Contrastive Learning with Knowledge-Enhanced Prompts for Insider Threat Detection

Haoyang Yu*†‡, Yunchuan Guo*†‡, jing Wang*†‡, Mengxiang Zhu*†‡, Yongqiang Xu*†‡, Zifu Li*†‡✉
*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
†School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
‡State Key Laboratory of Cyberspace Security Defense, Beijing, China
Email: {*yuhaoyang, guoyunchuan, wangjing2022, zhumengxiang, xuyongqiang, lizifu*}*@iie.ac.cn*

2025 International Joint Conference on Neural Networks (IJCNN) | 979-8-3315-1042-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/IJCNN64981.2025.11229300

***Abstract*—Insider threat detection is essential for protecting organizations from malicious or negligent insiders. This paper proposes a knowledge-enhanced self-contrastive learning framework for insider threat detection in multi-source user behavior graph scenarios. In the user behavior graph representation phase, a multi-head attention mechanism with relational encoding is used to explore user adjacency relations, with node connectivity guiding subgraph sampling. In the knowledge enhancement phase, self-contrastive learning aligns subgraph embeddings with behavior descriptions generated by a prompt template, enriching user behavior features. Finally, the dual-stage detection scheme filters anomalous users using a variational autoencoder and categorizes them through multi-class classification. Experimental results on the CERT insider threat dataset show that our scheme achieves 97.2% accuracy and an F1 score of 0.72, significantly outperforming existing schemes.**

***Index Terms*—Self-Contrastive Learning, Anomaly Detection, Insider Threat Detection, Knowledge-Enhanced**

I. Introduction

The insider threat, which is posed by individuals within an organization who have authorized access to its systems, data, or infrastructure, and who intentionally or unintentionally misuse this access to compromise information systems [1], has resulted in significant and catastrophic consequences. According to a cybersecurity report [1], 83% of organizations reported experiencing insider threats in 2024, a significant increase from 60% in 2023. The report emphasizes the critical role of anomaly detection in identifying and mitigating potential insider threats.

Currently, insider threat detection approaches can be roughly classified into two categories: Machine Learning (ML) [2]–[5] and Deep Learning (DL) [6]–[9]. ML has shown effectiveness in identifying anomalous patterns in user behavior. For example, Le et al. [3] aggregate user behaviors to extract serialized feature vectors and employ ML algorithms such as XGBoost for anomalous user detection. However, the effectiveness of this approach is highly dependent on the validity of feature extraction templates, and shallow ML models cannot efficiently leverage the high-dimensional, complex, heterogeneous, and dynamic nature of behavioral data [10]. DL schemes can learn higher-dimensional semantic representations from raw data. For instance, Villarreal-Vasquez et al. [9] utilize an LSTM model to identify high-dimensional features of insider threat events and analyze long-term dependencies to detect insider attacks. One of the prerequisites for DL to work effectively in anomaly detection is that DL models, consisting of millions of parameters, require large amounts of labeled data for effective training [10]. However, in the anomaly detection field, labeled data are scarce or even absent.

The challenges faced in insider threat detection can be summarized as follows.

- **Heterogeneous Data Representation:** To detect insider threat, heterogeneous data from various sources (e.g. device logs, web activities, file access records, emails, and employee information) have to be used. The diversity in data formats complicates their uniform representation and integration. Furthermore, complex interactions often fail to capture the full correlations and dependencies. Thus, the challenge lies in effectively representing user data and efficiently extracting features from extensive behavioral data to refine correlations and interactions.
- **Data Imbalance:** In practice, anomalous samples in insider threat scenarios are usually very sparse, leading to a highly imbalanced dataset. This imbalance makes it particularly difficult to learn effectively from the data, as classifiers often prioritize the majority class (normal users), leading to poor generalization for the minority class (anomalous users). As a result, the subtle but crucial patterns indicative of malicious behavior may go undetected, as they are often overshadowed by the overwhelming number of normal samples.
- **Coarse-grained label:** Existing datasets typically generalize labels for user categories, lacking detailed descriptions of the fine-grained features of user behavior. This labeling deficiency limits the model's ability to learn the intrinsic relations between users, which further complicates insider threat detection. Therefore, effectively learning and extracting useful features from datasets with coarse-grained labels presents a significant challenge in insider threat detection.

Our main contributions are as follows.

- We propose an effective framework for detecting insider

This work was supported by National Key Research and Development Program of China (No.2023YFB3107605), the National Natural Science Foundation of China (No.62202463, No.U23B2024)

[1]https://gurucul.com/2024-insider-threat-report/

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

threat users. In our framework, heterogeneous behavioral data are modeled as a user behavior graph. To construct the behavior subgraph, we first design involves a self-supervised subgraph sampling and encoding scheme to capture the dependencies between users and behaviors. Then, a Large Language Model (LLM) is used to extract descriptive information about user behaviors, and user behaviors are integrated into the user subgraph embeddings to enrich the behavioral features. Finally, a dual-stage detection scheme is designed: in the first stage, a variational autoencoder (VAE) is used to identify normal behavior patterns. In the second stage, a multi-class classifier is proposed to detect anomalous users that are selected in the first stage.

