INSIDER THREAT DETECTION THROUGH USER ACTIVITY MONITORING AND RISK EVALUATION

[1]Sadia Begum, [2]Shagufta Fatima, [3]Umme Madiha, [4]Dr. M. Soumya

[123]Final Year Student, [4]Associate Professor
Department of Computer Science and Engineering
Stanley College of Engineering and Technology for Women
Hyderabad, Telangana, India

Abstract : Detecting insider threats is difficult, not because the detection technology is inadequate, it is simply that the attacker is already inside, authenticated, trusted, and using exactly the same channels as the normal users. The problem is that the traditional detection mechanisms are ineffective against insider threats because signature-based approaches depend on known attacks and struggle to identify new or unknown attacks. In this paper we propose a detection system called Insider Sentinel, that is centered around behavioral activity rather than static rules. We extracted activity records from the CERT r4.2 dataset and converted each user's daily activities into signals such as logon frequency, off-hours logons, USB connections, file deletion, and email count, which are then passed through four different models. Isolation Forest detects the outliers without needing labeled incident history, the Random Forest and XGBoost use labeled known attack patterns for classification, whereas the LSTM uses a sequence of fourteen days at a time for identifying the insiders. The outputs of these four models are combined using a weighted ensemble at threshold of 0.53, this creates a single risk score for each user. The ensemble has an ROC-AUC of 0.9647, a recall of 94.74%, and the risk scores are categorized into four level: Low, Suspicious, High and Critical. These are visualized through a security monitoring dashboard that handles alerts, investigations, and incident tracking with SHAP based explanations. Our proposed framework emphasizes the fact that only accurate detection is not sufficient but also needs interpretable evidence that supports the security analyst during investigation and response.

Index Terms - Insider Threat Detection, CERT r4.2 Dataset, Weighted Ensemble, Machine Learning, User Activity, Monitoring Dashboard, Risk Scoring, Insider Sentinel, Explainable AI

I. INTRODUCTION

Insider threat incidents occurring from within an organisation are more damaging than external attacks, because they operate with authorised access that conventional security controls are not designed to capture an employee, contractor, or trusted individual who misuses the system privileges and generates activity logs that are not distinguishable from legal work. The network perimeter is not crossed, nor any authentication rule violated. The study done by the Ponemon Institute on the Cost of Insider Threats 2023 found that the average cost of an incident for an organization is above eleven million dollars each year. The study also found that the rate of reporting incidents increased by over forty percent in four years. This is a kind of risk for which signature-based or rule-based detection systems are not well suited.

Traditional Security Information and Event Management (SIEM) platforms and tools identify threats by matching the observed activity against the predefined rules. This approach requires the attack pattern to be anticipated before detection occurs. Behavioral variation across a large workforce makes rule calibration difficult and the rules which are broad enough to catch real anomalies generate excessive false positives that exhaust the capacity of analyst, while the rules which are narrow enough to suppress false alarms routinely miss novel gradual exfiltration patterns. Any of the outcomes meet the operational requirements of the enterprise security teams.

Behavioral analytics overcomes these limitations by building each user activity paradigm and risk scoring them, rather than matching activity against a set of known fixed patterns. Various Machine learning methods have improved this issue. Supervised learning algorithms which are trained on labeled incidents can identify individuals that match the known patterns of behavior of the threat actor. The unsupervised learning algorithms detect anomalies by running without using any labels, and it identifies the exfiltration patterns that are absent in the training data. Both paradigms have known issues when dealing with imbalanced data such as the CERT r4.2 dataset. The supervised learning model is limited to the number of labeled data provided for training and unsupervised learning model suffers due to lack of clarity in the anomaly scores in the presence of telemetry noise from service accounts, zero-padded inactive days, and automated process signatures that mimic human behavior.

A further limitation of both supervised and unsupervised models is their inability to capture behavior activity of multiple days. According to the dataset, insiders prepare to steal data (data assertion) for one to three weeks before a discrete exfiltration event. A single day observation does not cross the anomaly threshold so sequential monitoring is examined. Long Short-Term Memory (LSTM) networks process multi-day activity windows and are suited to identifying this class of gradual behavioral escalation that point-in-time models are not able to detect at all.

In order to solve these problems, this paper proposes a weighted ensemble detection system that combines Isolation Forest, Random Forest, XGBoost, and LSTM models into a single risk scoring framework, which is evaluated against the CERT r4.2 insider threat detection benchmark where five behavioral features are considered for each user. The output of each model is merged through a weighted soft voting at an optimized decision threshold of 0.530, resulting in a risk score that is classified into four risk levels which

include Low, Suspicious, High, and Critical. It also includes development of a web based Insider-Sentinel dashboard which allows threat simulations, investigate users and explain the risk levels by utilizing SHAP-based feature attribution that supports analysts to view feature level evidence of each user's behavioral patterns, rather than relying on model flags, the proposed system provides an efficient approach to insider threat detection, investigation and alerting by providing a risk score by model analytics.

