USER-BASED SEQUENTIAL MODELING WITH TRANSFORMER ENCODERS FOR INSIDER THREAT DETECTION

Mohamed Elbasheer
George Washington University
Washington D.C, USA
elbasheer@gwu.edu

Adewale Akinfaderin
George Washington University
Washington D.C, USA
waleakinfaderin@gwu.edu

July 11, 2025

ABSTRACT

Insider threat detection presents unique challenges due to the authorized status of malicious actors and the subtlety of anomalous behaviors. Existing machine learning methods often treat user activity as isolated events, thereby failing to leverage sequential dependencies in user behavior. In this study, we propose a User-Based Sequencing (UBS) methodology, transforming the CERT insider threat dataset into structured temporal sequences suitable for deep sequential modeling. We deploy a Transformer Encoder architecture to model benign user activity and employ its reconstruction errors as anomaly scores. These scores are subsequently evaluated using three unsupervised outlier detection algorithms: One-Class SVM (OCSVM), Local Outlier Factor (LOF), and Isolation Forest (iForest). Across four rigorously designed test sets, including combinations of multiple CERT dataset releases, our UBS-Transformer pipeline consistently achieves state-of-the-art performance—notably 96.61% accuracy, 99.43% recall, 96.38% F1-score, 95.00% AUROC, and exceptionally low false negative (0.0057) and false positive (0.0571) rates. Comparative analyses demonstrate that our approach substantially outperforms tabular and conventional autoencoder baselines, underscoring the efficacy of sequential user modeling and advanced anomaly detection in the insider threat domain.

1 Introduction

Insider threats pose a complex and challenging problem for organizations across government, public, and private sectors. The lack of sufficient data has made detecting such threats extremely difficult. However, over the last two decades, researchers have made significant progress in developing Machine Learning (ML) solutions that can effectively detect, prevent, and mitigate insider attacks. A 2023 report by the Ponemon Institute entitled "The Cost of Insider Risk" found that 64% of organizations worldwide consider ML to be an essential tool in addressing insider incidents [1]. This highlights the industry's confidence in the power of Machine Learning to tackle this pressing problem.

2 Insider Threats

The Cybersecurity and Infrastructure Security Agency (CISA) defines an insider as "*any person who has or had authorized access to or knowledge of an organization's resources, including personnel, facilities, information, equipment, networks, and systems*" [2].

CISA characterizes an 'insider threat' as the potential for such actors to utilize their access or understanding of an organization to cause harm. This harm could manifest through various means, including malicious, negligent, or unintentional actions that adversely affect the organization's Confidentiality, Integrity, and Availability (CIA) [2].

Furthermore, CISA outlines several manifestations of insider threats, ranging from espionage and terrorism to unauthorized information disclosure, participation in transnational organized crime, sabotage, workplace violence, and the deliberate or accidental compromise of organizational resources or capabilities.

Moreover, CISA categorizes insider threats into several types, including unintentional threats stemming from negligence or accidents and intentional threats, often called "malicious insiders." Among them, collusive threats involve insiders collaborating with external actors to harm the organization. In contrast, third-party insider threats arise from contractors or vendors who, though not formal organization members, have privileged access to essential assets and may pose direct or indirect threats [3].

Unlike external adversaries who encounter various barriers to entry, insiders inherently have legitimate access, placing them in a unique position to exploit the trust placed in them by the organization [4]. Their intimate understanding of internal systems equips them to inflict significant, often undetected damage [5].

An examination of the nature of insider threats reveals a range of motivations. At one end are the malicious insiders, individuals motivated by personal gain, financial incentives, or ideological beliefs [6]. Those actors misuse their privileged access for data theft, operational disruption, or espionage [5]. Their actions, concealed by their insider knowledge, often evade standard security measures, posing significant detection and prevention challenges [7].

Equally concerning are the unwitting insiders, often characterized by unintentional misconduct [4]. Through negligence or lack of awareness, those individuals inadvertently aid malicious actors. For instance, they might click on phishing links or unknowingly share sensitive information, thus creating entry points for external attackers to compromise the organization's systems [3].

3 Methodology

The transformer, developed by Vaswani et al. and published in the seminal paper "Attention Is All You Need" in 2017, is a groundbreaking development in the field of Natural Language Processing (NLP) and has gained widespread attention due to its exceptional performance in various language-related tasks [8].

The transformer offers a novel and effective alternative to the Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM), which were the predominant architecture used in many Insider Threat Detection (ITD) models. For instance, [9] used Convolution Neural Network (CNN) to analyze an image representation of audit logs, while [10] transformed similar data into multivariate time series and applied RNN. Another study by [11] employed LSTM to improve the insider detection rate using the CERT dataset. In addition, [12] utilized an LSTM auto-encoder to uncover hidden patterns from sequential user activities, then combined a Graph Neural Network (GNN) and CNN with an organizational graph to detect malicious insiders. Moreover, [13] proposed an LSTM-based anomaly detection system to address the insider threats data imbalance problem. Furthermore, [14] suggested an ensemble approach using stacked-LSTM and Gated Recurrent Unit (GRU)-based attention models trained on single-day sequential activity logs to detect insider threats.

Unlike LSTMs and RNNs, which recurrently process the input sequences, the transformer employs a self-attention mechanism that simultaneously evaluates the entire input sequence [15]. RNN and LSTM process the sequence one element at a time and rely on their hidden state to carry information from previously seen elements to the next steps [16]. This is fine for small variable-length sequences but can lead to issues like vanishing and exploding gradients when the sequence is long [17]. LSTMs are advanced version of RNNs designed to mitigate some of these issues by introducing gates that regulate the flow of information. However, despite their enhanced capability in managing long sequences compared to RNNs, LSTMs are still constrained by their inherent sequential processing nature which prevents them from fully leveraging the modern parallel computing architectures.

