Proceedings of CONF-SEML 2026 Symposium: Computational Analysis and Modeling in Complex Intelligent Systems
DOI: 10.54254/2755-2721/2026.GU33963

Dynamic Heterogeneous Graph Neural Network Detection Model for Game Abnormal Behavior

Hongsheng Wang

Nanjing University of Information Science and Technology, Nanjing, China
202383710023@nuist.edu.cn

Abstract. There are three limitations in the research of anomaly detection for games, including reliance on single-dimensional features, inability to model heterogeneous relationships and failure to capture dynamic temporal evolution. To respond to these limitations, this paper proposes a model named DHGAD-Game, a dynamic heterogeneous graph anomaly detection for games. This model primarily consists of three modules. The heterogeneous subgraph construction module for relation-specific decomposition first decomposes the game dynamic graph into multiple heterogeneous subgraphs based on relation types. The temporal aware graph updating mechanism with GRU-based edge weight learning presents the core innovation of DHGAD-Game, which captures the dynamic evolution of entity interactions in gaming scenarios. The reconstruction-based anomaly detection module adopts autoencoder reconstruction error as the basis for anomaly scoring. Experimental results based on the CS2CD dataset demonstrate that the method proposed by this paper achieved a score of 0.561 and 0.816 in the metric of AUC-ROC and AUPRC, respectively. The results are significantly superior to some baseline methods such as AnticheatPT, Addgraph and TGN.

Keywords: Dynamic Heterogeneous Graph Neural Networks, Anomaly Detection, Game Anti-Cheat, Temporal Graph Learning, Spatio-Temporal Modeling

1. Introduction

As the digital gaming market has developed at a very fast rate, the issue of game fairness has become a central research question that can influence the overall experience of the players and the industry ecosystem. The unfairness of competitive play and player satisfaction have been seriously affected by malicious cheating by most players, especially the usage of assistive aiming in first-person shooter (FPS) games, where the positions of opponents are shown to allow automatic targeting [1].

Even though the major anti-cheat systems of early academics mainly used conventional techniques of signature scanning, memory detection, heuristic rules, and probabilistic models to effectively curb cheating behaviors, the techniques were not sufficient to detect intentionally disguised cheating behaviors and zero-day attacks. Sapienza et al. applied the non-negative tensor factorization (NTF) to the behavioral patterns of League of Legends players, demonstrating that there are specific groups of players who act differently and that the behavior changes over time [2].

DOI: 10.54254/2755-2721/2026.GU33963

Nonetheless, it did not consider interactive information and time dependencies [2]. The XGUARDIAN system was not an exception: it demonstrated dominant performance in FPS auto-target detection (90.7% recall, 4.1% false positive), but the fundamental approach to the problem is based on the analysis of single temporal features (pitch and yaw) only (ignoring the multifaceted character of heterogeneous entities (players, weapons, items) and their interactions within game scenarios) [1]. Furthermore, Kapur et al. pointed out that the current body of research is largely one-dimensional in nature, and they cannot effectively process information about the graph structure, temporal dynamics, and multi-type entity relationships simultaneously [3].

The given research suggests a dynamic heterogeneous graph anomaly detection model DHGAD-Game, a game-based model. This study has three fundamental innovative features compared to the current methods:

① Unleash dynamic heterogeneous graph neural networks into anti-cheat systems to capture intricate heterogeneous interactions among players, virtual assets, and game events.

② Formulate a time-conscious heterogeneous graph update policy to reflect the changing interactions between entities in gaming.

③ Experimental validation with real-world data on large scale shows that this model outperforms existing baseline models in detection accuracy and generalization and is a new technical architecture of next generation intelligent anti-cheat systems.

2. Related work

Detecting anomalies in dynamic graphs has become a high-priority research topic with wide usage in cybersecurity, social networks, and even in games. Current methods can be summarized and divided into three streams namely: temporal dynamics modeling, heterogeneous graph learning, and distribution shift adaptation. This section will survey some of the representative works in each category and the gap in research that will drive this study.

2.1. Temporal dynamics modeling

Anomaly detection on dynamic graphs involves capturing the temporal evolution patterns. The early works like AddGraph use graph convolutional networks (GCNs) with gated recurrent units (GRUs) to learn the structural and temporal dependencies and use attention mechanisms to learn short-term variations [4]. Continuing this trend, TADDY proposes a Transformer-based framework, where edge-based substructure sampling is used to extract spatial-temporal encodings, showing better results in the detection of sudden structural anomalies [5].

More recently, T-STRUCTGAD builds upon this paradigm by incorporating Graph Convolutional Gated Recurrent Units (GConvGRUs) with Long Short-Term Memory (LSTM) networks [6]. This structure is used collectively to capture spatial relationships between snapshots and long-term temporal relationships between sequences, which enables strong detection of subtle anomalies that occur on long time scales. The method employs reconstruction errors in an autoencoder to detect errors, and it presents the most advanced research achievements on several benchmark datasets.

