Insider Threat Detection System Using XAI and LLM

Hemant Verma
Department of Computer Science and Information Systems
Birla Institute of Technology and Science, Pilani, Rajasthan, India
Email: h20240093@pilani.bits-pilani.ac.in

***Abstract*—Insider threats pose significant risks to organizations, as malicious actions by trusted employees are difficult to detect using traditional security measures. Insider threat detection systems which identify only the malicious users miss out on the importance of identifying the causes of maliciousness, which are of great importance. The paper presents an insider threat detection system that uses Deep-Neural network model, Explainable Artificial Intelligence (XAI) techniques and Large Language Models (LLMs) to improve both detection accuracy and interpretability. The proposed system combines LSTM-CNN based deep learning model for anomaly detection with post-hoc XAI methods to identify and explain suspicious user behaviors. Multiple LLMs are integrated into the framework to compare and generate best natural-language explanations of model outputs, aiding security analysts in understanding and responding to threats. The model generates test accuracy of 1.00 with malicious feature mapping using XAI techniques. The results indicate that combining XAI and LLM-based explanations can significantly enhance trust and transparency in insider threat detection systems.**

***Index Terms*—Insider threat detection, Explainable AI, Large Language Models, Cybersecurity, Anomaly detection**

I. Introduction

Insider threats, where trusted employees or users maliciously misuse their access privileges, remain one of the most challenging security issues for organizations. Effective insider threat detection is difficult because malicious insider behavior often resembles normal work activities, leading to high false negatives or false positives in conventional rule-based monitoring systems. Recent advancements in machine learning have improved anomaly detection capabilities for cybersecurity, enabling models to learn complex patterns of user behavior and identify deviations indicative of threats. However, a major drawback of many machine learning approaches is the lack of interpretability. Security analysts are often reluctant to trust "black-box" AI decisions without clear explanations, especially in high-stakes environments [4].

Explainable Artificial Intelligence (XAI) addresses this challenge by providing insights into model decisions. XAI techniques, such as SHAP (SHapley Additive exPlanations), Permutation feature importance and Counterfactual explanantions, help translate complex model outputs into human-understandable forms. In parallel, the emergence of Large Language Models (LLMs) like Qwen3 and other transformer-based models has opened new opportunities for generating coherent natural language summaries and explanations. LLMs have demonstrated the ability to interpret patterns and generate context-aware text, which can be applied to produce explanations of model findings in a security context.

In this paper, an insider threat detection system has been proposed that integrates a hybrid LSTM-CNN based deep neural network model [1] with XAI techniques and utilizes an LLM to deliver clear explanations. The goal is to not only accurately detect potential insider threats but also provide security officers with understandable justification for each alert. We evaluate the system on a publicly available insider threat dataset [3] and show that the approach yields high detection accuracy while significantly improving the transparency of the results.

II. Literature Review

[5] proposed a deep learning approach to detect anomalous network activity from system logs. They trained Recurrent Neural Networks (RNNs) to recognize characteristic of each user on a network and concurrently assessed whether user behavior is normal or anomalous. A common challenge noted in these studies is the class imbalance and scarcity of labeled insider attack examples, which can lead to high false positive rates. [1] have proposed an LSTM-CNN based hybrid architecture which can detect anomalous behavior through user actions. The LSTM is used to extract the features of user behavior. The CNN uses these features to find anomalous behavior. [6] have implemented an explainable AI based anomaly detection framework which identifies different events that impacts models' interpretability. In the context of Insider threats, the proposed work aims on combining rigorous XAI techniques and LLMs which would identify the causes behind the detected anomolous user's behavior and provide a narrative description of why a user was flagged as suspicious, in addition to raw numbers or scores.

III. Proposed Solution

The proposed insider threat detection system consists of three main components: (1) a deep learning-based detection model that flags potential insider threats from user activity data, (2) an explainability module using XAI techniques (such as SHAP value analysis and Permutation feature importance) to interpret the model's outputs, and (3) a Large Language Model integration that generates comprehensive natural language explanations for the detected threats.

