Improving DNS Exfiltration Detection via Transformer Pretraining

Miloš Tomić[1], Aleksa Cvetanović[1], and Predrag Tadić[1]

[1]*School of Electrical Engineering, University of Belgrade, Serbia*
{tomicmilos03, aleksa.cvetanovic99}@gmail.com, tadicp@etf.bg.ac.rs

***Abstract*—We study whether in-domain pretraining of Bidirectional Encoder Representations from Transformer (BERT) model improves subdomain-level detection of exfiltration at ultra-low false positive rates. Previous work mostly examined fine-tuned generic Transformers, but did not aim to isolate the effect of pretraining on the downstream task of classification. We therefore address this gap by developing a controlled pipeline where we freeze operating points on validation and apply them unchanged to the test set in order to measure the impact of pretraining. Our results show significant improvements in the left tail of the Receiver Operating Characteristic (ROC) against randomly initialized baseline. Additionally, within pretrained model variants, increase in the number of pretraining steps had the most impact when more labeled data were available for fine-tuning.**

Keywords—Binary classification, BERT, DNS-exfiltration, Masked Language Modelling, Pretraining

I. Introduction

The Domain Name System (DNS) is a common covert channel for data exfiltration because queries routinely traverse network boundaries and are weakly authenticated or authorized. Classical detectors mostly rely on handcrafted features of individual queries such as: string length, per-character entropy, number and shape of labels, or on streaming statistics [1], [2]. While such methods are effective at catching high throughput exfiltration, it has been shown that these models are vulnerable to low-rate and "slow" tunneling, especially to adversaries that mimic benign lexical statistics [3]. These findings motivate sequence models that would learn structure directly from subdomains [4]–[6], rather than relying on human feature-engineering, and raise a central question for this paper: *Does domain-specific masked language modelling (MLM) pretraining of a character-level BERT encoder causally improve detection of DNS exfiltration compared to training randomly initialized models?*

We aim to answer this by building a controlled pipeline that performs MLM pretraining on two statistically distinct datasets. The models are fine-tuned on classification, in order to measure performance against the randomly initialized baseline. To isolate the effect of pretraining, the main ablations are constructed under the same number of gradient updates against otherwise identical encoders. We retain duplicates on the training set to preserve observed frequencies, whereas we deduplicate at the string level on validation and test sets in order to measure generalization. Evaluation uses *frozen operating points*: thresholds τ_α are chosen on validation to satisfy $\mathrm{FPR} \leq \alpha$ for $\alpha \in \{1\%, 0.1\%\}$ and then are applied unchanged on test. We report normalized partial area under ROC curve across the left tail defined by α ($[0, \alpha]$), Recall at threshold α, and calibration (Brier score). For context only, we include a cross-corpus pretrained model (dataset B) and briefly verify distributional differences between these two datasets (length/depth/entropy).

II. Related Work

Early production systems rely on human-engineered lexical and header features (length, entropy, label count, digit ratio, etc.) and apply conventional machine learning (ML) or anomaly detection over data streams. Other methods such as sliding-window detectors can flag high-volume exfiltration with low false-positive rates, however, their performance degrades for slow tunneling. [1]

Moreover, Generative Adversarial Networks (GANs) have been used to synthesize DNS-like strings that mimic benign statistics while encoding malicious payloads. These models substantially reduce the accuracy of state-of-the-art feature-based detectors, underscoring the brittleness of human-engineered features [3].

Additionally, Transformers have been adapted to tokenized DNS strings and network-flow sequences. These hybrid designs augment attention encoders with 1-D CNN blocks or concatenate numerical/tabular features to token embeddings, yielding improvements over n-gram and CNN baselines [7]. On the other hand, graph-enriched approaches that augment BERT encoders such that they rely on similarity-based classification in the learned embedding space, have been shown to achieve higher accuracy when graph context is available [8]. More broadly, Transformers trained on packet/flow sequences demonstrate that self-supervision on network data is viable without manual feature extraction. [4], [5]

It has been shown that BERT-style encoders pretrained with MLM outperform character-level CNN/RNN baselines and network-security-tuned BERT variants across phishing/malware/DGA tasks [6]. Previous research argues that even modest in-domain pretraining can produce better representations than large, generic models, but prior work typically does not measure the causal impact of pretraining on classification task.