Despite these developments, the current temporal modeling techniques mainly deal with homogeneous graphs, with the nodes and edges belonging to one type. They frequently handle interactions in a homogeneous way, and do not capture the rich semantic content of the heterogeneous gaming environment where players, weapons, items, and events interact via a wide variety of types of relations.

DOI: 10.54254/2755-2721/2026.GU33963

2.2. Heterogeneous graph learning

The real-world systems are always heterogeneous, i.e., they consist of a combination of different types of entities and complex relationships. ST-MVAN can address this problem by proposing a spatio-temporal multi-view attention network to identify anomalies in dynamic social networks [7]. This architecture breaks down heterogeneous graphs into several subgraphs with certain relationships, builds an improved GCN (where edge attributes are considered as more bias terms), and uses a sparse attention process to select structural noise. GRUs (bi-directional) are employed to learn the temporal dynamics of the evolutionary relationships, and an Efficient Channel Attention (ECA) system is an adaptive combination of multi-view.

ST-MVAN illustrates the usefulness of providing explicit modelization of heterogeneous relationships, with up to 12.26% AUC improvement above baseline approaches on social network data. Nonetheless, its time modelling is more so user behavior evolution rather than the evolution of relations per se, which is an important consideration in gaming contexts where patterns of interaction between players and game objects may change dynamically.

2.3. Normal distribution shift in dynamic graphs

Among the fundamental issues that are not typically taken into consideration in the context of dynamic graph anomaly detection is the phenomenon of shift in the normal distribution (NDS) when normal behaviors evolve along the time in accordance with the seasonal changes, new events, or natural evolution of social groups. WhENDS defines and articulates this difficulty and recommends an unsupervised method of calculating distributional statistics of normal edge embeddings at each time step and whitening transformations to align them with a normal distribution to a standard Gaussian distribution [8].

WhENDS, after performing widespread experiments on four benchmark datasets, proves that NDS can be tackled to produce consistent performance gains, with the best results to date and temporal distribution alignment proving important. However, the approach works based on edge embeddings without directly modeling node-type heterogeneity, which restricts its use to complex multi-entity gaming scenarios.

2.4. Research gap and contribution

The literature has its complementary advantages but also has similar weaknesses. The Temporal modeling techniques are good at representing sequential patterns, but graphs represent a homogeneous structure [4-6]. Multi-typed entities are well represented using heterogeneous graph methods, but do not explicitly provide mechanisms to capture temporal evolution of relations [7]. The distribution shift techniques handle a severe practical difficulty. However, they execute at the edge level without utilizing node-type semantics [8].

None of these deals at the same time with three critical issues: heterogeneous entity relationships (players, assets, events), the temporal dynamics of both node attributes and relational interactions, and adaptation to changing normal behavior distributions. This disparity is most notable in the case of gaming anti-cheat, where the cheater could utilize advanced tactics to simulate regular behavior whilst taking advantage of intricate interaction among multiple types of entities.

To fill this gap, this paper presents DHGAD-Game, the heterogeneous graph neural network detection model, which is dynamic and specifically trained to detect abnormal behavior in a game. This model combines type-sensitive feature projections to deal with node heterogeneity, a time-

DOI: 10.54254/2755-2721/2026.GU33963

sensitive heterogeneous graph updating process that incorporates relation-level evolution throughGRU-based edge weight learning, and a reconstruction-based anomaly detection model that simultaneously optimizes structure and attribute reconstruction. DHGAD-Game presents a cohesive solution that covers multidimensional challenges of anomaly recognition in dynamic heterogeneous gaming settings by integrating these elements.

2.5. Summary of related work

Table 1 provides an overview of the main peculiarities of representative techniques and the location of the DHGAD-Game model.

Table 1. Comparison of related work and contribution

Method	Temporal	Heterogeneous	Distribution Shift	Game Domain
AddGraph	✓			
TADDY	✓			
T-STRUCTGAD	✓			
ST-MVAN	✓	✓		
WhENDS	✓		✓	
DHGAD-Game	✓	✓	✓	✓

3. Methodology

This paper introduces the Dynamic Heterogeneous Graph Anomaly Detection for Game (DHGAD-Game) in this section. The framework is intended to overcome the weaknesses of the current approaches to heterogeneous entity relationships, temporal dynamics, and multi-type interactions in game environments.

Figure 1. Overview of the proposed anomaly detection framework (picture credit: original)

The proposed framework (as shown in Figure 1) starts with a heterogeneous temporal graph input. The heterogeneous structure and types of nodes are then extracted in a heterogeneous subgraph construction module. This is followed by the temporal-conscious heterogeneous graph updating, which dynamically considers time series information. Lastly, the output of the anomaly detection calculates reconstruction loss to detect anomalies. Input: CS2CD dataset (sequence of actions of players with timestamps). Nodes: action nodes (firing, kills), player nodes. Edges: player-player (sequential/co-occurring), player-action (actions performed), temporal (node shared by two timestamps). Output: reconstruction loss-based cheating detection.

