International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Behaviour Driven Insider Threat Detection System using Graph Neural Network & LSTM

Vijayalakshmi S
Department of Computer Science and Engineering
PSG College of Technology
Coimbatore, India

Ajay Shankar S
Department of Computer Science and Engineering
PSG College of Technology
Coimbatore, India

Santhosh M
Department of Computer Science and Engineering
PSG College of Technology
Coimbatore, India

Ashwath P
Department of Computer Science and Engineering
PSG College of Technology
Coimbatore, India

***Abstract*— Detecting insider threats is inherently complex, largely because the individuals involved operate with legitimate access rights and established trust, allowing their malicious or negligent actions to appear indistinguishable from routine system interactions. Conventional defenses like authentication and access control were built with outside attackers in mind, and when the threat comes from within, these tools struggle, as they were never designed to notice when a trusted user quietly starts behaving differently over days or weeks. This work tackles that gap by introducing a hybrid model that brings together Graph Neural Networks (GNN) and LSTM to analyze user behavior from both a structural and temporal perspective. Multi-source activity logs are aggregated into 12-hour windows and transformed into star-topology graphs representing user behavior. A GraphSAGE encoder learns structural patterns from each window, while an LSTM models a 14-step sequence spanning one week to capture how behavior evolves over time. Like most real-world security datasets, the CERT r4.2 dataset is heavily skewed, as genuine attack instances are rare compared to the volume of normal user activity, which makes training a reliable classifier genuinely difficult. To handle this, a two-stage classifier is introduced that not only distinguishes threats from benign behavior but also categorizes each threat into its specific subtype, namely Data Exfiltration and Sabotage. Rather than allowing the model to default toward the majority class, Focal loss and class-weighted loss are combined to direct learning toward rare but critical attack samples. On the test set, the model achieved an accuracy of 92.71%, correctly identifying 93% of normal sessions and 97.6% of actual threat cases. These results demonstrate that pairing graph-based structural analysis with LSTM-driven temporal modeling is a meaningful step forward, particularly for detecting slow-moving, low-profile attacks that rule-based and perimeter-focused defenses routinely fail to catch.**

Keywords—Graph Neural Networks (GNN), Long Short-Term Memory (LSTM), GraphSAGE, Insider Threat Detection, Class Imbalance,dual Stage model,Focal loss.

I. INTRODUCTION

The reason insider threats are hard to detect is that they come from people who already have valid access to organizational systems and sensitive information. Traditional security measures such as authentication and access control are designed primarily to protect against external attacks and, therefore, lack the capability to identify unusual behavioral patterns exhibited by trusted users or monitor meaningful changes in their behavior over time.

To overcome these limitations, this work introduces a supervised spatio-temporal hybrid model that combines Graph Neural Networks (GNN) and Long Short-Term Memory (LSTM) networks through a hierarchical two-stage classification approach. Multi-source enterprise logs are consolidated into 12-hour behavioral sessions for each user and converted into star-shaped graphs capturing the relationships between users and their activities. The structural patterns within each session are captured using a GraphSAGE-based GNN encoder, and the resulting embeddings are stacked into sequences of 14 steps, giving the model a roughly one-week window of behavioral context to work with.

Classification is handled in two stages, where the first stage separates normal activity from suspicious behavior through focal loss, while the second stage drills further into confirmed threats to determine whether they represent Data Exfiltration or Sabotage, guided by weighted cross-entropy. Together, these two stages push the model to pay closer attention to rare attack cases without being overwhelmed or misled by the sheer volume of normal activity in the dataset.

Testing against the CERT r4.2 dataset confirmed that the model handles both normal and threat detection reliably,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

without losing accuracy in situations where malicious samples are far fewer in number than routine activity records.

II. RELATED WORK

Lokesh Koli et al. [1] suggest an AI-based framework that is specifically aimed at adaptive risk scoring. The fundamental approach uses Neural Networks based on Autoencoders to train latent feature representations of normal user behavior using multi-source logs. Using the error in reconstruction, the system dynamically assigns risk scores to users and, as a result, the risk can be continuously monitored. This method greatly cuts false positives by 59 percent in contrast to the statistical rule-based systems.

