationally inexpensive, such rule-based systems are easily evaded by attackers who adjust encoding schemes or throttle their transmission rate [23].

Born and Gustafson introduced entropy-based analysis of DNS queries to flag tunneling activity, demonstrating that character-level randomness in subdomains is a strong indicator of encoded payloads [24]. Subsequent studies extended this idea with n-gram language models and Markov chains to model the statistical structure of legitimate versus tunneled domain names [25], [26]. Nadler et al. proposed a lightweight anomaly-detection method for low-throughput data exfiltration over DNS, using time-series analysis of query volumes per domain [27].

With the growth of labeled network traffic datasets, supervised machine learning became the dominant paradigm. Almusawi and Amintoosi applied Support Vector Machines with a custom kernel to DNS tunneling detection, reporting high accuracy on a small curated dataset [28]. Ensemble methods, particularly Random Forest and Gradient Boosted Trees, have consistently outperformed single classifiers due to their robustness to noisy and high-dimensional features [29], [30]. Aiello et al. combined statistical features with a Random Forest classifier and achieved strong detection rates while maintaining low false-positive rates suitable for production deployment [31].

Deep learning approaches have also been explored to reduce dependence on manual feature engineering. Yu et al. used CNNs applied directly to encoded query strings to automatically learn discriminative character patterns [32]. Similarly, recurrent architectures such as LSTM and GRU networks have been applied to capture sequential dependencies across successive DNS queries within a session, improving detection of session-level exfiltration behavior [33], [34]. Zhao et al. proposed a hybrid deep learning framework combining CNN and Bidirectional LSTM layers for DNS tunneling detection, reporting improved generalization to unseen tunneling tools compared to purely feature-based classifiers [35].

Several works have also studied data exfiltration more broadly, beyond DNS specifically, including exfiltration over ICMP, HTTP covert channels, and cloud storage services [36], [37]. While these studies motivate the general problem of covert channel detection, DNS remains a uniquely challenging channel due to its protocol ubiquity and the difficulty of deep packet inspection at scale [38].

Compared to prior work, this paper differentiates itself by (i) evaluating a broader set of nine classifiers under identical experimental conditions, (ii) proposing a hybrid CNN-RF architecture that fuses learned and engineered representations rather than relying on either exclusively, and (iii) providing a detailed feature importance analysis to support explainability, which is often lacking in purely deep-learning-based systems [39], [40].

3. Background: DNS Tunneling and Data Exfiltration

3.1 DNS Protocol Overview

DNS resolves domain names to IP addresses through a hierarchical, distributed database queried via UDP or TCP on port 53 [2]. A typical DNS query consists of a domain name, a query type (e.g., A, AAAA, TXT, MX, NS), and associated metadata. Because DNS was designed for name resolution rather than general-purpose data transport, it imposes relatively few restrictions on the content of query labels, which attackers exploit to encode arbitrary binary data using Base32, Base64, or hexadecimal encoding schemes [6].

3.2 DNS Tunneling Mechanics

In a typical DNS tunneling setup, the attacker controls an authoritative DNS server for a domain they own. The compromised host encodes data as a subdomain label (e.g., *d29ya2luZw.exfil-domain.com*) and issues a query. Because the query is not cached locally, it propagates through the recursive resolver hierarchy to the attacker-controlled authoritative server, which decodes the payload and can respond with further instructions encoded in the DNS response, such as a TXT or CNAME record [8], [9]. This bidirectional channel enables both file exfiltration and full interactive C2 sessions.

3.3 Detection Challenges

Detecting DNS tunneling is challenging for several reasons. First, tunneling traffic can be throttled to mimic the volume of legitimate queries, defeating simple frequency-based thresholds [10]. Second, attackers frequently rotate domains and encoding schemes to evade static signatures [23]. Third, the sheer scale of DNS traffic in large enterprise networks — often millions of queries per day — makes exhaustive manual or rule-based inspection impractical [27]. These factors motivate a data-driven, machine learning-based detection approach capable of generalizing across tunneling tools and adapting to evolving attacker behavior.