We aim to bridge this gap by quantifying the effect of *subdomain-level* MLM pretraining on exfiltration detection using a controlled testing protocol.

III. Methods and Setup

A. Data Processing and Metrics

We use two sources: (A) a 24h Internet Service Provider (ISP) DNS request log from a Serbian national ISP augmented with controlled and synthetic DNS exfiltration traces (e.g., `iodine`, `DNSExfiltrator`) [9]; and (B) Duck's Party monthly web-crawl subdomains (unique counts with frequencies) [10]. For MLM pretraining we deduplicate at the string level (A: 590,238 uniques; B: 13,382,660 uniques). On the deduplicated corpora, dataset A has longer and deeper subdomains with higher per-character entropy than B (mean length 33.9 vs. 22.2; depth 3.16 vs. 2.82; entropy 3.63 vs. 3.33 bits). Two-sample Kolmogorov-Smirnov tests on length/labels/entropy reject equality (D=0.26-0.28, $p \ll 10^{-3}$), and we also highlight low lexical overlap: 2.64% of unique strings are shared.

Normalization. For MLM, we extract subdomains, lowercase, strip invalid entries, and deduplicate. Splits are partitioned into 80%/10%/10% (train/validation/test) fractions.

Classification set (derived from dataset A). We retain rows with valid binary labels $y \in \{0, 1\}$. Additionally, we *retain duplicates in training* to preserve the empirical per-query distribution—the distribution a deployed detector would see. In contrast, validation and test splits are *deduplicated at the string level* (each unique subdomain appears at most once) so that the reported metrics reflect generalization to *distinct* subdomains rather than being dominated by repeated copies of the same string. This prevents optimistic bias (e.g., inflated metrics on overweighed duplicates) and yields split-to-split comparability when the duplicate histogram is heavy-tailed.

Train split. We construct duplicate groups by exact string matching. Number of raw rows in the train split: 12270029 are grouped into 455046 unique string groups. Inflation ratio $\rho = \frac{12270029}{455046} = 26.96$. Group sizes are heavily-tailed: p50/p90/p99/max = 1/9/334/10000. Label counts with duplicates retained: benign 12,233,579 (99.703%), malicious 36,450 (0.297%). These prevalences are instance-weighted due to duplicates and are not directly comparable to the *deduplicated* validation/test statistics.

Validation split (deduplicated). After string-level deduplication we have $N_{\text{val}} = 56{,}880$ unique subdomains: benign 54,405 (95.65%), malicious 2,475 (4.35%).

Test split (deduplicated). $N_{\text{test}} = 56{,}880$ unique subdomains: benign 53,615 (94.26%), malicious 3,265 (5.74%). We construct the test set to be *disjoint* from pretraining train split, to prevent data leakage.

Metrics and evaluation. Let $s_\theta(x) \in [0, 1]$ denote the model score (estimated $\Pr(y{=}1 \mid x; \theta)$), and $\tau \in [0, 1]$ denote a decision boundary. The confusion counts $\text{TP}(\tau), \text{FP}(\tau), \text{TN}(\tau), \text{FN}(\tau)$ are standardly defined at operating point τ, where the positive estimates are given by $s_i \geq \tau$.