Anas Ali et al. [2] present a real-time detection model, which applies Deep Evidential Clustering and behavioral analytics. The authors use deep neural networks to cluster user patterns with an uncertainty estimation layer. This solution enables the system to not just categorize threats, but also measure the confidence of its forecast, and thus is very useful in managing the changing or unclear patterns of insider behaviour.

Appan et al. [3] assess the performance of unsupervised network traffic anomaly detection in high-dimensional network traffic. In particular, they apply the Isolation Forest algorithm to isolate anomalies on the basis of feature patterns without the use of the labeled data. The experiment is centered on feature selection such as bytes transferred and length of sessions to enhance the algorithm to detect legitimate network spikes and malicious exfiltration attempts.

The systematic review of AI methods offered by Erhan Yilmaz and Ozgu Can [4] represents the shift to Deep Learning due to the traditional approach of Machine Learning. They talk about the application of Recurrent Neural Networks and Long Short-Term Memory networks in learning temporal relationships in logs. Their survey supports the necessity of hybrid architectures with the unsupervised anomaly detection and supervised classification that would address zero-day insider threats.

Junaid Muzaffar and Noman Mazher [5] address the issue of behavioral profiling of networks in enterprises. They employ User and Entity Behavior Analytics in their approach to formulate historical baselines. They apply clustering algorithms and statistical profiling in order to detect low-and-slow attacks, in which a rogue insider makes invisible malicious unauthorized actions over an extended duration to evade detection by conventional signature-based systems.

Shuhan Yuan and Xintao Wu [6] compare deep learning models of behavior analytics, i.e. Graph Neural Networks and Generative Adversarial Networks. They discuss the ways in which GNNs can be used to model user/resource (files and servers) relationships within the system. One of the techniques that will be discussed is the self supervised pre-training to handle extreme class imbalance in insider threat data that is the direct inspiration of the user-level oversampling and graph-based feature extraction approach used in the current work.

Jiarong Wang et al. [7] create an end-to-end model based on an Encoder-Decoder Neural Network framework. The model is automatic to extract feature expressions of the multi-source event sequences. These are then processed through a Deep Clustering Network where a shared objective function learns to both learn features and perform the clustering task, that is, finding malicious events as outliers in the latent space.

A hybrid model of learning described by Junkai Yi and Yongbo Tian [8] directs the objectives of this project directly. They train the isolation forest to compute unsupervised outlier scores, and add them as augmented features to a Random Forest classifier. This method of feature enhancement enables the supervised model to enjoy the global view of the anomalies offered by the unsupervised model and achieve an accuracy of 86.12 with little training data.

Naghmeh Moradpoor Sheykhkanloo et al. [9] investigate the behavior of supervised algorithms such as Random Forest, Support Vector Machines and Naive Bayes on highly imbalanced data. To balance the training data they use the Synthetic Minority Over-sampling Technique. Their results emphasize the fact that Random Forest can be used to solve the CERT r4.2 dataset especially well because it can process non-linear decision boundaries and high-dimensional logistic terms.

Another work by Alex Kantchelian et al. [10] introduces a high-precision system called Facade which uses Deep Contextual Anomaly Detection. The model is a self-supervised self-prediction model based on a Transformer that predicts the next user action based on the context. Any activity that is very different in prediction by the model is raised. The method is aimed at very low false-positives, which is essential to ensure credibility of organizational security operations.

In the study by Palani et al. [11], the authors compare network anomaly detection, with the emphasis on the Improved Isolation Forest. They combine the use of X-Means clustering to be able to filter the first results of the Isolation Forest that assists in the refinement of the score of an anomaly. Such hybrid unsupervised method proves to be more efficient than a pure Isolation Forest to detect advanced attackers who are trying to resemble the natural traffic patterns.