3.1 Anomaly-based detection

Our research employed an insider threat anomaly-based detection approach using a novelty strategy. In this context, "Novelty" means the training dataset is devoid of any anomalies, enabling the model to learn and establish a baseline of "normal" user behavior. Our model predictive pipeline is depicted in Figure 1.

3.2 The environment

The specification of the system used for code development and data processing in this research is a dual-boot system running Windows 10 and Ubuntu 20.04 LTS. The hardware is a Dell Precision 1750 workstation with 128GB RAM, 6TB SSD storage, and an 8GB GPU. The software is Python 3.9 and PyTorch framework version 2.0.1+cu118 [18].

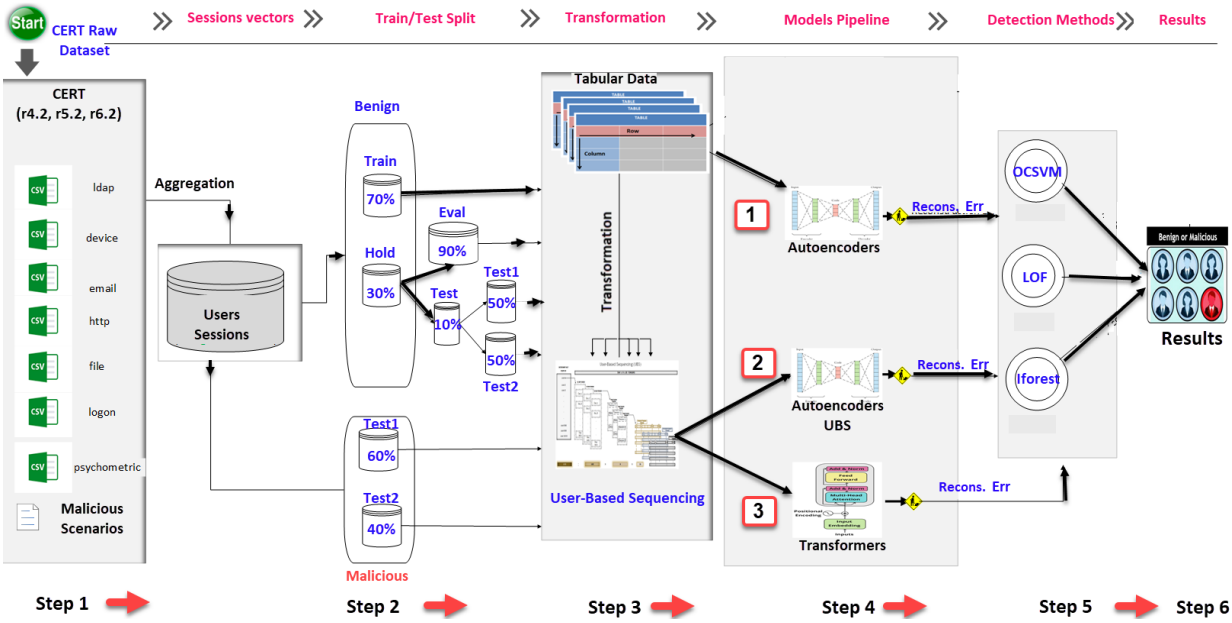

Figure 1: Model Pipeline

3.3 Data collection. preprocessing, and aggregation

The Computer Emergency Response Team (CERT) dataset, created by Carnegie Mellon University, is one of the most widely used datasets in insider threat research. It is a synthetic dataset, and publicly accessible [19]. The dataset mimics the log behaviors of a virtual organization and includes activity logs of 1000 insiders tracked over 17 months from January 2010 to May 2011. The dataset is available in several releases, created at different times, and includes multiple versions (e.g., r4.2, r5.2, r6.2, etc). Version r4.2 is classified as a "dense" dataset featuring numerous insider and malicious activities.

In our research, we used version r4.2 to train our model. CERT r4.2 has one of the largest malicious users and activities, 70 and 7323, respectively, and it is one of the widely used versions in research [20]. However, we are not restricted to using r4.2. Our model is designed to be used with any dataset with a structure similar to the CERT dataset. The only requirement is to transform the dataset using our novel User-Based Sequencing (UBS) structure.

3.4 Research pipeline

Considering the dataset consists of data in raw log format, we first decided to choose between three levels of granularity, session, daily, or weekly, to extract the correct features and build the appropriate data representation. We elect to use a session-based granularity and construct fixed-size vectors consisting of 35 encoded categorical and numerical features. Each numerical feature tracks the count of how many times a specific event was performed. We believe that a session-based granularity approach enables capturing subtle changes in user behavior that may go unnoticed when the data is aggregated over an extended period.

Secondly, we needed an approach to transform the extracted tabular data into a sequential format since this is the requirement for using the Transformer by design [21]. Therefore, we built a novel User-Based Sequencing technique to convert the extracted feature vector into a nested Python dictionary structure. A Python dictionary is a collection that associates keys with corresponding values. In our case, the keys represent employee IDs, while the values consist of three tuples. The first tuple denotes the day of the week corresponding to the dataset tracked days (1-501). In contrast, the second tuple is designed to accommodate the maximum number of sessions per day (9 in our experiment), and the third tuple tracks the 35 captured features we extracted. With this multi-index structure, we aim to tokenize all user activities sequentially and enable the Transformer to efficiently track the long-term dependency between days, sessions, and features.