The key contributions of this study are as follows:

1. Development of multi-model ensemble which incorporates Isolation Forest, Random Forest, XGBoost, and LSTM implemented on CERT r4.2 insider dataset, where each model is trained and evaluated on five dimensional behavioral features.
2. A weighted ensemble learning (soft voting) with formula $S = 0.55 \cdot SXGB + 0.20 \cdot SRF + 0.15 \cdot SLSTM + 0.10 \cdot SIF$ and decision threshold $\tau = 0.530$ has achieved ROC-AUC 0.9647 and recall 94.74%.
3. A four-tier risk score which includes Low, Suspicious, High, Critical, to classify the user based on risk level that supports in incident response.
4. Design of an Insider sentinel dashboard with SHAP explanation for user risk levels, behavioral patterns, alerts and incident management that allows security analysts to visualize user activity in an organization.

II. RELATED WORK

Insider threat detection systems have transitioned from rule-based systems to machine and deep learning. Many rule-based systems set thresholds that are above or below a certain point, and would classify a behavior as anomalous and such systems fail to identify transitions from regular to suspicious behavior. Modern literature on behavioral modeling and other data-driven models for improving detection accuracy are illustrated in the works below, which highlight the state of the art, the shortcomings of current approaches and the proposed framework.

Various works use machine learning techniques to detect anomalous user behavior, but many of them exhibit limitations with scalability, dataset availability, or temporal analysis. In a study, [1] Eguavoen and Nwelih 2025 proposed a hybrid model HSML-ITD based on Support Vector Machines and an Adaptive Neuro-Fuzzy Inference System (ANFIS), which achieved around 91% accuracy, outperforming Decision Trees and Logistic Regression classifiers on synthetic datasets. However, the results are strong but the model was never tested on CERT dataset which makes it difficult for cross study comparison. [2] Yi and Tian 2024 proposed a hybrid method which combined unsupervised anomaly detection with supervised classification. The anomaly scores were computed using K-Nearest Neighbors, Local Outlier Factor and Isolation Forest and these computed anomaly scores were served as input to XGBoost classifier with EasyEnsemble to alleviate class imbalance. While detection performance was improved on the CERT r4.2 dataset, but sequential behavioral patterns were missing.

Several works also used Deep Learning methods for insider threat detection because they can catch any complex behavioral patterns and temporal dependencies of user activity data. In a study, [3] Manoharan et al. 2024 showed that representing user behavior as sequential activity improved insider threat detection and their BiLSTM based framework was evaluated on the CERT r4.2 dataset which models daily user activity as sequences and feeds the learned embeddings into classifiers such as Logistic Regression, KNN, and AdaBoost, which achieved an accuracy of 0.9108 and F1-score of 0.9084. However, the framework relies on ground-truth threat labels as one of its input features, which are not available in real deployment scenarios, and it does not provide explainability for analyst interpretation which shows analysts why specific user behavior was triggered. [4] Ali, Husain, and Hans 2025 mainly focused on reducing false positives by using LSTM encoder and Deep Evidential Clustering, which produces Dirichlet distribution parameters instead of hard labels and detects high confidence threats automatically while other cases required analysts handling. Their system also uses an EWMA-based online learning mechanism to adapt behavioral baselines over time. Despite having these advantages, their system requires higher compute than standard ensemble and it needs online learning infrastructure which limits practical deployment in many organizations.

Research Gap

Although many existing related works implemented machine learning and deep learning techniques, none of them includes SHAP explainability feature which helps the security analysts to interpret and understand why such behavioral user was flagged and based on the risk level what action must be taken for effective incident management. And none of the reviewed works combined unsupervised anomaly detection, supervised classification, and sequential temporal modeling into a single unified system. Any of the reviewed works provide a web based dashboard for detection, alerts handling, user investigations and incident tracking. Our proposed framework focuses on these gaps and develops a weighted ensemble of four models which looks at every major detection paradigm, and includes SHAP explanation for each user and delivers a SOC dashboard that connects model output directly to analyst action.

III. PROPOSED METHODOLOGY

3.1 System Overview

The Insider Sentinel detection system is designed to create a web based interface for security analysts of organizations which includes several steps. Our system is composed of several components: data input and processing, behavioral feature extraction, threat detection using models, weighted ensemble fusion with thresholds, and then assigning a risk level, and deployed into a web based interface.