From $\{(y_i, s_i)\}_{i=1}^{N}$ on a split, we form the ROC curve. Because of extreme class imbalance, we define two low-FPR metrics to summarize the left tail of the ROC curve at evaluation and to compare models:

$$\text{Recall@}\tau_\alpha = \max_{\tau:\,\text{FPR}(\tau)\leq\alpha} \text{TPR}(\tau), \tag{1}$$

$$\text{pAUC@}\alpha(\text{norm}) = \frac{1}{\alpha}\int_0^\alpha \text{TPR}(u)\,du, \tag{2}$$

where $\alpha \in \{1\%, 0.1\%\}$. In practice we use trapezoidal integration over interpolated empirical ROC curve, inserting (0,0) if absent, and normalizing by α so that $\text{pAUC} \in [0, 1]$.

Frozen operating points (validation → test). To avoid test-tuning, we *freeze* thresholds on the validation split and *only* apply them on test. For FPR targets $\alpha \in \{1\%, 0.1\%\}$ we compute the thresholds on the validation split as:

$$\tau_\alpha = \arg\max_{\tau:\,\text{FPR}(\tau)\leq\alpha} \text{TPR}(\tau). \tag{3}$$

At this operating point we report Recall@τ_α (test), the realized FPR@τ_α (test), and confusion counts. We also report pAUC over $\text{FPR} \in [0, \alpha]$ (pAUC@α, norm), additionally we assess calibration with Brier score. [1]

B. Model Architecture and Training Setup

We use a character-level BERT model. Raw subdomains are tokenized over DNS-valid characters (a–z, digits, hyphen, underscore, etc.). The encoder has 12 layers with multi-head self-attention, pre-norm residual blocks, hidden size 768, 12 heads, with feed-forward size 3072. The classification [CLS] embedding feeds a binary classifier. For MLM, we use the same architecture but with an output head projecting to the tokenizer vocabulary space. We train both MLM and classifier using cross-entropy loss.

Pretraining. We pretrain the model on the self-supervised MLM task where a fraction of the input tokens are masked and the model is trained to predict omitted tokens. The masking procedure is the same as in the original BERT model: 15% of the tokens are selected for possible replacement, of which 80% are replaced with [MASK], 10% with a random token, and 10% are left unchanged. We pretrain the model on the in-domain corpus for 37.5k and 75k steps (PT-37.5k, PT-75k). We also pretrain the same architecture (HF-PT-37.5k) on a different dataset (B), that was introduced in III-A, to measure the effect of transfer from a larger, more heterogeneous corpus and domain mismatch.

Fine-tuning. Finally, we fine-tune the pretrained models on the binary classification task using the labels from corpus A. To isolate the effect of pretraining, we also train a randomly initialized model with the same architecture. We fine-tune all models for 112.5k steps (FT=112.5k) except the randomly initialized one which we train for 150k steps

[1] https://wandb.ai/lohsmi-school-of-electrical-engineering/Improving-DNS-Exfiltration-Detection

(FT=150k) to measure the impact under the same number of gradient updates. We experiment with three fractions of the labeled data: 10%, 25%, and 50% (stratified by label) alongside 100% to measure label efficiency of pretraining (fractions indicate the fraction of the ≈ 12.3M labels used).

Optimization. Briefly, we use AdamW optimizer with 1% weight decay, linear warmup (1% of total steps), and learning rate 5×10^{-5}. Batch size is 64 for both MLM and classification tasks. Hardware: 1× NVIDIA A100 (40 GB).

IV. Results

At the validation-frozen 0.1% FPR operating point, *in-domain* pretraining yields the highest malicious-class recall and the best calibration, while the cross-corpus model (HF-PT-37.5k) underperforms the randomly initialized baseline—highlighting the importance of domain match. As visualized in Fig. 1, PT-37.5k converts many false negatives into true positives at the cost of a modest increase in false positives at this strict threshold. Low-tail discrimination mirrors this trend: Table I reports higher pAUC@0.1% and pAUC@1%, alongside stronger recall at the frozen operating point. PT-37.5k model also yields the best calibration (Brier 9.7×10^{-4} vs. 1.3×10^{-3} for Randomly Initialized and 2.2×10^{-3} for HF-PT-37.5k).