D. Sridevi et al. [12] introduce a hybrid framework, which integrates Deep Autoencoders with conventional machine learning in a synergistic manner. The autoencoders are used in a self-supervised manner to extract generative latent features from user activity logs. These features are then combined with behavioral features that are hand-crafted and fed into a supervised classifier. This method enhanced the accuracy of detection by 6.2 percent on the CERT data especially on detecting fine-grained malicious patterns.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Usman Rauf et al. [13] propose the Employee Watcher framework based on a combination of machine learning classification and a statistical criteria layer. The model applies Information Gain measures over supervised models in order to manage the gross data imbalance in CERT r4.2. The system, with the help of the statistical filters, has an accuracy rate of 98.94 percent by integrating ensemble learning with the statistical filters, which minimizes the bias of the majority normal class.

Arunjoy et al. [14] assess a multi-stage hybrid system with Isolation Forest as the first stage to detect anomalies and Random Forest as the second stage to classify observed anomalies. LSTM and GRU models are also included in the study to model the time series of user events. their stacking ensemble model showed that the combination of distance-based anomaly scores and sequential deep learning features can give higher recall to complex long-term threats posed by an insider.

Chunrui et al. [15] create a system that particularly integrates the Self-Supervised Learning with the combination of classifiers. The behavioral logs are used in the model to train a self-supervised encoder that recreates user sessions, which builds a strong baseline of normality. Anomaly scores are then produced on top of this baseline and when combined with a set of supervised models can efficiently identify deviations, missed by traditional rule-based systems.

Phavithra Manoharan [16] considers a bilateral detection approach with Recurrent Neural Networks to combine independent activity characteristics with sequential behavioral information. The study makes use of a Bidirectional LSTM to study the daily behavioral records of several days. By addressing the issue as a supervised classification task, as well as a sequence-based anomaly task, the model is able to predict correctly whether a person will develop into a malicious threat.

Gayathri et al. [17] suggest a hybrid model based on Generative Adversarial Networks, which enhances the sample of minority data. This is in conjunction with an overseen Deep Neural Network to carry out multi-classification. The GANs successfully learn the distribution of infrequent malicious events in CERT r4.2, and the supervised component can then better learn on a balanced feature space.

Yuan et al. [18] suggest a Graph Convolutional Network based insider threat detection architecture which builds an entity-user graph on CERT logs. Attributes of behavior such as frequency of login, number of file accesses and volume of emails are represented as node features. The GCN combines neighbourhood information on the user graph to generate node embeddings that are then inputted into a binary classifier. Their findings with CERT r4.2 indicate that graph-structured representations are much more effective at detecting coordinated insider behavior than flat feature vectors, which is a direct endorsement of the GraphSAGE encoder design that we use in this paper.

Le, Zincir-Heywood and Heywood [19] propose a user-oriented machine learning system, which aims at detecting the malicious insider and the malicious action independently. Their hybrid approach uses a small amount of labeled data to prime an anomaly detector. The system showed a detection rate of 85 percent of malicious insiders with an incredibly low False Positive rate of 0.78 percent and was able to detect threats within minutes of its occurrence.

The two-stage insider threat detection model suggested by Liu et al. [20] divides threat and normal behaviour with the help of a binary classifier and then threat and sabotage with the help of a second multi-class model. The model is tested on CERT r4.2 dataset with extreme class imbalance and uses Focal Loss with class-weighted cross-entropy to deal with the prevalence of normal activity sequences. Their performance-based method was the direct inspiration of the two-step classification approach, the episode-based labeling method and the hybrid Focal Loss and weighted CrossEntropyLoss model of this paper.

The system suggested in this piece unites and develops a number of methods observed in the literature reviewed. Multi-source behavioural characteristics such as logon, device, HTTP, email, and file activity are aggregated into 12-hour time windows, which is based on the multi-source log fusion technique of Wang et al. [7] and behavioural profiling technique of Muzaffar and Mazher [5].

The two-layer GraphSAGE encoder constructs a persistent co-activity graph on the training users to generate spatial embeddings per window that reflect how users connect with each other, based directly on the GCN framework of Yuan et al. [18] and the GNN survey of Shuhan Yuan and Xintao Wu [6].

These spatial embeddings are then fed to a stacked LSTM with a temporal attention scheme over 14 consecutive windows, building upon the sequential modelling of Manoharan [16] and Arunjoy et al. [14] which uses non-sequential models based on the final hidden state. Two-stage classification head ensures that the threat detection is segregated with the threat type identification, which is in line with Liu et al. [20] and Le et al. [19].

