The ISC Int'l Journal of Information Security

ISeCure

October 2025, Volume 17, Number 3 (pp. 1–10)

http://www.isecure-journal.org

Presented at the ISCISC'2025 in Tehran, Iran.

An LSTM-DBSCAN Approach for Interpretable Insider Threat Detection via Behavioural Anomaly Analysis **

Mohammad Mohammadi [1], Moein Bannaye Zahmati [1], and Morteza Noferesti [1,*]

[1] *Department of Computer Engineering, Bozorgmehr University of Qaenat, Qaen, South Khorasan, Iran.*

ARTICLE INFO.

Keywords:
Insider threat detection, LSTM, DBSCAN clustering, MITRE ATT&CK, Behavioural anomaly analysis

Type:

doi:

Abstract

Insider threats pose a significant cybersecurity risk, as authorised users can exploit legitimate access to compromise sensitive systems and data. This paper proposes an integrated behavioural anomaly detection approach to address three critical challenges in AI-driven insider threat detection: lack of interpretability, misleading evaluation metrics, and misalignment with operational taxonomies. Our approach employs a three-stage pipeline: (1) an LSTM autoencoder to detect temporal anomalies in login patterns, (2) DBSCAN clustering to identify suspicious file access and device usage during anomalous sessions, and (3) DBSCAN-based URL analysis to uncover exfiltration patterns. By analysing behaviour across time, location, and web activity, this framework builds actionable threat chains mapped to MITRE ATT&CK techniques including `T1078`, `T1005`, `T1204.002`, `T1567.002`. It bridges the gap between theoretical models and the daily work of a Security Operations Center (SOC). In the data exfiltration scenario on the CERT R6.2 insider threat dataset, the proposed approach achieved a recall of 83.3% and an accuracy of 91.7% in classifying malicious days. The framework also provides interpretable alerts and maintains operational efficiency.

1 Introduction

Insider threats are a critical cybersecurity challenge because they originate from trusted individuals such as employees, contractors, or partners who can intentionally or accidentally misuse their legitimate access to cause harm. Unlike external attacks that exploit software vulnerabilities, insider threats originate from trusted users who already have legitimate access to sensitive systems and data [1]. Because these threats operate within normal access boundaries, traditional security measures like firewalls and intrusion detection systems often fail to detect them [2].

Insider Threat Detection (ITD) systems continuously analyse user behaviour to identify potential malicious or negligent activity by authorised personnel [3]. These systems operate primarily by detecting anomalies [4]. Using machine learning and statistical models, they first establish a baseline of nor-

* Corresponding author.

**The ISCISC'2025 program committee effort is highly acknowledged for reviewing this paper.

Email addresses: `mohammadi.ai.eng@gmail.com`, `moein.bannaye.zahmati@gmail.com`, `mnoferesti@buqaen.ac.ir`

ISSN: 2008-2045

mal user behaviour and then flag significant deviations [4]. Deviations from these baselines, such as unusual after-hours logins, unauthorised USB usage, or abnormal data transfers, trigger alerts for further investigation. Current ITD frameworks integrate User and Entity Behaviour Analytics (UEBA), endpoint detection and response (EDR), and data loss prevention (DLP) tools to provide a layered defense against insider threats [2].

ITD approaches primarily fall into three categories: feature engineering-based, sequence-based, and graph-based methods. Feature engineering-based methods rely on manually defined behavioural indicators (such as after-hours file access) combined with machine learning classifiers like Support Vector Machines (SVM) or Random Forests [2]. Sequence-based approaches model user activity sequences (sudh as daily logs of login times and file accesses) to detect temporal anomalies, often using Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks. Graph-based methods leverage relationships between users, devices, or activities, utilising Graph Neural Networks (GNNs) to identify anomalies in network structures [5].

Despite significant research efforts to apply AI algorithms, from classical machine learning [4, 6] to advanced deep learning [3, 5] in insider threat detection systems, three critical challenges persist, which we name *lack of interpretability*, *misleading evaluation*, and *taxonomy divide*, making them difficult to use in the real-world.

The first issue in applying AI to insider threat detection is the ***lack of interpretability*** in model outputs. Many AI systems, particularly complex deep learning models [3], generate results such as risk scores or anomaly flags without clear explanations [4]. The lack of actionable context (such as the reason for a flag or the specific triggering behaviour) makes these outputs difficult for Security Operations Center (SOC) analysts to investigate and respond. Without interpretability, security analysts cannot prioritise incidents, investigate root causes, or justify responses such as account suspension. Consequently, even highly accurate models become operationally ineffective, as their outputs remain abstract and cannot be translated into concrete security actions.

Another critical challenge is the ***misleading evaluation*** of AI models due to imbalanced datasets. Insider threat incidents are rare compared to normal user activity, leading to severe class imbalance [4, 7]. Many studies report high accuracy or precision, but these metrics are deceptive because a model can achieve them simply by always predicting "normal." Such evaluations mask poor performance in detecting actual threats, creating a false sense of reliability. Moreover, some methods label themselves as "anomaly detection" while relying on supervised learning with biased test sets, further obscuring their real-world applicability [1].