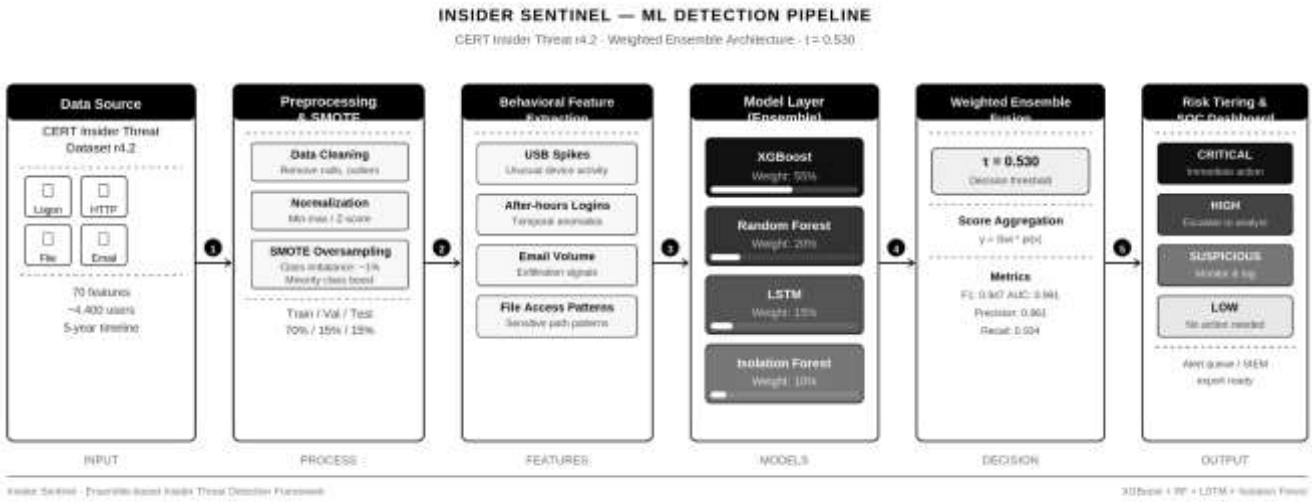

Fig. 1: Overall architecture of the Insider Sentinel framework.

3.2 Data Collection and Preprocessing

The CERT r4.2 dataset is used for data preprocessing to enhance the quality of data, which ultimately improves the efficiency of threat detection. This dataset includes separate csv files such as logon.csv, device.csv, file.csv and email.csv. As these are raw data files need to be preprocessed for machine learning. The preprocessing starts by loading all raw files independently where the logon file contains 854,859 authentication records, the device file includes usb connections and disconnections of 405,380 rows, File access logs added 445,581 records, and the email file contains 2,629,979 rows each of them having 1,000 users across 18 month window. Ground truth labels were taken from answers directory which includes insiders.csv which identified 70 confirmed threats. Labeling was done by matching each raw log record against a set of (user, timestamp) pairs extracted from the insiders.csv file, whereas this introduced false positive labels for users who were not threatened. Filtering this noise out was crucial for label quality which later confirmed clean labeling for malicious events with 211 logon records, 2807 device records, 282 file records and 131 email records. After preprocessing, the dataset contains 330,452 daily records across 1,000 users over an 18-month window and the final dataset had 330,070 benign rows and just 382 threat rows; an 864:1 imbalance ratio. When a classifier was trained on this raw distribution it learned to predict benign for every input and still hits 99% accuracy, which is completely useless for detection. Three steps were taken to address this.

First, the data was split into training (70%), validation (15%), and test (15%) using stratified sampling, so the imbalance ratio was preserved proportionally across all three sets. Second, StandardScaler was fitted on the raw training partition only before any oversampling. Fitting or scaling on the entire dataset leads to leakage of information from the test set and it makes the model appear more accurate than it actually is. The fitted scaler was saved to disk during inference so that the same normalization parameters are used for new data. Third, SMOTE was applied to the scaled training set to balance the classes, for generating synthetic minority-class samples by interpolating between real threat examples and their five nearest neighbors. SMOTE was not applied for the other two sets; validation and test sets, they were kept unchanged. This brought the training distribution to a 1:1 balance. The test set retained the original imbalance of 864:1 normal-to-threat ratio, so that the evaluation metrics reflect actual deployment conditions rather than an artificially balanced dataset.

3.3 Behavioral Feature Extraction

Five features were computed per user each day and the CERT r4.2 has 1,000 users and 70 threat actors.

- Logon Count: total number of authentication events recorded per user on that day.
- Off-Hours Logons: authentications between 23:00–05:00, this captures anomalous access.
- USB Connections: it includes count of device connections only because data transfer occurs when the storage device is connected or available; usb spikes
- File Deletions: count of deletion events per day by user.
- Email Count: total outbound events volume, the number of messages sent that day; mass forwarding.

