2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

CLUE: A High-Performance, Efficient, and Robust APT Detection Framework via Fine-Tuning Pretrained Transformer and Contrastive Learning

Wenzhuo Cui[1,2], Maihao Guo[3], Jingjing Feng[1,2], Shuyi Zhang[1,2], Zheng Liu[1,2], and Yu Wen[1]（✉）

[1] State Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{cuiwenzhuo, fengjingjing, zhangshuyi, liuzheng, wenyu}@iie.ac.cn
[3] Institute of Automation, Chinese Academy of Sciences, Beijing, China
maihao.guo@ia.ac.cn

Abstract. In recent years, provenance graph-based approaches have become the standard approach for Advanced Persistent Threat (APT) detection and investigation. However, existing studies face several challenges: (1) the high computational cost of the training process makes it difficult to update the model in a timely manner, leading to delayed attack detection; (2) the imbalance in training data results in a scarcity of attack samples, which negatively impacts model performance; and (3) high false positive rates hinder practical deployment in real-world applications. To address these challenges, we propose CLUE, a novel APT detection framework that enables high-quality, multi-granular detection. CLUE employs lemmatization techniques to normalize sequence data extracted from provenance graphs and then directly fine-tunes a pretrained Transformer model, significantly reducing both training time and dependence on scarce attack data. Furthermore, CLUE incorporates contrastive learning to enhance generalization capability in data-scarce scenarios by optimizing inter-sample distances while accelerating model convergence. Our evaluation of CLUE across 10 real-world APT attack scenarios demonstrates that compared to state-of-the-art methods, CLUE maintains superior detection performance while achieving a 7.4× reduction in average training time and requiring 45.2% less training data (particularly attack samples). These results validate CLUE's efficiency, robustness, and practical value in APT detection.

Keywords: APT detection, Provenance graph, Contrastive learning, Pretrained Transformer.

1 Introduction

Advanced Persistent Threats (APTs) have emerged as one of the most formidable challenges in the field of cybersecurity due to their stealth, complexity, and persistence[1].

These attacks are typically orchestrated by highly skilled adversaries who meticulously plan multi-phase infiltrations targeting critical infrastructures of corporations and government agencies. Once successfully executed, APTs can remain undetected for extended periods, causing significant damage to sensitive data and systems, leading to severe economic and societal consequences.

In recent years, data provenance technologies have found widespread application in the detection and investigation of APTs[2-17]. By analyzing audit logs to generate provenance graphs, these technologies enable the intuitive capture of causal relationships between system entities. This approach provides rich contextual information for APT detection, demonstrating significant advantages in enhancing detection performance and supporting post-attack investigations. Early research on APT detection[2-5] based on provenance graphs predominantly employed rule-based or signature-based methods, where audit data was matched against predefined rules to identify APT activities. However, as cyberattack techniques continue to evolve, these traditional methods have increasingly shown their limitations. They struggle to address the complex, dynamic nature of attack patterns and the rapidly shifting threat landscape, thus underscoring the urgent need for more advanced detection technologies.

Recently, deep learning-based approaches[6-17] for detecting APTs have gained increasing attention due to their powerful ability to model complex behaviors. These approaches significantly improve APT detection performance by modeling APT patterns or system behaviors and capturing the deep semantic relationships between system entities through classification or anomaly detection techniques. However, despite the promising results achieved by deep learning-based approaches, their widespread application remains constrained by three key limitations:

High Computational Cost: Existing APT detection models require building from scratch to learn the complex causal relationships, which imposes high demands on training time and computational resources. However, the frequent emergence of zero-day attacks necessitates rapid model updates. Prolonged training cycles can result in detection latency and increasing the risk of system exposure to threats.

Scarcity of Attack Data: Malicious behavior data is significantly less abundant than normal activity data, resulting in highly imbalanced training datasets. Existing methods typically rely on a small amount of domain-specific data for training, or use manually defined rules for data augmentation[8]. This leads to a lack of sufficient and effective training samples for the model to learn, thereby limiting its detection capability and generalization performance.

High False Positive Rate: Anomaly-based approaches[9, 10, 14], although not reliant on prior attack features, often yield high false positive rates. In practical applications, this high false alarm rate not only increases the workload of security analysts significantly but also risks overlooking actual attacks, which can further jeopardize system security.

To address these challenges, we propose CLUE, an APT detection approach based on the general pretrained Transformer. CLUE detects APT attacks through direct fine-tuning of general pretrained models combined with contrastive learning. Specifically, CLUE employs lemmatization to convert sequential data from audit-log-generated causal graphs into natural language sequences aligned with the RoBERTa model's vocabulary for model input. This approach effectively leverages the pretrained model's rich semantic knowledge acquired from large-scale general corpora.Unlike previous methods that use randomly initialized models, CLUE's direct fine-tuning of RoBERTa substantially reduces training time while mitigating detection capability and generalization issues stemming from scarce domain-specific data.