A critical gap persists between AI-driven insider threat detection research and operational security frameworks, which we call the ***taxonomy divide***. While organisations rely on frameworks like MITRE ATT&CK to classify and respond to threats [8, 9], most academic studies either ignore this standard or fail to align their detection methods with its taxonomy. Researchers frequently develop models that detect "anomalies" in isolation, without mapping findings to known adversarial techniques [2]. This omission makes it difficult for security teams to contextualise AI-generated alerts within their existing threat intelligence infrastructure.

This study focuses on a high-risk insider threat scenario: *an authorised user who logs in during non-working hours, introduces a USB device, and connects to an external URL to upload data.*

Figure 1 provides a detailed mapping of the observed insider threat actions to specific MITRE ATT&CK techniques. The insider leverages valid credentials (`T1078`) to log in during abnormal hours, exploiting authorised access. The attack uses a malicious USB device (`T1204.002`) to execute a harmful script. The insider then collects sensitive local data (`T1005`) from the system, a common step before exfiltration. Finally, the data is exfiltrated via a web service (`T1567.002`) to evade detection by blending with normal traffic.

In this paper, we present an integrated behavioural anomaly detection framework designed to overcome three critical barriers preventing AI adoption in operational security environments: *uninterpretable results*, *misleading performance metrics*, and *taxonomy divide*. Our approach correlates sequential indicators of malicious intent through three stages, each addressing these challenges directly:

First stage: An LSTM autoencoder builds personal behavioural baselines from user login sequences, detecting anomalies in access times or session patterns that indicate potential insider threats associated with MITRE ATT&CK `T1078` (Valid Accounts). Each detected anomaly includes contextual details (deviation type, baseline comparison, confidence score) and is treated as a threat candidate requiring verification through exfiltration analysis stages. This approach balances detection sensitivity with operational practicality by transforming raw anomalies into actionable security hypotheses rather than definitive threats.

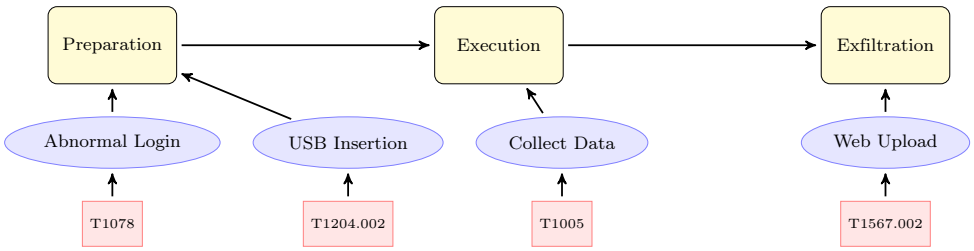

Figure 1. Insider Threat Attack Flow with MITRE ATT&CK Mapping

Second stage: This stage employs DBSCAN clustering to examine file access patterns during temporally suspicious sessions, specifically targeting data staging behaviours associated with MITRE ATT&CK techniques `T1005` (Data from Local System) and `T1204.002` (User Execution: Malicious File). This density-based method identifies clusters of high-risk behaviour. Specifically, it flags users who, following an anomalous login detected by the LSTM autoencoder, proceeded to access sensitive directories or removable storage devices. The clustering process serves as a critical filtering mechanism, reducing the initial set of temporal anomalies to a manageable subset of high-probability threats by analysing three key spatial features: (1) path sequences to classified data repositories, (2) frequency of removable device interactions, and (3) file type access patterns. For suspicious activity the cluster density increases and the system generates prioritised alerts containing both the original temporal anomaly context and newly identified event correlations.

Third stage: Our final stage applies DBSCAN to URL access logs during device-usage periods flagged as suspicious by the spatial clustering stage (`T1005/T1204.002`), targeting exfiltration patterns (`T1567.002`) without relying on external threat feeds [10].

The clustering only looks at suspicious web activity from unusual sessions (LSTM-flagged). We analyse two distinct feature sets to characterise this activity:

(1) Semantic URL Patterns: We apply TF-IDF to the domain and path segments of URLs to identify (a) connection frequency bursts to uncommon domains and (b) repeated access to known file-sharing services.
(2) Data Transfer Volume: We separately calculate the total bytes transferred per web session to flag sessions with abnormal data transfer volumes indicative of exfiltration. Our method detects data exfiltration by monitoring the destination and the volume of data being sent.

This work makes three key contributions to insider threat detection:

(1) Proposes the first insider threat detection system that sequentially correlates temporal (LSTM), spatial (DBSCAN), and exfiltration (URL clustering) anomalies into actionable attack chains, bridging the gap between academic models and SOC workflows.
(2) Maps all detected behaviours to specific techniques (`T1078, T1005, T1204.002, T1567.002`), enabling direct integration with enterprise threat intelligence platforms, which is a critical advance over "black box" AI approaches.
(3) Comprehensive evaluation of CERT Dataset v6.2 [7] showing the performance of the proposed approach in the real-world insider threat scenario.

The remainder of this paper is structured as follows: Section 2 analyses existing approaches to insider threat detection. Section 3 details our integrated methodology, including LSTM-based temporal anomaly detection and DBSCAN clustering for device and URL patterns. Section 4 presents comprehensive experiments on the CERT Dataset v6.2, comparing the detection performance with the baseline methods and validating the coverage of the MITRE technique. Finally, Section 5 summarizes our key findings and outlines directions for future work.