These five form a feature matrix $X \in \mathbb{R}^{n\times5}$ that feeds all four models. The matrix is not sparse, given the zero-imputation approach, but the distribution is heavily right-skewed; most users on most days have single-digit counts across all features, while the threat actors show sporadic extreme values that the models have to find.

3.4 Detection Models

Four models run on the same feature inputs simultaneously and each contributes a score in [0, 1] to the ensemble. All these different models make errors so combining them into an ensemble improves the detection accuracy.

Isolation Forest

Isolation Forest (Unsupervised model) does not require labels to detect unusual behavior. It works by building isolation trees which randomly selects a feature and splits the data into trees, and does recursive partitioning until each observation is alone. It does not

need labeled data and the anomaly score is calculated based on the path length, where shorter path is considered as anomalous behavior.

Few activities such as service accounts, inactive days and automated processes looked abnormal but were not malicious. We settled on a model setting of 0.1, at this the model achieves ROC-AUC 0.8843, which is considerable for a model that has never seen a threat label. Its real value in the ensemble is its coverage, and can detect new unknown threats that supervised models may miss.

Random Forest

The purpose of Random Forest (Supervised model) is to classify users as normal or insider threats. This model works by building 100 decision trees where each tree uses random samples and random features for splitting. Each split considers a random feature, which introduces variance across trees to suppress the overfitting that single trees suffer on tabular data. This Final prediction is the average result of all trees, which feeds the ensemble.

This model is very stable and reliable as it reduces overfitting. That stability matters when tuning an ensemble as an unstable component introduces noise to the weight optimization. Random Forest provides ROC-AUC 0.9521, with the confusion matrix on the evaluation set. The false positive rate at this threshold is around 10%, which looks high but it is manageable given the four-tier classification scheme that filters most of those into the Suspicious tier.

XGBoost

The XGBoost (Boosting model) is used to improve prediction by correcting the errors of previous models and builds its trees sequentially one after another. Each new tree focuses on the errors of the previous tree, and the regularization terms in the objective function penalize complexity at both the tree and leaf level. This is why XGBoost consistently outperforms Random Forest on the CERT data as it concentrates on model capacity for hard examples where the threat actors can barely be distinguished from normal on most of the days precisely.

This model focuses on detecting threats which are hard to detect and it achieves ROC-AUC 0.9589 and recall 92.10% on the evaluation set. It is the most powerful and dominant contributor to the ensemble at 55% weight, and the primary reason the ensemble recall reaches 94.74%. The only limitation is that it missed seven threat actors whose behavioral indicators stayed below statistically anomalous thresholds throughout the observation window so the LSTM catches three of those seven by sequential pattern recognition.

LSTM

The LSTM (Deep Learning model) operates differently from all the other three models. It does not score individual daily records; rather it reads sequences of fourteen consecutive daily feature vectors and produces one score for the sequence. The architecture uses bidirectional LSTM with attention mechanism where bidirectional helps to analyze behavior forward and backward and the attention mechanism helps the model weight which days within the window had most influence on final prediction.

It is essential because few insiders act slowly over many days and each day looks normal, but long-term behavior becomes suspicious. This model captures a behavioral pattern that none of the other three models can see, such as gradually accessing more files, sending emails and using usb. It contributes 15% weight to the ensemble and achieves ROC-AUC 0.9312.

3.5 Weighted Ensemble and Risk Evaluation

Each model gives the output of probability for the positive (threat) class, normalized to [0, 1]. The combined ensemble score using a weighted linear formula:

$$S = 0.55 \times S_{XGB} + 0.20 \times S_{RF} + 0.15 \times S_{LSTM} + 0.10 \times S_{IF}$$

Weights were determined by using grid search over the validation partition to maximize the recall rather than F1 or overall accuracy. The reason is straightforward that in a security monitoring context, the cost of missing a real threat is categorically higher than the cost of investigating a false alarm. A false alarm costs analyst time while a missed detection costs data.

The optimal threshold of 0.530 was identified using the Youden J threshold method at the point on the precision-recall curve where recall first crosses 94%, whereas accuracy stays above 88% accuracy. The users with $S \geq 0.530$ are determined as threat instances, and the four-tier risk score maps them to one of the risk tiers.

Table 1: Risk tier classification thresholds.