Thirdly, we use our novel UBS structure to train the Transformer model on only benign data and evaluate its performance based on its ability to reconstruct its input, with any deviations from the expected output flagged as potential anomalies [22]. Finally, instead of using conventional statistical methods like mean, standard deviation, and Median Absolute

Deviation (MAD), which can be skewed by data distribution and outliers. We elect to use unsupervised machine learning algorithms — Local Outlier Factor (LOF), One-Class Support Vector Machine (OCSVM), and Isolation Forest (iForest)—to evaluate the reconstruction errors. This approach provides a more precise and accurate method for identifying anomalies by leveraging the strengths of these algorithms to overcome the limitations of statistical techniques. Our proposed model pipeline is summarized in Figure 2.

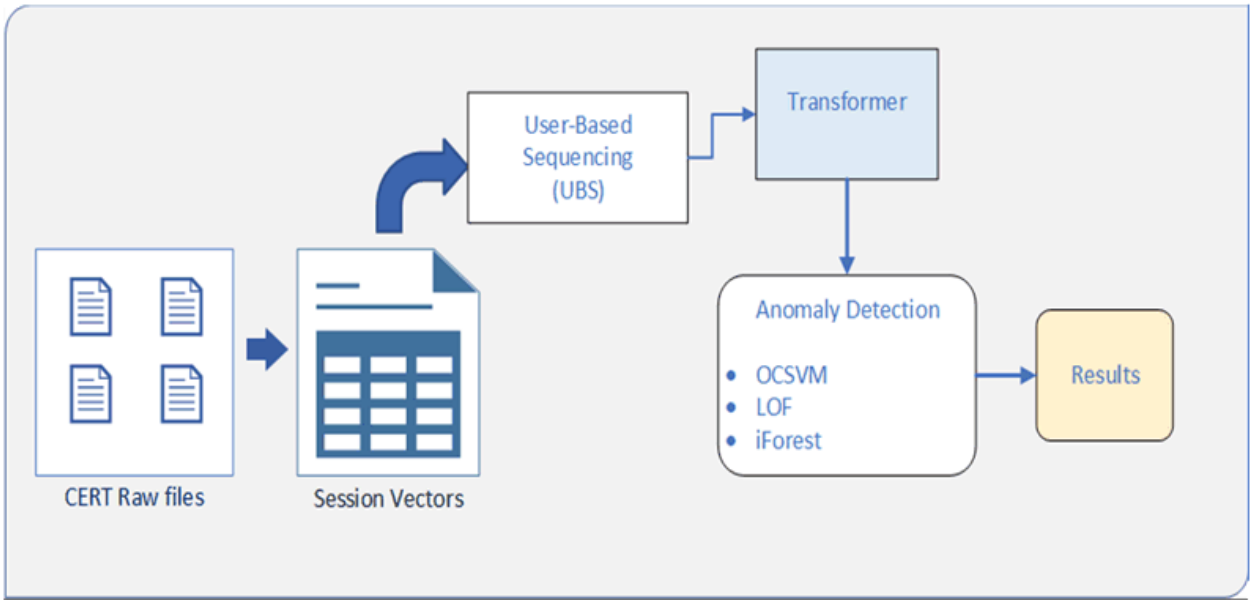

Figure 2: Methodology

3.5 User-Based Sequencing

User-Based Sequencing (UBS) is our novel technique to organize data based on the sequence of actions by users, transforming tabular data into a sequential format suitable for processing by our proposed model. For the CERT r4.2, UBS is built as a multi-index Python Dataframe and then converted into a tensor using a dictionary structure identified by the set of users U (1000), number of days D (501), Number of sessions per day S (9), and number of extracted features F (35), such that: For a single user u, the tensor that represents the [501,9,35] dimensions can be denoted as T_u. Hence, $T_u[i][j][k]$ represents the kth feature of the jth session on the ith day for user u. The entire user data dictionary can then be expressed as: user_data: U -> T. Where T is the tensor defining all possible tensors of shape [501,9,35]. It is important to note that S is set based on the maximum number of sessions extracted from the CERT dataset. To allow an additional buffer for future growth; we set S to 9. However, this can easily be increased if the number of sessions per user grows beyond that. Figure 3 depicts our UBS structure.

3.6 Experiment and model setup

Our research primarily uses the "Encoder" part of the vanilla transformer architecture. This is because we only aim to train the Transformer on what is "normal" and calculate the reconstruction errors from subsequent input. For our encoder architecture, we added an initial linear layer as an embedding layer to correctly project our User-Based Sequencing input dimensions. In addition, since the self-attention mechanism in the Transformer is inherently order-agnostic, we added positional encoding to our input embedding to provide the model with information about the order of the features in our UBS structure. Furthermore, we added a dropout layer after the Positional Encoding to serve as a regularization technique.