Moreover, CLUE introduces a contrastive loss function on top of the cross-entropy loss. By optimizing inter-sample relative distances, CLUE learns more discriminative embeddings. The introduction of contrastive learning enhances generalization in few-shot scenarios and accelerates convergence.

We evaluated CLUE in scenarios comprising four single-host attacks and six multi-host attacks[8], computing key evaluation metrics at three levels: event-level, entity-level, and sequence-level. Experimental results demonstrate that CLUE significantly outperforms existing state-of-the-art approaches in both detection performance and efficiency. Notably, CLUE achieves an average training time of merely 233.5 seconds, representing a 7.4× reduction compared to current methods. More importantly, despite using 45.2% less attack data (which is already scarce in practice), CLUE maintains exceptional detection performance, achieving 99.92% precision and 99.97% recall. CLUE also significantly reduces the false positive rate while maintaining outstanding detection capabilities, aligning with the practical needs of real-world applications.

In this work, we make the following contributions:

- We propose CLUE, a novel APT detection framework based on the general pretrained Transformer. The framework enables multi-granularity APT detection through direct fine-tuning of the pretrained RoBERTa model augmented with contrastive learning.
- We utilize lemmatization techniques to transform the raw sequences into natural language representations that are highly consistent with the vocabulary of the general pretrained language model. This enables direct fine-tuning on the pretrained RoBERTa model, significantly improving training efficiency while fully leveraging the rich semantic information embedded in the general pretrained Transformer, addressing the issue of domain-specific data scarcity.
- We design and introduce a contrastive learning loss function specifically tailored for pretrained Transformer models, in addition to the conventional cross-entropy loss. This function optimizes the embedding distances between samples, learning more pronounced feature differences between attack and normal behaviors. As a result, it improves detection performance and accelerates the convergence of the model.
- We perform a systematic evaluation of CLUE on ten real-world APT attack scenarios. Experimental results show that CLUE not only effectively detects APT activities, but also reduces training time and training data requirements, demonstrating its practical value in real-world applications.

2 Related Work

CLUE is mainly applied in the field of APT detection. Existing APT detection methods can be classified mainly into the following categories.

2.1 Rule-based Approaches

Most early studies matched attacks using predefined rules. Holmes[2] achieved precise positioning of the attack chain by constructing a multi-layer semantic association framework and using graph matching and false alarm suppression mechanisms. However, rule-based approaches rely on a large amount of manual updates to the rule base and are difficult to adapt to dynamic threat environments.

2.2 Learning-based Approaches

Log-based approaches. DeepLog[6] uses an LSTM network to model system log sequences and realizes anomaly detection through the joint analysis of log key-value pairs and metric indicators. AIRTAG[14] pretrains a BERT model on log data for log-level attack detection and investigation. DrSec[17] converts endpoint event sequences into distributed representations of processes by pretraining a language model.

Provenance-based Approaches. ATLAS[8] extracts normal and abnormal sequences from the provenance graph and uses LSTM to learn the relationships between sequences to identify attack events. Flash[16] achieves efficient and real-time APT detection by combining Word2Vec semantic encoding, GNN structure learning and embedding recycling techniques.

Although the learning-based approaches have made remarkable progress, there are still some problems. Unsupervised approaches such as AIRTAG[14] generate a large number of false alarms and are difficult to apply in practical scenarios. Approaches like DrSec[17] and ATLAS[8] have a high training time cost, which may even last for several days, potentially resulting in detection delays. The Word2Vec used by Flash[16] fails to dynamically capture context dependencies and has a relatively weak ability to model the temporal relationships of long-distance event sequences.

3 CLUE Overview

CLUE generates provenance graphs from audit logs and prunes them to capture the key causal relationships and operation sequences between subjects and objects. Subsequently, neighborhood graphs are constructed for both attack and non-attack entities within the provenance graph, with operations sorted in chronological order, resulting in attack and non-attack sequences that preserve causal relationships.

To improve the generalization ability and training efficiency, CLUE applies lemmatization to the extracted sequences, converting numerous unique tokens into standardized representations. This process ensures that the processed sequences are highly compatible with the vocabulary of the general pretrained RoBERTa model, thereby effectively utilizing its rich semantic information. Based on the processed sequence data, we propose a method that directly fine-tunes the general pretrained RoBERTa model without requiring additional domain-specific pretraining. This approach substantially reduces training time while fully exploiting the pretrained model's semantic representation capabilities, ensuring high-quality feature embeddings and effectively mitigating domain data scarcity.

Additionally, CLUE incorporates a contrastive learning loss function on top of the cross-entropy loss. By optimizing the relative distances between samples, the model learns more distinguishable embedding representations. This improvement significantly enhances the model's generalization performance in few-shot scenarios and strengthens its adaptability to novel attack patterns. Fig. 1 gives an overview of CLUE architecture.