- To efficiently represent heterogeneous behavioral data in a graph format, we employ a multi-head attention mechanism with fusion relation encoding to explore user adjacency relations. The node link count is used as a self-supervised signal to guide subgraph sampling. To address the issue of poor classification performance due to user sample imbalance and coarse-grained labels, we enhance user behavior knowledge using an LLM and adopt self-contrastive learning to incorporate behavioral descriptions into subgraph embeddings. This approach helps the VAE define the boundary of normal user behaviors and improves the accuracy of the classification model.
- Experimental results on the CERT insider threat dataset demonstrate that our approach achieves an accuracy of 97.2% and an F1 score of 0.72 in multi-class anomalous user detection. Comparative experiments show that our scheme outperforms existing approaches.

II. Related Work

Existing anomaly detection schemes for insider threats can be roughly divided into three categories: the baseline-based schemes, the classification-based schemes, and the graph neural network-based schemes.

In the first category, the pre-defined baselines, which are used to define normal behavior patterns, are constructed based on predefined criteria. Anomalies are detected by evaluating deviations that significantly differ from this normal pattern. Along this line, Gavai et al. [11] used supervised schemes, using expert-developed classifiers, with unsupervised schemes such as Isolation Forests for detecting insider threats in network logs. Tuor et al. [12] employed RNN and DNN schemes for anomaly detection. Lu and Wong [13] proposed a deep neural network scheme utilizing Long Short-Term Memory (LSTM) that models system logs as structured sequences to capture normal usage patterns and distinguish malicious activities. Although these schemes have made significant progress in detecting anomalous activities, they cannot effectively identify user behavior patterns with multiple interactions.

In the classification-based schemes, supervised learning, in which models train on prelabeled datasets, is designed to distinguish normal and anomalous behavior. These models can learn decision boundaries from statistical patterns, enabling them to classify user activities and detect potential threats or irregularities. Along this line, Le and Zincir-Heywood [14] applied unsupervised machine learning schemes for insider threat detection. Al-Shehari et al. [15] integrate Convolutional Neural Networks (CNN) with oversampling techniques, including Synthetic Minority Oversampling Technique (SMOTE), Borderline-SMOTE and Adaptive Synthetic Sampling (ADASYN), to enhance classification performance on imbalanced datasets. Nedelkoski et al. [16] adopted a hypersphere classification objective and designed a self-attention encoder network for logarithmic anomaly detection. Despite the application of classification-based schemes in anomaly detection, they still face challenges such as a lack of data features, data imbalance, the scarcity of malicious activities, and adaptive attacks.

Recently, Graph Neural Networks (GNNs) have been applied to anomaly detection tasks. By constructing a graph representing relations between user behaviors, these schemes map non-Euclidean (irregular spatial structure) data onto graph domains, capturing the associations between users and resources, interaction patterns in access activities, and structural dependencies in networked systems. For example, Graph Convolutional Networks (GCN) [17] effectively capture local information by aggregating weighted node features with those of their neighbors, achieving strong performance in node classification tasks. Graph Transformer Networks (GTN) [18] dynamically adjust graph structures by learning meta-paths, improving performance on complex graph tasks. GraphSAGE (Graph Sample and Aggregation) [19] employs sampling and aggregation strategies to efficiently process large-scale graph data, enhancing both the accuracy and scalability of node classification. However, GNNs face the issue of over-smoothing [20], where the aggregation of the representations of anomalies leads to averaging, making it more difficult to distinguish them.

III. Proposed Method

A. Overall Framework

As shown in Fig.1, our framework overview can be divided into three phases: user behavior graph representation, knowledge enhancement phase, and dual-stage anomaly detection phase. In the first phase, multi-source behavior data (user information and operation logs) are ingested and structured into a graph, where users, emails, devices, files, and web pages serve as nodes, while user access and operations on these entities define edge relations. After heterogeneous encoding, a graph encoder with a sampling aggregation strategy and fusion attention mechanism is designed for graph representation. In the knowledge enhancement phase, subgraph embeddings and user behavior graphs serve as input. The user behavior graphs are processed using a designed prompt template to generate user behavior feature descriptions. Then, self-contrastive learning is employed to achieve semantic alignment between subgraph embeddings and the generated descriptions, resulting in enhanced subgraph embeddings. Finally, in the dual-stage anomaly detection phase, enhanced subgraph embeddings of