The detection model is designed to capture temporal patterns in user behavior that indicate insider threats. The hybrid LSTM-CNN neural network inspired by previous sequence modeling work [1] has been used for the modeling task. The model takes as input a sequence of features that represent user activities. Three LSTM layers (with 60, 40, and 20 hidden units, respectively) are used to learn temporal representations of the activities. The final LSTM layer outputs are passed through a series of 1D convolutional layers (32 and 64 filters) with small kernel sizes, which act as a classifier to detect abnormal patterns in the sequence. A final dense layer with softmax activation produces an anomaly score or binary classification (malicious vs. normal) for the user or time window(session, day, week) in question.

Once the detection model identifies a suspicious user or activity sequence, the explainability module is triggered to provide insight into the model's decision. SHAP has been used to calculate feature importance values for the particular instance flagged. SHAP values correspond to the contribution of a particular feature to the model's output for that instance, based on the concept of Shapley values from cooperative game theory. In addition to SHAP, permutation feature importance is used to confirm which features most strongly influence the model globally.

The final component of our system uses different LLM(Qwen3, llama3.2 and deepseek-r1) integration to turn the above explanation information into a narrative report explaining the alert. The prompt for LLMs are carefully engineered to ensure the LLM output is factual (grounded in the features and their values) and useful (providing possible reasons why those features indicate a threat, and suggestions for further investigation). We ensure no new data is introduced by the LLM beyond the model's findings, to maintain correctness.

IV. Experimentation Details

To evaluate the proposed system, 3 experiments have been conducted on the widely-used CERT Insider Threat Dataset [3]. The CERT dataset is a synthetic collection of organizational data (logon records, file access logs, email communications, HTTP web browsing logs, etc.) with injected insider threat scenarios. Data from a specific release (r5.2) has been used which contains activities of hundreds of users over a period of 17 months, including a handful of malicious insiders who conduct attacks like data exfiltration and intellectual property theft. The dataset provides ground truth labels for users or sessions that were involved in malicious activities, allowing supervised learning and evaluation.

A. Data Preprocessing

The raw dataset contains multiple files (file, http, device, email and logon) which is merged and transformed into a unified sequence of daily user activities. For feature extraction, method proposed in [2] has been used. Each feature consists of user information – mostly categorical data encoded in numeric format for providing context for ML algorithms – and two types of features: • Frequency features, which are the count of different types of actions the user performed in the aggregation period, e.g., number of emails sent, number of file accesses after work hour, or number of Websites visited on a shared PC. • Statistical features, which are descriptive statistics, such as mean, median, standard deviation, of data. Examples of data that are summarized in statistical features are email attachment sizes, file sizes, and the number of words in Websites visited. Three levels of files were generated from feature extraction:

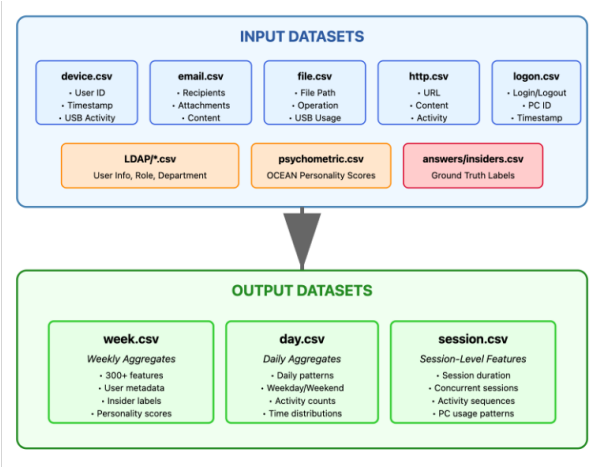

Fig. 1. Input files and output files obtained after feature extraction

Session-level Data - Session-level features capture immediate behavioral patterns, including login/logout sequences, file access patterns within sessions, resource utilization during active periods and command sequences and system interactions. Each session is represented as a sequence of activity vectors, maintaining the temporal ordering crucial for capturing suspicious action patterns.

