OPEN ACCESS

Kashf Journal of Multidisciplinary Research
Vol: 03 - Issue 04 (2026)
P-ISSN: 3007-1992 E-ISSN: 3007-200X
https://kjmr.com.pk

LEXIDNS-RISK: DEPLOYMENT-AWARE LEXICAL RISK MODELING FOR INTERPRETABLE MALICIOUS DOMAIN SCREENING

Wahaj Hassan Soomro[1], Muhammad Arslan Siddiqui[2]

[1]*Institute of Computer Science, Shah Abdul Latif University, Khairpur Mirs, Sindh, Pakistan.*
[2]*FAST NUCES, Karachi, Sindh, Pakistan.*

[*]***Corresponding Author:*** (wahajhassansoomro@gmail.com)

DOI:(*https://doi.org/10.71146/kjmr904*)

Article Info

Abstract

Malicious domains are widely used in phishing, malware delivery, command-and-control communication, and DNS-based abuse. This paper evaluates lightweight lexical-feature-based models for first-stage DNS-layer malicious-domain screening. Using a balanced dataset of two million labeled domains, we extract interpretable features from domain strings and compare Logistic Regression, Random Forest, Extra Trees, XGBoost, LightGBM, and a character n-gram TF-IDF baseline. Results show that lexical boosting models achieve strong clean-test performance. LightGBM obtains the best F1-score of approximately **0.997**, while XGBoost provides nearly identical accuracy with lower latency and smaller model size. Compared with the TF-IDF baseline, lexical boosting models achieve higher F1-score, lower false-positive rate, and lower inference cost. Feature-importance analysis shows that base-domain length, digit ratio, digit count, token length, and character diversity are the dominant signals. Robustness experiments show that clean-test accuracy alone is insufficient for deployment-oriented evaluation, as several domain perturbations reduce fixed-threshold performance. Threshold recalibration improves some stress cases, but subdomain-noise remains challenging. Overall, lightweight lexical models can support fast and interpretable first-stage DNS-layer screening under controlled evaluation, but external validation and robustness-aware modeling are required before stronger deployment claims can be made.

Keywords:
Malicious Domain Detection, DNS-layer screening, Lexical Feature Modeling, Interpretable Machine Learning, Robustness-Aware Cybersecurity

1. Introduction

Malicious domains remain a persistent component of modern cybercrime infrastructure. They are routinely used in phishing campaigns, malware delivery, command-and-control communication, spam redirection, credential theft, and other DNS-based abuse. Because domain resolution is often one of the earliest observable steps before a user or host interacts with malicious infrastructure, DNS-layer screening has become an important first-stage defense mechanism. Earlier systems such as Noto's, Kopis, and EXPOSURE showed that DNS-derived signals can reveal malicious infrastructure before it becomes widely visible in public blacklists [1]-[3]. Later work on DGA-based malware further demonstrated that attackers often generate or rotate domain names to evade static blocking and reputation mechanisms [4], [5]. This makes timely domain-level risk estimation useful, especially when a system must flag suspicious domains before richer content, traffic, or reputation evidence is available.

Traditional blacklist and reputation-based systems are valuable in practice, but they cannot fully cover newly registered, rapidly changing, or algorithmically generated domains. Machine-learning approaches have therefore been widely studied for malicious URL and domain detection, using lexical features, host-based information, DNS traffic statistics, registration metadata, and character-level models [6]-[10]. Among these, lexical features are especially attractive for lightweight screening because they can be extracted directly from the domain string. Features such as domain length, digit count, token structure, vowel ratio, character diversity, subdomain count, and top-level-domain properties require no page visit, no packet payload inspection, and no third-party intelligence feed. This makes them fast, low-cost, and relatively privacy-preserving compared with content- or behavior-heavy pipelines.

However, high clean-test accuracy alone is not enough for operational malicious-domain screening. A detector used in a DNS-layer setting must also be judged by false-positive behavior, latency, model size, threshold stability, interpretability, and robustness to domain-string perturbations. Recent deployment-oriented studies in neighboring security and ML screening tasks emphasize that practical risk-scoring systems should not only rank cases well, but also provide interpretable evidence, calibrated or threshold-aware scores, and evaluation protocols that reduce optimistic conclusions [11]-[13]. Similarly, benchmark-oriented work in other domains shows that evaluation hygiene, deployment cost, and robustness can be as important as raw accuracy when the goal is practical use rather than only model comparison [14], [15]. These lessons are directly relevant to malicious-domain detection, where a model with excellent random-split accuracy may still be brittle under domain obfuscation, dataset bias, or source shift.

This paper does not aim to replace DNS reputation systems, threat-intelligence feeds, or full enterprise security telemetry. Instead, it evaluates whether lexical domain features can provide a lightweight and interpretable first-stage risk signal for DNS-layer screening. We frame the task as a deployment-aware evaluation problem: given a domain string, can compact lexical models produce accurate, efficient, and explainable malicious-domain risk scores, and how stable are these scores under controlled domain-name perturbations?

To answer this question, we use a balanced dataset of two million labelled domains and extract 39 lexical features from each domain string. These features describe full-domain and base-domain length, digit usage, character ratios, token structure, character diversity, repetition patterns, subdomain indicators, and suspicious lexical tokens. We compare Logistic Regression, Random Forest, Extra Trees, XGBoost, LightGBM, and a character n-gram TF-IDF Logistic Regression baseline under the same threshold-tuned evaluation protocol. In addition to clean-test metrics, we report model size, inference latency, feature importance, dataset-bias diagnostics, compact top-k feature behavior, and robustness under domain obfuscation with fixed and recalibrated thresholds.

The results show that lexical boosting models achieve strong clean-test performance on the evaluated dataset. LightGBM obtains the highest F1-score of approximately 0.997, while XGBoost achieves nearly identical predictive performance with a smaller model and lower inference latency. Compared with the character n-gram TF-IDF baseline, lexical boosting models achieve higher F1-score, lower false-positive rate, and lower inference cost. Feature-importance analysis shows that base-domain length, digit ratio, digit count, token length, and character diversity are the dominant signals. At the same time, robustness experiments show that clean-test performance can overestimate deployment reliability: several perturbations reduce fixed-threshold performance, and subdomain-noise remains a difficult failure case. Threshold recalibration improves some stress conditions, but it does not fully remove robustness limitations.