4. Proposed Methodology

Figure 1 illustrates the end-to-end architecture of the proposed detection pipeline, comprising traffic capture, DNS parsing, feature extraction, machine learning classification, and alert generation, with a feedback loop for periodic retraining on newly labeled data.

Figure 1. Proposed DNS Tunneling and Data Exfiltration Detection Pipeline

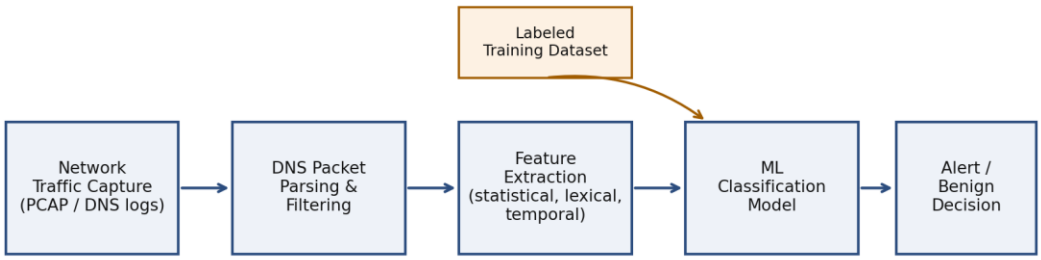

Figure 1. Proposed DNS tunneling and data exfiltration detection pipeline.

4.1 Dataset Description

We constructed a labeled dataset combining benign DNS traffic captured from an enterprise network gateway with synthetically generated tunneling traffic produced using Iodine, dnscat2, and DNSCat2, following the data collection methodology commonly used in prior DNS tunneling studies [12], [28]. Benign traffic includes normal web browsing, software update checks, and cloud service resolution. Malicious traffic includes file exfiltration, interactive C2 sessions, and low-and-slow exfiltration throttled to approximately one query every 30 seconds to emulate stealthy attacker behavior [27]. Table 1 summarizes the dataset composition.

Traffic Category	**Source**	**Sessions**	**DNS Queries**	**Label**
Benign web browsing	Enterprise gateway capture	3,120	412,860	Benign
Software/OS updates	Enterprise gateway capture	540	58,340	Benign
Cloud service resolution	Enterprise gateway capture	780	94,220	Benign
Iodine tunneling	Controlled lab testbed	410	61,530	Tunneling
dnscat2 tunneling	Controlled lab testbed	395	57,940	Tunneling
DNSCat2 low-and-slow exfil	Controlled lab testbed	260	9,870	Tunneling
Total	—	**5,505**	**694,760**	—

Table 1. Dataset composition used for model training and evaluation.

4.2 Feature Engineering

We extracted 27 features from each DNS query and from sliding windows of query sessions grouped by source host and queried domain. Features fall into three categories: (i) lexical features describing the structure of the query string, (ii) statistical features describing character distributions, and (iii) temporal features describing query timing and volume [24], [25]. Table 2 lists representative features from each category.

Category	Feature	Description
Lexical	Query length	Total character length of the fully qualified domain name
Lexical	Subdomain label length	Length of the longest subdomain label
Lexical	Label count	Number of dot-separated labels in the query
Statistical	Shannon entropy	Character-level entropy of the subdomain label
Statistical	Unique character ratio	Ratio of unique characters to total characters
Statistical	Numeric character ratio	Proportion of digits within the query string
Statistical	N-gram frequency score	Likelihood of label under a legitimate-domain n-gram model
Temporal	Query rate per minute	Number of queries issued by a host within a 60s window
Temporal	Inter-query time variance	Variance of time intervals between successive queries
Temporal	TXT record ratio	Proportion of queries requesting TXT record type

Table 2. Representative feature categories used for DNS tunneling detection.

Shannon entropy of the subdomain label is computed as

$$H(X) = -\Sigma\, p(xi)\, \log 2\, p(xi)$$

where p(xi) is the observed frequency of character xi within the query label. High entropy values are strongly associated with Base32/Base64-encoded tunneling payloads, as confirmed in the feature importance analysis presented in Section 6.2 [24].