Day-level Data - Day-level aggregation captures behavioral patterns across 24-hour periods. Features at this granularity include daily activity volume indicators, working hour deviation metrics, resource access frequency distributions and cross-system activity patterns. Day-level modeling facilitates the identification of threats that manifest through daily behavioral abnormalities, such as unusual work patterns or resource access volumes.

Week-level Data - Week-level features represent longer-term behavioral patterns like week-over-week activity trends, role-based behavioral consistency metrics, project and resource utilization patterns, organizational interaction networks This granularity enables the detection of slow-moving, strategic threats that develop across extended timeframes.

Fig. 1 illustrates the input files and the output files for feature extraction. For train and test splitting, since the dataset is highly unbalanced, separating malicious and normal users and then splitting them in 70:30 ratio was done.

B. Experiment 1 : Separate Modeling

In the Separate Modeling approach, three distinct LSTM-CNN models were developed which are operating at different

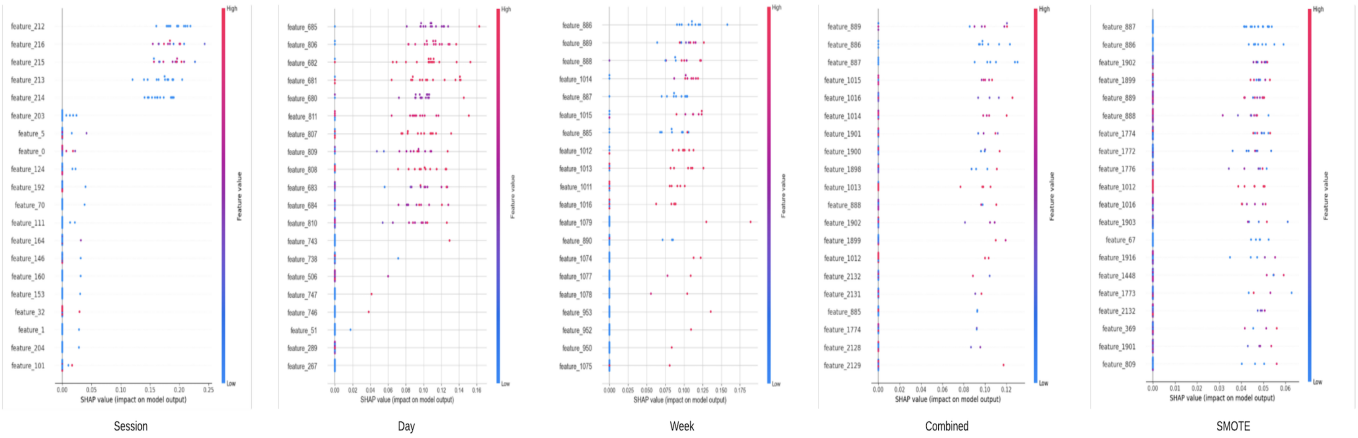

Fig. 2. SHAP summary plots showing the impact of top features on the model output. Each point represents a SHAP value for a feature in a specific instance (red indicates higher feature value, blue lower). Features are ordered by overall importance.

temporal granularities:

1. Session-level Model: Analyzes individual user sessions, capturing immediate behavioral patterns and anomalies within single login sessions.
2. Day-level Model: Aggregates user activities over entire days, detecting patterns that emerge across multiple sessions within a 24-hour period.
3. Week-level Model: Examines behavior over weekly timeframes, identifying slow-developing threat patterns that might be invisible at shorter timescales.

Each model is implemented using an identical LSTM-CNN architecture (three LSTM layers followed by CNN layers) but trained on different temporal aggregations of the same underlying user activity data. After training, the models are evaluated using SHAP and permutation feature importance which evaluates feature importance on per instance and globally respectively. Separate user- level reports are genrated using LLMs for each temporal granularity to provide operational flexibility and multi-scale threat coverage.

C. Experiment 2 : Combined Modeling

The combined modeling approach integrates session, day, and week-level behavioral data into a unified framework for insider threat detection. This multi-temporal view capitalizes on the complementary strengths of each time scale to create a more comprehensive threat detection system.