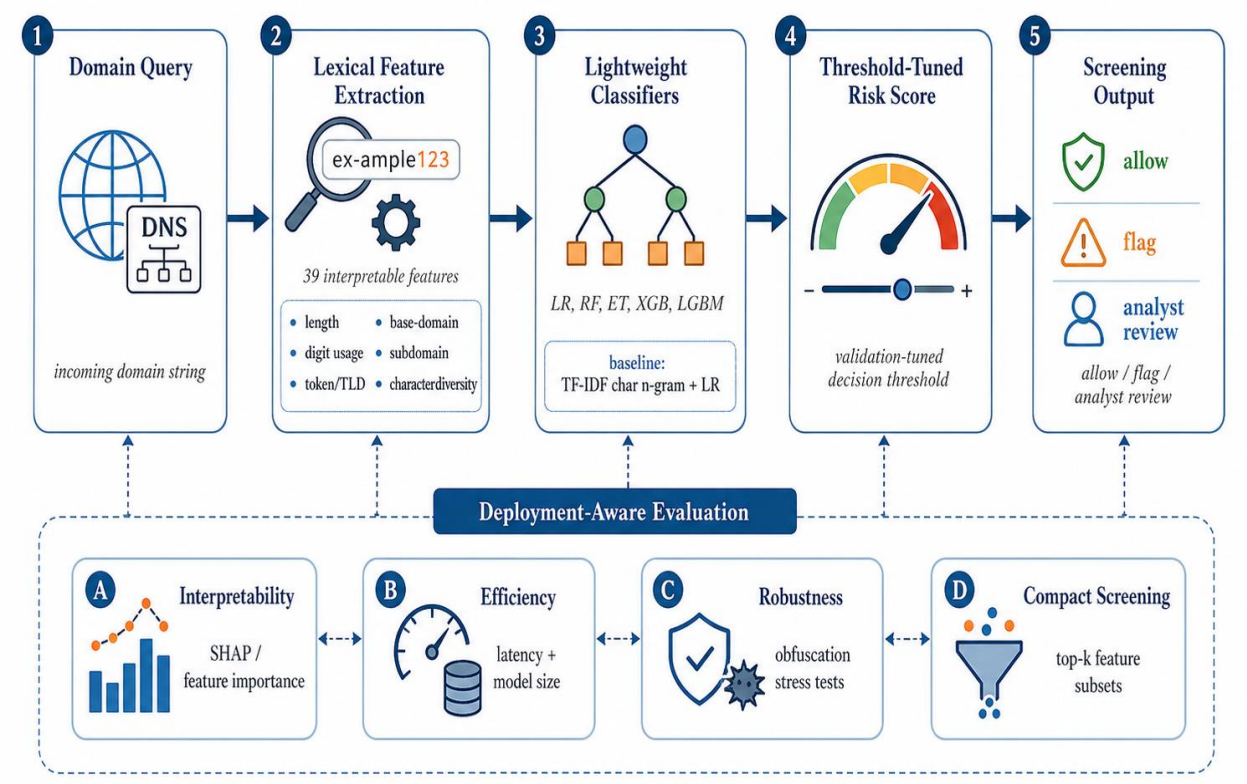

Fig. 1. Proposed lightweight DNS-layer malicious-domain screening workflow with lexical feature extraction, threshold-tuned risk scoring, and deployment-aware evaluation.

The main contributions of this study are:

1. **Large-scale lexical evaluation:** We conduct a controlled evaluation of malicious-domain screening on two million labelled domains using 39 interpretable lexical features.
2. **Model comparison under a common protocol:** We compare linear, tree-based, boosting-based, and character n-gram TF-IDF models using F1-score, ROC-AUC, PR-AUC, false-positive rate, false-negative rate, model size, and inference latency.
3. **Deployment-aware model selection:** We show that LightGBM provides the best clean-test F1-score, while XGBoost offers a stronger accuracy–efficiency trade-off for lightweight screening.
4. **Interpretability and dataset-bias analysis:** We use feature-importance analysis and class-wise feature-gap diagnostics to identify the lexical signals that drive model behavior.
5. **Robustness and threshold recalibration:** We evaluate domain perturbations such as digit substitution, hyphen insertion, vowel removal, character repetition, subdomain noise, and mixed obfuscation, comparing fixed clean thresholds with stress-specific recalibration.
6. **Compact feature study:** We analyze whether a reduced set of top lexical features can preserve most of the full-model performance, supporting lightweight deployment scenarios.

2. Related Work

2.1 Malicious Domain and DNS-Layer Detection

DNS has long been used by malicious actors to support phishing, malware distribution, botnet command-and-control, and infrastructure rotation. Early passive-DNS and reputation-based systems established that DNS behavior can reveal malicious activity beyond static blacklists. Noto's proposed a dynamic DNS reputation system based on passive DNS, network, and zone features, showing that malicious domains often exhibit distinguishable DNS behavior before appearing on public blacklists [1]. Kopis moved detection toward the upper DNS hierarchy and analyzed global query-resolution patterns to identify malware-related domains [2]. EXPOSURE used large-scale passive DNS analysis and extracted DNS-traffic features to detect malicious domains involved in real-world abuse [3]. These systems remain important because they show that DNS-layer evidence can support early risk estimation.

A related line of work focuses on DGA-based malware. DGAs generate many pseudo-random or algorithmically varied domains to maintain command-and-control connectivity and evade domain blocking. Pleiades showed that NXDomain patterns can help detect DGA-based malware without fully reversing the malware family [4]. FANCI later proposed a feature-based system for classifying NXDomain responses into benign and DGA-related domains using features extracted from individual domain names [5]. Deep learning approaches, including LSTM-based DGA classifiers, have also shown strong detection performance by learning character-level patterns directly from domain strings [8]. However, adversarial DGA work shows that domain-generation

strategies can be adapted to bypass learned detectors, motivating robustness-aware evaluation rather than clean accuracy alone [9].

Recent distributed deep forest work has framed malicious-domain detection as a DNS and big-data problem, using DNS traffic features and selective forest-based modeling for efficient detection. That work is relevant because it confirms the value of efficient tree/forest-style models for DNS security, but it focuses more on DNS traffic and distributed deep forest modeling. In contrast, the present paper studies a narrower lexical-only setting and asks whether domain-string features alone can provide fast, interpretable, and robustness-aware first-stage screening.

Prior work has established the importance of DNS-layer detection, passive DNS analysis, reputation signals, and DGA-focused modeling. The remaining practical issue is not simply whether malicious domains can be classified under a clean split, but whether lightweight screening models remain interpretable, efficient, threshold-stable, and robust enough to justify operational use as a first-stage signal.

2.2 Lexical and Character-Based Domain Modeling

Lexical features have been widely used in malicious URL and domain detection because attacker-controlled domains often differ from benign domains in string structure. Ma et al. showed that lexical and host-based URL properties can support malicious website detection beyond blacklists [6]. Sahoo et al. later reviewed machine-learning approaches for malicious URL detection and organized the field around feature representation, learning algorithms, and practical deployment challenges [7]. Typical lexical features include domain or URL length, number of digits, special-character usage, token count, subdomain depth, entropy-like randomness, suspicious words, and top-level-domain information. These features are attractive because they are inexpensive to compute and do not require visiting the webpage or observing long DNS histories.