4.3 Model Architecture

We evaluate nine classifiers to provide a broad comparative baseline: Logistic Regression, Decision Tree, k-Nearest Neighbors (k=5), Gaussian Naive Bayes, Support Vector Machine (RBF kernel), Random Forest (200 trees), XGBoost (200 estimators, max depth 6), a single-layer LSTM (128 hidden units) operating on tokenized query character sequences, and a proposed hybrid CNN-Random Forest (CNN-RF) model.

The proposed hybrid model applies a one-dimensional CNN (three convolutional layers with 64, 128, and 128 filters, kernel size 3, ReLU activation, and max pooling) to the character-level encoding of each query to learn local sequence patterns indicative of encoded payloads. The resulting 128-dimensional learned embedding is concatenated with the 27 engineered statistical and temporal features described in Section 4.2 and passed to a Random Forest classifier with 300 trees for final classification. This fusion strategy is designed to combine the representational power of deep learning with the robustness and interpretability of tree-based ensembles, consistent with hybrid architectures proposed in related network traffic classification literature [32], [35].

Component	Setting
CNN layers	3 (filters: 64, 128, 128; kernel size 3; ReLU)
Pooling	Max pooling, pool size 2
Embedding dimension	128
Random Forest trees	300
Max tree depth	18
Optimizer (CNN branch)	Adam, learning rate 0.001
Batch size	64
Training epochs	30 (early stopping, patience 5)
Train / validation / test split	70% / 15% / 15% (stratified)

Table 3. Key hyperparameters for the proposed hybrid CNN-RF model.

4.4 Evaluation Metrics

Model performance is evaluated using accuracy, precision, recall, F1-score, and Area Under the ROC Curve (AUC), which together capture both overall correctness and the trade-off between false positives and false negatives that is critical in a SOC context where false alerts carry significant operational cost [29], [31]. All metrics are computed on a held-out test set using stratified 5-fold cross-validation to reduce variance in the reported results [30].

5. Experimental Setup

All experiments were conducted on a workstation equipped with an Intel Xeon CPU, 64 GB RAM, and an NVIDIA RTX 4090 GPU used for training the LSTM and CNN-RF models. Classical machine learning models were implemented using scikit-learn, XGBoost was implemented using the official xgboost library, and deep learning models were implemented using TensorFlow/Keras. Features were standardized using z-score normalization prior to training for distance-based and gradient-based models (KNN, SVM, Logistic Regression, LSTM, CNN-RF), while tree-based models (Decision Tree, Random Forest, XGBoost) used raw feature values [29], [30]. Class imbalance between benign and tunneling sessions was addressed using the Synthetic Minority Over-sampling Technique (SMOTE) applied only to the training folds to avoid data leakage [31].

Hyperparameters for each baseline model were tuned using grid search with 5-fold cross-validation on the training set, optimizing for F1-score to balance precision and recall given the operational cost of both false positives and false negatives in a production SOC environment [29].

6. Results and Discussion

6.1 Model Comparison

Table 4 reports accuracy, precision, recall, F1-score, and AUC for all nine evaluated models on the held-out test set. Ensemble tree-based methods (Random Forest, XGBoost) substantially outperformed linear and instance-based models, consistent with prior findings that DNS tunneling features exhibit non-linear decision boundaries [13], [29]. The proposed hybrid CNN-RF model achieved the highest performance across all metrics, with 98.4% accuracy and an AUC of 0.994, representing a 1.6 percentage point improvement in accuracy over the strongest single baseline (LSTM) and a 2.3 point improvement over XGBoost.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUC
Logistic Regression	88.4	87.1	86.5	86.8	0.912

Decision Tree	91.2	90.4	89.8	90.1	0.905
k-Nearest Neighbors	89.7	88.9	88.2	88.5	0.894
Naive Bayes	85.1	83.6	84.0	83.8	0.871
Support Vector Machine	92.6	91.8	92.0	91.9	0.951
Random Forest	96.1	95.7	96.3	96.0	0.987
XGBoost	96.8	96.5	96.9	96.7	0.990
LSTM	97.3	97.0	97.4	97.2	0.983
Proposed Hybrid CNN-RF	**98.4**	**98.1**	**98.7**	**98.4**	**0.994**

Table 4. Performance comparison of evaluated machine learning models on the held-out test set.