2 Related Work

Modern insider threat detection (ITD) approaches predominantly employ machine learning. However, they face three persistent challenges: (1) *lack of interpretable outputs*, (2) *misleading evaluation metrics*, and (3) *failure to align with operational taxonomies* like MITRE ATT&CK.We evaluate existing methods against these specific challenges.

2.1 Interpretability Limitations

Deep learning has improved ITD by automating feature extraction from behavioural sequences, reducing the dependency on manual engineering [1, 3, 11]. However, this comes at the cost of interpretability:

Graph Neural Networks (LAN [1]) construct temporal graphs where nodes represent user activities

(logins, file accesses) and edges encode sequential relationships. The model uses multi-head attention to weight connections between distant events (a login at time t and file deletion at $t+k$). Although they effectively capture long-range dependencies, these graph embeddings (typically 128- to 256-dimensional vectors) lack interpretability, as they cannot be traced back to discrete malicious events like "CAD file copied to USB at 02:17."

Hybrid CNN-Transformers (CATE [11]) process log sequences through parallel streams: a 1D CNN extracts local patterns (bursts of file downloads) while a transformer layer models global dependencies using self-attention. Though achieving 96.18% F1-score on CERT r4.2, their classification heads output only binary labels (malicious/benign) and do not identify which specific technique (T1078, T1005, etc.) was detected.

Dehghan *et al.* in [12] present ProAPT, a novel framework that employs Deep Reinforcement Learning (DRL) to project and simulate the potential progression of Advanced Persistent Threats (APTs). The core innovation lies in modelling the attacker as a DRL agent that learns optimal, multi-stage attack paths within a network environment to achieve a specific goal. Moving beyond detection, this method focuses on forecasting threats, allowing teams to anticipate an attacker's next most likely moves from the existing network state. By effectively generating realistic attack narratives, ProAPT serves as a valuable tool for strategic security planning, penetration testing, and strengthening cyber defences by highlighting critical network vulnerabilities.

2.2 Misleading Evaluation Metrics

Current approaches often report inflated performance due to improper handling of class imbalance, particularly problematic in datasets like CERT r6.2, where malicious instances constitute only 0.003% of samples. EBIGAN [13] attempts to address this through a bidirectional GAN architecture that generates synthetic minority samples using both forward ($z \to x$) and backward ($x \to z$) latent space mappings. While achieving 98% precision on resampled CERT r6.1 data, this approach suffers from generating unrealistic behavioural sequences - such as producing 47 consecutive CAD file downloads at 1-second intervals - that distort the true characteristics of insider threats.

LAN [1] proposes a hybrid loss function combining focal loss ($\alpha = 0.25$, $\gamma = 2$) to down-weight well-classified majority samples with contrastive loss ($margin = 1.0$) to separate attack/normal embeddings in latent space. While theoretically sound, the approach's reported 9.92% AUC improvement is misleading as it was evaluated on artificially balanced test sets that do not reflect the extreme imbalance of real-world operational environments. This evaluation methodology masks the model's likely poor performance on genuine sparse threats, creating a false sense of efficacy. Taxonomy Misalignment

2.3 Taxonomy Misalignment

The disconnect between academic research and operational security frameworks remains a significant barrier to practical deployment. Stacked LSTMs [14] exemplify this gap through their sophisticated 3-layer architecture: a bottom LSTM processes raw log sequences, a middle GRU layer captures weekly patterns, and an attention head weights critical time steps. Despite its technical complexity, the model only outputs simple anomaly scores (on a 0–1 scale). It does not map these detections to specific ATT&CK techniques, leaving security analysts without actionable threat intelligence.

Methods based on the tri-training paradigm, such as the semi-supervised approach in [15], use ensemble classifiers (SVM, RF, MLP) to iteratively label undata, thereby reducing dependence on manual labels. Despite high detection rates (90%) with minimal labels (1%), such methods cannot differentiate between key techniques like `T1078` and `T1204`, making their outputs unsuitable for operational threat response workflows.

Our analysis reveals fundamental gaps in current insider threat detection research. These approaches share three critical limitations: (1) reliance on artificially balanced datasets that inflate performance metrics, (2) inability to map findings to MITRE ATT&CK techniques, and (3) computational complexity that hinders real-world deployment. Our LSTM-DBSCAN pipeline addresses these shortcomings by combining the temporal sensitivity of sequence modelling with the interpretability of density-based clustering, operating directly on raw imbalanced data while generating fully mappable threat chains. This approach bridges the divide between academic detection models and operational security frameworks, providing SOC teams with both technical detection capability and the contextual intelligence needed for effective response.

3 Proposed approach

To detect insider threats characterised by shifts in login patterns, device usage, and web activity, we developed a multi-stage methodology that integrates temporal anomaly detection with contextual analysis. The methodology leverages a pipeline of LSTM-based anomaly detection on pre-processed login events, fol-

lowed by DBSCAN clustering of device and web activities. It consists of three primary components: (1) pre-processing login events to extract temporal features, (2) identifying anomalous login sessions using an LSTM autoencoder, and (3) analysing device and web activities during anomalous sessions to uncover patterns indicative of insider threats, such as unauthorised device usage or suspicious web access.