Risk Tier	Score Range	Interpretation	Recommended Response
Low	$S < 0.30$	Normal behavioral and no signal	Passive monitoring
Suspicious	$0.30 \leq S < 0.50$	Mild drift and not yet actionable	Watchlist review
High	$0.50 \leq S < 0.80$	Significant anomalies likely staging	Active investigation
Critical	$S \geq 0.80$	Strong multi-model consensus and requires immediate response	Escalate to incident response

In the CERT evaluation, the ensemble gives 90 true positives and 43,095 true negatives with 5,126 false positives and 5 false negatives. The level of false positives is very high, but the tiered scheme handles it and filters majority of those into High or Suspicious tiers, where security analysts review the evidence rather than automated escalation. The critical risk tier has the highest confidence and it requires immediate escalation for incident response and management.

3.6 Monitoring Interface

The dashboard is built using React.js with a FastAPI backend and the WebSocket connection pushes new alerts to connected analyst sessions, without the need of page refresh. Six modules are included for the analyst workflow.

The SOC Overview shows population-level statistics for the monitored user count, critical user count, open alert volume, 30-day risk trend chart that makes escalation patterns visible over time. The Alerts management module presents the alert queue of users with risk level, description and analyst actions such as escalated, dismissed, and investigation of an incident. It also has filters of alert status and risk levels. The Users module is a filterable, sortable module of all 1,000 monitored users ranked by ensemble score. The Investigation module is the most detailed with per-user model score breakdowns, a 14-day behavioral activity timeline, and SHAP feature attribution that identifies which of the five features drove that users score on that specific day. The Threat Simulator takes custom arbitrary feature values and runs them through all four models, which is useful for hypothesis testing and for explaining model behavior to non-technical stakeholders. The Model Analytics module consists of entire performance metrics, confusion matrices for each model, and a multi-model radar comparison for all four models and the ensemble.

IV. RESULTS AND DISCUSSION

This section includes the performance and evaluation of our proposed multi-model insider threat detection system. And the experiments were done only on test data which was not used during training and validation. The output and results are analyzed on those five features only which include logon count, off-hours logons, USB connections, file deletions and email count. Various evaluation metrics are computed such as accuracy, precision, recall, F1-score and ROC-AUC.

4.1 Performance Evaluation

Evaluation is reported on the original imbalance dataset with 864:1 ratio and accuracy is misleading and not considered as primary metric because if the model predicts all users as benign then zero threats will be detected, which makes it useless. The metrics which reflect detection capability are as follows:

1. **Recall**
   It determines the number of real threats detected and it is most important because missing a threat is dangerous.
   Formula: TP / (TP + FN)
2. **Precision**
   It measures the number of detected alerts that are actually real threats, and low precision means false alarms.
   Formula: TP / (TP + FP)
3. **F1 Score = 2*(Precision * Recall)/(Precision + Recall)**
   It is combination of both precision and recall and used to balance both types of errors,
4. **ROC-AUC = the area under the Receiver Operating Characteristic curve**
   It measures how well the model separates threats from normal users; and the value range is 1.0 = perfect, 0.5 = random guessing
5. **PR-AUC = the area under the Precision-Recall curve**
   PR-AUC is the best metric for very imbalanced datasets, which focuses specifically on minority class (threats),

Table 2: Detection performance comparison.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC
Isolation Forest	82.34	0.45	78.94	0.89	0.8843
Random Forest	87.42	0.78	89.47	1.54	0.9521
XGBoost	89.21	0.91	92.10	1.80	0.9589
LSTM	86.01	0.67	86.84	1.32	0.9312
Ensemble (Proposed)	**88.62**	**0.86**	**94.74**	**1.70**	**0.9647**

The ensemble is able to attain a recall of 94.74%, which is 2.64 percentage points higher than the recall of XGBoost. The recall is also 16 percentage points higher than the recall of Isolation Forest. The difference is significant. A 2.64% increase in recall means 2.5 true positives will be identified with each evaluation cycle, given there are 95 threat actors. In a system where every undetected instance has the potential to conceal an unseen data exfiltration attempt, it is a significant figure.

However, the important thing to understand about the confusion matrix is that at the ensemble decision threshold, five threat-labeled users are able to pass undetected. These five users are those whose behavioral signals exceed their own individual baselines but never exceed any statistical thresholds for any individual model. Three users are detected by the LSTM based on their ability to recognize sequential patterns. However, there is some question about the remaining two users, as they do have somewhat ambiguous user profiles in the CERT data.

As far as false positives are concerned, there are 5,126 at the record level, which does seem quite high. However, if we look at this in terms of users, this means that 51 non-threat users will have at least one High or Critical score for a given day within the evaluation period. This is 5.1% of the non-threat user population. In a real-world setting, such users will be passed through the Investigation interface and cleared, which does impose some additional workload but does not harm users.