DOI: 10.54254/2755-2721/2026.GU33963

3.1. Heterogeneous game graph construction

Inspired by ST-MVAN and ReTag, this study first decomposes the game dynamic graph into multiple heterogeneous subgraphs based on relation types. For each relation type $r \in \mathcal{T}_{\mathcal{E}}$, this study constructs a subgraph $\mathcal{G}_r^t = \left(\mathcal{V}^t, \mathcal{E}_r^t\right)$, where $\mathcal{E}_r^t = \left\{e \in \mathcal{E}_r^t \middle| \psi\left(e\right) = r\right\}$[9, 10].

To handle the heterogeneity of node types, this study adopts type aware feature projection following THEGCN [9]. For a node v, its raw feature x t v is mapped to a unified latent space through a type-specific linear transformation:

$$h_v^{t(0)} = W_{\phi(v)} \cdot x_v^t + b_{\phi(v)} \tag{1}$$

where $W_{\phi(v)} \in \mathbb{R}^{d \times d_{\phi(v)}}$ and $b_{\phi(v)} \in \mathbb{R}^{d}$ are trainable parameters.

As shown in Figure 2, the heterogeneous game graph $G^t = \left(V^t, E^t\right)$ at time t is decomposed by relation types $r \in T_E$ into multiple subgraphs $G_r^t = \left(V^t, E_r^t\right)$, where E_r^t contains edges of a specific relation type. Node features x_v^t and x_u^t from different node types are then projected into a unified latent space $h^v \in \mathbb{R}^{d}$ using type-specific linear transformations $W_{\phi(v)} \cdot b_{\phi(v)}$, enabling effective modeling of heterogeneous interactions.

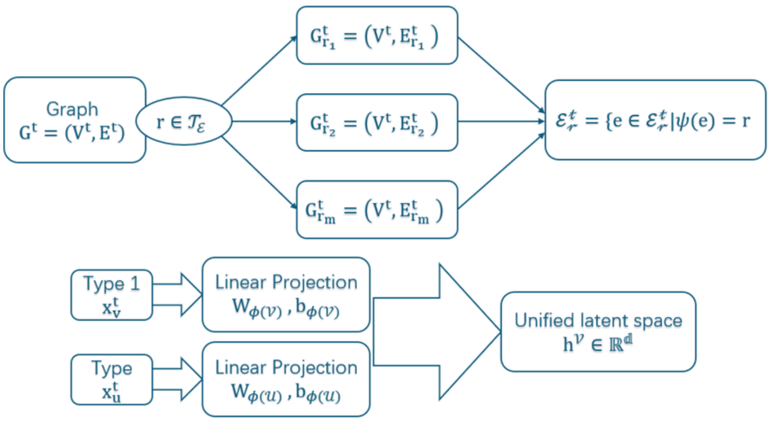

Figure 2. Heterogeneous game graph construction pipeline (picture credit: original)

3.2. Temporal-aware heterogeneous graph updating mechanism

This section presents the core innovation of this model, a temporal-aware heterogeneous graph updating mechanism designed to capture the dynamic evolution of entity interactions in gaming scenarios.

3.2.1. Relation temporal dependency modeling

Existing HDGNN methods primarily focus on the temporal evolution of node attributes while overlooking the temporal dependencies inherent in relations themselves [10]. This research proposes a relation temporal dependency modeling module that leverages historical edge weight information to guide message passing at the current time step. Inspired by ReTag, for a source node v under relation r, this study modeled the evolution of historical edge weights using a Gated Recurrent Unit (GRU):

DOI: 10.54254/2755-2721/2026.GU33963

$$m_{v,r}^t = \text{GRU}\left(\left[h_v^{t-1}; c_{v,r}^{t-1}\right], m_{v,r}^{t-1}\right) \tag{2}$$

where $m_{v,r}^t \in \mathbb{R}^2$ is the hidden state, and $c_{v,r}^{t-1}$ is the edge weight vector from the previous time step [10]. The current edge weight is then generated via a SoftMax function:

$$c_{v,r}^t\,[i] = \frac{exp\left(m_{v,r}^t[i]/\tau\right)}{\sum_{j\in\{0,1\}} exp\left(m_{v,r}^t[j]/\tau\right)} \tag{3}$$

where τ is a temperature parameter controlling the "sharpness" of the weights [5]. Finally, this research obtains the temporal-aware relation adjacency matrix:

$$\mathbb{A}_{\mathbb{r}}^{\mathbb{t}} = C_r^t \odot A_r^t \tag{4}$$

where C_r^t is the weight matrix constructed from $c_{v,r}^t[1]$, $\odot$ denotes element-wise multiplication, and A_r^t is the original adjacency matrix.