3.1 Login Event Pre-processing

The initial stage pre-processes login events to create a structured dataset of user login sessions, capturing temporal features that reflect deviations from standard working hours. This process transforms raw login and logoff events into session-level records suitable for anomaly detection.

3.1.1 Temporal Feature Engineering

For each login session, three temporal features are computed to quantify deviations from standard work hours:

- **Duration Minus 8 Hours**: The session duration (logoff time minus login time) is adjusted by subtracting 8 hours (28,800 seconds), representing a standard workday. The result is formatted as a signed timedelta (e.g., +02:30:00 or -01:15:00) to indicate whether the session exceeds or falls short of a typical workday.
- **Time to Start Work**: The time difference between the login timestamp and the next 7 AM (start of the workday). If the login occurs after 5 PM, the next 7 AM is on the following day; otherwise, it is on the same day. This feature flags logins that occur outside of a user's typical working hours.
- **Time to Finish Work**: The time difference between the logoff timestamp and the previous 7 PM (end of the workday). If the logoff occurs after 7 AM, the previous 7 PM is on the same day; otherwise, it is on the previous day. The feature measures how late a session extends beyond typical hours.

These features are converted to seconds for numerical processing and stored in an output file with columns: event ID, login timestamp, logoff timestamp, PC identifier, duration, time to start work, and time to finish work. This dataset enables the identification of sessions with unusual temporal patterns, such as after-hours logins, which are potential indicators of insider threats.

3.2 LSTM-Based Anomaly Detection

The second stage employs an LSTM autoencoder to detect anomalous login sessions. The LSTM model operates by first establishing a baseline of typical login behaviour. It then flags anomalous sessions, such as those occurring outside standard hours.

Assuming the sequence of events $SE_{t-1} =< E_1, ..., E_{t-1} >$ at timestamp $t-1$. the proposed approach predicts the next event E_t with (Equation 1).

$$p(E_t|SE_{t-1}) = \prod_{i=1}^{t} p(E_i|SE_{t-1}) \tag{1}$$

To calculate the probability of event E_t regarding (Equation 1), the proposed approach must store the sequence up to the timestamp $t-1$. The structure of an LSTM network is depicted in Figure 2, where E_i and Y_i are the input event and the model output, respectively, at timestamp i. F is the LSTM gate function. h_t and C_t are hidden layer vectors, and W^{HH}, W^{HX}, and W^{HY} are parametric matrices [16].

The encoder transform the received sequence of events into a single condensed context vector [17]. As this sequence is fed into the encoder, the hidden state of the LSTM changes following (Equation 1), where E_t represents each event and t indicates its respective position in the sequence. Once all network units have propagated their information, the encoder then outputs a context vector C representative of the entire input sequence [16]. The calculation of the hidden states, represented as h_i, is conducted using (Equation 2).

$$h_t = f_{enc}(W^{(HH)}h_{t-1} + W^{(HX)}sc_t) \tag{2}$$

The resulting context vector aggregates information from each event and delivers it to the decoder, which uses this integrated representation to generate its final prediction.

The decoder accepts the context vector C from the encoder and generates an output at every time step [17]. Unlike the encoder, which delivers an output only at the final step, the decoder creates output at each time step. Once prepared by the encoder, the context vector is delivered to the decoder as input [16]. Lastly, every hidden state h_i is computed using (Equation 3) with the weight W [17].

$$h_t = f_{dec}(W^{(HH)}h_{t-1}) \tag{3}$$

The proposed approach builds an LSTM autoencoder with two LSTM layers (64 and 32 units) for encoding and decoding, using mean squared error (MSE) loss and early stopping to prevent overfitting. The trained model predicts reconstructions for all sequences, and MSE scores are calculated. Sessions with MSE exceeding the threshold are marked as anomalous. Anomalous sessions are linked to their original indices, enabling traceability to specific login events.

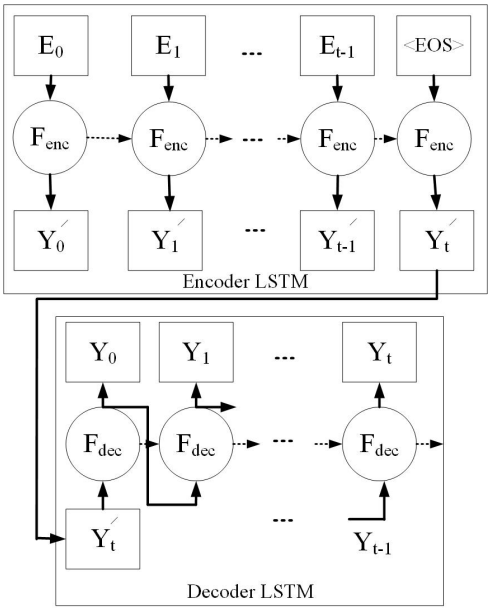

Figure 2. Overview of the LSTM with encoder and decoder components

3.3 Device and URL Activity Analysis

For each anomalous login, we analyse associated device and web activity to characterise potential insider threat behaviours, such as unauthorised device usage or suspicious web access. DBSCAN clustering is used to identify patterns in these activities.