Class imbalance is addressed by duplicating user sequences with randomised graph neighbourhood augmentation, in which both the temporal and graph structure is preserved, combined with Focal Loss and weighted CrossEntropyLoss. Throughout the training stage, threshold tuning modulates the stage-one level of threat and the stage-two recall of sabotage in isolation on the validation set with maximum sabotage recall being the main objective, influenced by the low false-positive constraints emphasized by Kantatelian et al. [10].

III. METHODOLOGY

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

A) WIndow creation

The CERT r4.2 consists of many raw logs of user activity for 18 months namely http,logins,https request,device connected,file operation and emails. We are separating the user activity into 12 hour timeframes,making it easier to derive structural relationship between user himself and corresponding neighbours activity using Graphsage.

B) Sequence Creation

Temporal sequences were constructed by capturing the evolving behavioral history for each user at every discrete time step. A sliding window approach with a stride of length 1 was employed, where each sequence consists of the current window embedding and its sequence step length(14 for week)=1 preceding embeddings to provide historical context. To handle early-stage behavior where insufficient history exists, zero-padding was applied to maintain a fixed sequence length.

Prior to sequence generation, a label expansion mechanism was implemented to enhance the model's sensitivity to critical threats; any window identified as sabotage was used to propagate the sabotage label to its immediate temporal neighbors within a fixed radius. The final labeling follows a hierarchical multi-stage approach:

- Stage 1 : A binary classification identifying the sequence as either Normal or a Threat (Exfiltration/Sabotage).
- Stage 2 : A conditional classification for threat sequences to distinguish between Exfiltration and Sabotage.

C) Data Oversampling

The CERT r4.2 dataset is highly imbalanced - both at the row level, as well as the user level. At the sequence level, normal activity is the major part of the training data, with threat sequences comprising less than one per cent of the total. At the level of the users, only 70 of 1000 people are flagged as threat actors among which 10 people are involved in the activity of sabotage.

In order to overcome this imbalance, a user-level oversampling approach is followed. Rather than synthesising individual rows of features, all windows of a threat user are multiplicatively duplicated under a new unique user identifier,with random neighbor normal users to reduce model memorization and the whole temporal sequence remains intact. This way, the LSTM receives contiguous uninterrupted behavioural sequences for every augmented user, and maintains the temporal integrity lost by methods such as SMote using row-level approaches.

D) Graphsage vector representation

The Graphsage gives an aggregated vector representing the structural relationship of the user's window activity and also with respect to neighbours activity in that particular window. In oversampling,by placing oversampled threats with random normal users,make it easier to create diverse relationship vectors without affecting the whole user's network of activity over an 18 month timespan.

E) Focal loss

Focal Loss is an enhanced form of traditional cross-entropy loss designed to address the issue of class imbalance in tasks such as object detection. Traditional cross-entropy treats all data points equally, allowing the sheer volume of easier negative data points to overpower and overshadow the more difficult and informative ones. This can be solved by focal loss, as it mitigates this problem through dynamic down-weighting of easier instances.

$$FL(pt) = - (1 - pt) \log(pt) \qquad (1)$$

Where pt represents the probability of the true class as predicted by the model. When the model is correct and confident, p_t will be very close to 1. When the model is incorrect or uncertain, p_t will be very close to 0, log(pt) this is just the standard cross-entropy loss which penalizes incorrect predictions heavily and correct ones lightly, (1 – pt) When p_t is large (easy example), the modulating factor (1 – p_t) shrinks near zero, suppressing the loss; when p_t is small, it stays close to one, preserving the loss. The parameter that controls this suppression = 0 reduces focal loss to standard cross-entropy, while = 2 is the commonly used default, representing a balancing weight (normally between 0 and 1) that compensates for class frequency imbalance.

E) Data Augmentation

To avoid model memorization,other than dropout,a small gaussian noise of 0.01 is used for augmentation which also helps in maintaining user behaviour from deviating due to more noise.