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

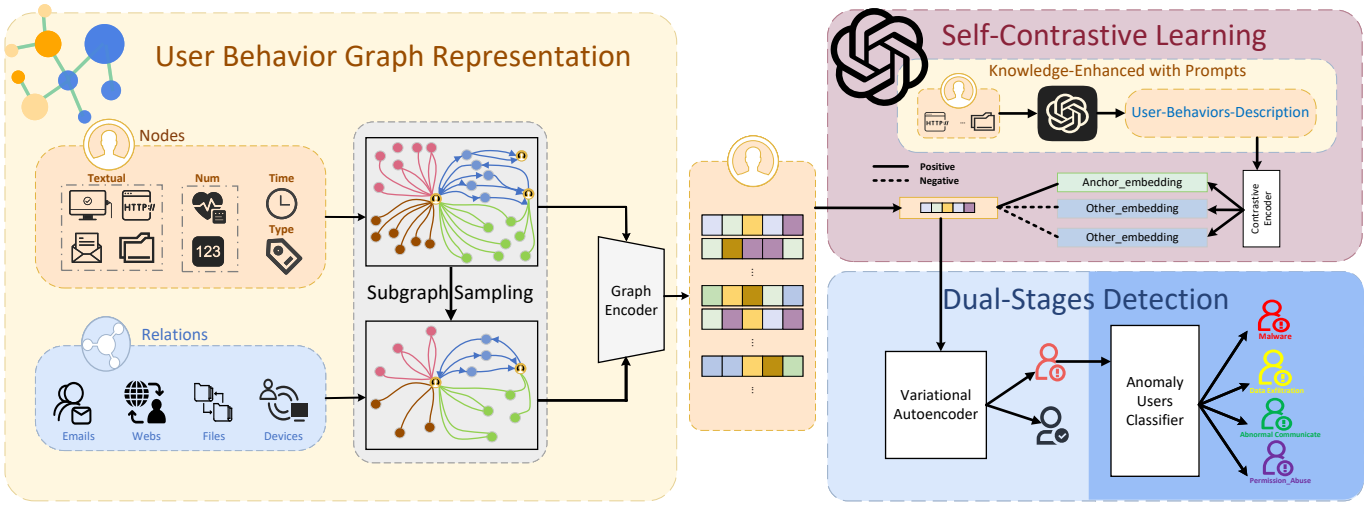

Fig. 1. Overview of our proposed framework, consisting of three phases: 1) User behavior graph representation, where multi-source data are transformed into a graph with nodes and edge relations; 2) Knowledge enhancement phase, utilizing self-contrastive learning for semantic alignment and enhancement of subgraph embeddings; and 3) Dual-stage anomaly detection phase, leveraging a VAE and multi-class classifier for effective anomalous user detection.

all normal users are first learned using a VAE to capture the distribution of normal behavior, thereby filtering out anomalous user samples. These samples are then processed by a multi-class classifier to achieve multi-category anomaly user detection.

B. User Behavior Graph Representation

In this subsection, we design a subgraph sampling scheme within a self-supervised learning framework, integrating heterogeneous encoding, multi-head attention, and node-link prediction. To address node label imbalance, it leverages node features and relation types to learn edge weights, enabling more representative and diverse sampling. Finally, a graph encoder with layer-wise aggregation generates user subgraph embeddings, preserving local and global structural dependencies.

1) Heterogeneous encoder: Since the data encompass various types of non-numeric information, different encoding schemes are required to process categorical features, textual data, numerical attributes, and temporal fields.

For categorical variables, which cannot be directly used in the model, we perform numerical encoding. We use One-Hot Encoding to convert categorical variables into binary vectors. Let $\mathcal{C}$ represent the category set, which can be the node type set $\mathcal{T}$ or the edge type set $\mathcal{R}$. The One-Hot Encoding for any category $C_j \in \mathcal{C}$ is defined as v_{type}.

Text data contain rich semantic information. To effectively capture its internal relations, we use a text encoding model to encode the text data into fixed-dimensional vectors.

$$v_{\text{text}} = f_T(\text{text}), \tag{1}$$

where f_T represents the RoBERTa [21].

For fields containing multiple text elements (e.g. URL, email content, and attachment), we first extract keywords and attachment information. We use the TF-IDF method to select the top N keywords with the highest frequency, denoted as x^{key}. We encode the keywords and then aggregate the embedding vectors by averaging:

$$v_{\text{text}}^{key} = \frac{1}{N}\sum_{i=1}^{N} f_T(x_i^{\text{key}}). \tag{2}$$

For numerical features, we apply the Standardization method, encoding them as v_{num} to eliminate the impact of differing feature scales during model training. In addition, time information is crucial to modeling user behavior patterns. We convert temporal fields into timestamps and datetime fields into Unix timestamps v_{time}.

In the node feature representation, we concatenate all the encoded features to form the final feature vector of node i:

$$v_i = [v_{\text{type}} \oplus v_{\text{num}} \oplus v_{\text{text}} \oplus v_{\text{time}}], \tag{3}$$

where $\oplus$ denotes the vector concatenation operation.

2) Relation encoder: In heterogeneous graphs, the nodes and edges are diverse, with different types of edges carrying distinct semantic relations. For example, in user behavior, "email sending and receiving" and "web page access" belong to two different types of relations. The former represents email contact between users, while the latter represents the interaction between users and web pages. This heterogeneity requires relation embeddings that can differentiate the semantics of different relation types, thus more effectively retaining key relations during sampling. Relation embeddings are assigned a unique embedding vector for each edge type $r \in \mathcal{R}$ to capture its semantic information. With $|\mathcal{R}|$ distinct relation types in the graph, each relation type r is associated with an embedding vector $e_r \in \mathbb{R}^{d_e}$.

The relation embeddings can be represented as:

$$E = \begin{bmatrix} e_1 & e_2 & \cdots & e_R \end{bmatrix}^T, \tag{4}$$

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

where $E \in \mathbb{R}^{R \times d_e}$ is the relation embedding matrix, and e_r is the embedding vector for relation type r.

3) Subgraph sampling in self-supervised attention mechanisms: User behavior graphs consist of a substantial number of nodes and associated edges, requiring effective distinction and utilization of information from different relations. Sampling nodes and edges from the original graph and aggregating them into representative small-scale subgraphs, we significantly enhance the computational efficiency and scalability of GNNs.