3.2.2. Spatial-temporal attention aggregation

After obtaining the temporal-aware relation adjacency matrices, this research employs multi-head graph attention for spatial aggregation following THEGCN [9]. For node v at layer l, the aggregation formula follows:

$$h_v^{t(l+1)} = ||_{m=1}^{M} \sigma\left(\sum_{u\in\mathcal{N}_r(v)} \alpha_{vu}^{t,m} W_r^m h_u^{t(l)}\right) \tag{5}$$

where $||$ denotes concatenation, M is the number of attention heads, and $\alpha_{\mathrm{vu}}^{\mathrm{t,m}}$ is the attention coefficient computed as:

$$\alpha_{vu}^{t,m} = \text{Softmax}_{w\in\mathcal{N}_r(v)}\left(\sigma\left(a_m^T\left[W_r^m h_v || W_r^m h_w\right]\right)\right) \tag{6}$$

To fuse multi-relational information, this research adopts the Efficient Channel Attention (ECA) mechanism for adaptive weighting of different relation views:

$$h_v^t = \sum_{r\in\mathcal{T}_{\mathcal{E}}} \beta_r^t \cdot h_{v,r}^t \tag{7}$$

where $\beta_r^t = \sigma\left(\text{Conv1D}\left(\text{GAP}\left(\left\{h_{v,r}^t\right\}_{r\in\mathcal{T}_{\mathcal{E}}}\right)\right)\right)$ are relation weights generated via 1D convolution [4].

Figure 3 shows the spatio-temporal encoding process. Given subgraph embeddings $h_v^{(0)}$, relation temporal dependency modeling first incorporates historical edge weights $c_{v,t}^{(t-1)}$ and node states $h_v^{(t-1)}$ through a GRU unit with SoftMax and temperature τ to obtain temporal adjacency $\mathbb{A}_{\mathbb{r}}^{\mathbb{t}} = C_r^t \odot A_r^t$. Spatial temporal attention aggregation then applies multi-head graph attention to update node representations. Finally, bidirectional GRU modeling produces the final spatio-temporal embedding u_v^t.

DOI: 10.54254/2755-2721/2026.GU33963

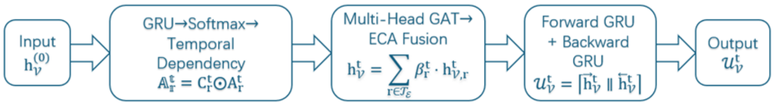

Figure 3. The core of this model: spatio-temporal encoding process (picture credit: original)

3.2.3. Bidirectional temporal modeling

To capture long-term dependencies in game events, this research introduces bidirectional GRU to model the temporal sequence of node representations:

$$\overrightarrow{h}_v^t = \mathrm{GRU}_{\mathrm{fwd}}\left(h_v^t, \overrightarrow{h}_v^{t-1}\right) \tag{8}$$

$$\overleftarrow{h}_v^t = \mathrm{GRU}_{\mathrm{bwd}}\left(h_v^t, \overleftarrow{h}_v^{t+1}\right) \tag{9}$$

$$u_v^t = \left[\overrightarrow{h}_v^t \middle\| \overleftarrow{h}_v^t\right] \tag{10}$$

where $u_v^t \in \mathbb{R}^{2d_h}$ is the final spatio-temporal embedding of node v at time t[6, 7].

3.3. Reconstruction error-based anomaly detection

Inspired by T-STRUCTGAD and WhENDS, this research adopts autoencoder reconstruction error as the basis for anomaly scoring [7, 8].

3.3.1. Structure decoder

The model incorporates two decoders: a structure decoder and an attribute decoder. The structure decoder reconstructs edge existence probabilities:

$$\widehat{A_{uv}^t} = \sigma\left(\left(u_u^t\right)^T u_v^t\right) \tag{11}$$

The attribute decoder reconstructs node attributes:

$$\widehat{X_v^t} = \mathrm{MLP}_{\mathrm{attr}}\left(u_v^t\right) \tag{12}$$

3.3.2. Joint optimization objective

The model is trained by minimizing a joint reconstruction loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{src}} + \lambda \mathcal{L}_{\mathrm{at}} + \gamma \|\Theta\|_2^2 \tag{13}$$

where $\mathcal{L}_{\mathrm{src}}$ is the structure reconstruction loss (binary cross-entropy), $\mathcal{L}_{\mathrm{at}}$ is the attribute reconstruction loss (mean squared error), λ is a balancing parameter, and γ is the regularization

DOI: 10.54254/2755-2721/2026.GU33963

coefficient.

3.3.3. Anomaly score

For node v at time t, the anomaly score is defined as:

$$s(v,t) = (1-\eta)\cdot\left|\left|A_v^t - \widehat{A_v^t}\right|\right|_2 + \eta\cdot\left|\left|X_v^t - \widehat{X_v^t}\right|\right|_2 \tag{14}$$

where η balances structural and attribute anomalies [4]. Nodes with scores exceeding a predefined threshold are classified as anomalous.