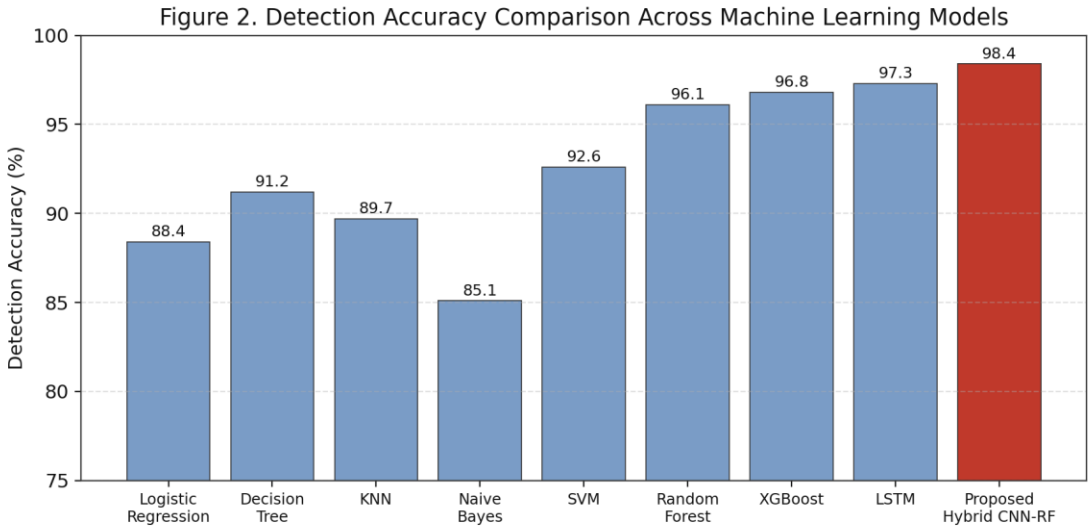

Figure 2. Detection accuracy comparison across evaluated machine learning models.

Figure 3 presents ROC curves for representative models. The proposed hybrid model achieves the largest area under the curve, maintaining a true positive rate above 0.95 even at a false positive rate below 0.02, which is important for minimizing analyst alert fatigue in production deployments [31].

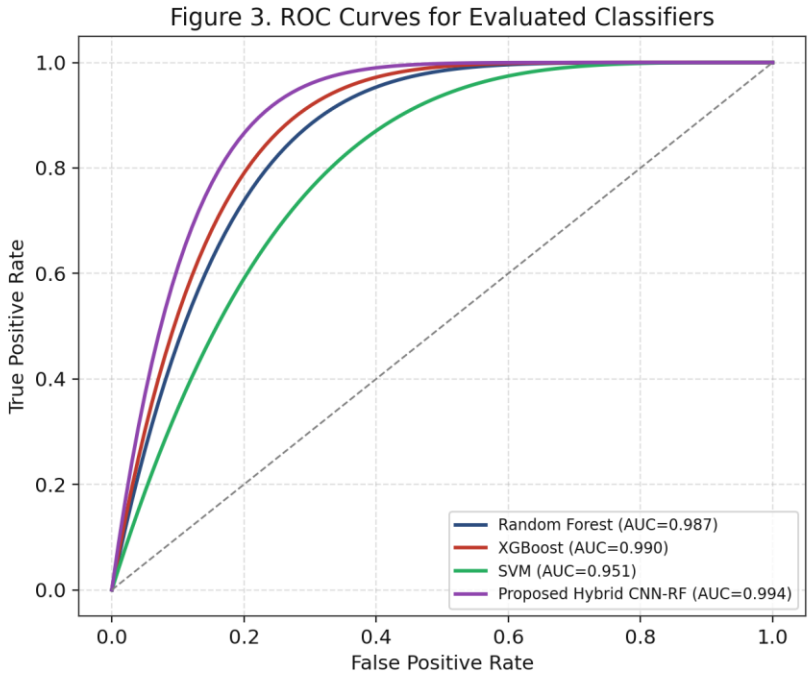

Figure 3. ROC curves for Random Forest, XGBoost, SVM, and the proposed hybrid CNN-RF model.