For completeness, we performed an experiment with 100k steps of pretraining on the dataset B (HF-PT-100k), and the results were comparable to the randomly initialized model. Therefore we further omit the HF model family for brevity, as the experiments suggest if bounded by the number of steps, random initialization is preferable. Specifically, the model attains pAUC@0.1% = 0.9714 and pAUC@1% = 0.9948.

Model	pAUC@0.1% (norm)	pAUC@1% (norm)	Recall@$\tau_{0.1\%}$	Recall@$\tau_{1\%}$
PT-37.5k	**0.9830**	**0.9951**	**0.9926**	**0.9979**
HF-PT-37.5k	0.9650	0.9896	0.9798	0.9942
Randomly Initialized	0.9790	0.9937	0.9853	0.9966

TABLE I: Low-FPR comparison on the held-out test set (100% train fraction).

A. Label Efficiency

Varying the supervision budget (10%, 25%, 50%, 100%), domain-matched pretraining consistently improves low-tail discrimination and probability quality. Compared to a randomly initialized encoder, PT-37.5k raises normalized area under ROC curve in the FPR∈ $[0, \alpha]$ tail consistently over both thresholds $\alpha \in \{0.1\%, 1\%\}$ at 10%, 25%, 50%, and 100% label fraction levels, respectively (Table II). We also note recall improvements at both operating points indicated by the positive difference in True Positives (TP). Brier score also decreases (improves) in every case (ΔBrier $\in [-0.0020; -0.0003]$), pointing to better calibration in addition to stronger discrimination.

We can also observe the effect of threshold transfer variance in the Table II. That is, the effect of the slight drift of the realized FPR on the test set with the threshold recorded on validation set. The drift lowers as the number of labels increases, and generally favors the pretrained model, except under extreme scarcity (10% labels), where we observe a small increase in realized FPR (ΔFPR@1% = +0.0042), trading +13 TP for +223 FP at the same fixed operating point. Crucially, once the label budget reaches 25%-50%, pretraining delivers strictly better performance at the same operating point, both higher recall and *lower* realized FPR with substantial confusion matrix gains (e.g., at 50%: +17 TP and −194 FP). Benefits persist (albeit smaller) at full-data training (100% labels). Together with the 0.1% FPR results in Fig. 1, these findings suggest that domain-matched pretraining gives the largest boost when labels are scarce, while remaining competitive even in the full-data regime.

Frac.	**Δ pAUC@0.1%**	**Δ pAUC@1%**	**Δ FPR@$\tau_{1\%}$**	**Δ TP@$\tau_{1\%}$**	**Δ FP@$\tau_{1\%}$**
10%	+0.1004	+0.0119	+0.0042	+13	+223
25%	+0.0751	+0.0143	−0.0022	+14	−117
50%	+0.0418	+0.0129	−0.0036	+17	−194
100%	+0.0040	+0.0015	−0.0003	+4	−18

TABLE II: Label efficiency: PT-37.5k − randomly initialized model on the held-out-test set (thresholds frozen on validation). Positive Δ means *pretrained* > *random.*

B. Pretraining Budget

Train frac.	Metric	PT-37.5k	PT-75k
100%	pAUC@0.1% (norm)	0.9830	**0.9892**
	pAUC@1% (norm)	0.9951	**0.9967**
	Recall@$\tau_{1\%}$ (test)	0.9979	**0.9982**
	FPR@$\tau_{1\%}$ (test)	0.0086	**0.0076**
50%	pAUC@0.1% (norm)	0.9845	**0.9866**
	pAUC@1% (norm)	0.9960	**0.9965**
	Recall@$\tau_{1\%}$ (test)	0.9975	**0.9985**
	FPR@$\tau_{1\%}$ (test)	**0.0053**	0.0093
25%	pAUC@0.1% (norm)	0.9866	**0.9867**
	pAUC@1% (norm)	0.9959	**0.9967**
	Recall@$\tau_{1\%}$ (test)	0.9979	**0.9991**
	FPR@$\tau_{1\%}$ (test)	**0.0077**	0.0092
10%	pAUC@0.1% (norm)	**0.9650**	0.9628
	pAUC@1% (norm)	0.9902	**0.9921**
	Recall@$\tau_{1\%}$ (test)	**0.9972**	0.9963
	FPR@$\tau_{1\%}$ (test)	0.0112	**0.0080**

TABLE III: Pretraining-budget scaling on the held-out test set.