Device activities are extracted from a dataset containing columns such as event ID, timestamp, user ID, PC identifier, activity type, and file paths accessed. Device events are filtered to match the anomalous login's PC and its active time window (login to logoff).

Filtered events are stored in a dataframe with an anomaly ID column linking them to the corresponding anomalous session. File paths are pre-processed by replacing delimiters with spaces to create space-separated tokens. A TfidfVectorizer with a custom token pattern (to preserve path components) vectorises the file paths. These vectors are then clustered using DBSCAN, configured with $\epsilon = 0.1$, minimum samples=2, and cosine similarity. Noise points (cluster -1) represent outlier paths, while clustered paths indicate common access behaviours during anomalous sessions. The results are visualised in a scatter plot. Event times are plotted on the x-axis and cluster IDs on the y-axis, with each cluster annotated by its most frequently accessed paths.

Web activities are processed from a dataset containing columns such as event ID, timestamp, user ID, PC identifier, URL, activity type, and content. To handle potentially large datasets, data is processed in chunks. For a given user during anomalous login sessions, web events are filtered to include those within the session's time window.

URLs are pre-processed to extract key components (domain and first two path segments) using regular expressions, removing protocols and query parameters (for example `example.com/path1 / path2?query` becomes `example.com path1 path2`). The cleaned URLs are vectorised using a TfidfVectorizer [18] with a maximum of 500 features and stop words (such as `com`, `net`, `org`) removed. DBSCAN clustering (with $\epsilon = 0.3$, minimum samples=3, and cosine similarity) groups similar URLs, identifying patterns such as frequent visits to external domains.

4 Evaluation

The insider threat detection approach was built and tested on an Ubuntu 20.04 LTS system, powered by an Intel Core i5-10400F CPU (6 cores, 2.9GHz base clock) and 16GB DDR4 RAM (2666MHz). All experiments utilised Python 3.11, incorporating libraries like pandas (v1.3.5) for data handling, scikit-learn (v1.0.2) for machine learning processes, TensorFlow (v2.8) for neural network models, and supporting tools such as numpy, matplotlib, and sklearn modules for data pre-processing, clustering (KMeans, DBSCAN), feature engineering (TfidfVectorizer, PCA), and result visualisation.

4.1 Dataset

The CERT r6.2 Insider Threat Test Dataset, developed by Carnegie Mellon University's CERT Division in collaboration with ExactData, LLC [7], serves as the principal benchmark for evaluating insider threat detection methodologies. This comprehensive synthetic dataset spans 18 months of simulated enterprise activity, encompassing approximately 137 million timestamped events across 4,000 user profiles. While the dataset models five distinct threat scenarios, our analysis emphasizes the first scenario, characterised by a data exfiltration pattern (valid account abuse, malicious file execution, and web exfiltration).

The dataset's forensic value derives from its multimodal composition: authentication patterns (3.5M logon/logoff events), removable media operations (1.5M device connections), web transactions (117M HTTP requests with full URL parameters), and file system interactions (2M operations). This coverage enables the detection of behavioural shifts, such as a user moving from normal work-hour activity to malicious after-hours actions that involve removable device usage and subsequent data exfiltration to external sites.

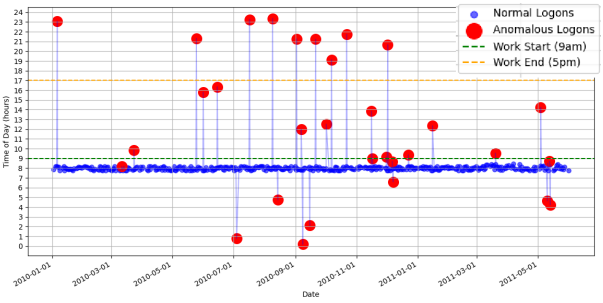

Figure 3. Results from an LSTM autoencoder applied to user `SHB3512` logon timestamps.

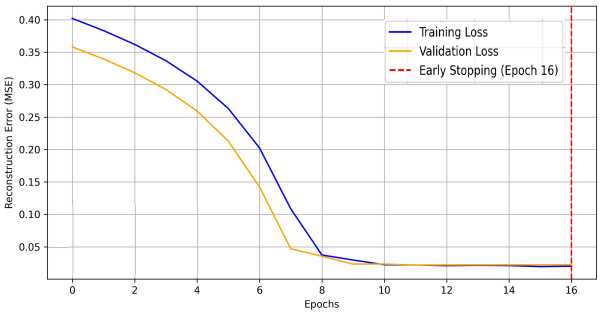

Figure 4. MSE value for the LSTM autoencoder over epochs for user `SHB3512`.