Furthermore, to ensure our model training is perfectly in sync with our UBS structure, we use a batch size of "1", referred to as User-Based Batching (UBB). This allows the model to learn from the entire user space at once. We used Mean Squared Error (MSELoss) as our Loss function and Adaptive Moment Estimation (ADAM) as our optimizer because it efficiently adjusts learning rates and uses fewer resources to converge [23]. In addition, We fine-tuned the transformer internals, such as the number of layers, nodes, dropout, and learning rate, by testing various combinations of these hyperparameters using a Cartesian product to ensure no repetition in these combinations [24]. Ultimately,

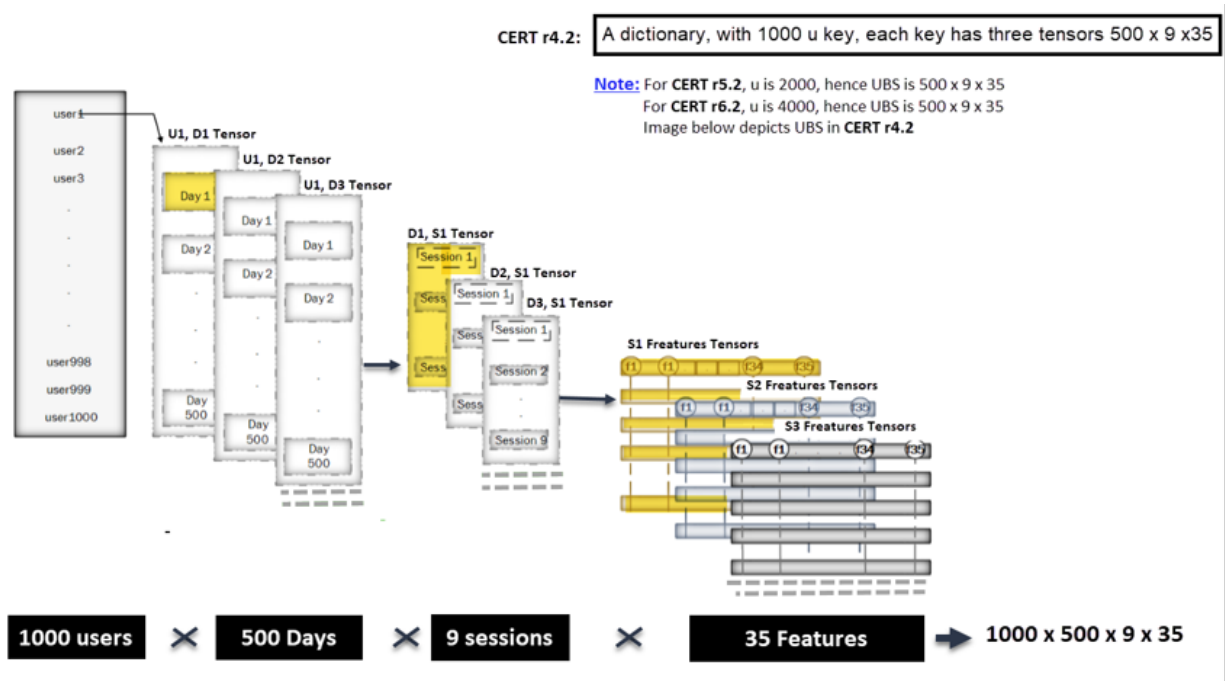

Figure 3: User-Based Sequencing pictorial representation

the best set of hyperparameters we experimented with yielded a model with 5,387,579 trainable parameters using six encoder blocks, eight multi-head attention, 0.1 dropouts, and a 0.00001 learning rate.

3.6.1 Detection Algorithms

The primary objective of anomaly detection is to discern instances that markedly diverge from established patterns of normal behavior. To this end, our models are exclusively trained on benign data to produce what is known as 'reconstruction errors.' These errors effectively gauge what the model has learned. Consequently, any significant differences in the reconstruction errors between training and testing—exceeding a predefined threshold—can be used to flag the record as anomalous [25]. Statistically, this is measured by how large the difference is from a predetermined threshold, and the threshold is commonly determined through methods like the mean, standard deviation, and MAD or using cross-validation or domain-specific knowledge. However, these manual methods are prone to outliers and may be heavily influenced by the data distribution. Inspired by the work of [26], [27], and [28], we decided to use LOF, OCSVM, and iForest to detect outliers in our reconstruction errors.

3.6.2 Testing setup

Our model testing comprises three combinations: First, we evaluate the Transformer against the Autoencoder (using Tabular data), referred to as AutoTAB. Second, we evaluate the Transformer against the Autoencoder (using UBS data), referred to as AutoUBS. Finally, we compared AutoTAB versus AutoUBS. For the experiment, we designed four testing sets. Tests-1 through Test-3 are designed to evaluate the individual CERT datasets (CERT r4.2. r5.2, r6.2). However, this introduced some limitations; for instance, in Test-3, there were only five malicious users compared to 120 benign ones (CERT r5.2), which can skew. To address this problem and ensure a more balanced and representative evaluation, we designed Test-4, which amalgamates users across all CERT dataset versions, culminating in a robust test set consisting of 210 benign and 174 malicious users.

3.6.3 Test Sets

As evident from the large number of benign versus malicious users in the CERT dataset—r4.2: 1000 versus 70, r5.2: 1901 versus 99, and r6.2: 3995 versus five users, we needed a way to construct test sets that are regardless of the CERT dataset version used. Table 1 summarizes the test we built to address this problem.

Table 1: *Model Test-Sets Summary*

Testset	Version	Benign Users	Malicious Users	Total Users
1	r4.2	30	70	**100**
2	r5.2	60	99	**159**
3	r6.2	120	5	**125**
4	r4.2, r5.2, r6.2	210	174	**384**

4 Results

During testing, our UBS data transformation has proven to be highly effective in improving the models' ability to learn from the embedded temporal and sequential patterns, regardless of the specific model employed. As demonstrated in tables below, we clearly see the differences in performance between the Transformer, Autoencoder-Tabular, and Autoencoder-UBS models. The results from all 4 test sets are shown in Tables 2 and 3