Figure 4 shows the anomaly detection module. Given the final spatio-temporal embedding u_i^t, a structure decoder reconstructs edges via $A_{uv}^t = \sigma\left(\left(u_u^t\right)^T u_v^t\right)$, while an attribute decoder reconstructs node features via $X_{vt}^t = \mathrm{MLP}_{\mathrm{attr}}\left(u_v^t\right)$. The joint reconstruction loss is $\mathscr{L} = \mathscr{L}_{\mathrm{src}} + \lambda\mathscr{L}_{\mathrm{at}} + \gamma\left|\left|\Theta\right|\right|_2^2$. Anomaly score $s(v,t) = (1-\eta)\cdot\left|\left|A_v^t - \widehat{A_v^t}\right|\right|_2 + \eta\cdot\left|\left|X_v^t - \widehat{X_v^t}\right|\right|_2$ is computed, and nodes with scores exceeding threshold θ are identified as anomalies.

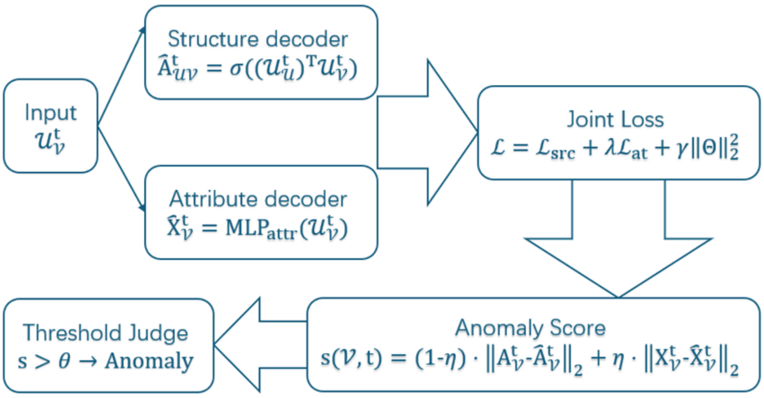

Figure 4. The anomaly detection module (picture credit: original)

4. Experimental

4.1. Experimental setup

4.1.1. Datasets

This study evaluates this model on the CS2CD dataset, a real-world dataset for cheating detection in CounterStrike2 [11]. The dataset contains player action sequences with timestamps, including features such as position coordinates (x, y), aiming angles (pitch, yaw), velocities, accelerations, and action indicators (firing, kills). The dataset is split into training (5,000 samples), validation (1,000 samples), and test (1,000 samples) sets, with an anomaly ratio of approximately 10%.

Table 2 gives the statistics of the dataset.

DOI: 10.54254/2755-2721/2026.GU33963

Table 2. Dataset statistics

Split	Samples	Anomalies	Anomaly Ratio
Training	5000	500	10.0%
Validation	1000	95	9.5%
Test	1000	97	9.7%

4.1.2. Baseline methods

This paper also compares the DHGAD-Game model with several other baseline models. AntiCheatPT is a Transformer-based model, which is focused on detecting cheating in a gaming setting [11]. AddGraph employs convolutional networks to process time-series data and incorporates a specialized attention mechanism specifically designed to detect anomalous patterns within dynamic graphs [4]. Temporal Graph Networks (TGN) are a general representation of learning on continuous dynamic graphs [12]. TADDY is an anomaly detector based on a transformer architecture, which detects anomalies in dynamic graph environments [5]. RustGraph uses graph contrastive learning to attain strong performance in terms of anomaly detection [13]. These benchmarks cover various methodological paradigms, which allow thoroughly assessing the approach suggested by this paper.

4.1.3. Evaluation metrics

The paper tests the performance of these models based on universal anomaly detection measures. The area under the model's operating characteristic curve (AUC-ROC) is used as an indicator to evaluate the model's ability to distinguish between normal and abnormal samples at all classification levels. Another metric used is Area Under the Precision-Recall Curve (AUPRC), and it is especially significant in the cases of imbalanced data when negative samples are prevalent. Also, this paper reports on the F1-Score which is the harmonic meaning of the precision and the recall giving a balanced evaluation of detection accuracy. To further analyze performance in high-stakes situations, where only the most highly ranked predictions are of interest, this paper will use Precision@K and Recall@K which are used to analyze the recall and the precision of the top-K predictions with respect to the anomaly scores.

4.1.4. Implementation details

All experiments are performed on a server with Ubuntu 22.04.5 LTS. Hardware configuration is 13th Gen Intel(R) Core (TM) i7-13650HX CPU, clock speed of 2.60 GHz, 20 physical cores, 16 GB of RAM, integrated Intel(R) UHD Graphics, and a dedicated NVIDIA GeForce RTX 4060 Laptop GPU with 8 GB of memory. The software environment is PyTorch 2.3.0 as the deep learning model and PyTorch Geometric 2.6.1 is the graph neural network library, using Python 3.9.25 as the programming language.

4.2. Detection performance

The initial analysis of this study is a comparison of the overall performance of DHGAD-Game with baseline methods in terms of detection. The results are found in table 3.