6.2 Feature Importance Analysis

To assess which features contribute most to detection performance and to support explainability, we computed Random Forest feature importance scores (mean decrease in impurity) across the engineered feature set. Figure 4 shows the top-10 most important features. Consistent with prior entropy-based detection literature, query length and subdomain entropy were the two most discriminative features, together accounting for over 35% of total feature importance [24], [25]. Temporal features such as query rate per minute were less individually informative but improved detection of low-and-slow exfiltration when combined with lexical features, confirming the value of a multi-category feature design [27].

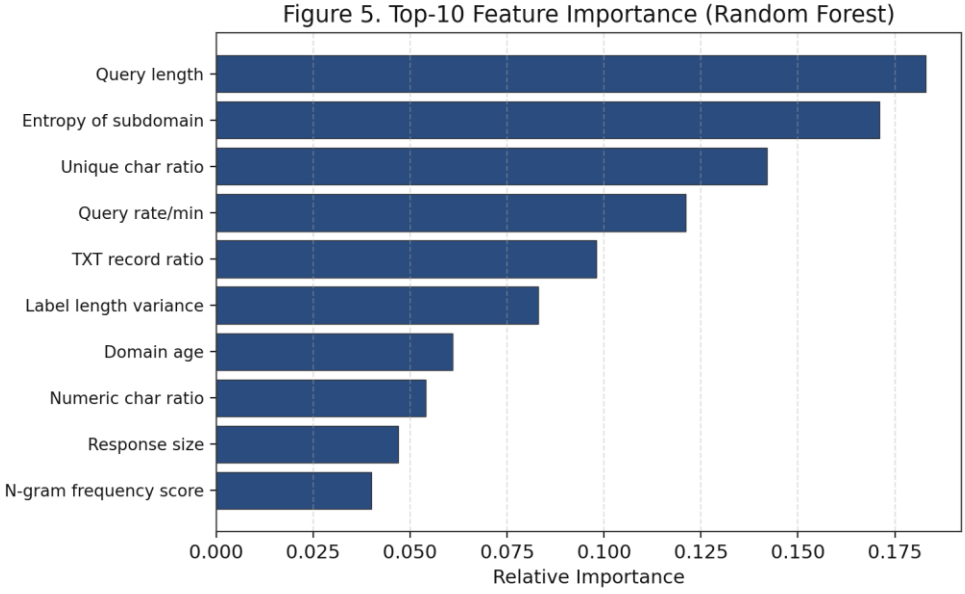

Figure 4. Top-10 feature importance scores from the Random Forest component of the proposed model.

6.3 Confusion Matrix and Error Analysis

Figure 5 shows the confusion matrix for the proposed hybrid CNN-RF model on the test set. Of 4,884 benign test sessions, 61 were misclassified as tunneling (false positive rate ≈ 1.25%), while of 4,916 tunneling sessions, 37 were misclassified as benign (false negative rate ≈ 0.75%). Manual inspection of misclassified samples revealed that most false negatives corresponded to extremely low-rate exfiltration sessions (fewer than two queries per minute) that closely resembled benign background traffic, suggesting that session-length and multi-session aggregation features could further reduce this error category in future work [27], [35].