This work advances insider threat detection in several important ways:

1. The model identifies complex behavioral patterns that would be invisible at any single time scale.
2. The divergence between XAI methods reveals the multi-faceted nature of insider threats.
3. The approach allows security teams to understand threats from both immediate actions and long-term behavioral changes.

D. Experiment 3 : Combined Modeling(SMOTE)

Class imbalance is a significant issue: in the prepared dataset, less than 0.5% of the instances correspond to malicious insider actions, reflecting the rarity of true threats. To test if this imbalance affects the model prediction, synthetic minority samples using SMOTE is done for the training and testing set. Specifically, the minority class was oversampled to have at least a few hundred examples. The results of this experiment reveals that class imbalance is not affecting the model prediction but with SMOTE more robust feature-outcome relationships can be observed.

V. Results and Discussion

A. Detection Performance

The proposed LSTM-CNN model achieved strong results on the insider threat dataset. Table I summarizes the performance metrics of the model.

TABLE I
Performance of Insider Threat Detection Models

Experiment	Test Accuracy
Separate Modeling	1.00
Combined Modeling	1.00
Combined Modeling(SMOTE)	1.00

B. XAI analysis

The SHAP evaluation results are given in Fig. 2. In the plot, the features are ranked by importance with the most important feature at the top. The higher feature value indicates that the higher presence of such feature is pushing the predictions toward ”malicious” while the lower feature value indicates that a low value of this feature increases the likelihood of classifying behavior as malicious.

The obtained SHAP values show features having high impact and a clear separation between important and unimportant features. The day-level values display more scattered

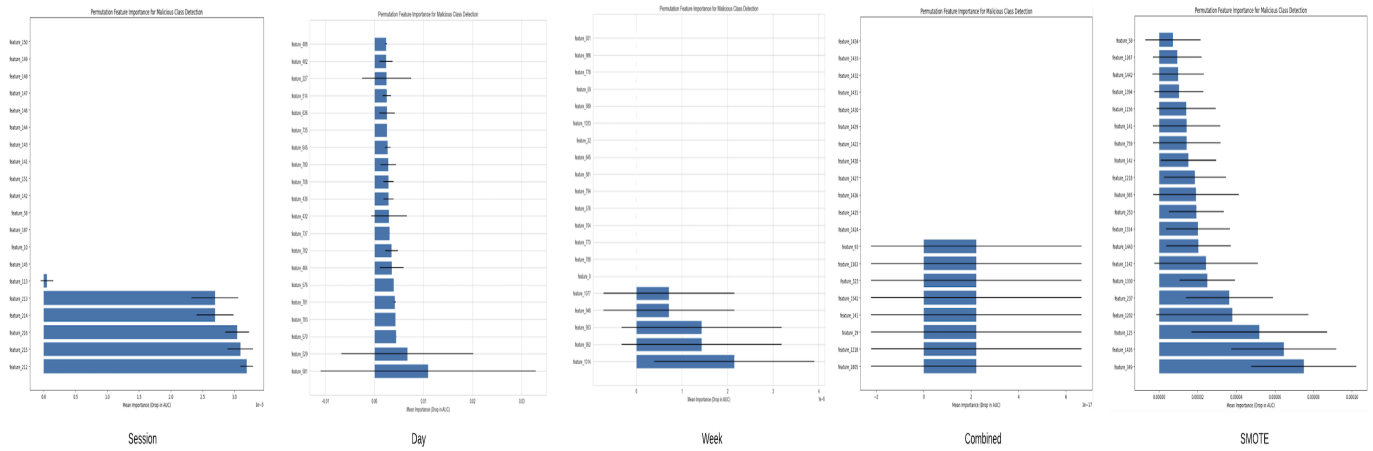

Fig. 3. Permutation feature importance plot showing the impact of top features on the model output. plot visually represents how much each feature contributes to a model's performance by measuring how much the model's accuracy decreases when a feature's values are randomly shuffled (permuted).

SHAP values across features which increases on with week-level model. Combined model exhibits broader distribution of SHAP values across more features, confirming the integrated approach captures more signals. The SMOTE model demonstrates the most balanced SHAP distribution with many features showing meaningful contributions, consistent with the permutation importance results.