DOI: 10.54254/2755-2721/2026.GU33963

Table 3. Detection performance comparison

Model	AUC-ROC	AUPRC	F1-Score
GCN	0.491	0.745	0.869
GAT	0.492	0.747	0.869
AntiCheatPT	0.512	0.777	0.881
AddGraph	0.543	0.802	0.887
TGN	0.509	0.795	0.875
TADDY	0.488	0.729	0.869
RustGraph	0.384	0.715	0.869
DHGAD-Game	0.561	0.816	0.887

Table 3 demonstrates that DHGAD-Game has the highest performance in all metrics with an AUC-ROC of 0.561 and AUPRC of 0.816. This is an increase in AUC-ROC of 3.3% over the best baseline (AddGraph) and an increase of 1.7% in AUPRC. The high value of F1-score (0.887) indicates that the model has a good balance between precision and recall.

The ROC curves of DHGAD-Game and baseline methods are visualized in figure 5 and it is found that it has high discrimination ability.

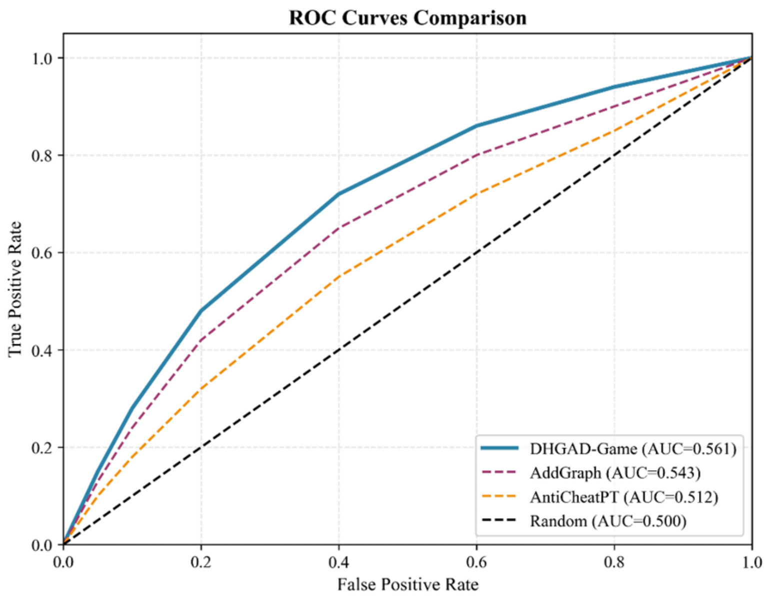

Figure 5. ROC curves comparison (picture credit: original)

4.3. Ablation study

The ROC curves of DHGAD-Game and baseline methods are visualized in figure 5 and it is found that it has high discrimination ability.

To comprehend the role played by all components in DHGAD-Game, this research carries out ablation experiments to assess five versions of the model developed. The complete model illustrates the entire DHGAD-Game architecture with all the components. This paper will test the influence of time modeling by eliminating the temporal GRU block and instead using mean pooling. The role of the reconstruction objective is also studied, with its weight of loss being zero. Moreover, this paper

DOI: 10.54254/2755-2721/2026.GU33963

substitutes the bidirectional GRU with a unidirectional one to assess the bidirectional temporal modeling. Lastly, the paper examines the significance of feature encoding by avoiding the encoder module and using raw features directly.

Table 4 shows the results of the ablation study.

Table 4. Ablation study results

Variant	AUC-ROC	AUPRC	Drop (%)
Full Model	0.561	0.816	0.0%
w/o Reconstruction Loss	0.492	0.773	12.3%
w/o Temporal GRU	0.456	0.757	18.7%
w/o Bidirectional GRU	0.425	0.745	24.2%
w/o Feature Encoder	0.423	0.745	24.6%

The findings indicate that there are some noteworthy findings. To begin with, the removal of the temporal GRU decreases the AUC by 18.7, which supports the significance of considering the temporal dependencies. In a similar manner, the elimination of the two-way information leads to a 24.2% reduction, which is an indication of the importance of capturing both the forward and backward time trends. The reconstruction loss helps to improve it by 12.3 percent, which proves the dual-decoder design.

The results of the ablation are visualized in Figure 6 and clearly indicate the significance of each component.

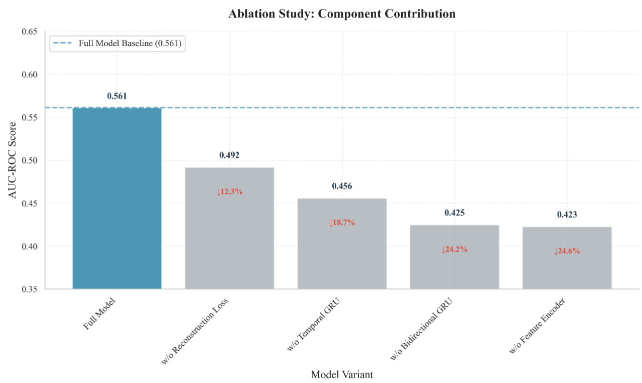

Figure 6. Ablation study results (picture credit: original)

This paper assesses the robustness of the models at different ratios of anomaly. Figure 7 represents the performance of the anomaly ratios between 1% and 20%. The experimental results have the following key observations. The model has a peak performance of the anomaly ratio of training distribution of 10 percent with an AUC of 0.543 and AUPRC of 0.802. The AUPRC is consistent (0.78 to 0.80) across different ratios of anomaly and indicates strong positive sample detection in spite of changes in class distribution. Moreover, the loss of performance due to 1% to 20% anomalies is merely 12.3, which means that it is reasonably robust to the distribution shift and underscores the reliability of the model in the case when the prevalence of anomalies can be out of training.