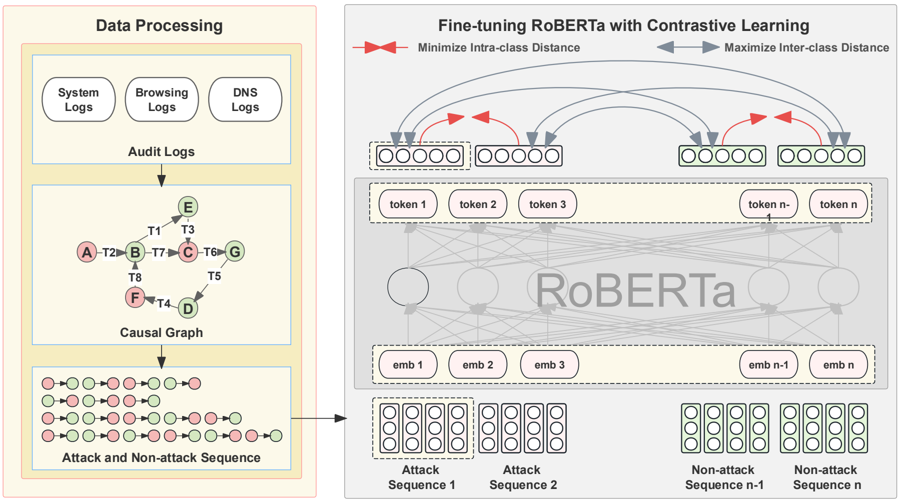

Fig. 1. Overview of CLUE.

4 Data Preprocessing

4.1 Log Preprocessing

CLUE first converts audit logs into causal graphs to capture the causal relationships and operation sequences between system entities (subjects and objects). It then prunes redundant nodes and edges from the graph to reduce computational overhead and extract key information[8]. The processed causal graphs effectively reveal the relationships between events in the logs, thereby supporting the modeling and detection of attacks and abnormal behaviors.

Specifically, the nodes in the causal graph represent entities within the system, while the edges capture the operations performed by subjects on objects, defined by the operation type (e.g., "read," "write") and timestamp, with the direction pointing from the subject to the object. During the pruning process, CLUE eliminates redundant edges by retaining only the first occurrence of an operation edge between a subject and an object, and merges similar nodes by combining multiple nodes with identical incoming and outgoing edges into a single node. This approach significantly reduces redundant information in the graph and optimizes its structure to facilitate subsequent analysis.

4.2 Sequence Construction

After preprocessing the logs, a simplified causal graph containing both attack and non-attack behaviors is obtained. Based on this causal graph, attack and non-attack sequences are extracted for model training.

Attack Sequence Construction. First, all attack entity nodes in the causal graph are aggregated to generate subsets corresponding to different attack behaviors. For each attack subset, a neighborhood graph is extracted to capture the causal relationships with other entities, thereby obtaining contextual information regarding the attack. The operations in the neighborhood graph are then sorted by timestamps to ensure that the operation order aligns with the actual temporal sequence. This process generates complete attack sequences. The process of constructing attack sequences is illustrated in parts C and D of Fig. 2.

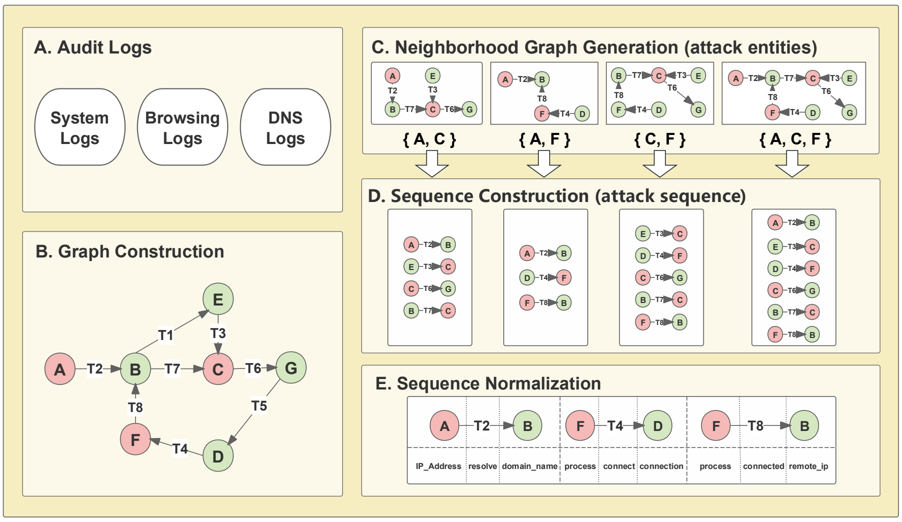

Fig. 2. Data preprocessing of CLUE.