The permutation feature importance results are given in Fig. 3. Permutation feature importance provides a global view of which features affect overall model performance while SHAP focuses on how features influence specific predictions for individual instances.

The permutation feature importance for session-level model shows a highly concentrated importance distribution with just a few features (at the bottom of the chart) dominating the model's performance. These few features have dramatically higher importance scores compared to other time scales. Day-level model exhibits a more distributed importance pattern with many features contributing moderately, and one standout feature at the bottom of the chart. The combined and SMOTE models display a broader and more balanced importance distribution with SMOTE model having the most balanced importance distribution.

Fig. 4 shows T-SNE visualizations obtained from model results and feature-user analysis. graphs for session, day, week and combined models show a clear separation between normal(blue) and malicious(red) which indicates high classification confidence. In session-level graph, normal data shows multiple distinct clusters for different session behaviors. In day-level graph normal and malicious centroids are relatively close, indicating less dramatic feature separation. Week-level graph indicate that feature extraction at week level produces less discriminative representations. Combined model shows clear class separation but with different topological structure than individual time scales. SMOTE graph shows complete left-right separation between classes along x axis. This creates a dramatically different feature space with extreme class separation.

C. LLM Integration

For Feature report generation, locally hosted LLMs have been used for feature analysis. The analysis generates three tier reporting structure -

Overall Explanation report - This report aggregates SHAP values across all detected malicious users. It identifies top 10 most influential features and provides domain expert interpretation of feature patterns

All User Summary report - This report contains the global analysis of all the malcious users. Serves as the entry point for investigation

Individual User Report - This report is the personalized analysis for each detected malicious user. It highlights user-specific feature importance values and tailored threat assessment and investigation recommendations.

The project employs 3 different LLMs for report generation with which a comparative analysis was performed as to which model produces the most accurate report. The following analysis were made -

Qwen3 - The project uses the model size with 8 billion parameters. This model produces the most accurate and comprehensive reports from XAI feature analysis. It has the highest feature interpretation accuracy and overall explanation coherence.

Deepseek-r1 - The project uses the model size with 7 billion parameters. This model has high feature interpretation accuracy but produces more informal responses which misses out on the formal technical aspects of the features.

Llama3.2 - The project uses Llama3.2 model with 3 billion parameters. This model produces more verbose explanations and is stronger at recommending security actions in comparison to deepseek-r1.

In summary, the integration of XAI and LLM in our system proved effective. This layered explanation approach

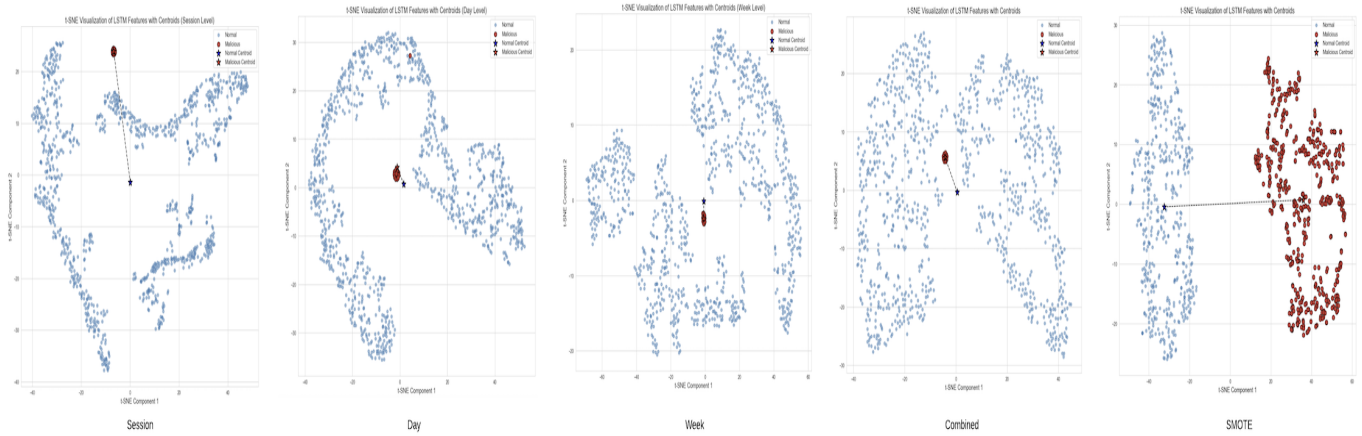

Fig. 4. T-SNE graph plots representing high-dimensional data in a low-dimensional space. Blue points indicate normal user activities and red points indicate malicious insider activities.

