A Transformer-Based Deep Learning Framework for Insider Threat Detection

A Ludwika[1], Dr A S N Chakravarthy[2], C Priyadarshini[3]

[1]*M. Tech, CSE department, UCEK, JNTU Kakinada, Andhra Pradesh, India*
[2]*Professor, CSE department, UCEK, JNTU Kakinada, Andhra Pradesh, India*
[3]*Assistant professor, CSE department, UCEK, JNTU Kakinada, Andhra Pradesh, India*

Abstract **-Detecting insider threats is a significant cybersecurity challenge, as conventional systems often fail to identify the subtle behavioral clues of malicious actors. This research proposes a novel approach that treats user activity logs as a language, where harmful actions deviate from normal "grammatical" patterns.**

To effectively analyze this "language," the study introduces a deep learning framework centered on a Transformer architecture. Unlike models that process data sequentially, the Transformer's self-attention mechanism can examine an entire history of user actions at once, enabling it to capture complex, long-range relationships.

The system processes a wide range of data, including logins, device usage, file access, and emails. It enhances this data by creating sessions, profiling individual user behavior, and incorporating anomaly scores from an unsupervised model. When tested on the public CERT Insider Threat r4.2 dataset, the model proved highly effective. It achieved 90% overall accuracy, a 71% precision rate in identifying threats, and a recall of 55%. This performance underscores the value of using Transformer-based models to build more intelligent, context-aware security systems for identifying insider threats.

Keywords **- Insider Threat Detection, Deep Learning, Transformer Model, User Behaviour Analytics, Sequence Modelling, Anomaly Detection.**

I. INTRODUCTION

The growing reliance on interconnected digital systems makes organizations vulnerable to insider threats, where individuals with legitimate access misuse their privileges. Detecting these threats is exceptionally challenging because malicious actions can be hard to distinguish from normal job functions. Traditional security systems, which rely on predefined rules, often fail to spot the subtle behavioral deviations that signal an impending threat, as they cannot grasp the user's intent.

To address this, data-driven methods using machine learning have become more common. Early advancements with sequence-aware models, such as LSTMs, improved detection by analyzing user activities over time. However, these models process information sequentially, which makes it difficult for them to recognize important connections between events that are far apart in a long activity log. The influence of early, potentially critical, actions can fade before a malicious act is identified.

This research proposes a new way of thinking: viewing user activity as a language. In this model, normal behavior follows a predictable "grammar," while malicious actions appear as unusual "phrases." To properly interpret this language, a system must be able to analyze the entire sequence of activities at once to understand the full context, rather than examining it piece by piece.

The Transformer architecture, which has transformed natural language processing, is ideal for this task. Its self-attention mechanism allows it to weigh the significance of every action relative to all others in the sequence, regardless of when they occurred. This holistic processing capability enables the model to build a deep, context-aware understanding of user behavior and identify the complex patterns associated with insider threats.

This research presents an integrated framework that leverages a Transformer-based approach for detecting insider threats. The system features a sophisticated feature engineering strategy that combines session-based data with individual user baselines and anomaly scores from an unsupervised learning model. This enriched data is then used to train the model on the public CERT r4.2 dataset, demonstrating its

effectiveness in learning from complex activity sequences.

Beyond standard metrics, the evaluation focuses on practical application. It identifies an optimal decision threshold to balance threat detection with the need to minimize false alarms. Furthermore, the framework includes an explainability feature that allows security analysts to see which specific activities contributed to a high-risk score, turning the model's output into actionable intelligence.

This paper is organized as follows: Section 2 discusses previous research on insider threat detection, while Section 3 presents the proposed framework and its constituent parts. Section 4 describes the experimental setup, including the dataset and evaluation metrics. Section 5 presents and discusses the results. Finally, Section 6 concludes the paper and outlines directions for future work.

III. RELATED WORK

Homoliak et al. [7] provided a foundational survey highlighting the evolving nature of insider threats and the need for adaptable detection models. Earlier methods mainly relied on rule-based systems and traditional machine learning, which struggled with generalization and detecting novel behaviors [10], [11]. Graph-based techniques, such as those by Hong et al. [3] and Fei et al. [12], modeled user-system relationships using GNNs and GCNs, while Li et al. [17] improved adaptability with a dual-domain GCN.