Table 2: *Summary of Results from Test-1, 2, and 3*

Model	Detection Method	A	P	R	F1	AUC	FNR
Test-1 (CERT r4.2)							
Transformer	OCSVM	**0.9900**	0.9859	**1.0000**	0.9929	1.0000	**0.0000**
	LOF	0.9900	0.9859	1.0000	0.9929	1.0000	0.0000
	iFOREST	0.9900	0.9859	1.0000	0.9929	1.0000	0.0000
Auto-TAB	OCSVM	0.6900	0.5857	0.5857	0.7257	0.8100	0.4143
	LOF	0.5900	0.8085	0.5429	0.6496	0.6900	0.4571
	iFOREST	0.7900	0.9623	0.7286	0.8293	0.9300	0.2714
Auto-UBS	OCSVM	0.9700	1.0000	0.9571	0.9781	0.9981	0.0429
	LOF	0.9100	1.0000	0.8714	0.9313	1.0000	0.1286
	iFOREST	0.9300	1.0000	0.9000	0.9474	1.0000	0.1000
Test-2 (CERT r5.2)							
Transformer	OCSVM	0.9308	0.9000	**1.0000**	0.9474	0.9000	**0.0000**
	LOF	0.9245	0.8919	1.0000	0.9429	0.8800	0.0000
	iFOREST	0.9245	0.8919	1.0000	0.9429	0.9000	0.0000
Auto-TAB	OCSVM	0.6101	0.9111	0.4141	0.5694	0.7600	0.5859
	LOF	0.5597	0.8085	0.3838	0.5205	0.6200	0.6162
	iFOREST	0.6918	0.8906	0.5758	0.6994	0.8400	0.4422
Auto-UBS	OCSVM	0.9434	0.9245	0.9899	0.9561	0.9806	0.0101
	LOF	0.9245	0.9143	0.9697	0.9412	0.9557	0.0303
	iFOREST	0.9308	0.9151	0.9798	0.9463	0.9286	0.0202
Test-3 (CERT r6.2)							
Transformer	OCSVM	**0.9440**	0.4167	**1.0000**	0.5882	0.9800	**0.0000**
	LOF	0.9440	0.4167	1.0000	0.5882	0.9700	0.0000
	iFOREST	0.9200	0.3333	1.0000	0.5000	0.9700	0.0000
Auto-TAB	OCSVM	0.9200	0.2727	0.6000	0.3750	0.8400	0.4000
	LOF	0.8720	0.0769	0.2000	0.1111	0.7600	0.8000
	iFOREST	0.8320	0.1667	0.8000	0.2759	0.9100	0.2000
Auto-UBS	OCSVM	0.9200	0.3333	1.0000	0.5000	1.0000	0.0000
	LOF	0.9360	0.3846	1.0000	0.5556	0.9967	0.0000
	iFOREST	0.9360	0.3846	1.0000	0.5556	0.9917	0.0000

In our approach, we utilize the reconstruction errors generated by two primary models, the Transformer and the Autoencoder. These models are designed to process input data and reconstruct it, with the reconstruction error serving as an indicator of how well the model has performed. A high reconstruction error typically suggests that the model has encountered an anomaly—something that deviates significantly from the normal patterns it has learned.

To further analyze these reconstruction errors, we feed them into three different anomaly detection algorithms: One-Class Support Vector Machine (OCSVM), Local Outlier Factor (LOF), and Isolation Forest (IFOREST). Each of these algorithms is specialized in identifying anomalies within a dataset, but they do so using different methodologies.

OCSVM: This algorithm creates a decision boundary around the normal data points, treating anything outside this boundary as an anomaly.

Table 3: *Summary of All Results from Test-4*

Model	Method	A	P	R	F1	AUC	FPR
Test-4 (All CERT datasets combined)							
Transformer	OCSVM	0.9505	0.9016	**1.0000**	0.9482	0.9600	**0.0905**
	LOF	0.9141	0.9325	0.8736	0.9021	0.9500	0.0524
	iFOREST	**0.9661**	0.9351	**0.9943**	0.9638	0.9500	**0.0571**
Auto-TAB	OCSVM	0.7318	0.8586	0.4885	0.6227	0.7900	0.0667
	LOF	0.6953	0.7822	0.4540	0.5745	0.6700	0.1048
	iFOREST	0.7552	0.7222	0.7471	0.7345	0.8600	0.2381
Auto-UBS	OCSVM	0.9427	0.9143	0.9770	0.9392	0.9900	0.0857
	LOF	0.9609	0.9818	0.9310	0.9558	1.0000	0.0143
	iFOREST	0.9688	0.9821	0.9483	0.9649	1.0000	0.0143

LOF: LOF detects anomalies by comparing the local density of a data point with that of its neighbors. Points that have significantly lower density than their neighbors are considered outliers.

IFOREST: Isolation Forest isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature.

Given that these algorithms are applied to the reconstruction errors—already processed by the Transformer and Autoencoder—it is possible that all three detection methods yield the same result. This can occur because the algorithms are evaluating the same underlying distribution of reconstruction errors, which have already been analyzed by the primary models.

For instance, in test1, all three algorithms might classify the same data points as anomalies. This agreement can happen because the reconstruction errors provided by the Transformer and Autoencoder models already highlight the anomalies effectively, making the subsequent detection task straightforward for the anomaly detection algorithms.

However, while these algorithms may agree in some cases, their individual methodologies and sensitivities can lead to different results in other scenarios, especially when the reconstruction errors contain more nuanced patterns. This is why using multiple detection algorithms can provide a more robust anomaly detection process, even if they occasionally produce the same results.