Non-Attack Sequence Construction. Since the goal is to learn the boundaries between attack and non-attack sequences, part of the non-attack entities is incorporated into the attack entities to create patterns that deviate from attack sequences, serving as the basis for extracting non-attack sequences. During non-attack sequence extraction, one non-attack entity is added to each attack entity subset to form non-attack entity subsets. Subsequently, we employ the same methodology used for attack sequence extraction to construct the non-attack sequences.

5 Fine-tuning General Pretrained Transformer with Contrastive Learning

5.1 Fine-tuning Pretrained RoBERTa

Sequence Normalization and Embedding. To improve the model's generalization ability and prevent overfitting, CLUE utilizes lemmatization techniques to normalize sequences extracted from audit logs into domain-generic vocabulary forms[15], which abstract deeper semantic information. This process not only mitigates overfitting and generalization issues but also ensures that the tokens in the extracted sequences align closely with the vocabulary of the pretrained RoBERTa model. E of Fig. 2 demonstrates an example.

Thanks to this strategy, CLUE can directly leverage the embedding representations of the open-source pretrained RoBERTa model (FacebookAI RoBERTa-base) by inputting the domain-specific attack and non-attack sequences. The rich semantic representation capabilities of RoBERTa provide the foundation for effective sequence embedding. Compared to previous studies using word2vec[16], Transformer embeddings through their dynamic word vector representations based on bidirectional context—offer a more precise representation of complex patterns and semantic differences within sequences. This significantly enhances the classifier's ability to differentiate between attack and non-attack sequences.

Fine-tuning a General Pretrained RoBERTa Model. In the APT detection field, previous works[14, 17] typically pretrain models using domain-specific data before fine-tuning for specific tasks. However, this approach faces two critical issues: First, domain pretraining is computationally intensive (e.g., DrSec required 7 days of pretraining[17]. Once application scenarios or attack patterns evolve, retraining becomes necessary, leading to detection delays that compromise defense effectiveness. Second, high-quality data is scarce in APT detection, with attack samples significantly fewer than normal samples. Domain pretraining tends to cause overfitting, impairing model generalization capabilities.

To address these challenges, we propose a strategy of directly fine-tuning a general pretrained model (FacebookAI RoBERTa-base). By leveraging lemmatized natural language representations of attack and non-attack sequence data, we bypass domain pretraining. This approach capitalizes on the existing deep linguistic capabilities of general language models, enabling rapid adaptation to specific detection tasks without domain-specific initialization.

5.2 Supervised Contrastive Learning Loss

In this section, we discuss the approach for learning discriminative representations for attack detection. Traditional cross-entropy loss often results in inadequate generalization ability and robustness in attack detection models[18-21], making them ineffective against novel attacks. To address this limitation, we introduce Supervised Contrastive Learning (SCL) and integrate it with the original cross-entropy loss. By optimizing the relative distances between samples through contrastive learning, we enhance feature discriminability, thereby significantly improving the attack detection performance in few-shot scenarios.

Supervised Contrastive Learning Loss (SCL Loss). The contrastive learning loss function is designed to enhance the separability of embedding features. CLUE employs a contrastive learning loss function, $\mathcal{L}_{SCL}$, specifically optimized for fine-tuning pre-trained language model[22]. This function not only ensures improved separability of embedding features but also integrates seamlessly with pretrained RoBERTa models. For a batch containing N samples, the SCL loss is defined as follows:

$$L_{SCL} = \sum_{i=1}^{N} -\frac{1}{N_{y_i}-1}\sum_{j=1}^{N} \mathbb{1}_{i\neq j}\,\mathbb{1}_{y_i=y_j}\log\frac{\exp\left(\frac{\Phi(x_i)\cdot\Phi(x_j)}{\tau}\right)}{\sum_{k=1}^{N}\mathbb{1}_{i\neq k}\exp\left(\frac{\Phi(x_i)\cdot\Phi(x_k)}{\tau}\right)}\ (1)$$

Here, $\Phi(x)$ represents the ℓ_2-normalized output of the pretrained encoder for input x; $\tau > 0$ is the temperature parameter used to adjust the separation strength between different samples; N_{y_i} denotes the number of samples with the same label y_i as sample x in the batch.

The SCL loss encourages samples of the same class to cluster in the embedding space, while simultaneously pushing samples of different classes farther apart (see Fig. 3).