Character-based models provide another way to represent domains. Instead of manually extracting lexical counts and ratios, character n-gram models and neural sequence models learn patterns from the domain string. TF-IDF character n-grams are simple, strong baselines because they capture local substrings, repeated patterns, and suspicious token fragments. LSTM and CNN-based DGA detectors extend this idea by learning sequential character representations directly [8], [16]. These models can reduce manual feature engineering, but they may require larger model footprints, more training data, or less transparent decision evidence than compact lexical features.

The present paper includes a character n-gram TF-IDF Logistic Regression baseline to make the comparison fairer. This is important because a lexical-feature model should not only beat weak classical baselines; it should also be compared against a strong string-based alternative. In our results, the TF-IDF baseline performs well, but lexical boosting models achieve higher F1-score, lower false-positive rate, and lower inference cost. This supports the paper's narrower claim: carefully designed lexical features remain competitive for controlled first-stage DNS screening, especially when efficiency and interpretability are part of the evaluation.

2.3 Interpretable and Deployment-Aware Cybersecurity ML

Operational cybersecurity screening differs from offline classification because the model output often becomes a triage signal. A high score may trigger review, blocking, escalation, or additional analysis. In such settings, discrimination metrics such as ROC-AUC and F1-score are useful but incomplete. The score must also be interpretable, threshold-aware, calibrated enough for risk communication, and efficient enough for high-throughput use. Prior work in Android malware filtering argues that real deployments need reliability diagnostics, calibrated probabilities, and robustness checks rather than only high ROC-AUC. Related risk-screening studies in healthcare fraud and Android malware detection show that thresholding, calibration, and interpretability can make model outputs more useful for operational review. Although these studies address different domains, their operational lesson is directly relevant: a screening model should produce useful risk evidence under realistic decision constraints.

Tree-based models are a natural fit for this setting. XGBoost and LightGBM are widely used for structured/tabular data because they often provide strong accuracy with manageable training and inference cost [17], [18]. SHAP provides a principled framework for explaining model predictions through additive feature attributions [19]. These tools allow a malicious-domain detector to be evaluated not only as a classifier, but also as an explainable risk-scoring component. In our study, feature-importance analysis identifies base-domain length, digit ratio, digit count, token length, and character diversity as key signals. This helps make model behavior understandable, while also exposing possible dataset bias.

Deployment-aware evaluation also requires attention to robustness and benchmark validity. The uploaded leakage-aware gesture benchmark shows how duplicate contamination and missing evaluation hygiene can inflate apparent performance, and it argues that deployment cost and robustness should be treated as first-class outcomes rather than supplementary details. A related Vector+SQL evaluation paper emphasizes that reporting only average behavior can hide important quality–latency trade-offs and that evaluation should be designed around the real operational condition of the system. The same principle applies to DNS-layer malicious-domain screening: a clean random split may show strong performance, but a deployment-oriented study should also test perturbations, threshold behavior, false positives, latency, and model size.

This paper follows that deployment-aware view. It does not introduce a new deep architecture or claim a complete DNS defense system. Instead, it evaluates a lightweight lexical screening pipeline across accuracy, efficiency, interpretability, calibration-related threshold behavior, and robustness. This framing is intentionally conservative: lexical models are positioned as first-stage risk-scoring components that can complement reputation feeds and broader telemetry, not replace them.

3. Dataset and Feature Extraction

This study uses a labeled malicious-domain dataset containing **2,000,000 domain records**. The dataset is class-balanced, with **1,000,000 benign domains** and **1,000,000 malicious domains**, where the label `0` represents benign and `1` represents malicious. The released CSV contained only

two fields: `Domain` and `Label`. Therefore, all predictive variables used in this study were extracted directly from the domain string rather than taken from precomputed metadata, DNS logs, WHOIS records, webpage content, or threat-intelligence feeds. This design keeps the evaluation focused on the question of whether lexical domain structure alone can support lightweight first-stage DNS-layer risk screening.

Figure 2 shows the class distribution of the dataset. The balanced class structure avoids the basic evaluation bias that can occur when one class dominates the dataset. However, class balance alone does not guarantee real-world generalization, because operational DNS traffic is usually highly imbalanced and changes over time. For this reason, the reported results should be interpreted as controlled benchmark evidence rather than direct proof of deployment performance.

A total of **39 lexical features** were extracted from each domain string. These features were designed to capture interpretable structural properties of domain names. They include full-domain and base-domain length features, digit and letter counts, vowel and consonant ratios, special-character usage, hyphen and dot patterns, top-level-domain length, token count, subdomain count, character-diversity indicators, repetition/run-length features, an entropy proxy, and a suspicious-token flag. The feature set is intentionally lightweight: all features can be computed from the domain string without resolving the domain, downloading page content, or querying external intelligence services.

The extracted features can be grouped into six main categories:

1. **Length and token-structure features**, including domain length, raw length, first-token length, base-domain length, token count, and TLD length.
2. **Character-count features**, including digit count, letter count, vowel count, consonant count, and special-character count.
3. **Ratio features**, including digit ratio, vowel ratio, consonant ratio, special-character ratio, hyphen ratio, dot ratio, and base-domain digit/vowel ratios.
4. **Base-domain and subdomain features**, including base-domain length, base digit count, base unique-character ratio, subdomain count, and dot count.
5. **Randomness and morphology features**, including unique-character count, unique-character ratio, entropy proxy, repeated-character count, maximum digit run, and maximum alphabetic run.
6. **Pattern indicators**, including binary flags for digit presence, hyphen presence, many-dot structure, IP-like format, and suspicious lexical tokens such as login, secure, verify, account, bank, wallet, and payment.