5 Analysis & Discussion

One of the primary goals we set in this research is to compare the effectiveness of the Transformer using our novel UBS to our baseline model—Autoencoder—which uses unstructured tabular data. To achieve this goal, we built four test sets (1 through 4) as documented in Table 1 and developed three models (Transformer using UBS, Autoencoder using Tabular, and Autoencoder using UBS)

We also built a fourth baseline model using OCSVM, LOF, and IFORST. However, we quickly realized that those models are ineffective as the data structure (UBS or tabulated) is not designed to accommodate such models. Therefore, we elect not to document their performance in this study. Furthermore, we also thought of running the Transformer on tabular data. However, again, we realized that the Transformer architecture is not designed to work well in non-sequential data. Hence, we abandon that idea early in our research.

Every test we have designed is based on a stratified splitting approach to ensure complete data separation and that the data in each set are unique; no overlap is present in any of the sets. Among all the test sets we built, Test-4 is the most comprehensive, combining user data from all CERT dataset versions.

As mentioned in the previous sections, we used several metrics, including accuracy, precision, F1, and recall, to measure the performance of the three anomaly detection algorithms employed in this study. However, given the severity of insider threats and the unique anomaly approach we designed, we decided that reducing the false negatives and maximizing recall should be prioritized over other metrics.

The results obtained by stacking the Transformer against both versions of the autoencoders are summarized in Tables 2 and 3. As previously demonstrated, the Transformer exhibited exceptional performance, outperforming all baseline models and comparable models reviewed in Chapter 2. Notably, our model achieved a recall of 99.43%, surpassing the DistilledTrans introduced by [20], which achieved a recall of 84.62%. Additionally, our Transformer model

outperformed the TRANLOG introduced by [29], which utilizes a similar anomaly approach and achieved only a 0.996% recall.

As demonstrated in Table 3, using Test-4 as the defacto test set, Transformer (OCSVM) stands out for its exceptional performance in recall and FNR, achieving a perfect score of 100% and 0.0%, respectively. These metrics are particularly relevant in insider threat scenarios where missing a true positive can have serious consequences. Despite the iFOREST's high accuracy and F1-score, the flawless recall and zero FNR of OCSVM make it the most effective algorithm.

- **Transformer (iFOREST) vs. Auto-TAB:**
  The Transformer OCSVM exhibits exceptional performance when compared to Auto-TAB. The Transformer model has delivered a remarkable 33.85% improvement in recall over the best-performing Auto-TAB model (iFOREST). This notable enhancement underscores the Transformer OCSVM's capability to identify all positive instances without any misses, as evidenced by its perfect recall rate and 0.0% FNR. Table 4 provides a clear breakdown of the percentage improvement of the Transformer over the Auto-TAB.

Table 4: *Improvement of Transformer over AutoTAB*

Metrics	% Point Change	Analysis
Accuracy	+27.92%	Substantial increase in overall model performance.
Precision	+29.48%	Notable improvement in true positive identifications.
Recall	+33.10%	Significant enhancement in identifying all actual positives.
F1 Score	+31.22%	Considerable betterment in the balance between precision and recall.
AUROC	+10.47%	Improved capability in distinguishing between the classes.
FPR	+ 76.01%	Reduction in the rate at which true positives are mistakenly overlooked.

- **Transformer (iFOREST) vs. Auto-UBS:**
  Despite the impressive improvements and high scores shown by the Auto-UBS model, particularly in AUROC and Precision, the flawless Recall and FNR demonstrated by the Transformer OCSVM are critical for applications where missing an anomaly could have dire consequences. The Transformer model with iFOREST boasts a 4.85% enhancement in recall over the top-performing Auto-UBS model (iFOREST). This improvement underscores the effectiveness of the Transformer iFOREST in achieving a perfect recall rate, surpassing the already impressive recall performance of Auto-UBS. Table 5 provides an overview of the percentage improvement of the Transformer compared to Auto-UBS.

Table 5: *Improvement of Transformer over AutoUBS*

Metrics	% Point Change	Analysis
Accuracy	-0.28%	A slight enhancement in the overall model performance by the Transformer over Auto-UBS.
Precision	-4.79%	A minor reduction in the proportion of true positive identifications over all positive predictions by the Transformer compared to Auto-UBS.
Recall	+4.85%	Superior ability of the Transformer to identify all actual positives.
F1 Score	-0.11%	A slightly better balance between precision and recall by the Transformer.
AUROC	-5.00%	A small decline in the Transformer's ability to distinguish between the classes compared to Auto-UBS.
FPR	4.27	(absolute) increase in the rate at which true positives are mistakenly overlooked by the Transformer compared to Auto-UBS.

The impressive performance of both the Transformer and Auto-UBS underscores the critical role our user-based sequencing plays in enabling both models to achieve better results by transforming the data so that each model can understand the data and extract meaningful patterns more effectively. The Transformer, with its ability to handle sequential data and capture long-range dependencies, and the Autoencoder, known for its feature extraction and reconstruction capabilities, both benefit immensely from the user-based sequencing approach.

- **AutoUBS vs. AutoTAB:**
  The Auto-UBS (iFOREST) shows notable improvements over the Auto-TAB (iFOREST) model across all metrics, affirming the benefit of our user-based sequencing in improving the Autoencoder detection capabilities. Table 6 shows the percentage of improvement of the Auto-UBS over the Auto-TAB.

Table 6: *Improvement of AutoUBS over AutoTAB*

Metrics	% Point Change	Analysis
Accuracy	+28.28%	A significant enhancement in overall model performance.
Precision	+35.97%	A better proportion of true positive identifications over all positive predictions.
Recall	+26.94%	A superior ability to identify all actual positives.
F1 Score	+31.31%	A better balance between precision and recall.
AUROC	+16.28%	An improved capability in distinguishing between the classes.
FPR	+ 93.99%	A substantial reduction in the rate at which true positives are mistakenly overlooked.

