LcaLLM: A Deep Latent Cross-Attention Framework for EDR Log Analysis with Large Language Model

Anonymous EMNLP submission

Abstract

Endpoint Detection and Response (EDR) systems play a critical role in safeguarding enterprises against sophisticated threats, particularly advanced persistent threats (APTs). However, detecting abnormal behaviors within long, complex and interdependent event sequences from EDR system log that remains a major challenge. Addressing these challenges, this paper introduces LcaLLM, an novel EDR log analytical framework leveraging the advanced capabilities of Large Language Models (LLMs) in understanding and representing extensive sequential data. LcaLLM proposes three distinguished contributions: (1) a Latent Cross-Attention (LCA) model architecture meticulously designed to enhance the representation of long EDR event sequence, (2) an Event Semantic Alignment mechanism that enriches structured EDR logs with nuanced natural language expressions, aligned with the input of language model for an improved interpretability, and (3) a Multi-Objective Loss Aggregation training approach that enables the model to learn deep complex relationships among EDR events. We also release **EDR47K-40F-v1.0**, a large-scale EDR dataset comprising over 47K event records, covering 40 threat families and normal activities. The LcaLLM framework not only outperforms traditional methods but also sets new benchmarks in threat detection accuracy and classification precision, achieving 98.32% accuracy in threat identification and a 96.73% success rate in classifying threats across 40 families. We further analyze the impact of latent size, layer depth, pooling strategies and robustness to dynamics. We open-source the dataset and code at: https://github.com/victorzhz19995/EDR_LcaLLM.

1 Introduction

In the rapidly evolving landscape of cybersecurity, the challenges in safeguarding organizations' digital assets have grown increasingly complex. As cyber threads become more sophisticated, traditional security measures such as firewalls and antivirus software are no longer sufficient to provide comprehensive protection against advanced persistent threats (APTs), zero-day exploits, and other modern attack vectors (Mei et al., 2021). Endpoint Detection and Response (EDR) systems have been developed as sophisticated tools that are designed to gather extensive data from devices across an enterprise network, providing deep insights into potential intrusions and advanced threats (Hassan et al., 2020). EDR solutions typically combine continuous monitoring and collection of data from endpoints with advanced analytics and automated response capabilities to identify and neutralize threats in real-time. Typically, these continuously collected data provide rich textural information that can be used to detect behavioral anomalies.

Traditional EDR systems predominantly rely on heuristic/rule-based approaches (Kaur et al., 2024), or machine learning models(Kaur and Tiwari, 2021). Heuristic/rule-based approaches ofent result in high false positive rates or fall to accurately detect complex threats such as APT attacks. These threats are highly sophisticated, designed to be stealthy and adaptive, easily bypassing static rule sets. Meanwhile, machine learning models necessitate extensive feature engineering, where the quality of the features significantly impacts model performance. Large Language Models (LLM) could be the elixir for those dilemmas. LLMs have shown significant capabilities in understanding and reasoning over textural data. These models are pre-trained on vast datasets and can be finetuned to downstream task with limited supervision. LLMs have already demonstrated success in detecting and classifying various types of network attacks, including DDoS attacks, man-in-the-middle (MITM) attacks, botnet traffic and so on (Wang et al., 2024b), (Piggott et al., 2023) (Moskal et al., 2023).

In this paper, we introduce the LcaLLM framework, which incorporates a novel Latent Cross-Attention (LCA) model architecture designed to enhance the contextualized representation of EDR sequences inside log data and improve the understanding of complex inter-dependencies among events. Currently LLMs typically rely on decoder-only transformer architectures with casual attention mask, which limits the model's visibility to only past tokens. This results in a "recency problem", where the representation of the entire sequence heavily depends on the <EOS>-last token. The LcaLLM framework addresses these challenges through LCA module, which employs latent cross-attention to effectively capture long-range dependencies and reduce reliance on the last token. We design three variations of the LCA-based model, integrating LCA module at different positions of the backbone model, and empirically evaluate their impact on the sequence representation. Additionally, to better align these structured logs with the pre-training data of language models, the LcaLLM framework incorporates an Event Semantic Alignment mechanism. It enriches the semantic meaning of EDR events. Additionally, our framework introduces a Multi-Objective Loss Aggregation method that integrates objectives from both unsupervised and supervised learning. This approach enables the model to learn the nuanced semantics of EDR event sequences while accurately identifying threats by leveraging relevant contextual information. By combining these techniques, LcaLLM not only captures the deep relationships between events but also significantly enhances the detection of sophisticated threats.

Furthermore, we constructed a log-based **EDR47K-40F-v1.0** dataset, which is the first and currently largest EDR dataset wih rich textural information on system level. It comprises over 47K EDR samples covering 40 threat families as well as normal activities. To support future research and advance AI-driven log analysis, we have made this dataset publicly available. Our contributions are as follows:

- We propose a novel model architecture with a Latent Cross-Attention (LCA) module that enhances contextualized representation of EDR event sequences by capturing long-range dependencies. We evaluate three LCA variants to assess their impact on sequence modeling.
- We introduce a multi-objective loss aggregation strategy that enables the model to learn semantic event patterns across diverse threat families and effectively detect threats
- We perform extensive evaluations across various architectures and training strategies in the challenging domain of EDR threat analysis, showing our LcaLLM framework outperforms existing baselines.
- We publicly release **EDR47K-40F-v1.0**, a large-scale dataset with over 40,000 EDR samples spanning 40 threat categories, providing a foundational resource for advancing AI-based log analytics research.