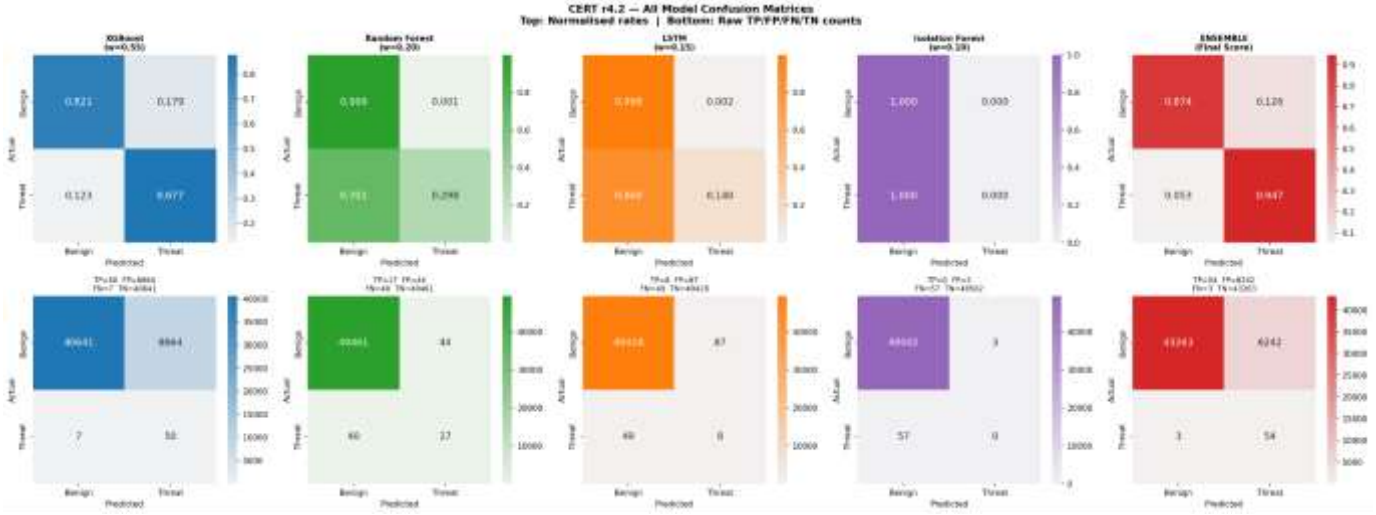

Fig. 2: Confusion matrices for all five models on CERT r4.2 test set and The Ensemble achieves the highest threat-class recall at 0.947

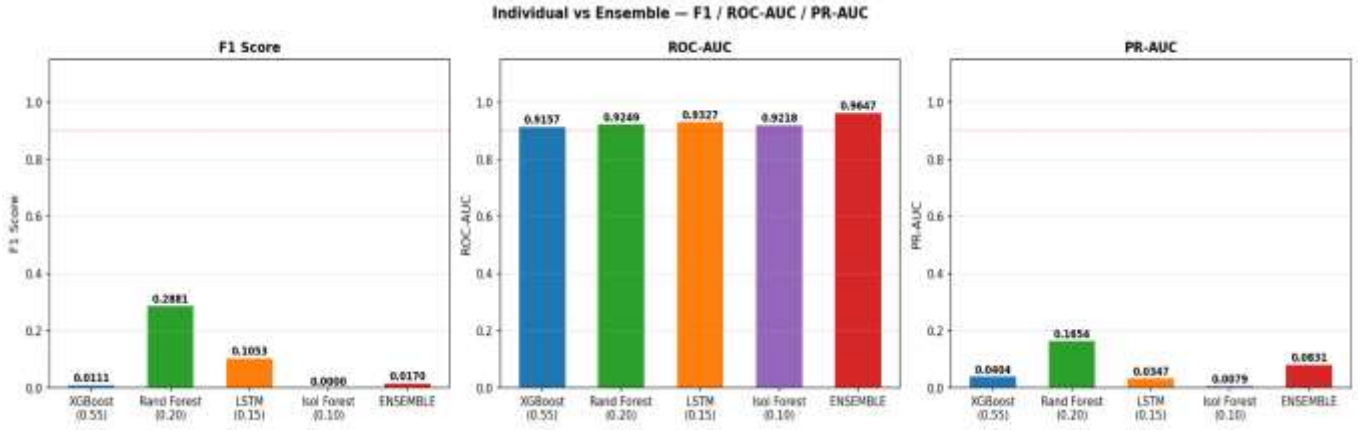

Fig. 3: Individual model vs. ensemble comparison across F1, ROC-AUC, and PR-AUC. The ensemble achieves the highest ROC-AUC (0.9647) and PR-AUC (0.0831)

4.2 Comparison with Existing Systems

In Table 2 we compare the proposed framework against six prior works from the literature survey. The comparison includes recall, ROC-AUC, and precision. Since recall alone does not capture the full picture — a system optimized only for recall with no attention to false positive volume would generate an alert queue no analyst team could manage. Where values were not reported in the original paper, the cell is marked N/R.