The above improvement underscores the substantial contribution our user-based sequencing made to enable the Auto-UBS to enhance its detection capabilities to identify anomalies compared to the Auto-TAB using Tabular data format.

5.1 Our Model versus Others

In Table 7, we have compared the performance of our Transformer with fifteen other baselines found in the literature for Insider Threat Detection. The top two results are highlighted, with the best result being boldfaced and the second-best result being underlined. An upward arrow indicates when the metrics have a higher value, it is better, while a downward arrow indicates that a smaller number is better.

Table 7: *Comparison to Other Baseline Models*

Model	DR (Recall) $^{\uparrow}$	FPR (%)$^{\downarrow}$	FNR $^{\downarrow}$
Original Transformer $^{+1}$ ([20])	80.77	X	0.1923
DistilledTrans $^{+2}$ ([20])	84.62	X	0.1538
BERT+FL $^{+3}$ ([20])	95.38	X	0.0462
Roberta+FL $^{+4}$ ([20])	94.62	X	0.0538
CAE ([30])	92.50	0.0510	0.0750
Stack CNN ([31])	95.00	X	0.0500
LSTM-AD ([13])	97.29	0.0380	0.0271
LAN ([32])	94.78	0.0120	0.0522
ITDBERT: Bi-LSTM ([33])	91.87	X	0.0813
Deep-Learning ([34])	92.00	0.0900	0.0800
Stacked BiLSTM & FNN ([35])	90.70	X	0.0930
AD-DNN ([36])	95.00	0.0400	0.0500
Our Transformer(iFOREST)	**99.43**	**0.0571**	**0.0057**

Based on the data shown in Table 7, our Transformer model has demonstrated superior performance compared to baseline models across important metrics such as Recall and FPR. Our model achieved a DR (Recall) of 99.43% and 100% using iForest and OCSVM, respectively, outperforming the transformer baseline models by significant margins (23.81% $^{+1}$, 18.17% $^{+2}$, 4.84% $^{+3}$, and 5.6% $^{+4}$). Unlike the baseline models, which relied solely on a single version of the CERT dataset, our results were derived from a combination of datasets, including CERT r4.2, r5.2, and r6.2. Additionally, we included FPR metrics in our evaluation, achieving rates between 0.0905% and 0.0571% using OCSVM and iFOREST detection algorithms, respectively. We believe that the omission of FPR in the baseline assessments is a significant oversight, given its critical relevance in the context of Insider Threat detection.

6 Limitations and future work

This study has a few limitations that affect the interpretation of its findings. One of the key limitations is that we relied on the CERT dataset. We chose these datasets because they are publicly available and designed to simulate real-world scenarios. However, they are synthetically created and might not entirely reflect real-world insider threat scenarios. Therefore, the extent to which our findings apply may be limited. Another challenge we faced was the significant imbalance in the dataset. Of the 1000 users in the CERT r4.2, only seventy were labeled as malicious. While this is sufficient for testing our detection rate, the small number of malicious users might have limited the depth of our analysis. Additionally, it is imperative to validate our research outcomes using real-world data. While the CERT datasets are useful for development and testing, they may not fully capture the complexity of user behavior in the context of insider threat detection. Furthermore, the technical aspects of this research were constrained by the computing resources available to us. We used the PyTorch framework on personal computers and Google Colab Pro to develop and test our

proposed Transformer. Using a different library or more powerful computational platforms might result in different outcomes.

To better grasp the Transformer's decision-making mechanisms, we highly recommend using some "Explainable AI" to explore the interpretability of the Transformer model. In this research, we only used "Explainable AI" to interpret the decision from the detection algorithms rather than the Transformer model itself.

7 Conclusion

Insider threats are a challenging problem for organizations of all kinds. The problem is further exacerbated because of the secretive nature of these threats, limited reporting mechanisms, and strict privacy laws, which led to a lack of sufficient data for research. Despite these challenges, over the two last decades, researchers have made some progress in developing Machine-Learning (ML) solutions to identify, prevent, and reduce insider risks. However, our literature review identified a significant gap in using Transformer Encoders in insider threat detection despite their proven success in other cyber-security fields. This research addresses this gap by building a predictive model using the Transformer architecture, novel User-Based Sequencing (UBS), and ML anomaly detection algorithms to improve insider threat detection capabilities. Our experiment showcased the impressive performance of our model using test sets based on individual CERT dataset versions, such as r4.2, r5.2, and r6.2, as well as a mixture of data from all CERT dataset versions. Our Transformer model achieved state-of-the-art performance, with a high accuracy rate of 96.61%, 99.43% recall, an F1-score of 96.38%, an AUROC value of 95.00%, and a low False Negative Rate (FNR) and False Positive Rate (FPR) of 0.0057 and 0.0571, respectively.

References

[1] Ponemon. Cost of insider risks, 2023.

[2] CISA - Defining Insider Threats. Defining insider threats, 2023.

[3] Harris Clyde Devince. Understanding controls to detect and mitigate malicious privileged user abuse. 4 2020.

[4] Software Engineering Institute. Common sense guide to mitigating insider threats, seventh edition. Technical report, Sep 2022.

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

[6] Michele Maasberg, Xiao Zhang, Myung Ko, Stewart R Miller, and Nicole Lang Beebe. An analysis of motive and observable behavioral indicators associated with insider cyber-sabotage and other attacks. *IEEE Engineering Management Review*, 48:151–165, 2020.