2 Background

EDR systems continuously monitor and log system-level events, generating telemetry data that captures detailed behavioral information like registry modification, configuration changes, and other relevant system activities (Arfeen et al., 2021). These rich textual logs are analyzed to uncover insights into endpoint behavior and potential security breaches. Traditional approaches have applied machine learning algorithms such as Random Forest, Naive Bayes, and SVM to log analysis (Revathi and Malathi, 2013). More recently, deep learning models like RNN, GRU, and LSTM have achieved notable performance in intrusion detection due to their ability to model sequential data (Radhi Hadi and Saher Mohammed, 2022). However, the effectiveness of both traditional and deep learning approaches heavily relies on handcrafted features such as TF-IDF and N-gram representations (Sharif et al., 2024). As EDR log volumes grow and attack patterns become more sophisticated, these methods face scalability challenges and require extensive feature engineering—an effort-intensive process dependent on domain expertise. This highlights the need for more intelligent, adaptive solutions capable of automatically learning meaningful representations from raw log data.

Pretrained language models (PLM) offer strong transfer learning capabilities and can be finetuned for wide range of downstream tasks. BERT-based architectures have demonstrated effectiveness in encoding textual inputs into meaningful representations. Telemetry data are often treated textual logs and transformed into token sequences compatible with language models. Ahmood Sharif et al. (Sharif et al., 2024) pretrained a RoBERT-

style model using masked language modeling and later fine-tuned it on small labeled datasets for specific detection tasks. However, a key limitation of BERT-based models is their restricted context length—typically limited to 512 tokens. This poses challenges when processing long EDR logs. To mitigate this constraint, Amit Portnoy et al. (Portnoy et al., 2024) proposed a sliding window approach, where each log is divided into approximately 80 segments. Each window's representation, captured via the CLS token, is passed through a bidirectional LSTM classification head. While effective in extending coverage, this method introduces information fragmentation by processing subsequences independently, rather than maintaining global context. Recent advances in large language models (LLMs), however, have dramatically increased context length limits—supporting up to 128K tokens or more (Qwen et al., 2025), (Ding et al., 2024). These extended-context models enable holistic processing of long EDR logs while preserving contextual coherence, thereby overcoming prior limitations and enabling richer sequence representation.

3 Method

3.1 Aligning EDR log to Natural Language

LLMs have already demonstrated remarkable capabilities in natural language processing. To leverage these capabilities, our first objective is to align sequential EDR log events with natural langauge representation. Some previous work have studied to reduce the semantic discrepancy between textual and temporal data modality to LLMs through direct prompting (Xue and Salim, 2023) (Xie et al., 2023) and aligning (Jin et al., 2024) (Liu et al., 2024) (Tang et al., 2024). The direct prompting constructs summarized prompts. The later requires fine-tuning and new design of the loss objectives. However, neither approach is well-suited to EDR logs due to their unique characterisitics **temporal dependency**, **sparse** and **structured**. EDR logs consist of sequences of timestamped events that reflect chronological activity, such as a "file write" following a "file open". Preserving this temporal order during preprocessing is critical. A typical EDR log sample may contain over 5,000 events, but only 10-15% are typically relevant for threat detection, making the date highly sparse. Each event is structured as attribute-value pairs (illustrated at Figure 1), capturing detailed properties like file paths, or registry values. Summarization risks losing critical

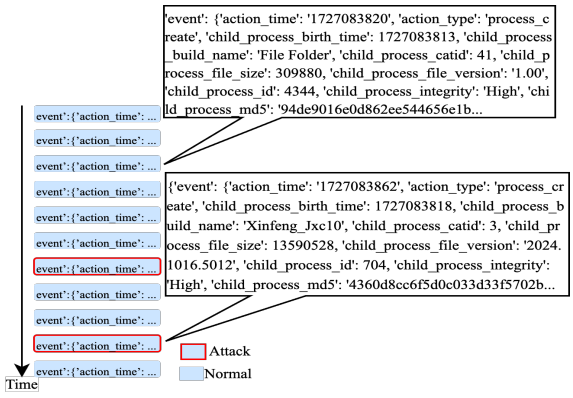


Figure 1: EDR events are in *json* format. Each EDR sample consists of many EDR events with various action. Each event is formed by multiple key-value pairs.

informaiton, and full alignment training introduces excessive computatioanl overhead.

To bridge the semantic gap without compromising detail or efficiency, we propose a lightweight textual alignment method. We observe that many EDR attributes use abbreviated forms (e.g., "$PPID$", "$zero\_day\_op$"), which can lead to ambiguity and hinder model understanding. To address this, we expand abbreviations into fully descriptive terms (e.g. "Parent Process ID", "zero day operation"). This improves interpretability and leverages the domain-agnostic knowledge acquired by LLMs during pre-training. Furthermore, not all attributes are equally relevant for threat analysis. By consulting security experts, we identified and removed unimportant attributes, reducing the average context length of EDR logs by approximately 63%. This selective pruning ensures efficient processing while retaining semantically rich, task-relevant information.

3.2 Large Language Model with Latent Cross-Attention (LCA) module

LLMs are typically based on the decoder-only architecture of vanilla Transformers with causual attention, which restricts token interactions by allowing each token at position i to only see preceding tokens $0, 1, \cdots, i-1$. This limits model's ability to capture global contextual information across the entire input sequence, leading to recency bias, where the sequence embedding disproportionately depends on the last token, diminishing the influence of earlier tokens.

Sentence embeddings aim to encode variable-length sentences into fixed-dimensional vectors

that preserve semantic similarity through distance metrics. They have been widely adopted in downstream tasks (Gupta et al., 2023; Khan et al., 2020; Wang and Koopman, 2017). Traditional approaches rely on high-quality labeled datasets and contrastive learning strategies to enhance embedding quality by pulling similar sentences closer and pushing dissimilar ones apart. Encoder-based models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) are commonly used due to their bidirectional attention mechanism, which enables richer contextual understanding, and their ability to produce high-dimensional embeddings that mitigate anisotropy through large-scale scaling. Despite these advantages, recent studies show that decoder-only LLMs can also achieve strong performance on text embedding benchmarks (Wang et al., 2024a), likely due to their massive pretraining data, multi-stage alignment processes, and extended context lengths. Nevertheless, they still face challenges related to recency bias (Lee et al., 2025), where the final <EOS> token dominates the sentence representation.