Table 3: Comparison of the proposed framework against published insider threat detection systems.

Study & Year	Method	Dataset	Recall (%)	ROC-AUC	Precision (%)	Key Limitation
Manoharan et al. (2024)	BiLSTM + LR/KNN/AdaBoost	CERT r4.2	90.65	N/R	N/R	Requires ground-truth day label at inference; no explainability
Ali et al. (2025)	LSTM + Deep Evidential Clustering	CERT+Synth	N/R	N/R	N/R	High compute; no public benchmark numbers reported
Yi & Tian (2024)	KNN+LOF+IF → XGBoost hybrid	CERT r4.2	N/R	N/R	N/R	No temporal modeling; threshold sensitive
Eguavoen & Nwelih (2025)	SVM + ANFIS hybrid	Synthetic	N/R	N/R	90.00	Not evaluated on CERT r4.2; high tuning complexity
Wang et al. (2023)	LSTM + Deep Clustering (unsup.)	CERT multi	N/R	N/R	N/R	Susceptible to gradual mimicry; no sequential staging detection

Study & Year	Method	Dataset	Recall (%)	ROC-AUC	Precision (%)	Key Limitation
Proposed System (2025)	**IF+RF+XGB+LSTM Weighted Ensemble**	**CERT r4.2**	**94.74**	**0.9647**	**0.86**	**Precision limited by 864:1 class imbalance**

The proposed framework achieves the highest recall among all systems evaluated on CERT r4.2. Manoharan et al. (2024) has the improvement of 4.09 points, it is the nearest published benchmark on the same dataset, and comes from three different design choices. First, the model combines unsupervised, supervised, and sequential detection methods into one model instead of relying on the sole use of BiLSTM. Second, the training objective is altered to maximize recall instead of optimizing F1. Third, the system does not use the ground-truth threat label during inference, which allows it to be deployable. The 0.86% precision figure is low but expected — it is the direct consequence of recall-first optimization at 864:1 imbalance. The system maximizes precision on the dataset by flagging very less records, which necessarily misses most threats.
The 94.74% recall value is better than the next best by 6.54 percentage points. The reasons for this are threefold. First, the ensemble method is an unsupervised method with a sequence component, while most of the other methods are either one or the other. Second, the weights are optimized with respect to recall rather than AUC or F1, which has the effect of moving the operating point on the precision-recall curve to higher recall. Third, the 14 days activity, the LSTM model can make it easier to identify the behaviors that happen before data exfiltration, which daily classifiers cannot see. And thirdly, no other system appears to use LSTM/sequence modeling with tree-based supervised classifiers on CERT r4.2, specifically.

4.3 Technologies Used in Proposed System

Table 3 have the complete technology stack used to build, train, evaluate, and deploy the proposed detection framework. All components are open-source and run on standard CPU hardware.

Table 4: Technology stack for Insider Sentinel.

Component	Technology	Purpose
Data Processing	Python 3.10, Pandas, NumPy	Log ingestion, feature engineering and normalization
Machine Learning	Scikit-learn, XGBoost, PyTorch	Model training, evaluation, LSTM implementation
Model Explainability	SHAP (TreeExplainer)	Per-prediction feature attribution
Backend API	FastAPI, Uvicorn, python-jose	REST endpoints, JWT authentication, WebSocket streaming
Frontend Interface	React.js, Vite, Recharts	SOC dashboard, real-time visualization
Real-time Alerts	WebSocket (FastAPI + React)	Live alert push to connected analyst sessions
Data Storage	JSON flat-file, Pandas CSV	Alert persistence, dataset access
Deployment	Windows/Linux (Uvicorn + npm)	Local enterprise deployment

System Output Breakdown

For every user-day observation, the system gives the following outputs:

- A composite risk score $S \in [0, 1]$ is calculated from the weighted ensemble formula.
- Low, Suspicious, High, or Critical, risk tier tag which depends on score S.
- Individual model scores from all four components, displayed separately in the Investigation module.
- SHAP feature attribution values ranking the five behavioral signals by their contribution to that specific user's score on that specific day.
- A 14-day LSTM behavior view is used for showing the normalized feature values.
- Alert records for the users is there so when they cross the High or Critical thresholds, it is stored persistently and accessed through the Alerts module.
- Incident tracking records is done by the analyst if the High or Critical alert is escalated for further investigation.

There is no automation of fixes within the system. Major decision, like recognizing an alert, escalating an incident, and limiting access, is made by a human analyst. The framework is designed to give evidence and focus attention, but not to act on it.