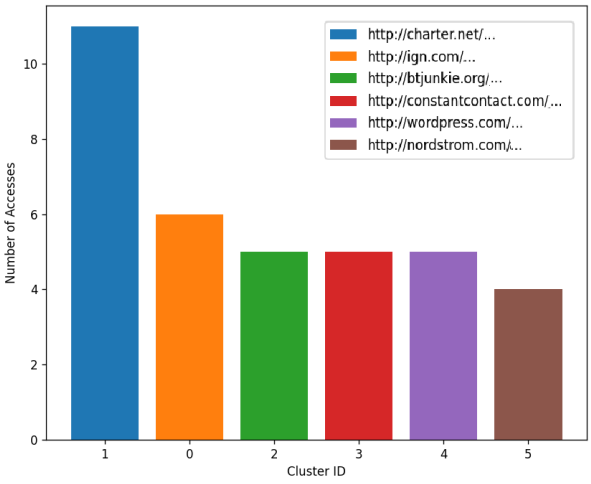

Figure 5. Anomalous URL clusters for user `ROR3483`.

4.2 Evaluation Metrics

The detection performance of the anomaly detection model was quantified by ensuring strict temporal alignment between model outputs and the ground truth provided in the CERT r6.2-1.csv file, where logged activities are considered malicious (attacks). The model aims to identify days with such activities as anomalies. For each day in the validation set, the following metrics were computed:

- *True Positives (TP)*: Counted when the model flags a day as anomalous, and that day is labelled as malicious in the r6.2-1.csv file.
- *False Positives (FP)*: Computed as days flagged as anomalous by the model without corresponding ground truth support.
- *False Negatives (FN)*: Recorded when a day with malicious activities is not flagged as anomalous by the model.
- *True Negatives (TN)*: Derived from benign days that the model correctly does not flag as anomalous.

4.3 Experiment results

Using an LSTM-based model, we detect anomalies by calculating the reconstruction error of session features. A total of 73,491 sessions are flagged as anomalous, corresponding to errors above the 95th percentile threshold. Figure 3 shows the system identifying deviations from standard work hours (9 AM–5 PM). It detects users like `SHB3512` by marking their anomalous login times as temporal outliers.

Figure 4 illustrates the training of our LSTM autoencoder for the user `SHB3512`. It demonstrates effective convergence through the evolution of training and validation loss (MSE) across epochs. Both curves show a characteristic exponential decay: rapid improvement over the first five epochs, followed by progressive stabilization. Optimal performance was reached at epoch 16, triggering the early stopping mechanism.

The temporal anomalies identified by our LSTM autoencoder are subsequently correlated with device activities from `device.csv` and web activities from `http.csv` occurring within the same session windows. Figure 5 presents the forensic analysis of anomalous web activities for user `ROR3483`, revealing six distinct threat clusters through DBSCAN-based URL pattern analysis.

Figure 6 presents the performance analysis of the proposed approach. The top visualisation compares the execution times of four key operations: training of the LSTM model per user (mean 0.0718s), LSTM inference per sample (mean 0.0001s), and DBSCAN clustering per user for both URLs (mean 0.0026s) and devices (mean 0.0034s). Clustering operations demonstrate consistent low-latency performance across different data modalities. The lower section of the figure displays critical alert statistics, showing that the system processed 73,491 total anomalies. These were organized into 26,558 URL-based clusters and 12 device-based clusters. This combination of performance metrics and operational results effectively illustrates the approach's efficiency in handling large-scale anomaly detection tasks while maintaining responsive processing times.

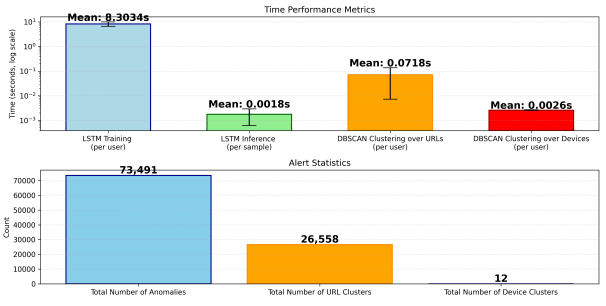

Figure 6. Proposed system performance and alert overview showing computational timings(top) and the number of detected anomaly and clusters (bottom).

4.4 Detected attack scenario

Our threat detection pipeline began by analysing login/logoff patterns using an LSTM autoencoder, which identified 73,491 anomalous sessions based on significant deviations from normal working hours and typical session durations. For these sessions, DBSCAN clustering identified two significant device-level anomaly clusters, which helped prioritise the investigation and narrow our focus from the overwhelming number of URL access patterns. These device anomalies were traced to user `ACM2278`(Figure 7). Cluster 1 consists of `UNKNOWN` paths exclusively associated with USB removal events. The user `ACM2278` exhibited suspicious behaviour, including repeated logins outside normal working hours—specifically, late-night and early-morning sessions between August 21–24, 2010 (see Figure 8). The timing of these anomalous logons precisely matched peaks in device activity, suggesting deliberate, unauthorised access. Further examination of associated URL clusters revealed high-risk web accesses, most notably visits to Wikileaks pages (Figure 9).

The combination of temporal anomalies, clustered device activity, and risky URL patterns points to a serious security breach. The after-hours access patterns suggest credential compromise, while the Wikileaks visits indicate potential data exfiltration attempts. The concentrated activity during August 21-24, 2010 represents the critical attack window, with the device clusters providing clear markers of malicious activity that might have been overlooked in URL analysis alone. This case demonstrates how multi-layered analysis - combining temporal, behavioural, and content indicators - can effectively uncover sophisticated attacks that might otherwise remain hidden in large datasets. The findings underscore the importance of monitoring both device activity and web access patterns for comprehensive threat detection.