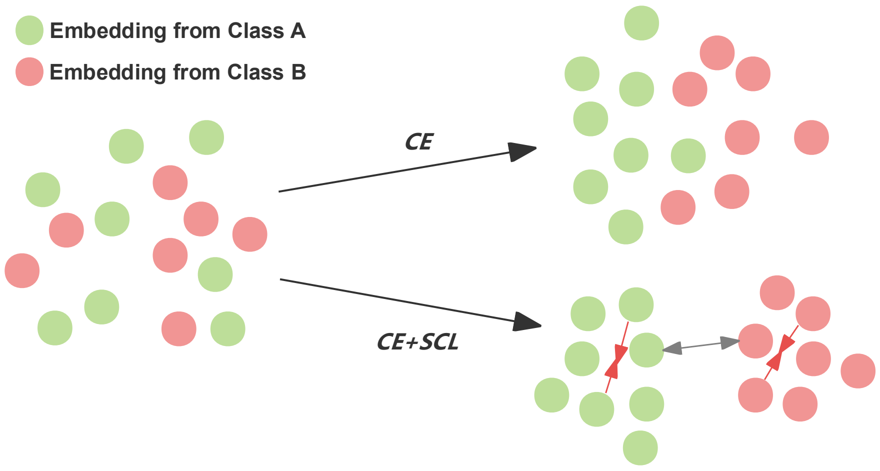

Fig. 3. Minimize intra-class distance and maximize inter-class distance with SCL loss.

Cross-Entropy Loss (CE Loss). While incorporating the supervised contrastive learning loss, we retain the conventional cross-entropy loss $\mathcal{L}_{CE}$ essential for model training to ensure classification accuracy. $\mathcal{L}_{CE}$ is defined as follows:

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\cdot\log\hat{y}_{i,c}(2)$$

Final Joint Loss Function $\mathcal{L}$. During training, the model jointly optimizes the cross-entropy classification loss $\mathcal{L}_{CE}$ and the contrastive loss $\mathcal{L}_{SCL}$, achieving a dynamic balance between feature learning and classification performance. The final joint loss function is defined as follows:

$$L = (1 - \lambda)L_{CE} + \lambda L_{SCL}(3)$$

Implementation on Pretrained RoBERTa. We utilize a pretrained RoBERTa model as the encoder $\Phi(\cdot)$ in $\mathcal{L}_{SCL}$, where the embedding of the [CLS] token is used as the representation of the input sequence. For each sequence in the batch, the CE and SCL losses are computed based on the output of the encoder. By adjusting the temperature parameter τ and the scalar weighting hyperparameter λ, we achieve a balance between classification accuracy and feature representation learning, thereby enhancing the model's discriminative capability for attack samples, generalization performance, and training convergence speed.

6 Evaluation

In this section, we provide a detailed description of the experimental setup and validate the effectiveness of the proposed method through extensive experiments. Subsequently, we evaluate the training efficiency and performance of the method and conduct ablation studies to analyze the importance of individual model components.

6.1 Experimental Settings

All experiments were performed on a computational server equipped with three NVIDIA Tesla V100 GPUs (each with 32GB memory), an Intel Xeon Gold 5218 CPU (64 cores, 2.30GHz), and 752 GB of RAM.

Dataset. We utilized the public ATLAS dataset (S1-S4 and M1-M6), a widely adopted benchmark in APT research, which covers diverse Advanced Persistent Threat (APT) attack scenarios. The dataset contains single-host attacks (S1-S4), multi-host attacks (M1-M6), and normal user activities. Each attack was simulated based on detailed APT activity reports to generate high-quality audit logs. Single-host attacks focused on one victim host, while multi-host attacks involved two hosts, with the second host simulating lateral movement targets. Attacks were performed in Windows 7 virtual machines, lasting approximately 1 hour each, followed by 24-hour audit log collection.

Model Configuration. Our classifier was implemented using Hugging Face's RoBERTa-base architecture[23]. All models were optimized via the Adam optimizer[24] with an initial learning rate of 1×10^{-5} and a linear decay schedule to zero. We employed a batch size of 32 and conducted fine-tuning for five epochs. Through extensive hyperparameter tuning, we determined that $\tau = 0.3$ and $\lambda = 2$ yielded optimal performance with $\mathcal{L}_{SCL}$, and consequently applied these values uniformly across all experiments.

Evaluation Metrics. We evaluated the performance of CLUE using widely adopted metrics in attack detection, including accuracy, precision, recall (True Positive Rate, TPR), False Positive Rate (FPR), and F1 score. Additionally, we incorporate efficiency metrics, such as training time, to holistically assess the model's performance.

6.2 Detection Performance

To evaluate CLUE’s detection performance, we conducted comprehensive testing at three levels: event-level, entity-level, and sequence-level. It should be emphasized that the test data was completely independent from the training data, encompassing previously unseen attack types and activities. This experimental setup effectively validates the model’s generalization capability on unseen test data, which holds significant importance for practical applications. Table 1 presents CLUE’s detection results at the entity-level and event-level. Even with a 45.2% reduction in training data, CLUE achieved precision, recall and F1-score of 99.92%, 99.97% and 99.95% respectively in event-level detection, performing comparably to ATLAS and significantly outperforming AIRTAG (see Fig. 4). These results fully demonstrate the robustness and efficiency of our approach.

Table 1. Entity-based and event-based investigation results.