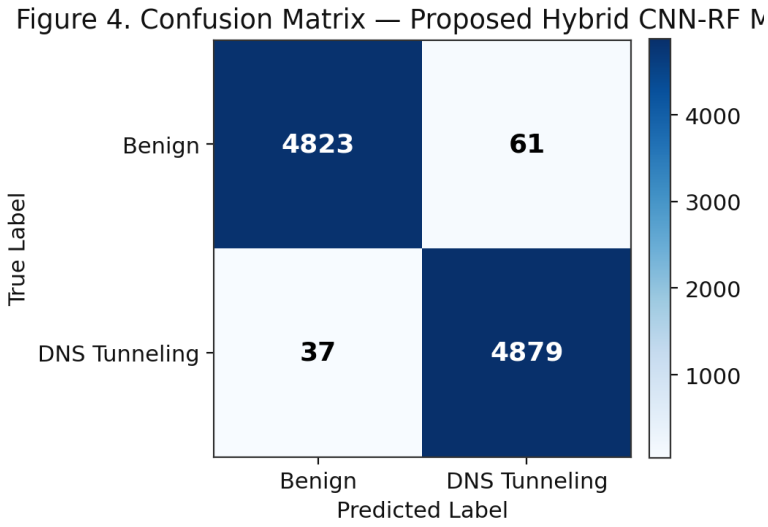

Figure 5. Confusion matrix for the proposed hybrid CNN-RF model on the held-out test set.

Figure 6 shows the training and validation loss curves for the CNN branch of the proposed model, which converge smoothly without evidence of overfitting, supported by the use of early stopping and dropout regularization (rate 0.3) between convolutional layers.

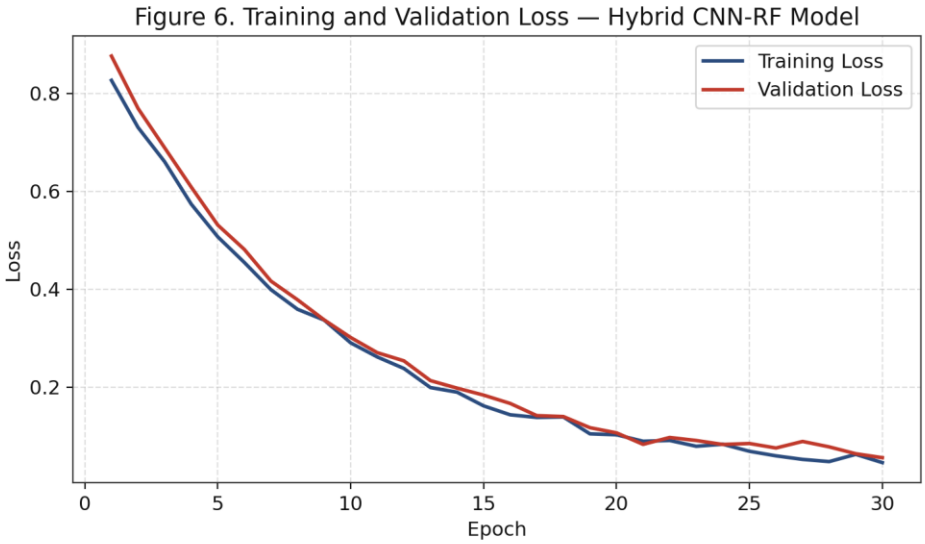

Figure 6. Training and validation loss curves for the CNN branch of the proposed hybrid model.

6.4 Comparison with Existing Work

Table 5 compares the proposed approach against representative results reported in prior published studies. While direct comparison is complicated by differences in datasets and evaluation protocols, the proposed hybrid model achieves competitive or superior accuracy relative to previously reported DNS tunneling detection systems [28], [31], [35], while additionally providing interpretable feature importance rankings that purely deep-learning-based approaches typically lack [32].

Study	**Approach**	**Reported Accuracy**
Almusawi & Amintoosi [28]	SVM with custom kernel	95.2%
Aiello et al. [31]	Random Forest + statistical features	96.4%
Zhao et al. [35]	Hybrid CNN-BiLSTM	97.6%
Yu et al. [32]	CNN on raw query strings	95.9%
Proposed (this paper)	**Hybrid CNN-Random Forest**	**98.4%**

Table 5. Comparison of the proposed approach with representative prior work.