At a high level, increasing the pretraining budget from 37.5k steps to 75k steps generally yields small but consistent numerical gains in the low-FPR region, with the clearest effect in the full-data regime. In Table III, the 75k-step-pretrained variant tends to outperform the 37.5k-step-pretrained model on both pAUC@0.1% (ultra-low tail) and pAUC@1%, with the gap most visible at 100% level. This is the regime in which the larger pretraining budget can be fully leveraged by the downstream supervised task.

For 25%-50% label levels, the differences between the two models are smaller and come with modest realized FPR increase, in contrast to the 100% level, where the 75k model delivers higher recall alongside reduction in the realized FPR. More precisely, at 25%-50% level we observe slight numerical gains in the partial AUC and recall, at the cost of the mentioned realized FPR increase. The observed

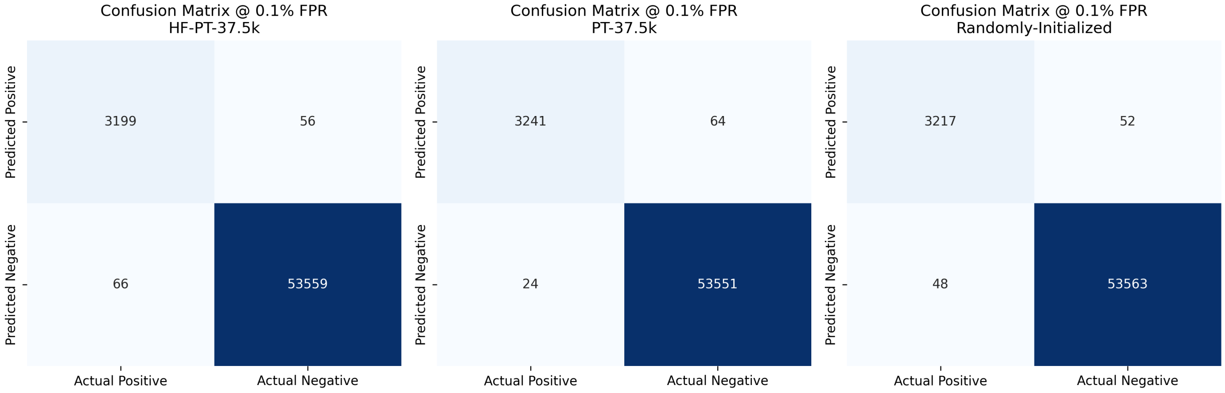

Fig. 1: Confusion matrix heatmaps at 0.1% FPR for Randomly initialized, PT-37.5k and HF-PT-37.5k (full-data training).

behaviour fits the intuition that, as the number of labels decreases, the gains from the longer pretraining cannot be fully utilized by the downstream classification task. At these intermediate budgets, the two models behave similarly, and it is difficult to differentiate between genuine increase in performance and training variance.

Finally, at 10% we observe the trade-off—one model performs better on pAUC@0.1%, the other on pAUC@1%, highlighting the sensitivity of pretraining to label budgets, especially under scarcity. At the given level the same pattern holds for the realized FPR@$\tau_{1\%}$ and Recall@$\tau_{1\%}$ on test. Taken together, these results suggest that longer pretraining is especially beneficial under higher label budgets, while at very low budgets the benefits are more mixed and can depend on the specific metric of interest.