ID	Entity-based Investigation Results			Event-based Investigation Results		
	Precision(%)	**Recall(%)**	**F1-score(%)**	**Precision(%)**	**Recall(%)**	**F1-score(%)**
M1	90.32	100.00	94.92	99.96	100.00	99.68
M2	97.30	100.00	98.63	99.96	100.00	99.98
M3	97.22	97.22	97.22	99.97	100.00	99.98
M4	100.00	96.43	98.18	100.00	99.82	99.91
M5	90.91	100.00	95.24	99.99	100.00	99.99
M6	93.18	97.62	95.35	99.68	99.99	99.84
S1	100.00	100.00	100.00	100.00	100.00	100.00
S2	92.31	100.00	96.00	99.97	100.00	99.99
S3	95.83	95.83	95.83	99.94	99.88	99.91
S4	80.77	100.00	89.63	99.75	100.00	99.88
Avg.	93.78	98.71	96.18	99.92	99.97	99.95

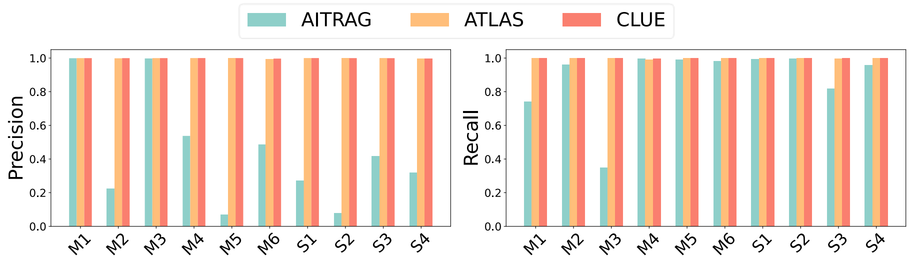

Fig. 4. Precision and recall of ATLAS, AIRTAG and CLUE.

Furthermore, unlike ATLAS and AIRTAG which focus solely on event-level evaluation, our work introduces additional sequence-level assessment. This capability connects multiple attack events to provide more comprehensive attack chain information, enabling security personnel to better understand and investigate complete attack scenarios. The sequence-level evaluation results are shown in Table 2.

Table 2. Training time for M1 to S4.

ID	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	Avg.
Accuracy	99.92	99.58	98.48	98.11	98.68	97.10	99.68	99.72	96.70	94.23	98.22
TPR(%)	95.18	96.34	93.83	96.20	94.87	95.71	95.24	95.35	95.24	93.75	95.17
FPR(%)	0.07	0.41	1.51	1.88	1.31	2.90	0.31	0.27	3.30	5.77	1.77

6.3 Efficiency Analysis

To evaluate CLUE's efficiency, we analyzed its time cost (see Table 3) and compared the training time requirements among CLUE, ATLAS, and AIRTAG (see Fig. 5). The results show that while maintaining comparable or superior detection performance, CLUE's training time is substantially shorter than both ATLAS and AIRTAG, requiring only 17.6% of ATLAS's and 51.8% of AIRTAG's training duration, which conclusively demonstrates CLUE's computational efficiency.

Table 3. Training time for M1 to S4.

ID	M1	M2	M3	M4	M5	M6	S1	S2	S3	S4	Avg.
Times(s)	164.9	227.6	274.9	475.6	106.8	261.7	296.4	171.3	202.9	153.6	233.5

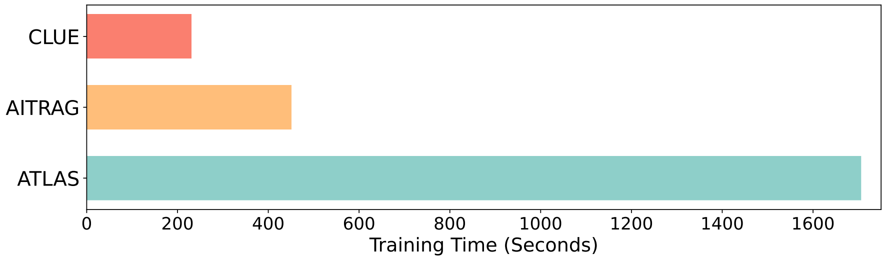

Fig. 5. Average time costs of ATLAS, AIRTAG and CLUE.

The enhanced training efficiency of CLUE confirms its suitability for APT detection scenarios that require frequent model updates. In practical enterprise applications where data volumes typically exceed experimental levels, CLUE's time-cost savings are even more pronounced for large-scale data scenarios.