In this work, we propose a new LLM architecture that incorporates the Latent Cross-Attention module to obtain better expressive EDR embedding. We explore the structural variation of applying the LCA module at different positions of the backbone LLM. In Figure 2, we illustrate three different structures (Bottom-LCA, Top-LCA, and Weigthed-LCA) that apply the cross-attention pooling on the hidden states of decoder-only LLM to obtain better expressive EDR embedding . Take the Figure 2(b) Bottom-LCA for example, it uses the output of the last layer from decoder-only LLM to form the query $Q \in R^{D_l * D_d}$, and trainable latent matrix M to form $K, V \in R^{D_s * D_d}$ after multiplying the attention weights W_k, W_v, where D_l is the sequence length, D_d is the model hidden dimension, and D_s the latent dimension size. The computation of cross-attention layer is very similar to normal attention computation:

$$O_{cross-attention} = Softmax(QK^T)V \quad (1)$$

, and then followed by a regular feedforward layer. Intuitively, K, V, produced by linear transformation of latent matrix M, can be viewed as an EDR dictionary, which is trained to encapsulate information about all threat families. The query matrix Q, on the other hand, is decoded from the current EDR sample and serves as a query to search this EDR dictionary. This intuition also applies to Figure 2(a), albeit with a different conceptual depths. In Figure 2(a), the LCA module is placed at a lower level of the LLM, which enforces the latent matrix to capture shallow concepts, such as syntactic features of EDR sequences. In contrast, Figure 2(c) introduces learnable weights applied to the hidden states of transformer layers. It is widely accepted that lower layers of an LLM tend to extract syntactic features of the input sequence, while upper layers focus on extracting semantic features. This design aims to aggregate information from different levels of the LLM, thereby providing a more fine-grained representation of EDR sequences by leveraging the multi-layered hierarchical features produced during the model's processing.

3.3 EDR sequence embedding tuning

To enforce LLM to learn the semantic of EDR log samples, we introduce the triplet objective into our learning objectives. Contrastive learning has been widely used in training sentence embedding model (Gao et al., 2021). For each training batch, it consists of a sentence s, a relative / positive document s^+, and a set of irrelevant/negative documents $s^- = \{s_1^-, s_2^-, \cdots, s_n^-\}$. The training objective commonly utilizes the InfoNCE loss (van den Oord et al., 2018), which learns the similarities between query and positive document and differences between query and a batch of negative documents. However, EDR sample has a large number of events that can easily go beyond 8K amount of tokens. Creating a batch of negative samples can be very expensive. **Triple loss** can be viewed as a special case of contrastive learning. Each data sample consists of an anchor sentence S_a, a positive sentence S_p and negative sentence S_n. In natural language, the positive sentence can be a rewritten sentence and expresses the same semantic meaning as the anchor sentence, and the negative sentence could be a totally different sentence on semantic meaning. In our case, we form the positive sample through injecting "noise" into anchor sample and the negative sample from other threat samples or normal samples. The objective of triple loss tunes the model such that the distance between S_a and S_p is smaller than the distance between S_a and S_p, illustrated at equation (2).

$$\mathcal{L}_{\text{triplet}} = \text{ReLU}\left(\cos(a, p) - \cos(a, n) + \epsilon\right), \quad (2)$$

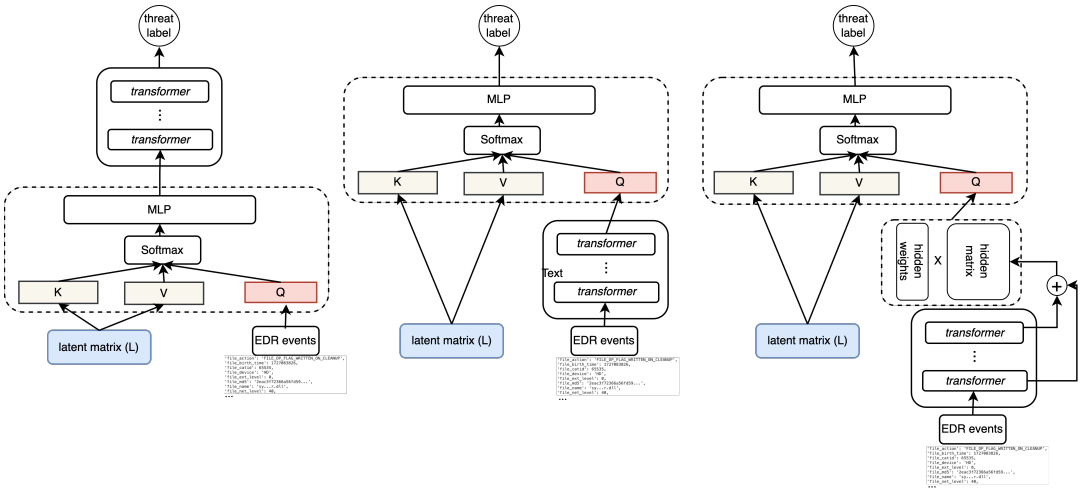


Figure 2: Structural exploration of the Latent Cross-Attention (LCA) LLM architecture. Figure (a) Bottom-LCA: feeds the output of LCA to LLM. Figure (b) Top-LCA: feeds the output of LLM as the input of LCA module. Figure (c) Weighted-LCA: combines all hidden states of each transformer layer with a learnable weights and then feed to LCA module.

where $a = \pi_{\theta,M}(S_a)$, $p = \pi_{\theta,M}(S_p)$, and $n = \pi_{\theta,M}(S_n)$. $||\cdot||$ is a distance measurement (i.e. cosine distance in our case), $\pi_{\theta,M}(S_{a/p/n})$ is the a sentence embedding outputted by model π with parameters of θ and latent matrix M. ϵ is the margin that measures that $\pi_{\theta,M}(S_p)$ is at least ϵ closer to $\pi_{\theta,M}(S_a)$ than $\pi_{\theta,M}(S_n)$. The triple loss objective also enforces the model to learn the semantic difference between EDR threats and capture the semantic meaning of the EDR log in the model's parameters.