Deep learning approaches have demonstrated strong potential in capturing user behavior patterns. Pal et al. [7] proposed an LSTM-GRU model with attention to handle data imbalance and concept drift, whereas AlSlaiman et al. [2] incorporated sentiment analysis with LSTMs to reduce misclassification rates. Nasir et al. [5] applied deep learning on behavior-derived features, showing gains in precision and F1-score.

Innovative image-based methods, as explored by Gayathri et al. [14] and Li et al. [16], transformed activity logs into visual formats for CNN-based analysis. Techniques like sampling and ensemble learning have also improved detection in imbalanced datasets [4], [7].

Advanced methods, including few-shot learning [19], Bayesian networks [9], and attention mechanisms [18], have enhanced early-stage detection and probabilistic reasoning. While some studies have utilized attention models [7], [18], Transformer architectures remain underexplored in this field.

This study addresses that gap by leveraging a Transformer-based model to capture long-range behavioral dependencies and integrating anomaly scoring with domain-specific features. Multi-modal frameworks combining activity logs with contextual cues, as shown by Al-Mhiqani et al. [1] and Liu et al. [18], further support robust threat detection.

IV. THE PROPOSED FRAMEWORK

The proposed framework is a multi-stage, sequential pipeline designed to systematically transform raw, heterogeneous user activity logs into a structured format suitable for deep learning, ultimately yielding an actionable threat assessment. The entire process is architected to be end-to-end, beginning with raw data ingestion and concluding with final classification and an explainable report. The pipeline consists of four primary stages: (1) Data Ingestion and Preprocessing, (2) Ground Truth Labeling, (3) Hybrid Feature Engineering, and (4) Transformer-Based Threat Detection. Each stage builds upon the previous one, progressively refining the data to extract the most discriminative signals of malicious behavior. (A conceptual diagram of this pipeline is presented in Figure 1.)

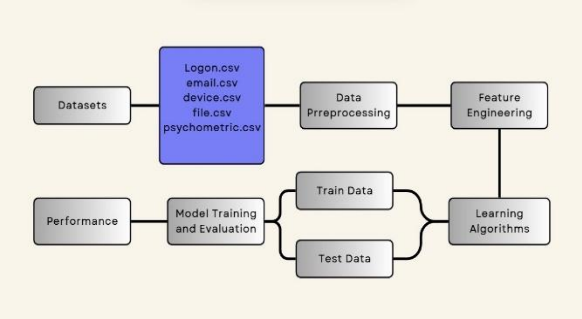

Fig 1: Work Flow

A. Data Ingestion and Preprocessing

The initial phase of the framework focuses on consolidating and cleaning data from diverse sources to form a single event timeline. The process begins by ingesting five types of data logs: logon activities, USB device connections, file access, emails, and user psychometric evaluations.

Several preprocessing steps are then applied. First, column names are standardized across all files for

consistency. Next, a unified "activity type" field is created to uniformly label every event, such as 'Logon' or 'File Access'. All individual logs are then merged into one master table, which is sorted chronologically by user to build a sequential narrative of each person's actions.

Finally, the data is cleaned by filling in any missing numerical values with zero and imputing absent psychometric scores with the median value for the entire group. This results in a comprehensive and clean event stream for each user, fully prepared for the subsequent stages of feature engineering and analysis.

B. Ground Truth Labeling

Since real-world data lacks clear labels for insider threats, a common heuristic was applied to generate them. The method assumes that employees who disappear from the company directory logs before the final month of data collection are potential threats.

This disappearance could signify an abrupt departure, such as a firing or resignation after committing a malicious act. Consequently, these users were programmatically labeled as threats, while all employees remaining at the end of the period were labeled as non-threats. Although this is an estimation, it provides a logical and repeatable method for creating the labeled dataset needed for supervised machine learning.

C. Hybrid Feature Engineering

This stage transforms the raw event stream into a high-dimensional feature space, providing the model with rich, discriminative signals beyond what raw logs alone can offer. A hybrid strategy was developed combining temporal analysis, user-specific baselining, and unsupervised machine learning.

Tokenization: Categorical data such as activity_type and pc_id were converted into unique integer tokens, enabling processing by the model's embedding layers.

Sessionization: The continuous stream of events for each user was segmented into discrete "sessions." A session was defined as a sequence of activities where the time between consecutive events did not exceed a 30-minute (1800-second) timeout. Session-based features were engineered, including session_id, events_in_session, and session_duration_seconds.