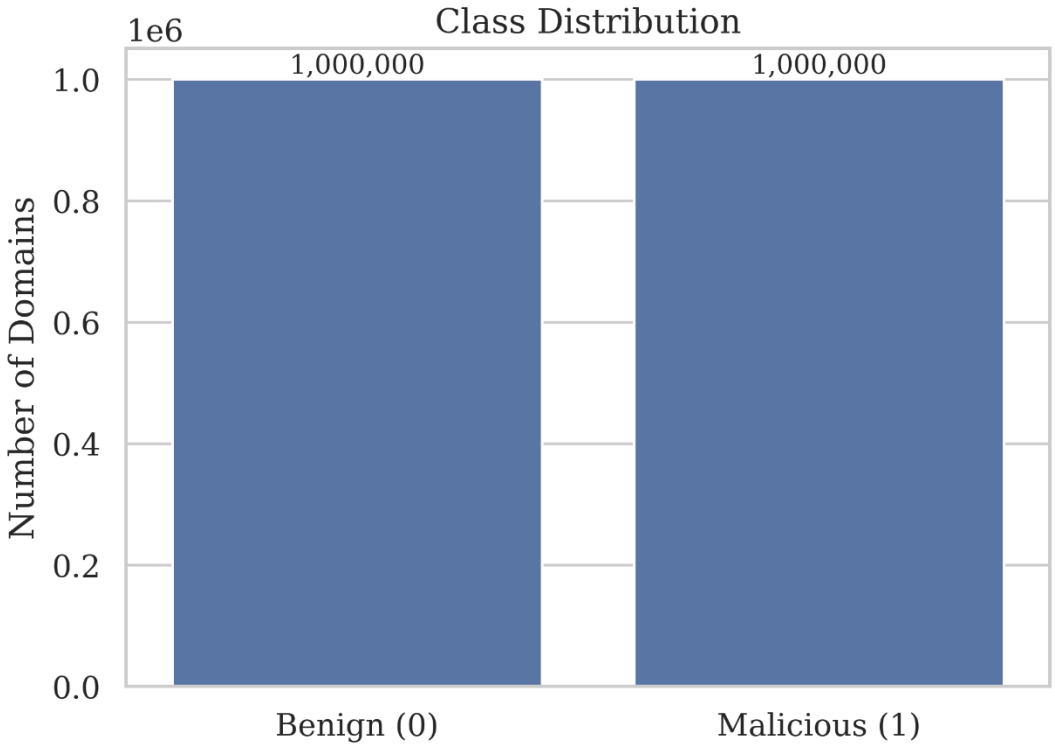

Fig. 2. Class distribution of the malicious-domain dataset, showing an equal number of benign and malicious samples.

To better understand the dataset before modeling, we also computed class-wise feature-gap statistics. Figure 3 reports the largest mean differences between benign and malicious domains. The diagnostic shows clear lexical separation between the two classes, especially in features related to domain length, base-domain length, digit usage, and character diversity. For example, malicious domains show substantially higher average domain length and digit usage than benign domains. This supports the use of lexical models, but it also introduces an important caution: the very strong clean-test results may partly reflect large class-wise lexical gaps in this dataset.

Therefore, this dataset is suitable for evaluating controlled lexical malicious-domain screening, but it has limitations. Since the dataset does not provide source metadata, collection time, DNS-resolution history, WHOIS information, attack family labels, phishing/malware/DGA subtype labels, or temporal splits, it cannot support temporal generalization, family-wise generalization, or source-wise robustness claims. In this paper, we therefore treat the dataset as a large controlled benchmark for lexical screening, not as complete evidence of real-world DNS deployment readiness.

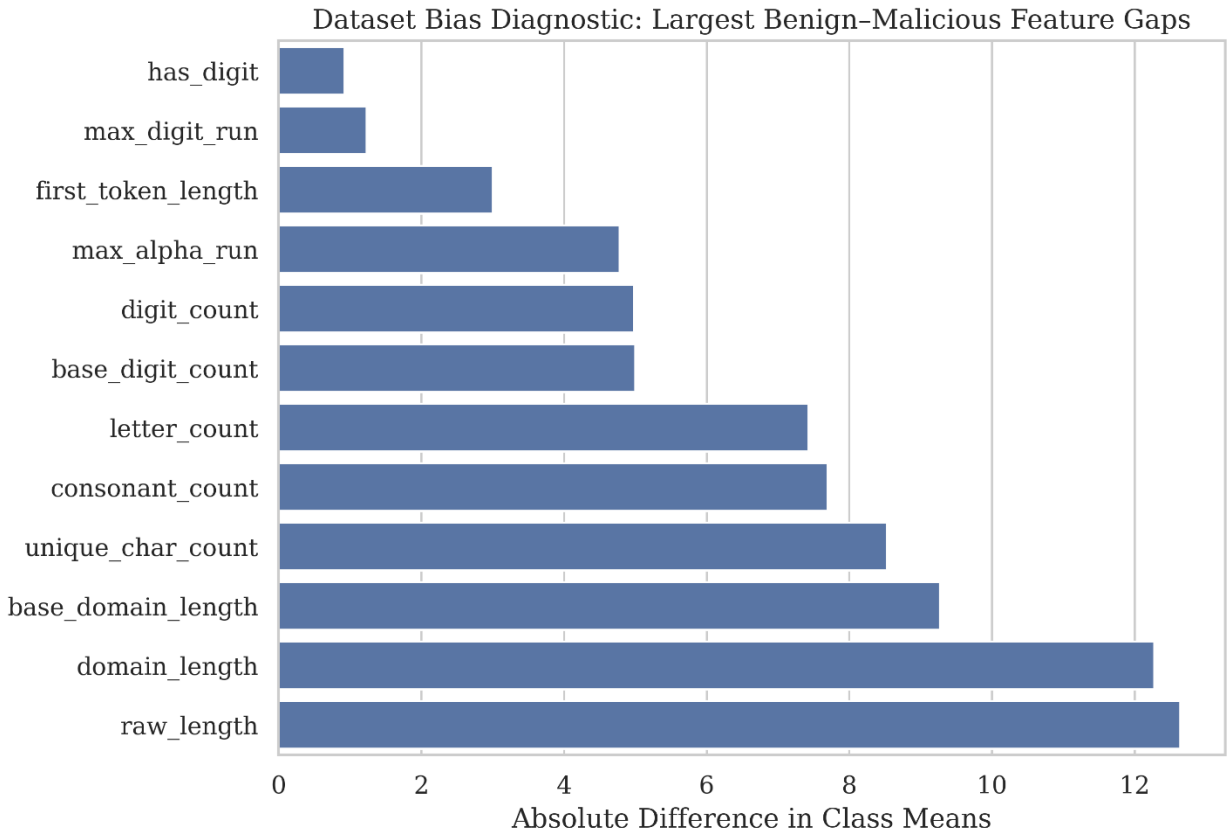

Fig. 3. Largest class-wise lexical feature gaps between benign and malicious domains.

4. Methodology

4.1 DNS-layer Screening Pipeline

The proposed workflow is designed as a lightweight DNS-layer screening pipeline, as shown in Fig. 1. Given an incoming domain query, the system first normalizes the domain string and extracts lexical features from it. These features are then passed to a lightweight classifier, which outputs a malicious-domain risk score. Instead of using the default probability threshold of 0.5, the score is converted into a final decision using a validation-tuned threshold. The domain can then be treated as benign, flagged for review, or passed to a stronger security layer for further inspection.