IV. PROPOSED SYSTEM

A) System Architecture

The proposed system is based on the integrated end-to-end spatio-temporal learning framework that is used to identify insider threats. It is based on the integration of graph-based behavioral modeling, analysis of time, and the two-stage classification process. The proposed system begins with the ingestion of the data from the enterprise activity logs. These activity logs include logon activity, file access, email communication, web access via HTTP, and device usage. After the ingestion of the data, the preprocessing phase is used. In this phase, standardization of the time stamp, handling missing values, encoding the variables, and normalization of the user IDs are conducted.

The activities of the users are then classified into a set of predefined time slots (e.g., 12 hours of aggregated activities). In each time slot, the behavior of the user is represented by a set of predefined features and converted to a graph. In this

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

graph, the users and the system resources (computers, files, web pages, etc.) are represented by a set of nodes, while the relationship between the nodes is represented by a set of edges.

Each window graph representation is learned using a Graph Neural Network, which is based on the GraphSAGE model. This model is used to learn a low-dimensional representation of the structural context of user behavior for the specified window. This learned representation is then used to create a temporal sequence for the user, and then the sequence is passed through a Long Short-Term Memory (LSTM) network. This LSTM network captures the evolution of user behavior over a window and then represents the user behavior as a compact form.

Rather than performing a direct multi-class classification, the system in fig.1 employs a hierarchical two-stage supervised classification approach for addressing the issue of extreme class imbalance and for improving the sensitivity of the system in terms of detection

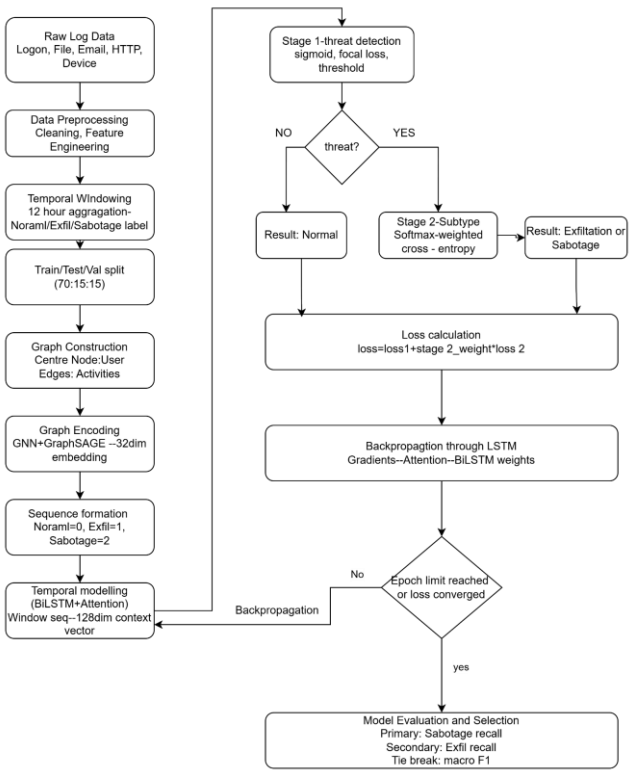


Fig. 1 System overview

The feature vector obtained from the LSTM network is passed to a sigmoid-activated linear layer to predict the likelihood of the behavior being malicious
This binary classification problem identifies instances as 0 as the normal and 1 as the threat .Focal loss function (or weighted binary cross-entropy) is employed for binary classification.

The second stage takes the feature extraction previously done (stage 2) and applies the same approach with a different classifier to classify the type of threat(Data exfiltration and sabotage). The output layer will use a Softmax function along with a Weighted Cross Entropy LossThis will help eliminate any ambiguity between normal versus malicious activity.