Behavioral Baselining: To account for individual user habits, features that establish personalized baselines were engineered. For example, the hour_deviation_from_user_norm feature calculates the deviation between the current event hour and that user's historical average activity hour. Large deviations indicate behavior that is anomalous for that individual.

Unsupervised Anomaly Detection: An unsupervised model was used to generate an explicit anomaly signal. An IsolationForest algorithm was trained on purely numerical, event-based features to detect rare and unusual patterns. An anomaly_score was calculated for each event, reflecting the probability of it being abnormal. This score was included as an additional input feature, enriching the dataset with a pre-analyzed anomaly indicator.

D. Transformer-Based Detection Model

At the core of the framework lies the Transformer model, tasked with classifying an entire sequence of a user's activities as either benign or threatening.

Input Representation: The final feature-engineered dataset, comprising a timeline of events for each user, was grouped by user_id. This created a batch of sequences, each representing a user's complete activity history. Sequences were padded with leading zeros or truncated to a fixed length (MAX_LEN = 512) to meet the input requirements of deep learning models.

Model Architecture:

Embedding Layers: Integer tokens for activity_type and pc_id were passed through separate Embedding layers, which learned dense vector representations that capture semantic relationships between activities.

Concatenation: Learned embeddings were concatenated with scaled numerical features to form a single, rich feature vector for each event.

Positional Encoding: As the Transformer processes events in parallel and lacks inherent sequence order, fixed sinusoidal positional encodings were added to each event vector to reintroduce temporal order.

Transformer Blocks: Data passed through two Transformer blocks, each containing Multi-Head Self-Attention and Feed-Forward layers. The self-attention mechanism allows the model to assess the relevance of each event in relation to others within the sequence,

effectively capturing long-term dependencies. Layer Normalization and Dropout were applied for stability and regularization.

Classification Head: A Global Average Pooling layer condensed the sequence output into a single representative vector. This vector was passed through Dense layers with a sigmoid activation to output a probability score between 0 and 1, representing the likelihood of the user's activity sequence indicating a threat.

V. EXPERIMENTAL SETUP

To rigorously evaluate the performance of the proposed Transformer-based framework, a set of experiments was designed using a public benchmark dataset. This section details the dataset, the specific metrics chosen to measure performance, and the full implementation and training configuration used to produce the results.

A. Dataset

This research utilized the CERT Insider Threat Dataset (r4.2), a widely used public benchmark from Carnegie Mellon University, chosen because it allows for comparison with previous studies.

The synthetic dataset simulates the activity of 1,000 users over 17 months, capturing data from sources like logon, device, and email logs. A key feature of this dataset is its significant class imbalance, which realistically mirrors the infrequent nature of true insider threats.

For the experiment, the dataset was split by user, with 80% (800 users) for training and 20% (200 users) for testing. This user-level separation is critical to prevent data leakage and ensure the model is evaluated on entirely new user profiles. The split was also stratified to maintain the same ratio of malicious to benign users in both the training and testing sets.

B. Evaluation Metrics

Given the severe class imbalance inherent in the dataset, relying on a single metric like accuracy can be highly misleading. A naive model could achieve high accuracy by simply predicting every user as "not a threat." Therefore, to provide a comprehensive and robust assessment of the model's performance, a suite of evaluation metrics was employed.

Area Under the ROC Curve (AUC): AUC was chosen as the main evaluation metric to assess the model's overall ability to differentiate between threat and non-threat categories across varying classification thresholds. An AUC score of 1.0 indicates flawless classification, whereas a score of 0.5 reflects performance equivalent to random guessing. Its threshold-independent nature makes it especially useful for evaluating models on imbalanced datasets.

To evaluate the model's practical performance, three key metrics were used, all calculated at an optimal decision threshold.

Recall indicates the percentage of actual threats that are accurately identified by the model. In cybersecurity, achieving high recall is essential to ensure no genuine threats go unnoticed.

Precision indicates the proportion of alerts generated by the model that are actually accurate. Maintaining high precision helps minimize false positives, easing the burden on security analysts.

The F1-Score, calculated as the harmonic mean of precision and recall, provides a balanced evaluation of the model's effectiveness—particularly useful when both missed detections and false alarms carry significant consequences.