3.4 Robust threats classification tuning

One of model's expected capabilities is to correctly detect the threat family from EDR log. The naive approach is to fine-tune LLM as a "giant" classifier. The output embedding layer is a linear transformation layer that maps the model's last hidden states to vocabulary size. Instead of mapping to vocabulary size, the output linear layer maps to the number of classes. We use the first token outputted by model as the threat classification results. We use the cross-entropy loss objective on the first token and aim to algin it as a multi-class classification problem. The LLM encodes the input EDR log and classifies into corresponding thread family. The cross-entropy aims to tune the model to be able to generate the correct threat family.

$$\mathcal{L}_{CE}(S) = argmin_{(\theta,M)} \sum_{i=1}^{|V|} y_i * log(\pi_{\theta,M}(\widehat{y_i}|S)) \quad (3)$$

y_i is the ground truth token id, threat family in our case. $S = [e_0, e_1, e_2, \cdots, e_{n-1}]$ is the EDR input event sequence with length n. $e_{i,0\leq i<n}$ is an EDR log event. $\pi_{\theta,M}(\widehat{y_i}|S)$ is the probability of the threat class outputted by model π with parameters θ and latent matrix M. $|V|$ is the total number of threat classes.

To enforce model learning and classify threat classes based on the semantic information of EDR events, we use aggregated loss by merging triplet loss and cross-entropy loss. In real scenarios, many non-related events often exist between threat-related events. The positive sample is constructed by imputing various non-related events between the threat-related events. The positive sample can also be utilized as training samples that have the same threat class as the anchor sample. The aggregated loss (l_{toal}) for latent cross-attention LLM is computed as followings:

$$\mathcal{L}_{total} = \mathcal{L}_{triplet}(S_a, S_p, S_n) + \mathcal{L}_{CE}(S_a) + \mathcal{L}_{CE}(S_p) \quad (4)$$

, where $l_{triplet}(S_a, S_p, S_n)$ is the triplet loss with anchor, positive and negative EDR sequence, $l_{CE}(S_a)$, $l_{CE}(S_p)$ are the cross-entropy loss of anchor and positive sample. $\pi_{\theta,M}$ denotes the model π with parameters θ and latent matrix M. The θ and M will be tuned based on the multi-objective loss. The loss of positive sample allows model to learn from noise and improve the robustness of threat classification. The learning objective of aggregate loss enables the model to classify threat families under noisy conditions based on the se-

mantic meaning of EDR events.

4 Experiments

4.1 Dataset

Large-scale EDR dataset with covering various threat families is essential for training and validating algorithms. However, the open release of EDR datasets are scarce (Sharif et al., 2024) (Alsaheel et al., 2021), (Zengy et al., 2022), and (Dong et al., 2023). To alleviate the urgen needs of open large-scale EDR dataset, we constructed **EDR47K-40F-v1.0**, a total amount of 47,537 EDR log samples with covering 40 threat families. For the raw EDR log, we first perform Event semantic Alignment to enrich EDR samples with more semantic meaning. We split the dataset into train set (40,405) and test-set (7,132), while ensuring the each threat family also split as the same ratio. The introduction of our EDR log collection process and **EDR47K-40F-v1.0** dataset are provided at Appendix A.1.

We propose the following three approaches for introducing noise into EDR samples to construct the positive training sample, aimed at enhancing the model's robustness: 1. **Imputation.** We impute extra non-related events between each threat-related events . 2. **Swap.** For each EDR log sample, we randomly swap the events to break the dependency 3. **Shuffle.** Each EDR event is a structured attribute-value pairs. We randomly shuffle the order of attributes at each event. We present the detailed data augmentation at Appendix A.2.

4.2 Experimental Setup

We use Qwen2.5 3B Instruct (Qwen et al., 2024) as the backbone model. Qwen2.5 model series have achieved the state-of-the-art performance in various benchmarks. We conduct our experiments on two nodes with 16 NVIDIA H100 80GB GPUs. We select the 32K context length to ensure that we can maximize the coverage of events in each EDR sample, while balancing the training costs. We implemented our latent cross-attention with flash-attention 2 to obtain a higher training efficiency (Dao, 2024). We comprehensively evaluated our framework from two perspectives: model architecture and training approaches. For model architecture, we first compare the LCA model with traditional model architectures in EDR practices. We implemented the LCA module with one latent cross-attention layer and one transformer layer. We conducted the comparison experiments with the traditional machine learning and deep leanring models (details of implementation are presented at Appendix A.3).

Our evaluation metrics include binary classification evaluation with accuracy, threat recall (sensitivity), false positive rate (false alarm), and multi-class classification accuracy for identifying specific threat families. Table 1 depicts the performance of traditional approaches and variational latent cross-attention LLM modules trained and evaluated with EDR47K-40F-v1.0 dataset. The binary classification is evaluated by an additional test dataset consisting of 5000 samples, each labeled only as attack or non-attack. This additional dataset is designed to assess the model's performance in distinguishing between malicious and benign activities.

Table 1 shows that models with the Latent Cross-Attention (LCA) module outperform traditional approaches, highlighting the LCA's effectiveness in learning deep intrinsic relationship matrices among threat families and integrating this knowledge via cross-attention on EDR sequences. Top and Weighted LCA models excel across all metrics, suggesting that LLM works better as extracting textual features and integrated with the intrinsic knowledge by LCA module. However, Weighted LCA underperforms due to potential noise from linearly combining outputs of all layers, which may demand more training data. Conversely, Bottom LCA introduces latent features at early stages, possibly diluting positional information and hindering competitive performance.