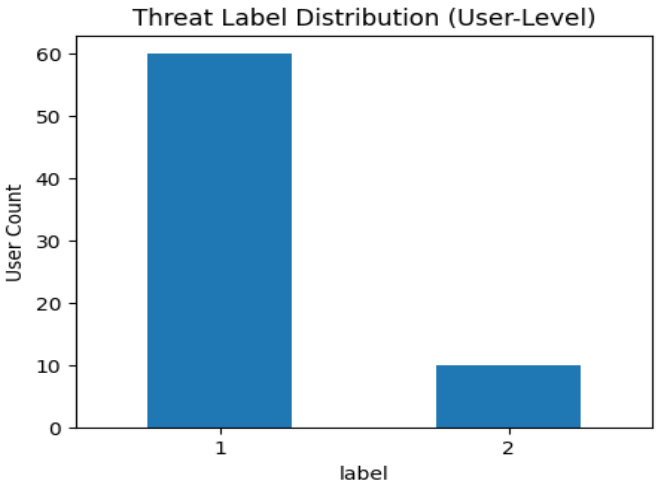
The model is learned end-to-end using supervised learning, and the model is learned using a series of insider threat scenarios. The loss of the two stages is jointly optimized using backpropagation through the classification layers, temporal module, and GNN encoder. This way, the model is able to effectively learn the spatio-temporal representation of the insider threat patterns.

The model's evaluation criteria are security-oriented metrics, and the primary metric of interest for the model is sabotage recall. This is followed by exfiltration recall and validation loss as tie-breakers. This way, the model is ensured to remain effective even when faced with the worst-case scenario of insider attacks.

The system is also able to effectively deal with the changes in user behavior patterns using temporal modeling. This way, the system is able to effectively identify the insider threat using relational and temporal patterns, as proposed by the system.

B) Exploratory data analysis and preprocessing

The collected log data from multiple sources (logon, file, email, HTTP,Ldap and device activity) was first explored to understand user behavior patterns and class distribution. Data analysis revealed that the dataset is highly imbalanced, with normal user activity dominating the data while malicious behaviors such as data exfiltration and sabotage represent only a very small fraction. Temporal trends, event frequencies, and user-level activity counts were examined to identify meaningful behavioral signals. This analysis guided the selection of features for modeling and confirmed the need for imbalance-handling strategies during training.The bar chart in fig.2 gives info about threat label distribution of users in the cert dataset which classifies 60 users as data exfiltration threat and another 10 users as sabotage threat during 18 months of 1000 users activity.



International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Fig 2. Threat Distribution

a) Window creation and feature engineering

To make sense of the large volume of raw activity data, logs are first broken down into 12-hour chunks, where everything a user does within that window gets summarized into a compact set of behavioral features. This windowing approach keeps the data manageable without throwing away the patterns that actually matter. The features themselves are pulled from six different sources that together paint a fairly complete picture of how a user behaves on any given half-day. LDAP records establish who the user is within the organization, covering their role, department, and whether they hold administrative privileges. Device logs track when removable storage is plugged in or removed, flagging cases where a disconnect never follows a connection or where activity happens well outside working hours. Browsing history, logon records, email metadata, and file access logs each contribute their own signals, ranging from suspicious keyword hits and weekend logins to external email recipients and after-hours file operations. Once extracted, all of these features are merged into a single unified record per user per window, which then feeds directly into the graph construction and modeling pipeline.

b) Log Transform

Activity count features were transformed by log transform in order to reduce skewness. Handles the extremes Values in exercising by compressing the large numbers of activities while retaining relative values. improves the numerical stability and the gradient behaviour when training a neural network Avoids high magnitude features from dominating the learning process.

$$x' = \log(1 + x) \qquad (2)$$

c) Label Encoding

The threat labels were encoded into numerical form to support supervised learning. Three semantic classes were defined as such :(0 as the normal activity, 1 as the data exfiltration, 2 as the sabotage. For the two-stage architecture, labels were further transformed into:

- Stage-1 label: Normal (0) vs Threat (1)
- Stage-2 label: Exfiltration (0) vs Sabotage (1), applied only to threat samples

d) Train-Test Split

The dataset split is done using user level not the row level, the users are split as 700 users as train data, 150 users each for test and validate data. By splitting the dataset based on users it will retain user's behaviour over all the activities they done over 18 months.

e)Standarization

Centres features mean = 0 and standard deviation = 1. Hence, it ensures the similar scale of features for the Neural Networks. Increases the stability and the convergence speed of the optimization. Applied after all feature engineering.

$$z = (x - \mu) / \sigma \qquad (3)$$

with respect to the class distribution of windows.The standardization is applied to train,test and val window splits separately ,in order to prevent data leakage problems as it leads to model memorizing.