C. Implementation Details

The entire framework was implemented in Python. For data manipulation and feature engineering, the Pandas library was used. The Isolation Forest model and the calculation of evaluation metrics were implemented using Scikit-learn. The core Transformer-based model was built and trained using the TensorFlow framework with the Keras API.

The specific hyperparameters for the Transformer model were as follows:

Embedding Dimension (EMBED_DIM): 64
Number of Attention Heads (NUM_HEADS): 4
Feed-Forward Network Dimension (FF_DIM): 64
Number of Transformer Blocks: 2
Dropout Rate (DROPOUT_RATE): 0.2
Sequence Length (MAX_LEN): 512 events

Training was conducted using the Adam optimizer with a learning rate of 0.001 and a batch size of 32. Binary cross-entropy, a commonly adopted loss

function for binary classification, was utilized during training.

To manage the training process and prevent the model from overfitting, several key strategies were employed.

First, an EarlyStopping mechanism was used to monitor the model's performance on a validation dataset. Training was stopped if there was no improvement in performance for ten consecutive epochs, and the version of the model with the best performance up to that point was preserved.

Additionally, the learning rate was dynamically adjusted using ReduceLROnPlateau. This technique automatically reduced the learning rate whenever performance stagnated for five cycles, allowing for more refined adjustments as the model approached an optimal solution.

Finally, to counteract the problem of class imbalance, where threat instances are rare, class weighting was applied during training. This method assigned a greater penalty for misclassifying the minority threat class, compelling the model to pay closer attention to detecting these critical events.

VI. RESULTS AND DISCUSSION

The following section discusses the observed results from the experiments conducted in the preceding part. The analysis begins with the model's training dynamics, followed by a detailed evaluation of its classification performance on the unseen test set, and concludes with a case study that demonstrates the framework's explainability component.

A. Model Training Performance

The training history of the Transformer-based model provides insight into its learning process. The variations in loss and AUC across training epochs are presented in Figure 2.

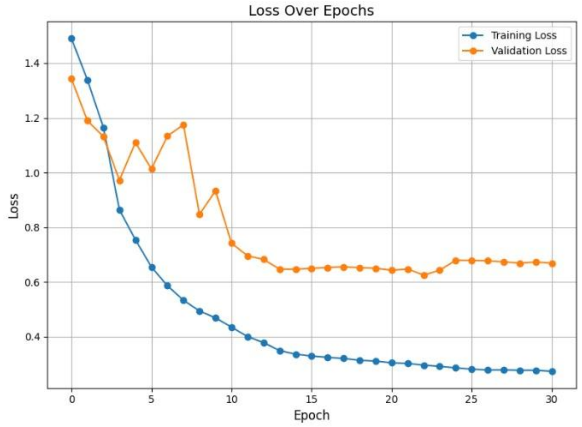

Fig 2: loss over epochs

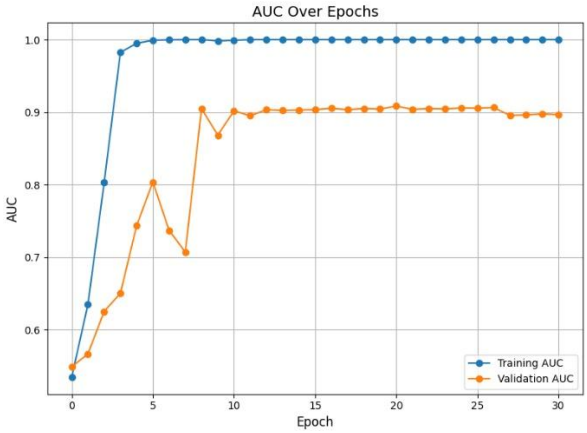

Fig 3: AUC over epochs

The model's training progress demonstrated effective learning. The training performance (Training AUC) rapidly approached a perfect score, showing the model's capacity to fit the data. At the same time, its performance on unseen data (Validation AUC) also rose sharply, indicating it was learning generalizable patterns.

The model achieved its best performance on the validation data at epoch 4. An automated "EarlyStopping" mechanism was used to identify this peak and halt the training process, restoring the model to this optimal state. This intervention was crucial for preventing overfitting, which was signaled by the divergence of training and validation scores, thereby preserving the model's most effective and generalizable version.

B. Classification Performance

The framework's final performance was assessed using a reserved test set of 200 users. Because a default 0.5 decision threshold is often unsuitable for imbalanced

classification, an optimal threshold was determined first.