User graphs often contain multiple types of relations (such as "access," "communication," etc.), and the attention mechanism assigns different weights to different nodes or edges, enabling the model to dynamically focus on the most relevant parts for the task. For each attention head h, the attention weight $\alpha_{i,j}^{(h)}$ of the edge (i, j) is calculated as follows:

$$\alpha_{i,j}^{(h)} = \frac{\exp\left(\frac{(W_h^Q v_i)^\top (W_h^K v_j) + e_r^\top W_h^R e_r}{\sqrt{d_k}}\right)}{\sum_{k \in \mathcal{N}(i)} \exp\left(\frac{(W_h^Q v_i)^\top (W_h^K v_k) + e_k^\top W_h^R e_k}{\sqrt{d_k}}\right)}, \quad (5)$$

where W is the projection matrix, and d_k is the dimension of the key vector. The final attention weight $\alpha_{i,j}$ is obtained by averaging the weights from all heads:

$$\alpha_{i,j} = \frac{1}{H} \sum_{h=1}^{H} \alpha_{i,j}^{(h)}. \quad (6)$$

Since the input nodes are predominantly normal samples with no clear class distinction, direct training limits the model's ability to capture node importance and relation patterns. To overcome this challenge, this paper designs a self-supervised learning task, in which the model predicts the actual number of connections for each node based on node characteristics and edge importance. By guiding the model predictions to be as close as possible to the true number of connections, the multi-head attention mechanism is encouraged to learn edge weights that better reflect the importance of the nodes.

$$L_{degree} = \frac{1}{N} \sum_{i=1}^{N} \left(d_i^{pred} - d_i^{true}\right)^2, \quad (7)$$

where d^{pred} is the predicted degree of node i, and d^{true} is the true degree of node i. N represents the total number of nodes. By passing d_i^{pred} through the regression layer using the node embedding v_i, we obtain:

$$d_i^{\text{pred}} = W_{\text{deg}} v_i + b_{\text{deg}}, \quad (8)$$

where W_{deg} and b_{deg} are learnable parameters.

Based on the sampling probability, we sample neighbor nodes and edges from each relation type to construct subgraphs containing diverse relations. The probability distribution for sampling from each relation type r is defined as $P(r)$, and can be calculated as:

$$P(r) = \frac{\exp(\alpha_r)}{\sum_{k=1}^{R} \exp(a_k)}, \quad (9)$$

where α_r represents the attention weight of relation type r. $\mathcal{R}$ represents the set of all relation types.

The sampling process for node i's neighbors from each type of relations is described below:

$$\mathcal{N}(i) = \bigcup_{r \in \mathcal{R}} \text{Sample}(\mathcal{N}_r(i), k_r). \quad (10)$$

Eq. 10 describes the process of randomly sampling k_r nodes from the neighbor node set $\mathcal{N}_r(i)$ of different relation types r based on the weight distribution $P(r)$, to ultimately construct the neighbor node set $\mathcal{N}(i)$ of node i. It ensures that relation types with higher attention weights are more likely to be sampled.

4) Subgraph embedding: We update node features using the aggregation scheme based on GraphSAGE [19], generating node embeddings through layer-wise message passing. The feature vector of node i in the k-th layer is:

$$v_i^{(k)} = \sigma\left(W^{(k)} \cdot g\left(v_i^{(k-1)}; \bar{v}_{\mathcal{N}(i)}^{(k-1)}\right)\right), \quad (11)$$

where $W^{(k)}$ is the learnable weight matrix in the k-th layer, σ is the ReLU activation function, and $g(\cdot)$ represents the concatenation operation, used to combine the node's own features with the aggregated features of its neighbors.

Then, the overall embedding of the subgraph u is generated via attention-based pooling:

$$u = \sum_{r \in \mathcal{R}} \sum_{i \in \mathcal{N}_r(i)} \frac{\exp(\text{MLP}_t(v_i)) v_i}{\sum_{j \in \mathcal{N}_r(i)} \exp(\text{MLP}_t(v_j))}. \quad (12)$$

C. Contrastive Learning with Knowledge-Enhanced

Relying solely on node connection count prediction in self-supervised tasks may overlook rare but important low-connection nodes and fail to efficiently leverage the behavioral features within the samples. To address this, we propose a self-contrastive learning with knowledge-enhanced prompt scheme. By guiding the prompts to enrich internal labels, this approach improves the detection of anomalous users with insider threats, helping the model better understand and capture subtle differences in user behaviors.

1) User behavior description generation: To leverage the powerful prior knowledge of LLM to enhance user behavior features, we design a prompt template to detect and extract ***Description*** from user behaviors. Specifically, we input a formatted behavior graph of a user, transform it into natural language input for the LLM, and design a prompt template that instructs the LLM to describe the user based on their basic information and behavior patterns, and identify potential anomalous behaviors.