We then compared three training strategies: cross-entropy loss, dual-stage training, and multi-objective loss aggregation. Dual-stage training involves unsupervised contrastive learning followed by supervised classification. We aimed to determine how multi-objective loss affects training—whether it hinders convergence or enhances performance. Experiments using Qwen 3B and Top-LCA models, selected for their superior performance, revealed that multi-objective loss aggregation yields the best results by effectively leveraging positive and negative samples, leading to more directed gradients. As shown in Table 2, introducing additional loss objectives consistently improves performance, with multi-objective loss providing balanced enhancements across all metrics and ensuring a more stable training process. These experiments were conducted both with and without the LCA module to demonstrate the generalization

Table 1: The evaluation of traditional approaches and variational latent cross-attention LLM modules in detecting threats and threat families among 40 threat families and white EDR log. For decoder-only based models, we applied the <EOS>-last pooling to obtain the sequence embedding representation. We applied the multi-objective loss aggregation on all the models except the TF-IDF + DT. TF-IDF is a statistical measurement for textural feature exaction. We present the evaluation results of each threat family at Append A.3

Avg. Accuracy	Threat Accuracy	Recall	False Positive Rate	Model name
0.8021	0.9114	0.8992	0.0764	TF-IDF + DT
0.8345	0.9302	0.9324	0.0720	Bi-LSTM
0.8622	0.9338	0.9596	0.0920	BERT + LSTM
0.9353	0.9672	0.9680	0.0336	Qwen2.5 3B
0.9577	0.9784	0.9784	0.0232	Qwen2.5 3B (Bottom LCA)
0.9673	**0.9832**	0.9868	**0.0204**	Qwen2.5 3B (Top LCA)
0.9604	0.9786	**0.9996**	0.0424	Qwen2.5 3B (Weighted LCA)

Table 2: The impact of different training strategies to model's performance: ①cross-entropy loss, ②dual-stage training and ③multi-objective loss aggregation

Avg. Accuracy	Threat Accuracy	Recall	False Positive Rate	Model name
0.8789	0.8874	0.9676	0.1928	Qwen2.5 3B with ①cross-entropy loss
0.9145	0.9376	0.9680	0.0928	Qwen2.5 3B with ②dual-stage training
0.9353	0.9672	0.9680	0.0336	Qwen2.5 3B with ③multi-objective loss aggregation
0.8970	0.9212	0.9684	0.1260	Qwen2.5 3B Top LCA with ①cross-entropy loss
0.9471	0.9726	0.9840	0.0416	Qwen2.5 3B Top LCA with ②dual-stage training
0.9673	**0.9832**	**0.9868**	**0.0204**	Qwen2.5 3B Top LCA with ③multi-objective loss aggregation

capabilities of our proposed methods.

4.3 Ablation Study

We conduct all ablation studies on Top LCA model architecture with multi-objective loss aggregation training approaches.

The impact of LCA module. We hypothesized that the latent matrices M form a latent space that captures intrinsic relationships among EDR threat families, tuned progressively through backpropagation during training. This latent space functions as a knowledge base for EDR log, with the LLM acting as an encoder to extract features from log samples. We investigated whether the size of this latent matrix and the number of layers in the LCA module influence the model's ability to represent EDR sequences.

The number of layers in the LCA module signifies varying levels of information processing. A single layer within the LCA module can be interpreted as performing latent attention-based pooling. Conversely, two or more layers operate as a information fusion mechanism, integrating diverse features and relations from EDR samples and latent space. To evaluate the impact of the number of layers in LCA module, we maintained a fixed dimensionality $D_s = 2048$ while varying the number of layer L within the LCA module. Table 3 show that latent matrix size has minimal impact on threat detection performance but positively affects threat family recognition—higher dimensions M allow to encode more discriminative features. However, increasing the number of LCA layers provided no significant gains, due to the model already has many attention layers.

The impact of pooling methods. To evaluate the impace of different pooling methods, we conducted experiments comparing <EOS>-last, mean, self-attention, and weighted-mean pooling methods. We demonstrated the performance of different pooling methods in Table 4. The weighted-mean pooling outperformed mean pooling, <EOS>-last pooling and self-attention pooling in most metrics. This performance difference can be attributed to the <EOS>-last methods' heavily depends on the last token's embedding, which introduces a recency bias, while mean pooling dilutes important token information by averaging all token embeddings. The LCA module and LLM already have many attention-based layers, the self-attention pooling does not provide additional improvement for sequence representation, and even slightly reduce the performance (same as the number of layers in LCA module).

Robustness to EDR sequence length. EDR events often span long time periods, with sophisticated threats interspersed among numerous normal activities, creating complex and variable tempo-

Table 3: The evaluations of different latent matrix sizes and number of layers in LCA module on model's performance in EDR threats identification and classification. For latent matrix $M \in R^{D_s * D_d}$, D_d must be consistent with model hidden size. We varied D_s to explore the impact of latent matrix: M with sizes of $128 * 2048, 896 * 2048, 2048 * 2048, 5120 * 2048, 8196 * 2048$. We also evaluated the different number of layers (L) in LCA module.

Avg. Accuracy	Threat Accuracy	Recall	False Positive Rate	Model name
0.9644	0.9818	0.9908	0.0272	$R^{128*2048}$
0.9658	0.9836	0.9928	0.0256	$R^{896*2048}$
0.9673	0.9832	0.9868	**0.0204**	$R^{2048*2048}$
0.9782	**0.9866**	**0.9992**	0.0260	$R^{5120*2048}$
0.9785	0.9860	0.9964	0.0244	$R^{8196*2048}$
0.9644	0.9742	0.9716	0.0232	L = 1
0.9673	**0.9832**	**0.9868**	0.0204	L = 2
0.9668	0.9828	0.9820	**0.0164**	L = 3
0.9641	0.9738	0.9723	0.0248	L = 4

Table 4: The evaluations of <EOS>-last, mean, and weighted mean pooling methods on model's performance. We select the 2-layer bottom LCA model with latent matrix $M \in R^{2048*2048}$. We applied the multi-objective loss aggregation training approach for all the pooling methods.