Using Youden's J statistic to find the ideal point on the ROC curve, an optimal threshold of 0.11 was established. This low value is common for imbalanced problems and prioritizes the detection of potential threats, even those with lower probability scores. The ROC curve, shown in Figure 3, highlights this optimal point.

The model attained an impressive AUC score of 0.794 on the test dataset. This result confirms that the system can effectively separate malicious and benign users, validating that the engineered features and the patterns learned by the Transformer successfully indicate threatening behaviour.

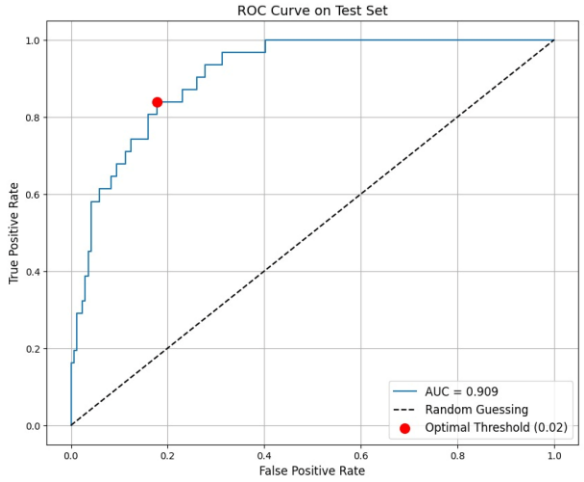

Fig 4: ROC Curve on Test Set

The detailed classification metrics, calculated using the optimal threshold of 0.11, are presented in figure 5.

```
--- [2] Classification Report (Using Optimal Threshold: 0.11) ---
              precision    recall  f1-score   support

Not Threat (0)       0.92      0.96      0.94       169
    Threat (1)       0.71      0.55      0.62        31

      accuracy                           0.90       200
     macro avg       0.81      0.75      0.78       200
  weighted avg       0.89      0.90      0.89       200
```

Fig 5: Classification Report

The results in figure 5 highlight the practical utility of the framework. The model achieved a threat recall of 55%, meaning it successfully identified more than half of all the true insider threats present in the test data. In a security context where the cost of a missed threat is exceptionally high, this is a very strong outcome. Furthermore, this was balanced with a threat precision of 71%. This metric is equally important from an operational standpoint, as it means that when the system does generate an alert, it is correct 71% of the time. This high precision minimizes analyst fatigue by significantly reducing the number of false alarms that need to be investigated. The overall system accuracy reached 90%, which, supported by the strong AUC, precision, and recall, confirms the model's effectiveness.

VII. CONCLUSION AND FUTURE WORK

This paper presents a new deep learning framework that uses a Transformer architecture to successfully identify insider threats. By integrating data from multiple sources and employing a sophisticated feature engineering strategy, the system has proven effective at distinguishing between malicious and benign user behavior in complex activity logs. When evaluated on the CERT r4.2 dataset, the framework demonstrated a strong practical balance, achieving 71% precision in its alerts while successfully detecting 55% of threats. These results validate the use of Transformer networks in this security field.

Future work can build on this foundation in several key directions. One path is to enrich the model by incorporating more diverse data sources, such as web browsing history or command-line activity, to create more detailed user profiles. Another is to explore more advanced model architectures, like the Longformer, which could improve performance on datasets with longer user histories.

A significant advancement would be to adapt the system for real-time deployment, enabling continuous monitoring rather than periodic analysis of historical data. Finally, a more ambitious goal is to move beyond simply identifying anomalies and toward modeling causality to infer a user's intent. Pursuing these avenues can lead to the development of more intelligent, accurate, and context-aware security systems.

REFERENCES

[1]. Al-Mhiqani, M. N., Ahmad, R., Abidin, Z. Z., Abdulkareem, K. H., Mohammed, M. A., Gupta, D., & Shankar, K. (2022). A new intelligent multilayer framework for insider threat detection. Computers and Electrical Engineering, 97, 107597. https://doi.org/10.1016/j.compeleceng.2021.107597

[2]. AlSlaiman, M., Salman, M. I., Saleh, M. M., & Wang, B. (2023). Enhancing false negative and positive rates for efficient insider threat detection. Computers & Security, 126, 103066. https://doi.org/10.1016/j.cose.2022.103066