As shown in Fig.2, for user i, the generated description Des_i is expressed as:

$$Des_i = LLM\left(P\left(G_i\right)\right), \quad (13)$$

where G_i represents the user behavior graph, and P represents the prompt template function, which converts the subgraph into a natural language prompt. (The specific template for P is shown in the INPUT part of Fig.2.) Using a pre-trained

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

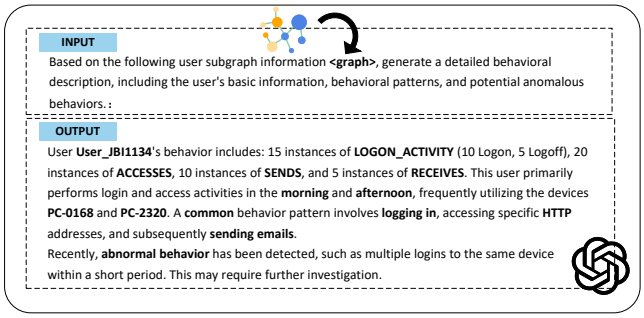

Fig. 2. Prompt Template and Description Generation

text encoding model, we convert the behavior description generated by the LLM into a fixed-dimensional embedding vector, denoted as Z_i. This can be represented as follows,

$$Z_i^T = f_T\left(Des_i\right), \tag{14}$$

where f_T uses the RoBERTa model trained with SimCSE [22] for encoding.

2) Self-contrastive learning: Incorporating behavior descriptions into user subgraph embeddings for knowledge enhancement has become a key challenge. Existing schemes label users based on behavior descriptions and train the model to perform multi-label classification on user behavior graphs. However, this approach overlooks the potential correlations between labels, leading to overfitting on high-frequency labels while neglecting low-frequency ones. This exacerbates the over-smoothing problem, resulting in similar embedding vectors.

To address these issues, we design a framework based on self-contrastive learning to optimize the embedding space by aligning user subgraph embeddings with behavior description embeddings. This scheme effectively enhances the model's ability to distinguish between differences in user behaviors.

Since user subgraph embeddings u and description embeddings z originate from different feature spaces, we perform linear transformations to project them into a unified embedding space. The transformed embeddings are denoted as $\tilde{u}_i$ and $\tilde{z}_i$. The transformation process is achieved through learnable parameter matrices and bias vectors, as shown in the following equations:

$$\tilde{u}_i = W_{\tilde{u}} \cdot u_i + b_{\tilde{u}}, \tag{15}$$

$$\tilde{z}_i = W_{\tilde{z}} \cdot z_i + b_{\tilde{z}}, \tag{16}$$

where u_i and z_i are the original user subgraph embeddings and description embeddings of user i, and $W_{\tilde{u}}$, $W_{\tilde{z}}$, $b_{\tilde{u}}$, and $b_{\tilde{z}}$ are learnable weight matrices and bias vectors corresponding to each embedding.

For the loss function $\mathcal{L}$, we use cosine similarity as the similarity measure. In a batch of N users, we define positive sample pairs as the transformed embeddings $(\tilde{u}_i, \tilde{z}_i)$ of the same user, and negative sample pairs are the embeddings from different users. For each positive sample pair $(\tilde{u}_i, \tilde{z}_i)$, the loss function is defined as:

$$\mathcal{L} = -\log \frac{\exp(\text{sim}(\tilde{u}_i, \tilde{z}_i)/\tau)}{\sum_{j=1}^{N} \exp(\text{sim}(\tilde{u}_i, \tilde{z}_j)/\tau)}, \tag{17}$$

where τ is the temperature parameter, which adjusts the smoothness of the similarity distribution.

By minimizing the contrastive loss, user embeddings with the same behavior description tend to get closer, generating discriminative user embeddings. Meanwhile, the model learns behavioral features that better match the actual user behaviors. This knowledge-enhanced self-contrastive learning scheme, utilizing LLM, plays a crucial role in subsequent classification tasks, as it effectively distinguishes behavior patterns between users and is essential for defining the boundaries of normal user behavior.

D. Dual-Stage Anomaly Detection

Existing anomaly detection schemes that rely on a single model often struggle to balance detection accuracy and classification granularity. To address this, we propose a dual-stage detection scheme that combines one-class classification and multi-class classification to improve the effectiveness of anomalous user detection and the precision of classification.

1) One-classifier stage: In this stage, we first utilize a Variational Autoencoder [23] for one-class classification to filter out potential anomalous users. VAE, as a generative model, learns the latent distribution of the data by maximizing the likelihood of observed data. It can effectively capture the behavior patterns of normal users, thereby identifying anomalous users who deviate from these patterns. VAE consists of an encoder and a decoder. The input is the enhanced user subgraph embedding $\tilde{u}$. The mathematical representations of the encoder and decoder are as follows:

$$q_\phi(z|\tilde{u}) = \mathcal{N}\big(z; \mu_\phi(\tilde{u}), \sigma_\phi^2(\tilde{u})I\big), \tag{18}$$

$$p_\theta(\tilde{u}|z) = \mathcal{N}\big(\tilde{u}; \mu_\theta(z), \sigma_\theta^2(z)I\big), \tag{19}$$

where the latent variable z is the hidden space representation of $\tilde{u}$, with ϕ and θ being the parameters of the encoder and decoder.

To enable the VAE to effectively distinguish between normal and anomalous users, we train the VAE using only the samples of normal users. The training objective is to maximize the lower bound of the normal user data, known as the Evidence Lower Bound (ELBO):

$$L(\theta, \phi; \tilde{u}) = \mathbb{E}_{q_\phi}\big[\log p_\theta(\tilde{u}|z)\big] - \text{KL}\big(q_\phi(z|\tilde{u}) \| p(z)\big), \tag{20}$$

where $\mathbb{E}$ denotes expectation, and KL represents the Kullback-Leibler divergence.