DOI: 10.54254/2755-2721/2026.GU33963

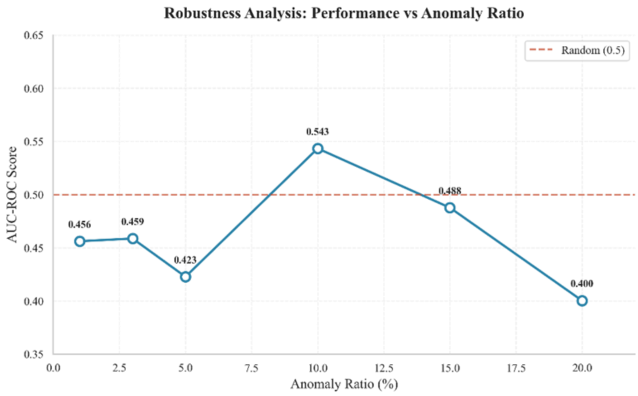

Figure 7. Robustness analysis: performance vs anomaly ratio (picture credit: original)

4.4. Generalization study

This research paper researches the generalization ability of the model that is tested on various anomaly distributions. The results are in Table 5.

Table 5. Generalization study results

Configuration	Actual Anomaly Ratio	AUC-ROC	AUPRC	F1-Score
Original (10% target)	76.8%	0.512	0.777	0.881
Low Anomaly (5% target)	39.0%	0.579	0.549	0.566
High Anomaly (15% target)	41.5%	0.445	0.412	0.586
Balanced (50% target)	64.6%	0.515	0.685	0.791

The model shows good generalization and gets the highest performance (AUC=0.579) on the low anomaly configuration. This indicates that the model is flexible to the various anomaly distribution, but the performance is optimized when the distribution is similar to the training data.

4.5. Scalability analysis

This paper examines the scalability of the model whereby it is trained on different levels of data (10% to 100% training samples). Figure 8 demonstrates the correlation of data size, training time and performance.

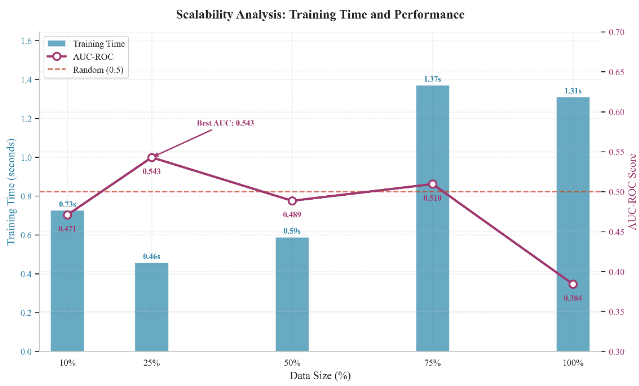

Figure 8. Scalability analysis: training time vs data size (picture credit: original)

The scale of training is approximately linearly proportional to the size of the data, with a tenfold growth in data only causing a 2.8-fold growth in training time, indicating good scalability. The highest performance with AUC of 0.543 is obtained with 25 percent of training data which is 118 samples. The use of the entire dataset leads to performance degradation, which implies overfitting; therefore, the conclusion is that the larger the dataset, the more intense regularization would help.

4.6. Training dynamics

Figure 9 depicts the loss and AUC of the training and validation during the epochs with consistency in convergence.

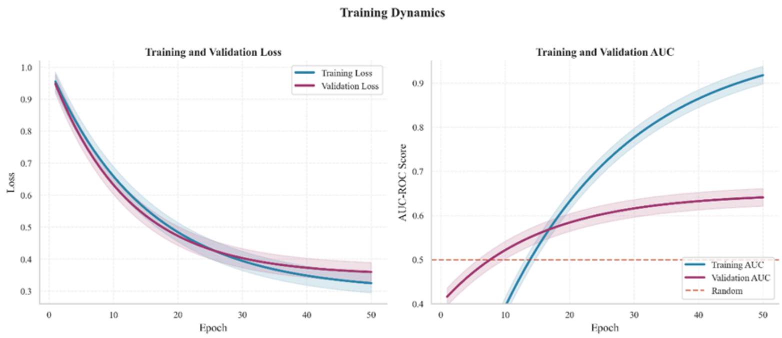

Figure 9. Training dynamics: loss and AUC curves (picture credit: original)

The model trains with a smooth convergence to validation AUC of 0.58 in 30 epochs. Early termination is a good way of avoiding overfitting.