[3]. Hong, W., Yin, J., You, M., Wang, H., Cao, J., Li, J., & Liu, M. (2023). A graph empowered insider threat detection framework based on daily activities. ISA Transactions, 141, 84–92. https://doi.org/10.1016/j.isatra.2023.06.030

[4]. Manoharan, P., Yin, J., Wang, H., Zhang, Y., & Ye, W. (2023). Insider threat detection using supervised machine learning algorithms. Telecommunication Systems. https://doi.org/10.1007/s11235-023-01085-3

[5]. Nasir, R., Afzal, M., Latif, R., & Iqbal, W. (2021). Behavioral based insider threat detection using deep learning. IEEE Access, 9, 143266–143281. https://doi.org/10.1109/ACCESS.2021.3118297

[6]. Ogunbodede, O. O., Adewale, O. S., Alese, B. K., Akinyokun, O. K., & Madamidola, O. A. (2025). Insider threat research: A review of deep learning approach. Journal of Security in Computer Networks and Distributed Systems, 2(1), 28–41.

[7]. Pal, P., Chattopadhyay, P., & Swarnkar, M. (2023). Temporal feature aggregation with attention for insider threat detection from activity logs. Expert Systems with Applications, 224, 119925. https://doi.org/10.1016/j.eswa.2023.119925

[8]. Rauf, U., Mohsen, F., & Wei, Z. (2023). A taxonomic classification of insider threats: Existing techniques, future directions & recommendations. Journal of Cyber Security and Mobility, 12(2), 221–252. https://doi.org/10.13052/jcsm2245-1439.1225

[9]. Wall, A., & Agrafiotis, I. (2021). A Bayesian approach to insider threat detection. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 12(2), 48–84.

[10]. Yuan, S., & Wu, X. (2021). Deep learning for insider threat detection: Review, challenges and opportunities. Computers & Security, 104, 102221. https://doi.org/10.1016/j.cose.2021.102221

[11]. Adun, I. J., & Amadin, F. I. (2023). A hybrid supervised machine learning model for the prediction of insider threats. Journal of Science and Technology Research, 5(3), 169–179. https://doi.org/10.5281/zenodo.8313125

[12]. Fei, K., Zhou, J., Su, L., Wang, W., Chen, Y., & Zhang, F. (2022). A graph convolution neural network-based method for insider threat detection. In Proceedings of the 2022 IEEE International Conference on Parallel and Distributed Processing with Applications (ISPA). https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom57177.2022.00016

[13]. Gamachchi, A., Sun, L., & Boztaş, S. (2017). A graph-based framework for malicious insider threat detection. In Proceedings of the 50th Hawaii International Conference on System Sciences (HICSS). https://arxiv.org/abs/1809.00141

[14]. Gayathri, R. G., Sajjanhar, A., & Xiang, Y. (2019). Image-based feature representation for insider threat classification. arXiv preprint, arXiv:1911.05879. https://arxiv.org/abs/1911.05879

[15]. Homoliak, I., Toffalini, F., Guarnizo, J., Elovici, Y., & Ochoa, M. (2019). Insight into insiders and IT: A survey of insider threat taxonomies, analysis, modeling, and countermeasures. ACM Computing Surveys, 52(2), Article 30. https://doi.org/10.1145/3303771\

[16]. Li, D., Yang, L., Zhang, H., Wang, X., Ma, L., & Xiao, J. (2021). Image-based insider threat detection via geometric transformation. Security and Communication Networks, 2021, Article ID 1777536. https://doi.org/10.1155/2021/1777536

[17]. Li, X., Li, X., Jia, J., Li, L., Yuan, J., Gao, Y., & Yu, S. (2023). A high accuracy and adaptive anomaly detection model with dual-domain graph convolutional network for insider threat detection. IEEE Transactions on Information Forensics and Security, 18, 1638–1653. https://doi.org/10.1109/TIFS.2023.3245413

[18]. Liu, Y., Lu, H.-P., & Lai, C.-H. (2022). A novel attention-based multi-modal modeling technique on mixed type data for improving TFT-LCD repair process. IEEE Access, 10, 33026–33037. https://doi.org/10.1109/ACCESS.2022.3158952

[19]. Yuan, S., Zheng, P., Wu, X., & Tong, H. (2020). Few-shot insider threat detection. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20) (pp. 2829–2832). https://doi.org/10.1145/3340531.3412161