After training, the VAE is used to reconstruct the embeddings of new users and calculate the anomaly score based on the reconstruction probability as an assessment of anomaly,

$$\text{Anomaly Score} = -\log p_\theta(\tilde{u}|z). \tag{21}$$

A lower reconstruction probability indicates that the user's behavior pattern deviates from the latent distribution of normal users, potentially marking the user as anomalous.

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

2) Multi-class classifier stage: After filtering out potential anomalous users in the first stage, the second stage focuses on classifying these users into specific categories. The multi-class classifier can employ a variety of algorithms such as Support Vector Machine (SVM), Random Forest (RF), or Deep Neural Network (DNN). We select RF as the multi-class classifier, as it provides the best classification performance in our experiments.

The training objective of the multi-class classifier is to minimize the cross-entropy loss function, with M representing the number of training samples and K representing the number of classes. The function is defined as:

$$L_{\text{CE}} = -\frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K} y_{m,k}\log \hat{y}_{m,k}, \tag{22}$$

where $y_{m,k}$ is the indicator variable of the true label belonging to class k, Specifically, if sample m belongs to class k, then $y_{m,k} = 1$, otherwise $y_{m,k} = 0$. $\hat{y}_{m,k}$ is the predicted probability that sample m belongs to class k, as output by the model. This probability is derived using a Softmax function applied to the output logits of the classifier.

The multi-class classification stage plays a critical role in fine-grained anomaly detection of users. Unlike binary anomaly detection, which merely aims to identify whether a user is anomalous, multi-class classification allows the model to categorize anomalous users into distinct types, such as data leak perpetrators, IT destroyers, or other forms of abnormal users. This level of detailed classification provides actionable security insights, helping security personnel take appropriate actions against different types of threats.

IV. Experiments

A. Insider Threat Dataset

The CERT insider threat dataset is a publicly available dataset [24]. The version of the dataset used in this study is 5.2 (CERT r5.2), which records an organization with 2,000 employees over a period of 18 months. CERT r5.2 includes user activity logs, categorized as: login/logout, emails, web-pages, files, and device connections, along with organizational structure and user information. Each malicious insider in CERT r5.2 belongs to one of the four common insider threat scenarios: data leakage (scenario 1), intellectual property theft (scenarios 2 and 4), and IT sabotage (scenario 3).

In the data processing process, we constructed user nodes, email nodes, file nodes, HTTP nodes, and device nodes, and annotated edges between nodes based on user activities. For example, the sending and receiving of emails connect user nodes to email nodes.

B. Baselines

To evaluate the performance of our model, we selected two main categories of baseline models for comparison: one comprising end-to-end anomaly classification models, and the other consisting of internal combination models for dual-stage detection. In the end-to-end anomaly classification models, we compared five different models: one Machine Learning method (Random Forest), two types of Deep Learning Networks (CNN [15] and LSTM [9]), and two types of Graph Neural Networks (GCN [17] and RAT [25]). In the internal combination models for dual-stage detection, we used Isolation Forest (IF) as the model for the first stage, and selected other machine learning schemes, such as SVM and XGBoost, for comparison in the second stage.

C. Implementation details

Experiments were conducted on a computing cluster equipped with 4 NVIDIA 4090 GPUs (24GB VRAM each), an Intel Xeon Platinum 8352V CPU (2.10GHz, 32 cores) and 256GB DDR4 RAM. The operating system used was Ubuntu 20.04 LTS (64-bit), with Python 3.10 and PyTorch 2.1.2. To ensure reproducibility, random seeds and data shuffling were fixed consistently across all experiments. We selected ChatGPT-4o as the Large Language Model to enhance knowledge.

To validate the effectiveness of our approach, in the first stage, 80% of the labeled normal user samples were randomly selected as the training set, with the entire user dataset used as the test set. In the second stage, the anomalous users detected in the first stage were used as input and 80% of these samples were labeled to create a new training set. To ensure data balance and fairness in the experiments, the proportion of each anomalous class in the input data for the second stage was kept consistent with the original dataset (29:10:30:30). Furthermore, the SMOTE oversampling technique [26] was applied to the second anomalous class to improve the performance of the model.

The experimental parameters were as follows: 5 types of nodes, 9 types of relationships, 32 attention heads (H=32), a temperature parameter τ of 0.1, a learning rate of 1e-4, and 50 epochs for the dual-stage training with early stopping. The threshold was set at 90, the random seed at 42, the batch size at 256, and the AdamW optimizer [27] was used. These configurations ensured the stability and reliability of the experiment results.

D. Effectiveness Evaluation of Our Scheme

1) Overall performance comparison: Table I summarizes the overall performance of the anomalous user detection schemes of insider threats, demonstrating that our proposed model (VAE + RF) outperforms all baseline models. Specifically, the VAE + RF model achieves an accuracy of 97.2% and an F1 score of 0.723, both of which are the highest among all the models compared, with the F1 score improving by 1.7% over the best-performing baseline.