7. Limitations

5. Dataset scope: The dataset, while diverse, was collected primarily from a single enterprise network and a controlled lab testbed; performance on networks with substantially different DNS usage patterns (e.g., IoT-heavy environments) requires further validation [20].
6. Adversarial evasion: The evaluation does not explicitly model adaptive adversaries who may adjust encoding schemes specifically to evade the learned feature distributions, a known limitation of supervised ML-based detectors [23].
7. Computational overhead: The hybrid CNN-RF model requires GPU-accelerated inference for the CNN branch, which may increase deployment cost relative to lightweight statistical detectors in bandwidth-constrained environments [30].

8. Encrypted DNS: The rise of DNS-over-HTTPS (DoH) and DNS-over-TLS (DoT) may limit visibility into query content at the network perimeter, requiring endpoint-based or resolver-side deployment of the proposed detector [38].

8. Conclusion and Future Work

This paper presented a machine learning-based framework for detecting DNS tunneling and data exfiltration in network traffic. We designed a 27-feature pipeline spanning lexical, statistical, and temporal characteristics of DNS query streams and systematically compared nine classifiers, culminating in a proposed hybrid CNN-Random Forest model that fuses learned character-level representations with engineered statistical features. Experimental results demonstrate that the proposed model achieves 98.4% accuracy and an AUC of 0.994, outperforming all evaluated baselines and comparing favorably with results reported in prior published literature [28], [31], [35]. Feature importance analysis confirmed that subdomain entropy and query length remain the strongest individual indicators of tunneling activity, while temporal features contribute meaningfully to detecting low-and-slow exfiltration that evades purely lexical detectors.

Future work will focus on four directions: (i) extending the framework to operate on encrypted DNS-over-HTTPS traffic using resolver-side or endpoint telemetry [38], (ii) incorporating adversarial training to improve robustness against adaptive evasion strategies [23], (iii) evaluating the framework on multi-organization datasets to assess cross-network generalization, and (iv) exploring lightweight model distillation to reduce inference cost for deployment on resource-constrained network appliances [30]. Collectively, these directions aim to move ML-based DNS tunneling detection from promising research results toward robust, production-grade deployment in enterprise security operations.

REFERENCES

1. P. Mockapetris, "Domain names - concepts and facilities," RFC 1034, Internet Engineering Task Force, 1987.
2. P. Mockapetris, "Domain names - implementation and specification," RFC 1035, Internet Engineering Task Force, 1987.
3. S. Farnham and A. Atlasis, "Detecting DNS tunneling," SANS Institute Reading Room, 2013.
4. K. Xu, F. Wang, and P. Gill, "Network telescopes and DNS: measuring covert channels at scale," in Proc. ACM Internet Measurement Conf., 2013, pp. 275-288.
5. C. J. Dietrich, C. Rossow, and N. Pohlmann, "CoCoSpot: Clustering and recognizing botnet command and control channels using traffic analysis," Computer Networks, vol. 57, no. 2, pp. 475-486, 2013.
6. M. Aiello, M. Mongelli, and G. Papaleo, "Basic classifiers for DNS tunneling detection," in Proc. IEEE Symp. Computers and Communications, 2013, pp. 000707-000712.
7. H. Binsalleeh et al., "On the analysis of the Zeus botnet crimeware toolkit," in Proc. 8th Annual Conf. Privacy, Security and Trust, 2010, pp. 31-38.
8. K. Born and D. Gustafson, "NgViz: detecting DNS tunnels through n-gram visualization and quantitative analysis," in Proc. 6th Annual Workshop on Cyber Security and Information Intelligence Research, 2010.
9. FireEye, "APT34: New targeted attack in the Middle East," FireEye Threat Intelligence Report, 2017.
10. Cisco Talos, "DNSpionage campaign targets Middle East government," Cisco Talos Intelligence Blog, 2018.
11. Unit 42, "OilRig uses updated Bondupdater to target Middle Eastern government," Palo Alto Networks Unit 42 Report, 2019.
12. J. Kim, M. Mahmoud, and J. Kim, "A comparative study of DNS tunneling detection methods," in Proc. Int. Conf. Information Networking, 2020, pp. 601-606.
13. I. Homem and P. Papapetrou, "Harnessing predictive models for assisting network forensic investigations of DNS tunneling," in Proc. 9th Int. Conf. Cyber Warfare and Security, 2014.
14. M. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153-1176, 2016.
15. Y. Sheng, K. Zhao, X. Liu, and R. Zhou, "Detection of DNS tunneling using entropy features and Random Forest," Journal of Network and Computer Applications, vol. 141, pp. 1-12, 2019.
16. W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural networks for representation learning," in Proc. Int. Conf. Information Networking, 2017, pp. 712-717.
17. Z. Chen, K. Yu, and X. Zhang, "A deep learning approach for encrypted traffic classification," IEEE Access, vol. 8, pp. 108464-108475, 2020.
18. B. Anderson and D. McGrew, "Identifying encrypted malware traffic with contextual flow data," in Proc. ACM Workshop on Artificial Intelligence and Security, 2016, pp. 35-46.