The resulting confusion matrix, presented in Figure 10, quantifies the model's ability to classify days as malicious or benign for user `ACM2278` based on in-

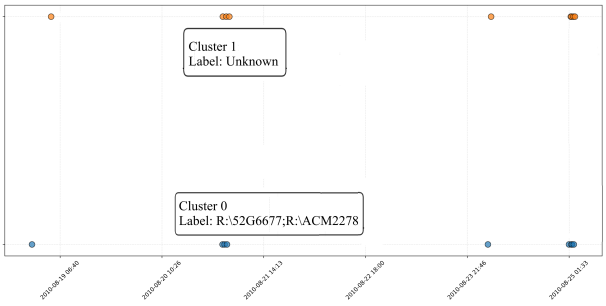

Figure 7. DBSCAN clusters of device activity timestamps, highlighting the anomaly pattern for user ACM2278.

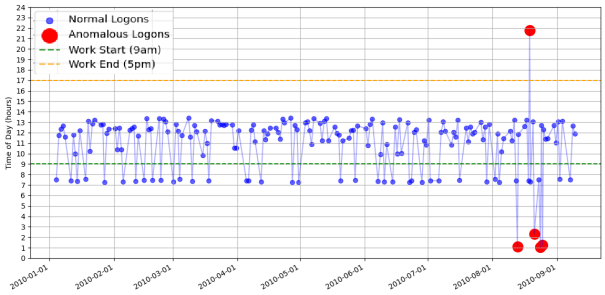

Figure 8. Anomalous logon events (red) revealing after-hours access patterns (ACM2278).

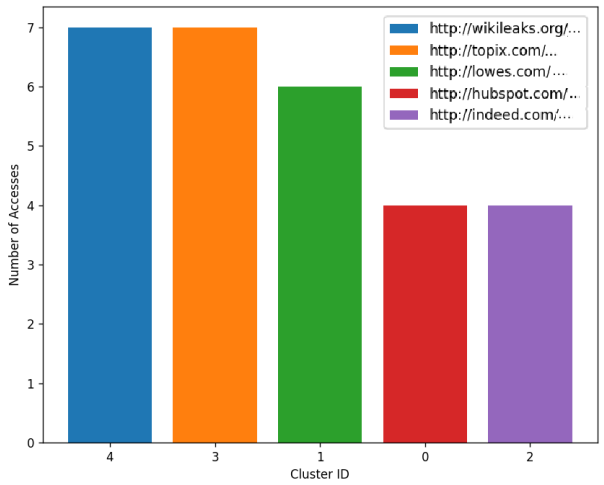

Figure 9. Suspicious URL clusters accessed during anomaly periods, showing Wikileaks visits (ACM2278).

tegrated web and device logs. A day is labelled as malicious if the CERT r6.2-1 ground truth file indicates one or more malicious activities occurred on that day. For user ACM2278, activity was observed over 157 days. The model correctly identified 139 benign days (True Negatives), demonstrating robust performance in recognizing normal behaviour. With only 1 False Negative, the model exhibits a recall of 83.3%, effectively capturing nearly all actual malicious days. The 5 True Positives confirm the model's success in detecting numerous attack instances. Overall, this performance yields a accuracy of 91.7%, highlighting the

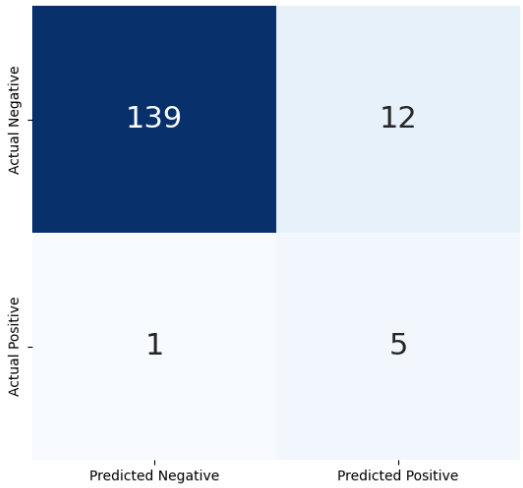

Figure 10. Confusion matrix for web and device log integration (user CMPACM2278).

model's reliability in minimizing missed detections while maintaining strong predictive power, which is critical for practical insider threat detection.

5 Conclusion and Future Work

In this paper, we presented a multi-stage insider threat detection approach that combines temporal anomaly detection with behavioural clustering to identify malicious activity in enterprise logs. Our approach first flags suspicious login sessions using an LSTM autoencoder, then correlates these anomalies with device and URL access patterns using DBSCAN clustering. This layered methodology successfully uncovered a real-world attack scenario involving after-hours logins, USB-based data staging, and exfiltration to high-risk web destinations. By integrating temporal, device, and web analysis, our system provides security teams with actionable, context-rich alerts while reducing false positives. The experimental results on the CERT r6.2 dataset demonstrate the effectiveness of our approach in detecting sophisticated insider threats.