Compared to other models, RAT achieves the highest precision at 86.8%, due to its effective use of an attention mechanism that captures dependencies between nodes in the graph structure and facilitates global information propagation. However, the recall rate of the RAT is 59. 6%, indicating a more conservative approach. The lower recall rate may be attributed to the insufficient sensitivity of the graph neural network to anomalous patterns, emphasizing the benefit

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT MODELS

Model	Precision (%)	Recall (%)	Accuracy (%)	F1-Score
RF	35.70	46.50	87.30	0.404
CNN	81.00	51.50	90.05	0.630
LSTM	79.10	53.50	93.15	0.639
GCN	74.30	55.60	95.75	0.638
RAT	**86.80**	59.60	92.95	0.706
(VAE+SVM)	63.20	60.60	96.30	0.619
(VAE+XGBoost)	67.40	64.60	96.70	0.659
(IF+SVM)	62.10	64.60	96.30	0.632
(IF+RF)	68.00	**70.70**	96.90	0.694
(VAE+RF)	84.00	63.60	**97.20**	**0.723**

of knowledge-enhanced user behavior descriptions through prompts, which can capture more detailed behavioral patterns.

Although (IF + RF) achieves the highest recall rate at 70.7%, its precision is lower, resulting in more false positives where normal users are misclassified as anomalous. This is because IF lacks the ability to learn latent features from the data, relying on tree-based partitioning for anomaly detection, which affects its precision.

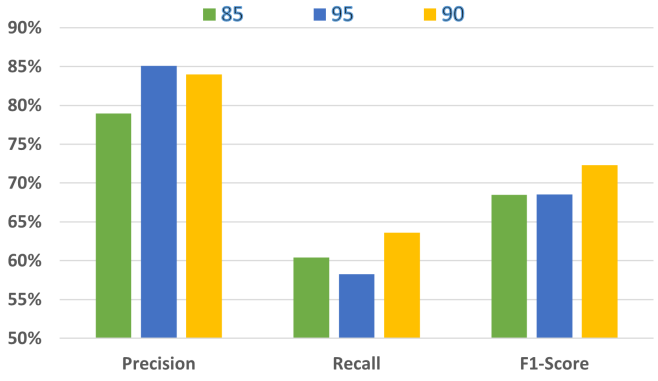

Fig. 3. Performance Comparison of Different Thresholds

2) Threshold performance comparison: As shown in Fig. 3, when the threshold is set at 90%, the model achieves the highest F1 score and recall. This indicates that at this threshold, the model can effectively distinguish between positive and negative samples without making overly aggressive judgments about "anomalous" users. This setting helps identify most insider threats while minimizing unnecessary investigations.

When the threshold is reduced to 85%, the model tends to classify users as "anomalous" more easily, which increases the recall, allowing it to identify more true anomalies. However, this also results in more normal users being misclassified as anomalies, thereby reducing precision and negatively impacting the overall F1 score.This is suitable for high-risk scenarios like financial fraud detection, where detecting anomalies takes priority despite some false positives.

On the other hand, when the threshold is raised to 95%, the model becomes more cautious, requiring stronger evidence before classifying a user as "anomalous." This may slightly improve precision, as fewer normal users are mistakenly flagged as anomalies. However, this conservative approach leads to missing more true anomalies, causing a decline in recall and ultimately lowering the F1 score. This is ideal for low-risk settings, such as website user behavior monitoring, where minimizing false positives ensures a smooth user experience.

TABLE II
ABLATION STUDY WITH DIFFERENT DETECTION THRESHOLDS AND WITHOUT SELF-CONTRASTIVE LEARNING

Number	GR	CL	Acc(%)	F1
1	✓		96.5	0.685
2		✓	95.8	0.650
3	✓	✓	97.2	0.723

E. Ablation Study

To evaluate the effectiveness of graph representation learning and self-contrastive learning, we conducted an ablation study on the CERT r5.2 dataset, with the results shown in Table II. Each row in the table corresponds to a different configuration (indicated by a ✓) and its associated accuracy and F1 score from the ablation experiment.

In the first row, the model uses only graph representation learning, achieving an accuracy of 96.5% and an F1 score of 0.685. This result highlights the ability of graph representation learning to capture complex relationships within the data and contribute to effective classification.

The second row applies only self-contrastive learning, yielding an accuracy of 95.8% and an F1 score of 0.650. While self-contrastive learning improves the model's ability to distinguish between different classes, its performance is slightly lower compared to the graph representation learning approach.

In the third row, both graph representation learning and self-contrastive learning are combined, achieving the highest

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.

accuracy of 97.2% and an F1 score of 0.723. This configuration demonstrates the synergistic effect of the two schemes, with their combination leading to a significant boost in model performance by enhancing feature extraction and promoting more robust learning.

Overall, the experimental results clearly indicate that the combination of graph representation and self-contrastive learning provides a substantial performance improvement over the individual schemes, highlighting the complementary strengths of both approaches.

V. Conclusion

This paper proposes a knowledge-enhanced self-contrastive learning framework for insider threat detection, suitable for multi-source user behavior graph scenarios. The framework consists of three main components: user behavior graph representation, knowledge enhancement, and dual-stage anomaly detection.