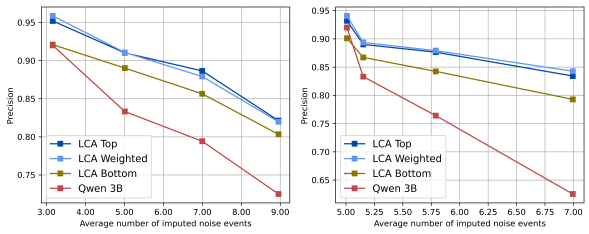
Avg. Accuracy	Threat Accuracy	recall	false positive rate	Model name
0.9673	0.9832	0.9868	**0.0204**	<EOS>-last
0.9774	0.9874	0.9988	0.0240	mean
0.9658	0.9812	0.9840	0.0288	self-attention
0.9760	**0.9880**	**0.9996**	0.0236	weighted-mean

ral patterns. To evaluate our model's robustness under such realistic conditions, we test it on elongated EDR sequences by injecting additional normal events between threat-related ones. The number of inserted events follows a normal distribution with parameters μ (mean) and σ (standard deviation), which we vary to assess performance under increasing sequence length and event-spacing randomness. We evaluate Qwen2.5 3B, Bottom-LCA, Top-LCA, and Weighted-LCA models. Increasing μ tests the model's ability to handle longer contexts, while varying σ simulates irregular event spacing, mimicking real-world noise and uncertainty. As shown in Figure 6, all models experience performance degradation with longer sequences, but LCA-based models demonstrate significantly greater resilience to both sequence elongation and random event distribution. This indicates their superior capability in capturing long-range dependencies and maintaining accuracy under noisy temporal structures.

5 Conclusion

In this paper, we propose LcaLLM, a novel framework for detecting malicious behaviors in endpoint detection and response (EDR) logs using large language models. LcaLLM introduces a latent cross-attention module that addresses the "recency" problem and overcomes limitations of causal masking in decoder-only LLMs. It also presents the capabilities of better capturing complex event relationships and an Event Semantic Alignment mechanism to bridge the gap between raw EDR logs and natural language inputs. These innovations enable superior performance in both threat detection and threat family classification. To advance research in this domain, we introduce EDR47K-40F-v1.0, the largest publicly available EDR dataset with over 47,000 samples across 40 threat families. Our work represents a significant step toward more effective, AI-driven endpoint security solutions. Future efforts will focus on expanding the dataset's scale and diversity, as well as exploring efficient context-length extension techniques.

Figure 3: The number of imputed noise events among threat-related events to the performance of model's performance. We study the impact of μ = 3, 5, 7, 9 with σ = 3 (left), and μ = 5 with $\sigma = 1, 3, 5, 7$ (right).



6 Limitations

Our EDR logs were collected using a kernel-based sandbox environment designed to simulate real endpoint conditions. Despite this, the real world encompasses hundreds of known threats and numerous unknown ones. While the LcaLLM framework shows excellent performance in detecting and recognizing threats from our current dataset, its generalization capabilities need further evaluation. Specifically, more diverse datasets and comprehensive experiments are required to assess LcaLLM's effectiveness in identifying potential unknown threats beyond the discovered threat families. This will ensure that the model remains robust and adaptable to the evolving threat landscape.

References

Abdulellah Alsaheel, Yuhong Nan, Shiqing Ma, Le Yu, Gregory Walkup, Z. Berkay Celik, Xiangyu Zhang, and Dongyan Xu. 2021. ATLAS: A sequence-based learning approach for attack investigation. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3005–3022. USENIX Association.

Asad Arfeen, Saad Ahmed, Muhammad Asim Khan, and Syed Faraz Ali Jafri. 2021. Endpoint detection response: A malware identification solution. In *2021 International Conference on Cyber Warfare and Security (ICCWS)*, pages 1–8.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens.

Feng Dong, Liu Wang, Xu Nie, Fei Shao, Haoyu Wang, Ding Li, Xiapu Luo, and Xusheng Xiao. 2023. DISTDET: A Cost-Effective distributed cyber threat detection system. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 6575–6592, Anaheim, CA. USENIX Association.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *CoRR*, abs/2104.08821.

Vishal Gupta, Ashutosh Dixit, and Shilpa Sethi. 2023. An improved sentence embeddings based information retrieval technique using query reformulation. In *2023 International Conference on Advancement in Computation Computer Technologies (InCACCT)*, pages 299–304.

Wajih Ul Hassan, Adam Bates, and Daniel Marino. 2020. Tactical provenance analysis for endpoint detection and response systems. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1172–1189.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. 2024. Time-llm: Time series forecasting by reprogramming large language models.

Gautam Karat, Jinesh M. Kannimoola, Namrata Nair, Anu Vazhayil, Sujadevi V G, and Prabaharan Poornachandran. 2024. Cnn-lstm hybrid model for enhanced malware analysis and detection. *Procedia Computer Science*, 233:492–503. 5th International Conference on Innovative Data Communication Technologies and Application (ICIDCA 2024).

Harmionee Kaur and Richa Tiwari. 2021. Endpoint detection and response using machine learning. *Journal of Physics: Conference Series*, 2062:012013.

Harpreet Kaur, Dharani Sanjaiy, Tirtharaj Sl, Rohit Kumar Paul, Thakur, Vijay Kumar Reddy, Jay Mahato, and Kaviti Naveen. 2024. Evolution of endpoint detection and response (edr) in cyber security: A comprehensive review. *E3S Web of Conferences*.

Atif Khan, Qaiser Shah, M. Irfan Uddin, Fasee Ullah, Abdullah Alharbi, Hashem Alyami, Muhammad Adnan Gul, and Furqan Aziz. 2020. Sentence embedding based semantic clustering approach for discussion thread summarization. *Complex.*, 2020.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. Nv-embed: Improved techniques for training llms as generalist embedding models.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. 2024. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Yangyang Mei, Weihong Han, Shudong Li, and Xiaobo Wu. 2021. A survey of advanced persistent threats attack and defense. In *2021 IEEE Sixth International Conference on Data Science in Cyberspace (DSC)*, pages 608–613.