This pipeline is intentionally framed as a **first-stage screening mechanism**, not as a standalone blocking system. In practical DNS security settings, lexical evidence alone is not sufficient to replace threat-intelligence feeds, reputation systems, DNS traffic analysis, or analyst review. The purpose of this design is to test whether domain-string structure can provide a fast and interpretable early risk signal before more expensive analysis is applied.

4.2 Lexical Feature Extraction

For each domain, we extracted **39 lexical features** directly from the domain string. These features were selected to represent interpretable structural properties of domain names rather than opaque learned embeddings. The feature set includes length-based features, digit and character counts, ratio-based features, base-domain properties, TLD length, subdomain count, repetition patterns, an entropy proxy, and suspicious-token indicators.

The motivation is that malicious domains, especially phishing or DGA-like domains, often differ from benign domains in visible string structure. For example, they may contain longer base names, more digits, unusual token patterns, higher character diversity, or suspicious words such as `login`, `secure`, `verify`, `account`, `bank`, and `payment`. At the same time, these features are computationally cheap and can be extracted without resolving the domain, visiting a webpage, or querying external services. This makes them suitable for a lightweight screening setting.

4.3 Models

To evaluate whether lexical features are competitive for malicious-domain screening, we compared several model families under the same train–validation–test protocol:

- **Logistic Regression**, as a simple and interpretable linear baseline.
- **Random Forest**, as a classical bagging-based tree ensemble.
- **Extra Trees**, as a randomized tree ensemble with low tuning complexity.
- **XGBoost**, as a gradient-boosted tree model with strong tabular-data performance.
- **LightGBM**, as an efficient boosting model designed for fast training and high predictive performance.
- **TF-IDF character n-gram Logistic Regression**, as a strong string-based baseline.

The TF-IDF character n-gram model was included to avoid comparing lexical features only against weak baselines. Character n-grams can capture local string patterns directly from the domain name and are commonly effective for URL/domain classification. Therefore, this baseline helps test whether handcrafted lexical features remain useful when compared with a competitive string-level representation.

4.4 Evaluation Protocol

The dataset was split into training, validation, and held-out test partitions using stratified sampling. The validation set was used only for threshold selection, while the final metrics were reported on the held-out test set. We report F1-score, ROC-AUC, PR-AUC, false-positive rate, false-negative rate, model size, and per-domain inference latency.

4.5 Threshold Tuning

All models produce a probability-like malicious-domain score. Instead of applying the default threshold of **0.5**, we tuned the decision threshold on the validation set using **F1-score**. This is

important because the best operational threshold may differ from 0.5, especially when the cost of false positives and false negatives is not identical.

For each model, candidate thresholds were scanned on the validation set, and the threshold producing the highest F1-score was selected. The selected threshold was then fixed and applied to the held-out test set. This avoids optimistic test-set thresholding and gives a more realistic estimate of how the model would behave after validation-based calibration.

4.6 Robustness and Recalibration Protocol

To examine whether clean-test performance remains stable under domain-string variation, we designed a controlled robustness protocol. The goal was not to fully simulate adaptive attackers, but to test how sensitive lexical models are too simple perturbations that change the visible form of a domain. Six stress settings were used:

- **Digit substitution**, replacing characters such as o, i, e, and a with visually similar digits.
- **Hyphen insertion**, inserting a hyphen inside the base domain.
- **Vowel removal**, removing vowels from the base-domain string.
- **Character repetition**, duplicating a character inside the domain.
- **Subdomain noise**, adding a suspicious-looking prefix such as login-secure.
- **Mixed obfuscation**, combining multiple perturbations into a stronger stress case.

For each stress setting, we evaluated two threshold policies. First, we used the clean validation threshold to measure how the model behaves when deployed without adjustment. Second, we recalibrated the threshold using perturbed validation data and then evaluated it on perturbed test data. This comparison shows whether performance degradation comes mainly from poor score ranking or from threshold shift under distribution change.

This robustness protocol provides a practical sensitivity check. It does not claim to cover all attacker behaviors, but it helps identify whether a lexical detector is stable under common domain-string modifications and whether threshold recalibration can recover part of the lost performance.

5. Results and Discussion

5.1 Clean Test Performance

The clean held-out evaluation shows that all lexical models learned the task well, but the boosting models gave the strongest overall performance. As shown in **Fig. 4**, LightGBM achieved the best clean-test result, with F1 = 0.9972, ROC-AUC = 0.9999, and PR-AUC = 0.9999. XGBoost followed closely with F1 = 0.9971 while using a smaller model and lower latency. The difference between LightGBM and XGBoost is therefore very small in predictive terms.

Fig. 4. Clean held-out performance of lexical classifiers and the TF-IDF character n-gram baseline.

The character n-gram TF-IDF Logistic Regression baseline also performed strongly, which confirms that domain strings contain useful class-discriminative patterns. However, it underperformed the lexical boosting models in F1-score, false-positive rate, and inference latency. This result suggests that carefully designed lexical features remain competitive against a strong string-based baseline, especially when efficiency and interpretability are part of the evaluation.

5.2 Efficiency and Deployment Trade-off

Accuracy alone is not sufficient for DNS-layer screening, where a model may need to process a large number of domain queries with limited delay. For this reason, we also compared model size and per-domain inference latency. As shown in **Fig. 5**, LightGBM achieved the highest F1-score, but XGBoost offered the most attractive accuracy-efficiency trade-off.