In the user behavior graph representation stage, multi-source behavior data are integrated, and high-quality graph embeddings are generated through heterogeneous encoding, sampling aggregation strategies, and a fusion attention mechanism. In the knowledge enhancement stage, self-contrastive learning is employed to align subgraph embeddings with behavioral description embeddings, significantly improving the semantic consistency of the embeddings. In the anomaly detection stage, a VAE is first used to learn the behavior distribution of normal users to filter potential anomaly samples. Then, a multi-class classifier is used to detect anomalies of multiple categories within these samples.

Experimental results demonstrate that the proposed scheme achieves high accuracy in detecting anomalous users in the CERT insider threat dataset, significantly outperforming existing detection schemes. This indicates that the proposed framework has significant application value in improving the accuracy of user behavior analysis. Future work will focus on enhancing adaptability to behavior-level anomaly detection and integrating real-time detection mechanisms to improve the robustness of insider threat detection.

References

[1] S. E. Institute, "Common sense guide to mitigating insider threats, seventh edition," Carnegie Mellon University, Software Engineering Institute's Digital Library, Software Engineering Institute, Tech. Rep., Sep 2022, accessed: 2024-Dec-28. [Online]. Available: https://insights.sei.cmu.edu/library/common-sense-guide-to-mitigating-insider-threats-seventh-edition/

[2] M. A. Haq, M. A. R. Khan, and M. Alshehri, "Insider threat detection based on nlp word embedding and machine learning," *Intell. Autom. Soft Comput*, vol. 33, no. 1, pp. 619–635, 2022.

[3] D. C. Le, N. Zincir-Heywood, and M. I. Heywood, "Analyzing data granularity levels for insider threat detection using machine learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 30–44, 2020.

[4] N. Elmrabit, F. Zhou, F. Li, and H. Zhou, "Evaluation of machine learning algorithms for anomaly detection," in *2020 international conference on cyber security and protection of digital services (cyber security)*. IEEE, 2020, pp. 1–8.

[5] A. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, "Machine learning for anomaly detection: A systematic review," *Ieee Access*, vol. 9, pp. 78 658–78 700, 2021.

[6] R. Nasir, M. Afzal, R. Latif, and W. Iqbal, "Behavioral based insider threat detection using deep learning," *IEEE Access*, vol. 9, pp. 143 266–143 274, 2021.

[7] M. N. Al-Mhiqani, R. Ahmed, Z. Z. Abidin, and S. Isnin, "An integrated imbalanced learning and deep neural network model for insider threat detection," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 1, 2021.

[8] Y. Wang, J. Zhang, S. Guo, H. Yin, C. Li, and H. Chen, "Decoupling representation learning and classification for gnn-based anomaly detection," in *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, 2021, pp. 1239–1248.

[9] M. Villarreal-Vasquez, G. Modelo-Howard, S. Dube, and B. Bhargava, "Hunting for insider threats using lstm-based anomaly detection," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 451–462, 2023.

[10] S. Yuan and X. Wu, "Deep learning for insider threat detection: Review, challenges and opportunities," *Computers & Security*, vol. 104, p. 102221, 2021.

[11] G. Gavai, K. Sricharan, D. Gunning, R. Rolleston, J. Hanley, and M. Singhal, "Detecting insider threat from enterprise social and online activity data," in *Proceedings of the 7th ACM CCS international workshop on managing insider security threats*, 2015, pp. 13–20.

[12] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," in *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[13] J. Lu and R. K. Wong, "Insider threat detection with long short-term memory," in *Proceedings of the Australasian Computer Science Week Multiconference*, ser. ACSW '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3290688.3290692

[14] D. C. Le and N. Zincir-Heywood, "Anomaly detection for insider threats using unsupervised ensembles," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1152–1164, 2021.

[15] T. Al-Shehari, M. Kadrie, M. N. Al-Mhiqani, T. Alfakih, H. Alsalman, M. Uddin, S. S. Ullah, and A. Dandoush, "Comparative evaluation of data imbalance addressing techniques for cnn-based insider threat detection," *Scientific Reports*, vol. 14, no. 1, p. 24715, 2024.

[16] S. Nedelkoski, J. Bogatinovski, A. Acker, J. Cardoso, and O. Kao, "Self-attentive classification-based anomaly detection in unstructured logs," in *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2020, pp. 1196–1201.

[17] J. Jiang, J. Chen, T. Gu, K.-K. R. Choo, C. Liu, M. Yu, W. Huang, and P. Mohapatra, "Anomaly detection with graph convolutional networks for insider threat and fraud detection," in *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, 2019, pp. 109–114.

[18] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," *Advances in neural information processing systems*, vol. 32, 2019.

[19] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[20] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International conference on machine learning*. PMLR, 2019, pp. 6861–6871.

[21] Y. Liu, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, vol. 364, 2019.

[22] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," *arXiv preprint arXiv:2104.08821*, 2021.

[23] D. P. Kingma, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[24] B. Lindauer, "Insider threat test dataset," *Carnegie Mellon University Dataset*, 2020.

[25] S. Feng, Z. Tan, R. Li, and M. Luo, "Heterogeneity-aware twitter bot detection with relational graph transformers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 4, 2022, pp. 3977–3985.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[27] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization. 7th int," in *Conf. Learn. Represent. ICLR*, 2019.

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on January 23,2026 at 04:47:35 UTC from IEEE Xplore. Restrictions apply.