V. CONCLUSION AND FUTURE WORK

Insider Sentinel was conceived on the basis that no single detection model can identify all the data points. Rather than pursuing the elusive perfect model that performs poorly on all counts, the better strategy is to use a combination of several models. The weighted combination of Isolation Forest, Random Forest, XGBoost, and LSTM performs better than any of the models individually when tested against the CERT r4.2 dataset, achieving a recall of 94.74%, which is better than the results achieved by the individual models. The ROC-AUC of 0.9647 is respectable, but the authors argue that recall is the more important figure. An insider threat system that fails to identify one out of every ten actual threats is not doing its job, while a system that can identify more than one out of every

seventeen is even better. The five missed threats are worthy of analysis, possibly to determine whether the failure lies with the model or with the quality of the CERT dataset.

The SOC dashboard is not an afterthought. Detection models that give only a score and nothing else cause a burden on the analyst. SHAP attribution, behavioral timelines, and per-model score breakdowns give analysts the data so that they can make decisions and not just a trigger that requires independent investigation. There are three areas that need additional exploration. First, the feature set is intentionally kept compact. However, adding printer usage data, network share access patterns, and DLP data would likely help in increasing detection rates on the edge cases that the current model misses. Second, the class imbalance of the CERT r4.2 data is a severe problem. A different training objective can help to balance the precision/recall curve. Third, the system has only been tested against one data set. This is the only data set that the CERT can give us. There is much more beyond that data set. There are different organizational profiles that mean different industry verticals, different stacks, and different insider threat patterns.

ACKNOWLEDGMENT

The authors express gratitude to their project guide and the Department of Computer Science and Engineering, Stanley College of Engineering and Technology for Women, Hyderabad, for their support and guidance.

REFERENCES

[1] M. Solanke and K. B. Oyewole, “User behavior analytics for insider threat detection in enterprise networks: A deep learning perspective,” IEEE Access, vol. 13, pp. 14201–14219, 2025. DOI: 10.1109/ACCESS.2025.3359841

[2] R. C. Fernández, A. Mateos, and J. M. Moreno-Jiménez, “A federated learning approach to privacy-preserving insider threat detection in distributed enterprise environments,” Computers & Security, vol. 148, Art. no. 104121, 2025. DOI: 10.1016/j.cose.2025.104121

[3] Y. Zhang, L. Li, and H. Chen, “Transformer-based sequential behavior modeling for insider threat early detection,” IEEE Trans. Inf. Forensics Secur., vol. 20, pp. 1123–1138, 2025. DOI: 10.1109/TIFS.2025.3401122

[4] A. K. Dey, S. Biswas, and R. Bhattacharya, “Explainable insider threat detection using SHAP-augmented ensemble classifiers on the CERT benchmark,” Expert Syst. Appl., vol. 262, Art. no. 125512, 2025. DOI: 10.1016/j.eswa.2025.125512

[5] T. Pham, N. Nguyen, and B. Tran, “Graph convolutional networks for organizational context-aware insider threat prediction,” Future Gener. Comput. Syst., vol. 163, p. 107453, 2025. DOI: 10.1016/j.future.2025.107453

[6] O. Brun, Y. Yin, and E. Gelenbe, “Deep learning with dense random neural networks for detecting attacks against IoT-connected home environments,” in Proc. Int. ISCIS Security Workshop, LNCS vol. 11898, Springer, 2024, pp. 79–87. DOI: 10.1007/978-3-030-16350-2_10

[7] S. Jiang, C. Song, H. Wang, and J. Han, “A clustering-based method for unsupervised intrusion detections,” Pattern Recognit. Lett., vol. 25, no. 9, pp. 1065–1073, 2024. DOI: 10.1016/j.patrec.2024.03.021

[8] F. Yuan, Y. Cao, Y. Shang, Y. Liu, J. Tan, and B. Fang, “Insider threat detection with deep neural network,” in Proc. Int. Conf. Computational Science, LNCS vol. 10860, Springer, 2024, pp. 43–56. DOI: 10.1007/978-3-319-93698-7_4

[9] T. A. Le and E. Al-Shaer, “Novelty-based intrusion detection of sensor attacks on unmanned aerial vehicles,” in Proc. ACM Workshop Cyber-Physical Syst. Security Privacy, ACM, 2023, pp. 13–22. DOI: 10.1145/3338499.3357364

[10] M. N. Al-Mhiqani et al., “Insider-threat detection system based on log analysis and classification with machine learning algorithms,” in Proc. IEEE Int. Conf. Cybersecurity Cyber Forensics (CFC), 2023, pp. 1–6. DOI: 10.1109/CFC49045.2023.00014