can greatly enhance insider threat analysis workflows by not only automating detection but also automating part of the investigation process.

VI. Conclusion and Future Works

We presented an insider threat detection system that combines a deep learning model with explainable AI techniques and explanations based on large language models. By employing SHAP and other interpretability methods, we ensure that each alert can be traced back to specific contributing factors. The incorporation of an LLM to generate narrative explanations further bridges the gap between data science output and human-centric insight, translating raw features and statistics into meaningful assessments of risk. For future work, the aim is to fine-tune domain-specific LLMs for cybersecurity so that explanations can become even more tailored to ensure efficiency in addressing evolving cybersecurity needs.

References

[1] Fangfang Yuan, Yanan Cao, Yanmin Shang, Yanbing Liu, Jianlong Tan and Binxing Fang, "Insider Threat Detection with Deep Neural Network" *Computational Science – ICCS 2018: 18th International Conference, Wuxi, China, June 11–13, 2018, Proceedings, Part I*, pp. 43-54 .

[2] D. C. Le, N. Zincir-Heywood and M. I. Heywood, "Analyzing Data Granularity Levels for Insider Threat Detection Using Machine Learning" in *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 30-44, March 2020.

[3] CERT Insider Threat Center, "Insider Threat Test Dataset (r5.2)," Carnegie Mellon University, 2016. [Online]. Available: http://www.cert.org/insider_threat/

[4] O. Yilmaz and O. Can, "Unveiling Shadows: Harnessing Artificial Intelligence for Insider Threat Detection," *Engineering, Technology & Applied Science Research*, vol. 14, no. 2, pp. 13341–13346, 2024.

[5] C. E. Tuor, K. M. Hoff, B. A. Stevens, J. S. Osborne, and E. N. Michael, "Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data," in *MILCOM 2017 - 2017 IEEE Military Communications Conference*, Oct 2017, pp. 1–6.

[6] K. Alam, K. Kifayat, G. A. Sampedro, V. Karovič and T. Naeem, "SXAD: Shapely eXplainable AI-Based Anomaly Detection Using Log Data," in IEEE Access, vol. 12, pp. 95659-95672, 2024.

Appendix A
Model Implementation and Sample Report

Layer (type)	Output Shape	Param #
input_layer_4 (InputLayer)	(None, 1, 222)	0
lstm_12 (LSTM)	(None, 1, 60)	67,920
dropout_12 (Dropout)	(None, 1, 60)	0
lstm_13 (LSTM)	(None, 1, 40)	16,160
dropout_13 (Dropout)	(None, 1, 40)	0
lstm_14 (LSTM)	(None, 1, 20)	4,880
conv1d_8 (Conv1D)	(None, 1, 32)	672
max_pooling1d_8 (MaxPooling1D)	(None, 1, 32)	0
batch_normalization_8 (BatchNormalization)	(None, 1, 32)	128
conv1d_9 (Conv1D)	(None, 1, 64)	2,112
max_pooling1d_9 (MaxPooling1D)	(None, 1, 64)	0
batch_normalization_9 (BatchNormalization)	(None, 1, 64)	256
flatten_4 (Flatten)	(None, 64)	0
dense_8 (Dense)	(None, 64)	4,160
dropout_14 (Dropout)	(None, 64)	0
dense_9 (Dense)	(None, 2)	130

Total params: 96,418 (376.63 KB)

Trainable params: 96,226 (375.88 KB)

Non-trainable params: 192 (768.00 B)

Fig. 5. Model Implementation

Fig. 6. Sample report generated via LLM