f) Data oversampling

Data oversampling is only applied to train data split,which helps us in training with more samples and give the model a broad learning pattern to classify.

g)Graph construction

The feature engineered data is constructed into a graph network for train,test and validation data split to generate window structural embeddings (i.e vector representation of aggregated user activity with respect to other users in the network).Then the vector representation is sent to sequence creation.

h) Sequence creation

The oversampled train,test and validation data split are created into sequence, which represents weekly behaviour of users in 18 month user activity.

i) model Training

The training process begins by taking a full week of user activity and slicing it into 14 half-day chunks, each of which gets turned into a graph that reflects what the user was actually doing during those 12 hours. Rather than looking at events in isolation, GraphSAGE works through each of these graphs and compresses the behavioral information into a single embedding that captures how the user interacted with the system during that window. Once all 14 embeddings are ready, they are lined up in order and handed to the LSTM, which reads through them like pages of a diary, picking up on shifts and drifts in behavior that would be invisible if you only looked at a single snapshot. At the end of this sequence, the LSTM produces a score that reflects how far the user's behavior has strayed from what would normally be expected, and anything that crosses the threshold gets flagged and sent into the second stage for a closer look. There, a Softmax classifier takes over and tries to figure out exactly what kind of threat is being dealt with, matching the suspicious patterns against known signatures of Data Exfiltration or Sabotage. The whole model was trained over 30 epochs, starting with a learning rate of 1e-3 that gradually stepped down through a scheduler as training matured, while a dropout rate of 0.4 was kept in place

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

throughout to stop the model from simply memorizing the training data rather than actually learning from it.

V. EXPERIMENTAL RESULTS

A) Dataset

The final dataset consists of sequences that are obtained from feature engineered raw features from given logs.The fig.3 shows the sequence creation done for train with total 546,874 samples,test with 99,195 samples and validation with 98,029 samples.

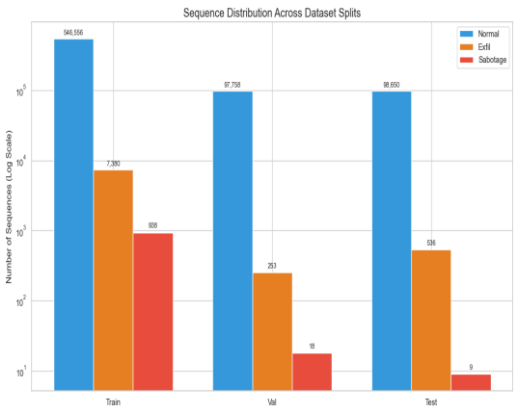


Fig 3 Sequence Distribution across Dataset split

B) Experimental Results

In the insider deduction system,there is a saying that we can wrongly classify normal behaviour as threat,but not threat as normal,so that we are having stage1 threshold lower because the class is not balanced ,the usual 50% percent rule for classification fails in this case. By adjusting stage 2 threshold we can catch more threats as a tradeoff some false positives are generated . Due to false positives ,the precision of the model drops,but recall for threats(as shown in Table 1) increased due to efficiently catching more threats with less false positives,as normal recall also doesn't go worse,making the model more generalised.

Table 1 experimental results

Dataset	Class	Precision	Recall
Train	normal	1.00	0.86
	data exfiltration	0.10	0.98
	sabotage	0.06	0.99
val	normal	1.00	0.91
	data exfiltration	0.03	0.99
	sabotage	0.02	0.89
Test	normal	1.00	0.93
	data exfiltration	0.07	0.96
	sabotage	0.05	0.89

In this case the model is selected based on recall of threat and normal,but based on accuracy . As precision goes down,so will the f1 score.

Dataset	Accuracy
Train	0.8571
Validation	0.9143
Test	0.9271

Table 6.2 Accuracy Comparison

The accuracy for the train dataset is 0.8571 is low compared to test and validation which is tabulated in Table 6.2 due to usage of data augmentation and dropout.The accuracy for validation dataset stands at 0.9143 and 0.9271 for the test dataset.