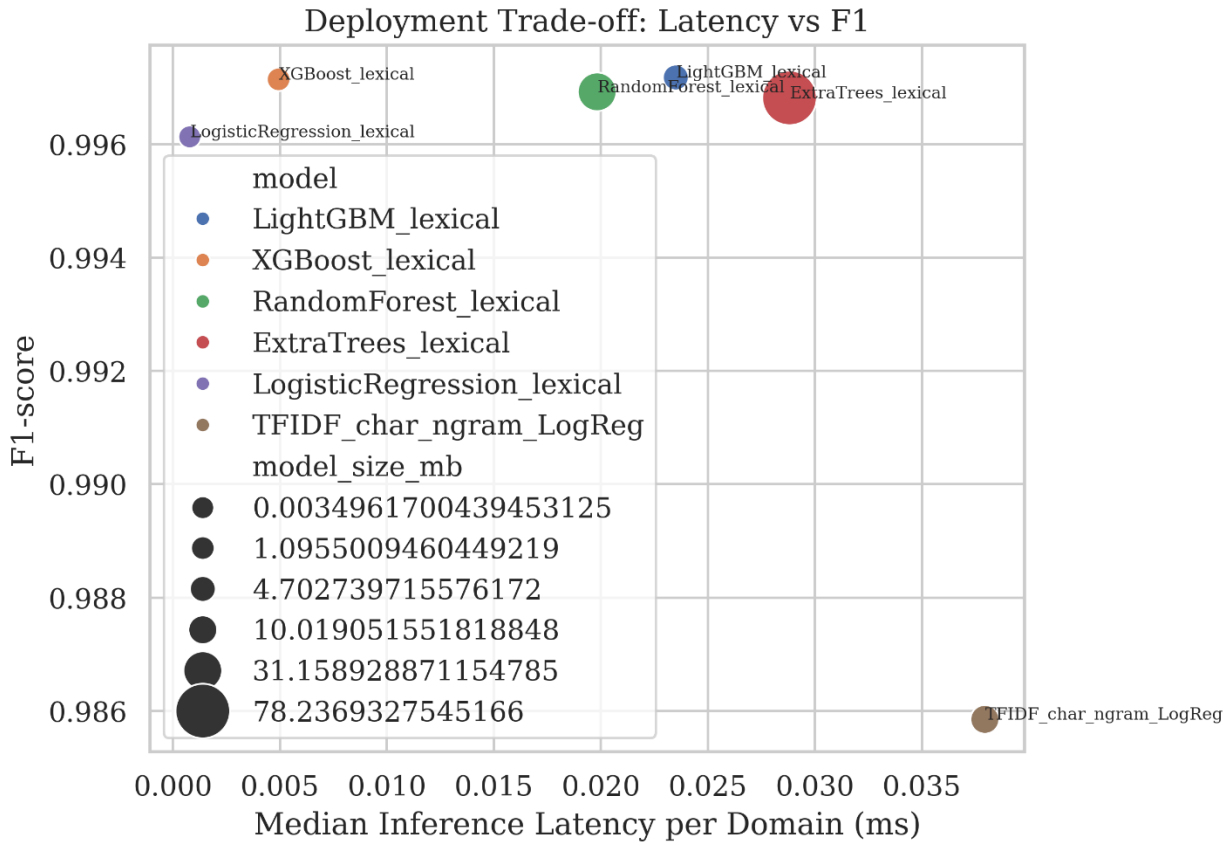

Fig. 5. Deployment trade-off between F1-score, inference latency, and model size.

XGBoost reached almost the same F1-score as LightGBM while using a smaller model and lower inference latency. In practical terms, this makes XGBoost a strong candidate for resource-constrained or high-throughput first-stage screening. Random Forest and Extra Trees also performed well, but their larger model sizes made them less attractive for lightweight deployment.

These latency values indicate feasibility for batch screening or first-stage DNS risk scoring. However, they should not be interpreted as complete production-throughput measurements. A real DNS deployment would still require system-level integration, caching behavior, network overhead analysis, and throughput testing under realistic query loads.

5.3 Interpretability and Feature Behavior

To understand what the best lexical model learned, we analyzed feature importance using SHAP-based attribution. As shown in **Fig. 6**, the most influential features were **base-domain length**, **base digit ratio**, **base digit count**, **first-token length**, and **unique character count**. These features are meaningful in the context of malicious-domain screening because many suspicious or algorithmically generated domains tend to contain unusual length patterns, digit-heavy strings, or higher character diversity.

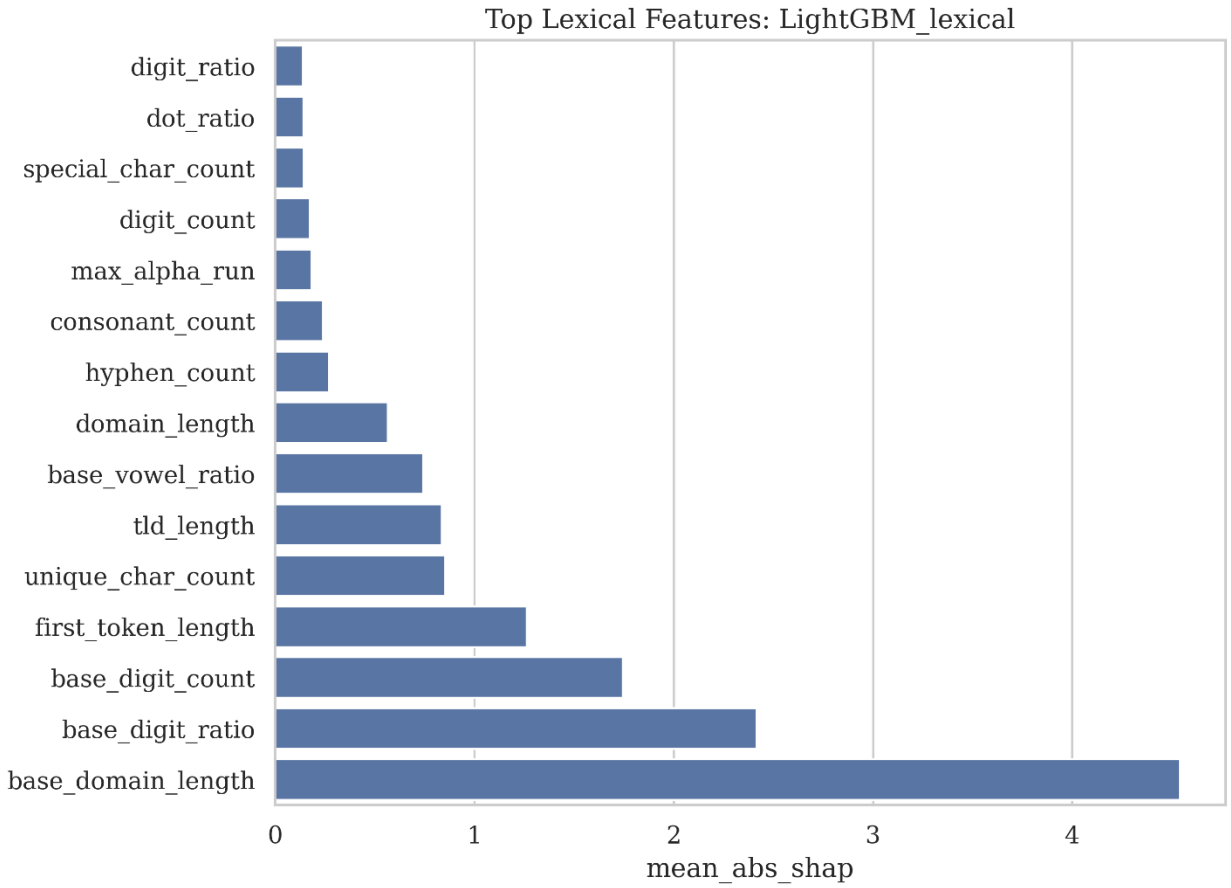

Fig. 6. Feature-importance analysis of the best lexical model. Base-domain length, base digit ratio, base digit count, first-token length, and unique character count are the dominant signals, indicating that the model relies on interpretable domain-string structure.