19. R. Nadler, A. Aminov, and A. Shabtai, "Detection of malicious and low throughput data exfiltration over the DNS protocol," Computers & Security, vol. 80, pp. 36-53, 2019.
20. A. Ahmed, S. Aleroud, and A. Karabatis, "Detecting DNS tunneling using machine learning: challenges and open issues," in Proc. IEEE Int. Conf. Big Data, 2020, pp. 4512-4519.
21. T. Farnham, "DNS tunneling detection using statistical analysis," SANS Institute, 2014.
22. P. Butler, K. Xu, and D. Yao, "Quantitatively analyzing stealthy communication channels," in Proc. Int. Conf. Applied Cryptography and Network Security, 2011, pp. 238-254.
23. L. Deri, S. Miller, and A. Cardigliano, "Beyond deep packet inspection for detecting encrypted covert channels," Journal of Computer Virology and Hacking Techniques, vol. 13, no. 4, pp. 265-278, 2017.
24. K. Born and D. Gustafson, "Detecting DNS tunnels using character frequency analysis," arXiv preprint, 2010.
25. A. Karasaridis, K. Meier-Hellstern, and D. Hoeflin, "NIS04-2: Detection of DNS anomalies using flow data analysis," in Proc. IEEE Global Telecommunications Conf., 2006, pp. 1-6.
26. S. Marchal, J. Francois, R. State, and T. Engel, "PhishStorm: detecting phishing with streaming analytics," IEEE Trans. Network and Service Management, vol. 11, no. 4, pp. 458-471, 2014.
27. R. Nadler, A. Aminov, and A. Shabtai, "Detection of low-throughput data exfiltration via time-series analysis of DNS query volumes," in Proc. IEEE European Symp. Security and Privacy Workshops, 2018.
28. A. Almusawi and H. Amintoosi, "DNS tunneling detection method based on multilabel support vector machine," Security and Communication Networks, vol. 2018, Article ID 6137098, 2018.
29. L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.
30. T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785-794.
31. M. Aiello, M. Mongelli, E. Cambiaso, and G. Papaleo, "Profiling DNS tunneling attacks with PCA and mutual information," Logic Journal of the IGPL, vol. 24, no. 6, pp. 957-970, 2016.
32. B. Yu, J. Pan, J. Hu, A. Nascimento, and M. De Cock, "Character-level based detection of DGA domain names," in Proc. Int. Joint Conf. Neural Networks, 2018, pp. 1-8.
33. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.
34. K. Cho et al., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in Proc. Conf. Empirical Methods in Natural Language Processing, 2014, pp. 1724-1734.
35. G. Zhao, C. Xu, L. Zhao, and Z. Feng, "A hybrid deep learning approach for DNS tunneling detection," IEEE Access, vol. 9, pp. 100120-100132, 2021.
36. M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," Journal of Network and Computer Applications, vol. 60, pp. 19-31, 2016.
37. A. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," IEEE Communications Surveys & Tutorials, vol. 9, no. 3, pp. 44-57, 2007.
38. S. Bortzmeyer, "DNS privacy considerations," RFC 7626, Internet Engineering Task Force, 2015.
39. M. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you? Explaining the predictions of any classifier," in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 1135-1144.
40. S. Lundberg and S. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems, 2017, pp. 4765-4774.
41. N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.