[7] Joshua Glasser and Brian Lindauer. Bridging the gap: A pragmatic approach to generating insider threat data. pages 98 – 104, San Francisco, CA, United states, 2013.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[9] Krunal Randive, R. Mohan, and Ambairam Muthu Sivakrishna. An efficient pattern-based approach for insider threat classification using the image-based feature representation. *Journal of Information Security and Applications*, 73, 2023.

[10] Abdullah Alshehri. Relational deep learning detection with multi-sequence representation for insider threats. *International Journal of Advanced Computer Science and Applications*, 13:758–765, 2022.

[11] Muhanned AlSlaiman, Mohammed I. Salman, Mariam M. Saleh, and Bin Wang. Enhancing false negative and positive rates for efficient insider threat detection. *Computers and Security*, 126, 2023.

[12] Wei Hong, Jiao Yin, Mingshan You, Hua Wang, Jinli Cao, Jianxin Li, Ming Liu, and Chengyuan Man. A graph empowered insider threat detection framework based on daily activities. *ISA transactions*, 141:84–92, 2023.

[13] Miguel Villarreal-Vasquez, Gaspar Modelo-Howard, Simant Dube, and Bharat Bhargava. Hunting for insider threats using lstm-based anomaly detection. *IEEE Transactions on Dependable and Secure Computing*, 20(1):451 – 462, 2023.

[14] Preetam Pal, Pratik Chattopadhyay, and Mayank Swarnkar. Temporal feature aggregation with attention for insider threat detection from activity logs. *Expert Systems with Applications*, 224:119925, 8 2023.

[15] Ross Gruetzemacher and David Paradice. Deep transfer learning & beyond: Transformer language models in information systems research. *ACM Computing Surveys (CSUR)*, 54(10s):1–35, 2022.

[16] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.

[17] Vijaya Ravindra Sagvekar and Prashant Sharma. Word embedding attention and balanced cross entropy technique for sentiment analysis. *Multiagent and Grid Systems*, 19(1):23 – 42, 2023.

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[19] Brian Lindauer. Insider threat test dataset, 2020.

[20] Wang and A El Saddik. Dtitd: An intelligent insider threat detection framework based on digital twin and self-attention based deep learning models. *IEEE Access*, 11:114013 – 114030, 2023.

[21] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[22] Yunseung Lee and Pilsung Kang. Anovit: Unsupervised anomaly detection and localization with vision transformer-based encoder-decoder. *IEEE Access*, 10:46717 – 46724, 2022. Anomaly detection;Anomaly localizations;Features extraction;Images reconstruction;Location awareness;MVTecAD;Task analysis;Transformer;Vision transformer;.

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Ole Agesen. The cartesian product algorithm: Simple and precise type inference of parametric polymorphism. In *ECOOP'95—Object-Oriented Programming, 9th European Conference, Åarhus, Denmark, August 7–11, 1995 9*, pages 2–26. Springer, 1995.

[25] Zhe Zhao and Bangyong Sun. Hyperspectral anomaly detection via memory-augmented autoencoders. *CAAI Transactions on Intelligence Technology*, 8(4):1274 – 1287, 2023. Abnormal samples;Auto encoders;Background reconstruction;Hyperspectral anomaly detection;Hyperspectral Data;Low dimensional;matrix;Memory modules;Performance;Reconstruction error;.

[26] Diana Haidar and Mohamed Medhat Gaber. Adaptive one-class ensemble-based anomaly detection: An application to insider threats. volume 2018-July, Rio de Janeiro, Brazil, 2018.

[27] Andrew Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern, and Weng-Keen Wong. A meta-analysis of the anomaly detection problem. *arXiv preprint arXiv:1503.01158*, 2015.

[28] Filip Wieslaw Bartoszewski. *Machine learning and anomaly detection for insider threat detection*. PhD thesis, Heriot-Watt University, 2022.

[29] Hongcheng Guo, Xingyu Lin, Jian Yang, Yi Zhuang, Jiaqi Bai, Bo Zhang, Tieqiao Zheng, and Zhoujun Li. Translog: A unified transformer-based framework for log anomaly detection. 2021.

[30] Yichen Wei, Kam-Pui Chow, and Siu-Ming Yiu. Insider threat prediction based on unsupervised anomaly detection scheme for proactive forensic investigation. *Forensic Science International: Digital Investigation*, 38, 2021.

[31] A. Anju and M. Krishnamurthy. M-eos: modified-equilibrium optimization-based stacked cnn for insider threat detection. *Wireless Networks*, 30(4):2819 – 2838, 2024.

[32] Xiangrui Cai, Yang Wang, Sihan Xu, Hao Li, Ying Zhang, and Xiaojie Yuan. Lan: Learning adaptive neighbors for real-time insider threat detection. *arXiv preprint arXiv:2403.09209*, 2024.

[33] Weiqing Huang, He Zhu, Ce Li, Qiujian Lv, Yan Wang, and Haitian Yang. Itdbert: Temporal-semantic representation for insider threat detection. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–7. IEEE, 2021.

[34] Rida Nasir, Mehreen Afzal, Rabia Latif, and Waseem Iqbal. Behavioral based insider threat detection using deep learning. *IEEE Access*, 9:143266–143274, 2021.

[35] Shuang Song, Neng Gao, Yifei Zhang, and Cunqing Ma. Britd: behavior rhythm insider threat detection with time awareness and user adaptation. *Cybersecurity*, 7(1), 2024.

[36] Al-Mhiqani, Rabiah Ahmed, Z.A. Zainal Abidin, and S.N. Isnin. An integrated imbalanced learning and deep neural network model for insider threat detection. *International Journal of Advanced Computer Science and Applications*, 12(1):573 – 577, 2021.