This result supports the interpretability motivation of the study. Instead of relying only on opaque embeddings, the model uses visible lexical signals that can be inspected and explained. For example, a high-risk score can be linked to a long base domain, a high digit ratio, or an unusual token structure. Such explanations are useful for analyst-facing triage because they provide a simple reason why a domain was flagged.

At the same time, this behavior also creates a boundary for the model. Lexical signals may be less effective for compromised legitimate domains, benign-looking phishing domains, or carefully crafted impersonation domains that avoid obvious randomness. Therefore, these features should be treated as risk indicators rather than definitive proof of maliciousness.

5.4 Dataset-Bias Analysis

The dataset-bias diagnostic provides an important explanation for the very high clean-test results. Benign and malicious domains differ strongly in several lexical dimensions, especially domain

length, base-domain length, digit count, and character diversity. These large class-wise gaps make the classification problem highly separable under a random train–test split.

This finding is useful but also cautionary. It explains why multiple models, including Logistic Regression, achieved near-saturated performance. If simple linear and tree-based models all perform extremely well, the dataset likely contains strong lexical regularities that make clean classification easier. Therefore, the clean-test results should be interpreted as evidence of strong lexical separability in this dataset, not as proof that the model will generalize equally well to all real-world DNS traffic.

This is one reason why the paper includes robustness tests and reports dataset-bias diagnostics rather than relying only on accuracy. The goal is to make the evaluation more transparent and avoid overstating the deployment claim.

5.5 Robustness under Domain Obfuscation

The robustness experiments show that clean-test performance does not automatically translate into stable behavior under domain perturbations. As shown in **Fig. 7** and **Fig. 8**, the model remained stable under character repetition and showed moderate degradation under digit substitution and vowel removal. These cases suggest that the learned lexical representation can tolerate some small string changes.

Fig. 7. Robustness comparison under fixed clean threshold and stress-recalibrated threshold.

The more difficult cases were hyphen insertion, mixed obfuscation, and subdomain noise. Under the fixed clean threshold, hyphen insertion caused severe performance degradation, but stress-specific threshold recalibration recovered a substantial portion of the lost performance. Mixed obfuscation showed a similar pattern: fixed-threshold performance was weak, but recalibration improved the result considerably. This indicates that in some perturbation settings, the model still preserves useful ranking information, but the clean threshold becomes poorly matched to the shifted score distribution.

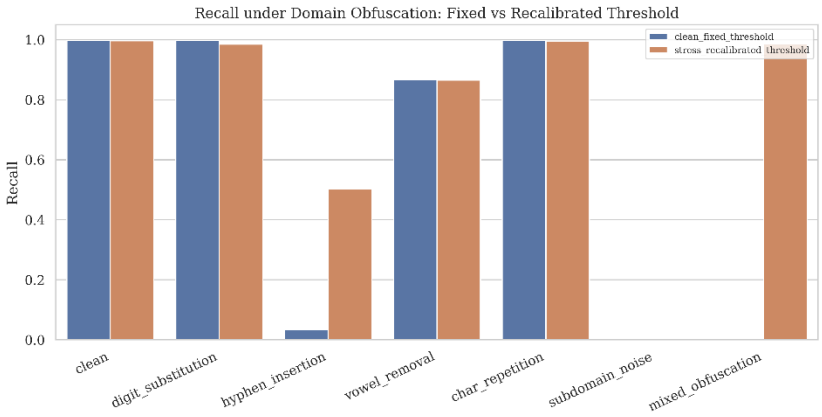

Fig. 8. Recall behavior under fixed and recalibrated thresholds across perturbation settings.

Subdomain noise remained the strongest failure case. Even after recalibration, performance stayed poor compared with other perturbations. This suggests a structural weakness in the current lexical representation: adding suspicious-looking subdomain prefixes can change the feature distribution in a way that the model does not handle reliably. For DNS-layer deployment, this is an important finding because real domains often include subdomains, tracking prefixes, service labels, or attacker-controlled hostnames.

Robustness testing is one of the most important parts of this study. It shows that threshold recalibration can improve several stress cases, but it also reveals that lexical screening still needs stronger subdomain-aware modeling and external validation before stronger deployment claims can be made.

5.6 Compact Feature Study

The compact feature experiment evaluates whether the full 39-feature representation is necessary. As shown in **Fig. 9**, the top-10 and top-20 feature subsets recovered most of the full-model performance. The top-5 feature set was weaker but still achieved strong performance, while adding more top-ranked features gradually closed the gap to the full model.

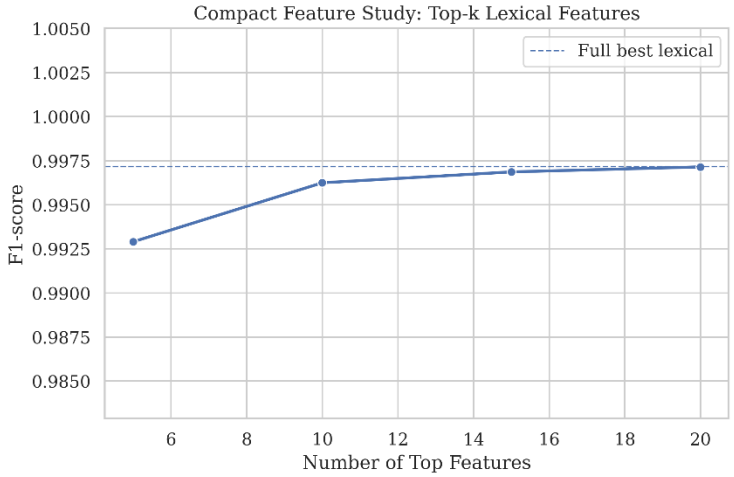

Fig. 9. Compact top-k feature study showing performance using reduced lexical feature subsets.

This result supports the lightweight deployment motivation of the paper. A reduced feature set can retain high detection performance while simplifying computation and model interpretation. In practical settings, this is useful because a compact feature extractor is easier to implement, audit, and maintain. It also confirms that much of the predictive signal is concentrated in a small group of lexical features, especially base-domain length, digit-related features, token length, and character-diversity measures.

The compact-feature result should still be interpreted within the same dataset limitation. It shows that compact lexical screening is feasible under the evaluated benchmark, but external validation would be required to confirm whether the same reduced feature set remains reliable on independently collected domains.

6. Limitations

This study has several limitations. First, the experiments are based on a single public dataset. Although the dataset is large and balanced, no independent external validation set was available. Therefore, the reported results mainly reflect controlled benchmark performance rather than cross-dataset generalization.

Second, the dataset provides only Domain and Label fields. It does not include source information, collection time, DNS history, WHOIS records, malicious-family labels, or separate phishing, malware, and DGA categories. As a result, this study cannot support temporal, source-wise, or family-wise generalization claims.