V. Conclusion

We showed that character-level BERT encoders *pretrained in-domain* with MLM materially improve DNS-exfiltration detection where it matters most—at ultra-low FPRs. Subjected to a deployment-faithful protocol that freezes operating points on validation (for $\alpha \in \{1\%, 0.1\%\}$) and transfers them to test, in-domain pretraining delivers higher recall at the same budget, stronger tail separation (normalized pAUC@α), and better calibration (Brier) than randomly initialized models. The results point to gains being the largest under scarce labels yet persisting at full data. By 25%-50% label budgets the metrics generally favor pretraining, with the extreme scarcity 10%-label caveat, where we observe slightly higher realized test FPR traded for more true positives.

As shown in Section IV, cross-corpus pretraining on a different subdomain distribution is comparable to random initialization, but does not match the in-domain gains, underscoring the value of domain match. Scaling the pretraining budget further improves tail metrics when more labels are available. Overall, domain-matched self-supervision seems to be a label-efficient and deployment-aligned path to robust DNS exfiltration detection at very low FPRs, attaining superior results to the randomly initialized baseline.

VI. Acknowledgements

The authors acknowledge the Government Data Center, Kragujevac, Serbia for providing access to its HPC cluster.

Predrag Tadić received funding from the taxpayers of the Republic of Serbia, through the Ministry of Science, Technological Development and Innovation, under contract number 451-03-136/2025-03/200103.

References

[1] K. Žiža, P. Tadić, and P. Vuletić, “Dns exfiltration detection in the presence of adversarial attacks and modified exfiltrator behaviour,” *Int. J. Inf. Secur.*, vol. 22, no. 6, p. 1865–1880, 2023. [Online]. Available: https://doi.org/10.1007/s10207-023-00723-w

[2] Y. Ozery, A. Nadler, and A. Shabtai, “Information-based heavy hitters for real-time dns data exfiltration detection and prevention,” 2023. [Online]. Available: https://arxiv.org/abs/2307.02614

[3] A. Fahim, S. Zhu, Z. Qian, C. Song, E. Papalexakis, S. Chakraborty, K. Chan, P. Yu, T. Jaeger, and S. V. Krishnamurthy, “Dns exfiltration guided by generative adversarial networks,” in *2024 IEEE 9th European Symposium on Security and Privacy (EuroS&P)*, 2024, pp. 580–599.

[4] X. Meng, C. Lin, Y. Wang, and Y. Zhang, “Netgpt: Generative pretrained transformer for network traffic,” *arXiv preprint arXiv:2304.09513*, 2023. [Online]. Available: https://arxiv.org/abs/2304.09513

[5] L. D. Manocchio, S. Layeghy, W. W. Lo, G. K. Kulatilleke, M. Sarhan, and M. Portmann, “Flowtransformer: A transformer framework for flow-based network intrusion detection systems,” 2024. [Online]. Available: https://arxiv.org/abs/2304.14746

[6] A. E. Mahdaouy, S. Lamsiyah, M. J. Idrissi, H. Alami, Z. Yartaoui, and I. Berrada, “Domurls_bert: Pre-trained bert-based model for malicious domains and urls detection and classification,” 2024. [Online]. Available: https://arxiv.org/abs/2409.09143

[7] H. Li, Z. Li, S. Zhang, and X. Pu, “Malicious dns detection by combining improved transformer and cnn,” *Scientific Reports*, vol. 14, 12 2024.

[8] Y. Tian and Z. Li, “Dom-bert: Detecting malicious domains with pre-training model,” in *Passive and Active Measurement: 25th International Conference, PAM 2024, Virtual Event, March 11–13, 2024, Proceedings, Part I.* Berlin, Heidelberg: Springer-Verlag, 2024, p. 133–158. [Online]. Available: https://doi.org/10.1007/978-3-031-56249-5_6

[9] K. Ziza, P. Vuletić, and P. Tadić, “Dns exfiltration dataset,” 2023. [Online]. Available: https://doi.org/10.17632/c4n7fckkz3.3

[10] nyuuzyou, “Subdomain Statistics from scanner.ducks.party,” https://huggingface.co/datasets/nyuuzyou/subdomains, 2025.