The next steps involve validating the framework on real-world, anonymised enterprise data to assess its performance in noisy environments and against a broader range of threats, including sabotage and credential theft. A key future direction is engineering a transition from batch processing to a low-latency, real-time streaming architecture using technologies like Apache Flink or Kafka. This will enable incremental model updates and near-real-time detection. Finally, we will develop an advanced alert triage system with root-cause explainability to mitigate alert fatigue and provide analysts with actionable intelligence.

References

[1] Xiangrui Cai, Yang Wang, Sihan Xu, Hao Li, Ying Zhang, Zheli Liu, and Xiaojie Yuan. Lan: learning adaptive neighbors for real-time insider threat detection. *IEEE Transactions on Information Forensics and Security*, 2024.

[2] Usman Rauf, Fadi Mohsen, and Zhiyuan Wei. A taxonomic classification of insider threats: Existing techniques, future directions & recommendations. *Journal of Cyber Security and Mobility*, 12(2):221–252, 2023.

[3] Shuhan Yuan and Xintao Wu. Deep learning for insider threat detection: Review, challenges and opportunities. *Computers & Security*, 104:102221, 2021.

[4] Fatima Rashed Alzaabi and Abid Mehmood. A review of recent advances, challenges, and opportunities in malicious insider threat detection using machine learning methods. *IEEE Access*, 12:30907–30927, 2024.

[5] Chunrui Zhang, Shen Wang, Dechen Zhan, Tingyue Yu, Tiangang Wang, and Mingyong Yin. Detecting insider threat from behavioral logs based on ensemble and self-supervised learning. *Security and Communication Networks*, 2021(1):4148441, 2021.

[6] S Asha, D Shanmugapriya, and G Padmavathi. Malicious insider threat detection using variation of sampling methods for anomaly detection in cloud environment. *Computers and Electrical Engineering*, 105:108519, 2023.

[7] Brian Lindauer. Insider Threat Test Dataset. 9 2020.

[8] Bader Al-Sada, Alireza Sadighian, and Gabriele Oligeri. Mitre att&ck: State of the art and way forward. *ACM Computing Surveys*, 57(1):1–37, 2024.

[9] P Rajesh, Mansoor Alam, Mansour Tahernezhadi, A Monika, and Gm Chanakya. Analysis of cyber threat detection and emulation using mitre attack framework. In *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, pages 4–12. IEEE, 2022.

[10] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. Reading the tea leaves: A comparative analysis of threat intelligence. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 851–867, Santa Clara, CA, August 2019. USENIX Association.

[11] Haitao Xiao, Yan Zhu, Bin Zhang, Zhigang Lu, Dan Du, and Yuling Liu. Unveiling shadows: A comprehensive framework for insider threat detection based on statistical and sequential analy-

sis. *Computers & Security*, 138:103665, 2024.

[12] Motahareh Dehghan, Babak Sadeghian, Erfan Khosravian, Alireza Sedighi Moghaddam, and Farshid Nooshi. Proapt: Projection of apts with deep reinforcement learning. *ISeCure*, 17(1), 2025.

[13] P Lavanya, H Anila Glory, and VS Shankar Sriram. Mitigating insider threat: a neural network approach for enhanced security. *IEEE Access*, 2024.

[14] Preetam Pal, Pratik Chattopadhyay, and Mayank Swarnkar. Temporal feature aggregation with attention for insider threat detection from activity logs. *Expert Systems with Applications*, 224:119925, 2023.

[15] Duc C Le, Nur Zincir-Heywood, and Malcolm Heywood. Training regime influences to semi-supervised learning for insider threat detection. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 13–18. IEEE, 2021.

[16] Raghav Bhardwaj, Morteza Noferesti, Madeline Janecek, and Naser Ezzati-Jivan. Emd-scs: A dynamic behavioral approach for early malware detection with sonification of system call sequences. In *2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 1728–1737. IEEE, 2023.

[17] Yue Guan, Morteza Noferesti, and Naser Ezzati-Jivan. A two-tiered framework for anomaly classification in iot networks utilizing cnn-bilstm model. *Software Impacts*, 20:100646, 2024.

[18] Vipin Kumar and Basant Subba. A tfidfvectorizer and svm based sentiment analysis framework for text data corpus. In *2020 national conference on communications (NCC)*, pages 1–6. IEEE, 2020.

Mohammad Mohammadi is a Master's student in Artificial Intelligence at Bozorgmehr University of Qaenat. He holds a Bachelor's degree in Computer Engineering, which he began at the University of Sistan and Baluchestan before completing his studies at Bozorgmehr. He is passionately interested in the potential of AI to develop innovative solutions that streamline processes and enhance the quality of life.

Moein Bannaye Zahmati is currently pursuing a Master of Science in Artificial Intelligence at Bozorgmehr University of Qaenat, building upon a Bachelor's degree in Computer Engineering earned from the same institution. His academic AI research is underpinned by a strong foundation in software development, complemented by specialised, hands-on expertise in back-end engineering gained during his undergraduate studies.

Morteza Noferesti is a computer engineer and academic who earned his BS from Shiraz University of Technology, followed by his MS and PhD from Sharif University of Technology. He further advanced his research as a postdoctoral fellow in Performance Engineering at Brock University before joining the faculty at Bozorgmehr University of Qaenat. His research, documented in numerous international publications, focuses on computer networks, performance analysis, applied artificial intelligence, and vulnerability analysis.