Third, the clean-test performance should be interpreted carefully. The dataset-bias analysis showed large lexical differences between benign and malicious domains, especially in length, digit usage, and character diversity. These gaps may partly explain why several models achieved near-saturated clean-test performance.

Fourth, robustness remains challenging. The model handled some perturbations reasonably well, but subdomain-noise and certain obfuscation patterns caused clear performance degradation. This suggests that stronger subdomain-aware modeling is needed.

Finally, the proposed framework should be treated as a first-stage DNS-layer risk scorer, not as a complete DNS firewall or a replacement for threat intelligence. In practice, lexical screening should be combined with DNS telemetry, reputation feeds, domain registration signals, network behavior, and analyst review.

7. Conclusion

This study evaluated lightweight lexical models for malicious-domain detection in a DNS-layer screening setting. Using a balanced dataset of two million labelled domains, we extracted 39 interpretable features from domain strings and compared linear, tree-based, boosting-based, and character n-gram models under the same threshold-tuned protocol.

The results show that lexical models can achieve strong clean-test performance on this controlled benchmark. LightGBM produced the highest F1-score, while XGBoost achieved nearly the same detection performance with a smaller model size and lower inference latency. This makes XGBoost especially attractive when the goal is fast first-stage screening rather than only maximizing accuracy.

The interpretability analysis also gave useful insight into model behavior. Base-domain length, digit ratio, digit count, token length, and character diversity were the most influential signals. These features are easy to compute and explain, which supports the use of lexical models as analyst-facing risk-scoring tools. At the same time, the dataset-bias analysis showed large lexical gaps between benign and malicious domains, suggesting that the high clean-test scores should be interpreted with caution.

The robustness experiments were important because they showed that clean accuracy does not guarantee stable behavior under domain-string changes. Some perturbations caused major performance drops under the fixed clean threshold. Threshold recalibration improved several stress cases, but subdomain-noise remained a difficult failure mode. This indicates that future systems need stronger subdomain-aware features and broader validation.

Overall, the findings suggest that lightweight lexical models are promising for interpretable first-stage DNS-layer malicious-domain screening. However, they should not be treated as a complete replacement for DNS reputation systems, threat intelligence, or broader security telemetry. External validation, temporal testing, and improved robustness under subdomain and obfuscation shifts are needed before stronger real-world deployment claims can be made.

References

[1] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a dynamic reputation system for DNS," in *Proc. 19th USENIX Security Symposium (USENIX Security 10)*, Washington, DC, USA, Aug. 2010.

[2] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou II, and D. Dagon, "Detecting malware domains at the upper DNS hierarchy," in *Proc. 20th USENIX Security Symposium (USENIX Security 11)*, San Francisco, CA, USA, Aug. 2011.

[3] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "EXPOSURE: Finding malicious domains using passive DNS analysis," in *Proc. Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, Feb. 2011.

[4] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon, "From throw-away traffic to bots: Detecting the rise of DGA-based malware," in *Proc. 21st USENIX Security Symposium (USENIX Security 12)*, Bellevue, WA, USA, Aug. 2012, pp. 491–506.

[5] S. Schüppen, D. Teubert, P. Herrmann, and U. Meyer, "FANCI: Feature-based automated NXDomain classification and intelligence," in *Proc. 27th USENIX Security Symposium (USENIX Security 18)*, Baltimore, MD, USA, Aug. 2018, pp. 1165–1181.

[6] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: Learning to detect malicious web sites from suspicious URLs," in *Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, Paris, France, 2009, pp. 1245–1254, doi: 10.1145/1557019.1557153.

[7] D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL detection using machine learning: A survey," *arXiv preprint arXiv:1701.07179*, 2017.

[8] J. Woodbridge, H. S. Anderson, A. Ahuja, and D. Grant, "Predicting domain generation algorithms with long short-term memory networks," *arXiv preprint arXiv:1611.00791*, 2016.

[9] H. S. Anderson, J. Woodbridge, and B. Filar, "DeepDGA: Adversarially-tuned domain generation and detection," in *Proc. ACM Workshop on Artificial Intelligence and Security (AISec)*, Vienna, Austria, 2016, pp. 13–21, doi: 10.1145/2996758.2996767.

[10] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, Banff, AB, Canada, 2007, pp. 649–656, doi: 10.1145/1242572.1242660.

[11] S. A. Mangi, S. Rajper, N. A. Shaikh, and N. Maitlo, "Efficient malicious domain detection using a distributed deep forest algorithm," *Preprints.org*, Sep. 2025, doi:10.20944/preprints202509.0573.v1.

[12] I. Hyder, R. A. Shaikh, R. H. Arain, Z. Hussain, and B. Raza, "Audit-ready healthcare fraud screening: Split-safe provider aggregation and explainable boosted risk triage," *Southern Journal of Computer Science*, vol. 2, no. 1, pp. 18–28, 2026.

[13] B. Raza, A. Maitlo, Z. H. Shar, and I. Hyder, "Operational Android malware filtering: Calibrated probabilities and distribution-free guarantees," *Kashf Journal of Multidisciplinary Research*, vol. 2, no. 12, pp. 58–73, 2025.

[14] B. Raza, S. Rajper, N. A. Shaikh, Z. H. Shar, and I. Hyder, "Parsimonious gesture benchmarking for duplicate-contaminated touchless document interaction," *Spectrum of Engineering Sciences*, vol. 4, no. 4, pp. 917–932, 2026, doi: 10.5281/zenodo.19690462.

[15] S. Bibi, F. A. Rajput, M. Younis, S. Bibi, and B. Raza, "Vector+SQL retrieval with selectivity workloads: Measuring tail latency and quality under filtered Top-K," *VFAST Transactions on Software Engineering*, vol. 14, no. 1, pp. 335–349, 2026, doi: 10.21015/vtse.v14i1.2353.

[16] 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 (IJCNN)*, Rio de Janeiro, Brazil, 2018, pp. 1–8.

[17] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[18] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems 30 (NeurIPS)*, Long Beach, CA, USA, 2017, pp. 3146–3154.

[19] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30 (NeurIPS)*, Long Beach, CA, USA, 2017, pp. 4768–4777.

[20] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proc. 22nd Int. Conf. Machine Learning (ICML)*, Bonn, Germany, 2005, pp. 625–632, doi: 10.1145/1102351.1102430.

[21] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Machine Learning (ICML)*, Sydney, Australia, 2017, pp. 1321–1330.

[22] J. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large Margin Classifiers*, A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. Cambridge, MA, USA: MIT Press, 1999, pp. 61–74.

[23] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[24] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006, doi: 10.1007/s10994-006-6226-1.

[25] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, 2011, doi: 10.1145/1961189.1961199.