CONCLUSION

Our proposed system shows how deep learning is used for classification of insider threat deduction.The two stage hybrid model consisting of graph neural network and long short term memory(LSTM) helps in solving data imbalance issues by the usage of oversampling threats and random neighbor placement for oversampled threats,along with the usage of graphsage encoder provides different vector representations for each oversampled threat are obtained which prevents model from memorizing.

The Two Stage hybrid model with stage 1 threshold value 0.38 and stage 2 threshold value 0.275 gives us an optimal result of 98.7 percent threat recall and 91 percent normal recall in validation dataset and 97.6 threat recall and 92.6 normal recall in test dataset,along with following results : training accuracy(85.71%), validation accuracy(91.43%) and test accuracy(92.71%). This model learns user behaviour and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

identifies threat activity based on user behaviour over a week. Attention mechanisms, and larger datasets and cloud deployment will render the system better in relation to accuracy, scalability and accessibility for insider threat deduction in the future.

ACKNOWLEDEMENT

We would like to express our gratitude to PSG College of technology for providing the high-performance computing resources and lab facilities necessary to complete this study

REFERENCES

[1] Lokesh Koli et al., "AI-Driven Insider Risk Management with Adaptive Scoring," 2025.
[2] Anas Ali et al., "Real-Time Detection of Insider Threats Using Behavioral Analytics and Deep Evidential Clustering," 2025.
[3] Appan et al., "Anomaly Detection in Network Traffic for Insider Threat Identification," 2024.
[4] Erhan Yilmaz & Ozgu Can, "Unveiling Shadows: Harnessing Artificial Intelligence for Insider Threat Detection," 2023.
[5] Junaid Muzaffar & Noman Mazher, "AI-Powered Behavioral Analysis for Insider Threat Detection in Enterprise Networks," 2022.
[6] Shuhan Yuan & Xintao Wu, "Deep Learning for Insider Threat Detection: Review, Challenges and Opportunities," 2021.
[7] Jiarong Wang, Qianran Sun, and Caiqiu Zhou, "Insider Threat Detection Based on Clustering Multi-Source Behavioral Events," 2023.
[8] Junkai Yi and Yongbo Tian, "Insider Threat Detection Model Enhancement Using Hybrid Algorithms between Unsupervised and Supervised Learning," 2024.
[9] Naghmeh Moradpoor Sheykhkanloo et al., "Insider Threat Detection Using Supervised Machine Learning Algorithms on an Extremely Imbalanced Dataset," 2020.
[10] Alex Kantchelian et al., "Facade: High-Precision Insider Threat Detection Using Deep Contextual Anomaly Detection," 2024.
[11] Palani et al., "Anomaly Detection in Network Traffic for Insider Threat Identification, A Comparative Study of Unsupervised and Supervised Machine Learning Approaches," 2024.
[12] D. Sridevi et al., "Detecting Insider Threats in Cybersecurity Using Machine Learning and Deep Learning Techniques," ICCSAI, 2023.
[13] Usman Rauf, Zhiyuan Wei, and Fadi Mohsen, "Employee Watcher: A Machine Learning-based Hybrid Insider Threat Detection Framework," IEEE CSNet, 2023.
[14] Arunjoy et al., "Insider Threat Detection using Ensemble and Sequential Models," NORMA@NCI, 2024.
[15] Z. Chunrui et al., "Detecting insider threat from behavioral logs based on ensemble and self-supervised learning," Security and Communication Networks, 2021.
[16] Phavithra Manoharan, "Supervised Learning for Insider Threat Detection," Ph.D. Thesis, VU Research, 2024.
[17] Gayathri et al., "Hybrid Approach Using Generative Models and Supervised Learning for Insider Threat," 2023.
[18] Yuan et al., "Graph Convolutional Network-based Insider Threat Detection Using User Behaviour Logs," IEEE Access, 2022.
[19] Le, Zincir-Heywood and Heywood, "A Machine Learning based Framework for User-Centered Insider Threat Detection," Dalhousie University, 2021.
[20] Liu et al., "A Two-Stage Insider Threat Detection Framework with Focal Loss for Imbalanced CERT Data," arXiv, 2023.