Stephen Moskal, Sam Laney, Erik Hemberg, and Una-May O'Reilly. 2023. Llms killed the script kiddie: How agents supported by large language models change the landscape of network threat testing.

Brett Piggott, Siddhant Patil, Guohuan Feng, Ibrahim Odat, Rajdeep Mukherjee, Balakrishnan Dharmalingam, and Anyi Liu. 2023. Net-gpt: A llm-empowered man-in-the-middle chatbot for unmanned aerial vehicle. In *2023 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 287–293.

Amit Portnoy, Ehud Azikri, and Shay Kels. 2024. Towards automatic hands-on-keyboard attack detection using llms in edr solutions.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report.

Mhmood Radhi Hadi and Adnan Saher Mohammed. 2022. A novel approach to network intrusion detection system using deep learning for sdn: Futuristic approach. In *Machine Learning amp; Applications*, CMLA 2022. Academy and Industry Research Collaboration Center (AIRCC).

S Revathi and A. Malathi. 2013. A detailed analysis on nsl-kdd dataset using various machine learning techniques for intrusion detection. *International journal of engineering research and technology*, 2.

Mahmood Sharif, Pubali Datta, Andy Riddle, Kim Westfall, Adam Bates, Vijay Ganti, Matthew Lentz, and David Ott. 2024. Drsec: Flexible distributed representations for efficient endpoint security. *2024 IEEE Symposium on Security and Privacy (SP)*, pages 3609–3624.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. 2024. Salmonn: Towards generic hearing abilities for large language models.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving text embeddings with large language models.

Shenghui Wang and Rob Koopman. 2017. Semantic embedding for information retrieval. In *BIR@ECIR*.

Tongze Wang, Xiaohui Xie, Lei Zhang, Chuyi Wang, Liang Zhang, and Yong Cui. 2024b. Shieldgpt: An llm-based framework for ddos mitigation. *Proceedings of the 8th Asia-Pacific Workshop on Networking*.

Qianqian Xie, Weiguang Han, Yanzhao Lai, Min Peng, and Jimin Huang. 2023. The wall street neophyte: A zero-shot analysis of chatgpt over multimodal stock movement prediction challenges.

Hao Xue and Flora D. Salim. 2023. Promptcast: A new prompt-based learning paradigm for time series forecasting.

Jun Zengy, Xiang Wang, Jiahao Liu, Yinfang Chen, Zhenkai Liang, Tat-Seng Chua, and Zheng Leong Chua. 2022. Shadewatcher: Recommendation-guided cyber threat analysis using system audit records. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 489–506.

Ke Zhang, Jianwu Xu, Martin Renqiang Min, Guofei Jiang, Konstantinos Pelechrinis, and Hui Zhang. 2016. Automated it system failure prediction: A deep learning approach. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1291–1300.

A Appendix

A.1 EDR Log Collection

We developed a program to continuously monitor real-time system activities, including random registry modifications, file creation or deletion on sensitive location, suspicious network requests, and so on. To be more elaborate, we setup multiple isolated environments for emulating Windows 10 system configurations with typical enterprise intranet setups and commonly used applications. We simulate comprehensive attack scenarios such as malware injection, privilege escalation, and lateral movement. These simulated attacks are executed using automated scripts along with injected malware samples, generating a significant volume of system event data.

To gather raw data, we utilize the Windows kernel callback mechanism. This data is subsequently transmitted to a user-space program through memory-mapped files and high-performance pipelines, ensuring real-time storage in JSON format. To uphold data integrity and enable real-time processing, our method employs a

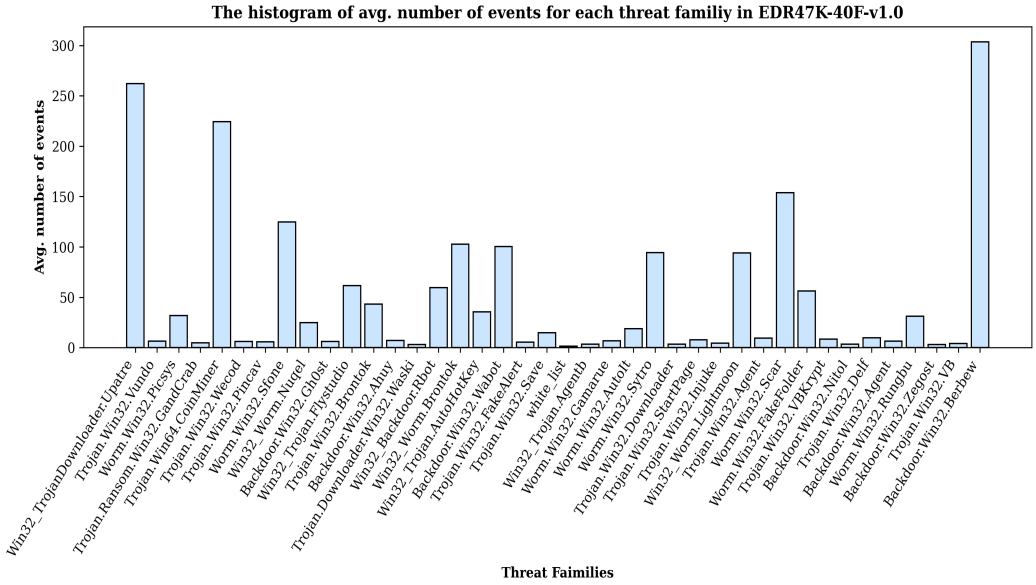


Figure 4: The average number of events in each EDR sample across the 40 collected threat families within the sandbox environment. We classify the EDR event sequence to its corresponding threat family based on the behavior of events.

highly efficient non-paged memory pool at the kernel level for caching events, which markedly reduces context-switching overhead. Furthermore, we have introduced a ring buffer to streamline data transmission, effectively preventing potential event loss or bottlenecks. We list the average number of events in each EDR sample across 40 collected threat families after Event Semantic Alignment at Figure 4. We observe the number of events are varying across different threat families.