6.4 Ablation Study

Comparison of Domain-Specific Pretraining and General Fine-Tuning. To evaluate the impact of domain-specific pretraining, we compared two methods: (1) directly fine-tuning the general pretrained RoBERTa model (FacebookAI RoBERTa-base) and (2) pretraining on domain-specific data (lemmatized attack and non-attack sequences) followed by fine-tuning. Results, as shown in Table [tab:comparison_models], demonstrate that the detection performance of the domain-specific pretrained model is nearly identical to that of directly fine-tuned general RoBERTa, indicating that the CLUE framework effectively leverages the semantic representations of general pretrained RoBERTa without requiring additional domain-specific pretraining. Efficiency analysis further highlights the advantage of direct fine-tuning, which achieved approximately 17.9× faster training speed compared to the domain-specific pretraining approach.

Table 4. Table captions should be placed above the tables.

Model	Training Time(s)	Accuracy(%)	Recall(%)
Fine-tuning General Large Model	**144.02**	97.10	95.17
Domain Pretraining + Fine-tuning	2572.51	**97.88**	95.71

These results demonstrate that directly fine-tuning the general pretrained RoBERTa model is an efficient and feasible solution for APT detection. By eliminating the domain-specific pretraining step, CLUE not only significantly reduces training time but also mitigates the risk of overfitting to domain-specific data. In dynamic environments where frequent model updates are required to address emerging APT attacks, the proposed CLUE method demonstrates exceptional adaptability and practical value.

Impact of Contrastive Loss. To evaluate the impact of the proposed contrastive learning loss, we compared the model's performance using only the cross-entropy loss with that when combining cross-entropy loss and contrastive learning loss. We selected the S4 dataset, which exhibits relatively low baseline performance, to provide a more intuitive analysis of the performance improvements introduced by the contrastive learning loss.

The experimental results demonstrate that incorporating the contrastive learning loss significantly improves the model's performance across all metrics (see Table 5). In the sequence-level evaluation, TPR improved by 3.12% and FPR decreased by 7.29%, indicating that the model's ability to distinguish between attack and non-attack sequences was notably strengthened, while false positives were significantly reduced. This improvement is particularly crucial for practical attack detection applications, as a lower false positive rate translates to higher detection reliability and reduced resource wastage. Furthermore, the experiments showed that incorporating the contrastive learning loss reduced training time by 11.31%, indicating that our approach not only improves model performance but also significantly enhances training efficiency.

Table 5. Table captions should be placed above the tables.

Metric	**CE Loss**	**CE+SCL Loss**	**Improvement**
Accuracy (%)	86.95	**94.23**	**7.28**
TPR (%)	90.63	**93.75**	**3.12**
FPR (%)	13.06	**5.77**	**7.29**
Training Time (s)	521.72	**475.59**	**46.13**

Comparison of Contrastive Learning Methods. We further compared the impact of different contrastive learning methods on model performance by evaluating two commonly used contrastive loss functions: Margin Loss[25, 26] and our SCL Loss, as shown in Table 6. The results demonstrate that the proposed SCL Loss outperforms Margin Loss in terms of detection performance, further validating the effectiveness and applicability of the contrastive learning approach used in this study.

Table 6. Comparison of contrastive loss functions.

ID	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**S1**	**S2**	**S3**	**S4**
Margin Loss(%)	99.62	99.58	**99.24**	97.57	98.31	96.54	99.58	99.34	**97.80**	86.95
SCL Loss(%)	**99.92**	**99.58**	98.48	**98.11**	**98.68**	**97.10**	**99.68**	**99.72**	96.70	**94.23**

7 Conclusion

We propose an APT detection framework, CLUE, which enables multi-granularity detection at the entity, event, and sequence levels. CLUE leverages lemmatization techniques to normalize sequence data extracted from provenance graphs into forms consistent with the vocabulary of general pretrained models, thereby utilizing the embedding representations and pretrained parameters of the general RoBERTa model to directly fine-tune it. Additionally, we introduce a contrastive learning loss function to further enhance feature separability.

We evaluated CLUE on 10 real-world APT attack scenarios. Experimental results demonstrate that, compared to state-of-the-art methods, CLUE achieves superior detection performance while significantly reducing training time and data requirements. These results highlight CLUE's excellent balance between efficiency and performance, showcasing its strong practical value in real-world applications.