A.2 EDR log Augmentation

We present the details of constructing positive training samples, aiming to improve the robustness of model in understanding the complex and dynamic patterns inside EDR event sequence. We proposed three data augmentation approaches including imputation, swap and shuffle and randomly applied to EDR sample with the probability weight [0.4, 0.2, 0.4]. We set a low probability for swap, due to it may break the attack chain, which will make the threat loss its semantic meaning. Assume we have an EDR event sequence $S = [e_0, e_1, \cdots, e_{n-1}]$, which contains only threat-related events.

Imputation. We define a probability p_{impute} that random number of normal events will be imputed between e_i, e_j events, where $0 \leq i, j < n, j = i + 1$ including the beginning and the end. We set $p_{impute} = 0.7$ and the number of events will be imputed is randomly chose between $[1, L_{max} - len(S)]$, where L_{max} (set to 128) the max number of events for EDR sequence and $len(S)$ is current number of events in the sequence. If $len(S)$ is larger than L_{max}, then it will perform other two approaches with probability of [0.3, 0.7].

Swap. We randomly select a pivot event $e_i, 2 \leq i < n$. Then, there is a probability p_i that e_i will be firstly exchanged with e_{i-1}, and a probability p_j that the pivot event will be exchanged with e_{j-2}. We set the p_i and p_j as 0.3 and 0.15.

Shuffle. Each EDR event is a structured attribute-value pairs. The order of these keys are not meaningful. To make model neglect the order of attributes, we randomly shuffle the order of attributes at each event.

A.3 Baseline Approach Details

TF-IDF + Decision Tree (DT) depends on the features extracted from EDR logs. The textual features such as term frequency-inverse document frequence (TF-IDF) are commonly used as the EDR log feature representations (Zhang et al., 2016), (Sharif et al., 2024). For fair comparison, we produced these features with Qwen's tokenizer and apply the Decsion Tree classification algorithm as the baseline result.

CNN + Bi-LSTM has been proven to be effective in identifying attacks in long-spanning high-dimensional sequence data (Karat et al., 2024). The convolution neural network reduces the dimension

Table 5: The precision of classifying the EDR logs for each threat family. The label (name) of each threat is based on the log behavior patterns.

Malware Family	Precision	Malware Family	Precsion
Worm.Win32.Gamarue	0.9940	Trojan.Win32.Brontok	0.9889
Trojan.Win32.VB	0.9889	Backdoor.Win32.Gh0st	0.9889
Win32_Trojan.Agentb	0.9389	Trojan.Win32.FakeAlert	0.9770
Worm.Win32.Sfone	0.9556	Win32_Trojan.AutoHotKey	0.9056
Trojan.Win32.FakeAlert	0.9770	Trojan.Win32.FakeAlert	0.9770
Trojan.Ransom.Win32.GandCrab	0.9722	Backdoor.Win32.Nitol	1.0000
Worm.Win32.AutoIt	1.0000	Trojan.Win32.Downloader	0.9944
Worm.Win32.Sytro	0.9556	Trojan.Win32.VBKrypt	0.9944
Backdoor.Win32.Zegost	0.9944	Win32_Worm.Lightmoon	0.9611
Trojan.Win64.CoinMiner	0.9944	Trojan.Win32.Save	1.0000
Win32_Backdoor.Rbot	0.9056	Worm.Win32.FakeFolder	0.8278
Trojan.Win32.Pincav	0.9389	Backdoor.Win32.Berbew	0.8944
Trojan.Win32.Vundo	0.9934	Win32_Worm.Brontok	0.9833
Worm.Win32.Scar	1.0000	Backdoor.Win32.Ahuy	0.9389
Worm.Win32.Picsys	1.0000	Win32_TrojanDownloader.Upatre	0.9278
Win32_Worm.Nuqel	0.9556	Backdoor.Win32.Agent	0.9500
Trojan.Downloader.Win32.Waski	1.0000	white_list	0.9832
Trojan.Win32.Agent	0.9278	Worm.Win32.Rungbu	1.0000
Win32_Trojan.Flystudio	0.9389	Trojan.Win32.StartPage	0.9556
Trojan.Win32.Delf	1.0000	Backdoor.Win32.Wabot	0.9667
Trojan.Win32.Wecod	1.0000	Trojan.Win32.Injuke	1.0000

of the input sequence and the bidirectional LSTM model is capable of capturing the dependencies in sequential data. We adopted the Qwen's embedding layer within the LSTM model.

Bert + LSTM, proposed by (Portnoy et al., 2024), encode the EDR log sample with sliding window approach. For the classification head, they attach a bidirectional LSTM layer of each window $[CLS]$ token. We experimented with different number of layers of LSTM model and report the optimal performance.

Qwen2.5 3B can be easily transformed to a classification model by modifying the output size of "lm_head" layer from vocabulary size to number of threat families. We also add two extra transformer layers in Qwen2.5 3B instruct model to keep the model size as the same as LCA-based models.

A.4 Evaluation Results Details

We present detailed classification results for the model's performance on EDR logs across 40 threat families in Table 5. The model achieves a precision score exceeding 0.9 for the majority of threat families. However, "Worm.Win32.FakeFolder" and "Backdoor.Win32.Berbew" have precision scores below 0.9. Specifically, "Worm.Win32.FakeFolder" is frequently misclassified as "Win32_Worm.Lightmoon", while some "Backdoor.Win32.Berbew" samples are classified as "Trojan.Win32.VB". Upon closer examination of the misclassified log samples, it was observed that these threats exhibit similar behavior patterns, leading to confusion. For instance, both "Worm.Win32.FakeFolder" and "Win32_Worm.Lightmoon" involve attack events primarily categorized as "CreateProcess", "PeFileWritten", and "LoadImage". This similarity in behavioral characteristics makes it challenging for the model to accurately distinguish between them.