Acknowledgments. I would like to express my gratitude to my fellow students and my teacher. They have provided great assistance for this work.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Alshamrani, A., Myneni, S., Chowdhary, A., Huang, D.: A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. IEEE Communications Surveys & Tutorials 21(2), 1851-1877 (2019)
2. Milajerdi, S.M., Gjomeno, R., Eshete, B., Sekar, R., Venatakrishnan, V.: Holmes: real-time apt detection through correlation of suspicious information flows. In: 2019 IEEE symposium on security and privacy (SP). pp. 1137-1152. IEEE (2019)
3. Hossain, M.N., Milajerdi, S.M., Wang, J., Eshete, B., Gjomoemo, R., Sekar, R., Stoller, S., Venkatakrishnan, V.: Sleuth: Real-time attack scenario reconstruction from {COTS} audit data. In: 26th USENIX Security Symposium (USENIX Security 17). pp. 487-504 (2017)
4. Hassan, W.U., Bates, A., Marino, D.: Tactical provenance analysis for endpoint detection and response systems. In: 2020 IEEE symposium on security and privacy (SP). pp. 1172-1189. IEEE (2020)
5. Milajerdi, S.M., Eshete, B., Gjomoemo, R., Venkatakshnan, V.: Poirot: Aligning attack behavior with kernel audit records for cyber threat hunting. In: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security. pp. 1795-1812 (2019)
6. Du, M., Li, F., Zheng, G., Srikumar, V.: Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In: Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. pp. 1285-1298 (2017)
7. Han, X., Pasquier, T., Bates, A., Mickens, J., Seltzer, M.: Unicorn: Runtime provenance-based detector for advanced persistent threats. arXiv preprint arXiv:2001.01525 (2020)
8. Alsaheel, A., Nan, Y., Ma, S., Yu, L., Walkup, G., Celik, Z.B., Zhang, X., Xu, D.: Atlas: A sequence-based learning approach for attack investigation. In: 30th USENIX security symposium (USENIX security 21). pp. 3005-3022 (2021)
9. Zergy, J., Wang, X., Liu, J., Chen, Y., Liang, Z.L.: Shadewatcher: Recommendation-guided cyber threat analysis using system audit records. In: 2022 IEEE symposium on security and privacy (SP). pp. 489-506. IEEE (2022)
10. Wang, S., Wang, Z., Zhou, T., Sun, H., Yin, X., Han, D., Zhang, H., Shi, X., Yang, J.: Threatrace: Detecting and tracing host - based threats in node level through provenance graph learning. IEEE Transactions on Information Forensics and Se - curity 17, 3972 - 3987 (2022)
11. Liu, F., Wen, Y., Zhang, D., Jiang, X., Xing, X., Meng, D.: Log2vec: A hetero - geneous graph embedding based approach for detecting cyber threats within en - terprise. In: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security. pp. 1777 - 1794 (2019)
12. Li, Z., Cheng, X., Sun, L., Zhang, J., Chen, B.: A hierarchical approach for advanced persistent threat detection with attention - based graph neural networks. Security and Communication Networks 2021(1), 9961342 (2021)
13. Kapoor, M., Melton, J., Ridenhour, M., Krishnan, S., Moyer, T.: Prov - gem: Automated provenance analysis framework using graph embeddings. In: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 1720 - 1727. IEEE (2021)
14. Ding, H., Zhai, J., Nan, Y., Ma, S.: Airtag: Towards automated attack investigation by unsupervised learning with log texts. In: 32nd USENIX Security Symposium (USENIX Security 23). pp. 373 - 390 (2023)
15. Xu, Z., Fang, P., Liu, C., Xiao, X., Wen, Y., Meng, D.: Depcomm: Graph summarization on system audit logs for attack investigation. In: 2022 IEEE symposium on security and privacy (SP). pp. 540 - 557. IEEE (2022)

16. Rehman, M.U., Ahmadi, H., Hassan, W.U.: Flash: A comprehensive approach to intrusion detection via provenance graph representation learning. In: 2024 IEEE Symposium on Security and Privacy (SP). pp. 3552 - 3570. IEEE (2024)
17. Sharif, M., Datta, P., Riddle, A., Westfall, K., Bates, A., Ganti, V., Lentzk, M., Ott, D.: Drsec: Flexible distributed representations for efficient endpoint security. In: 2024 IEEE Symposium on Security and Privacy (SP). pp. 3609 - 3624. IEEE (2024)
18. Cao, K., Wei, C., Gaidon, A., Arechiga, N., Ma, T.: Learning imbalanced datasets with label - distribution - aware margin loss. Advances in neural information processing systems 32 (2019)
19. Liu, W., Wen, Y., Yu, Z., Yang, M.: Large - margin softmax loss for convolutional neural networks. arXiv preprint arXiv:1612.02295 (2016)
20. Zhang, Z., Sabuncu, M.: Generalized cross entropy loss for training deep neural networks with noisy labels. Advances in neural information processing systems 31 (2018)
21. Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., Fergus, R.: Training convolutional networks with noisy labels. arXiv preprint arXiv:1406.2080 (2014)
22. Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., Fergus, R.: Training convolutional networks with noisy labels. arXiv preprint arXiv:1406.2080 (2014)
23. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
24. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
25. Zhou, W., Liu, F., Chen, M.: Contrastive out-of-distribution detection for pre-trained transformers. arXiv preprint arXiv:2104.08812 (2021)
26. Zou, J., Guo, M., Tian, Y., Lin, Y., Cao, H., Liu, L., Abbasnejad, E., Shi, J.Q.: Semantic role labeling guided out - of - distribution detection. arXiv preprint arXiv:2305.18026 (2023)