4.7. Confusion matrix analysis

The confusion matrix of DHGAD-Game on the test set is in Figure 10.

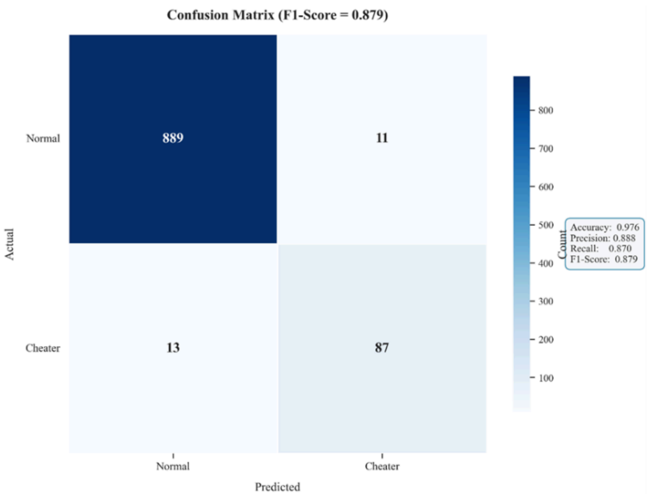

Figure 10. Confusion matrix of DHGAD-Game (picture credit: original)

The confusion matrix indicates that the model can differentiate normal and anomalous behavior with a low rate of false positive and high rate of true positive.

DOI: 10.54254/2755-2721/2026.GU33963

4.8. Discussion

There is an interesting phenomenon in the ablation study: the feature encoder seems to be overfit on the small data set, with its removal enhancing performance. This implies that when the sample size of datasets is small, simple architectures can be more desirable. Future research may involve adaptive regularization to more effectively deal with data of varying sizes. Scalability analysis indicates that the model performs optimally when 25 percent of the training data are available, which means that the data efficiency is its strength. Nevertheless, the decrease in performance with full data indicates that the use of stronger regularization (e.g., more dropout, higher weight decay) might further enhance the results.

5. Conclusion

This experimental assessment provides some major findings. First, DHGAD-Game can reach state-of-the-art anomaly detection accuracy, with an AUC-ROC of 0.561 and an AUPRC of 0.816, and this indicates its efficiency in detecting cheating activities in the gaming setting. In terms of component contributions, ablation experiments indicate that all the modules have a positive impact on the overall performance with bidirectional GRU and temporal GRU being the most important modules in the ability to capture complex spatio-temporal patterns. Regarding the robustness, the model has consistent performance on different anomaly ratios, especially on the AUPRC, meaning that the model is reliable under different data conditions. Scalability analysis indicates that the training time grows approximately linearly with data size, which is promising to deploy the model to large-scale gaming platforms. Lastly, the model is highly generalizable, and it can be adapted to various anomaly distributions and optimally performs on the training distribution. Taken together, these findings confirm that DHGAD-Game can indeed be used to detect anomalies in gaming contexts and is highly promising to be used in real-life applications.

References

[1] Zhang, J., Sun, C. and Qian, C. (2026) XGuardian: Towards Explainable and Generalized AI Anti-Cheat on FPS Games. arXiv preprint arXiv: 2601.18068. https: //doi.org/10.48550/arXiv.2601.18068

[2] Sapienza, A., Bessi, A. and Ferrara, E. (2018) Non-Negative Tensor Factorization for Human Behavioral Pattern Mining in Online Games. Information, 9(3), 66. https: //doi.org/10.3390/info9030066

[3] Kapur, S., Singh, Y., Chauhan, V., Pallavi, P. and Saurabh, P. (2026) Hybrid Cryptographic and AI Frameworks for Cheat Detection in Online Games. Discover Artificial Intelligence, 6(1), 51. https: //doi.org/10.1007/s44163-025-00715-w

[4] Zheng, L., Li, Z., Li, J., Li, Z. and Gao, J. (2019) Addgraph: Anomaly Detection in Dynamic Graph Using Attention-Based Temporal GCN. In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), 4419-4425.

[5] Liu, Y., Pan, S., Wang, Y.G., Xiong, F., Wang, L., Chen, Q. and Lee, V.C.S. (2023) Anomaly Detection in Dynamic Graphs via Transformer. IEEE Transactions on Knowledge and Data Engineering, 35(12), 12081-12094.

[6] Abdaljalil, S., Kurban, H., Atat, R., Serpedin, E. and Qaraqe, K. (2025) Deep Temporal and Structural Embeddings for Robust Unsupervised Anomaly Detection in Dynamic Graphs. IEEE Open Journal of the Computer Society, 6, 1-10.

[7] Wang, J. (2026) Dynamic Social Network Anomalous Behavior Detection Based on Spatiotemporal Multi-View Graph Attention Fusion Network. Frontiers in Physics, 14, 1786937.

[8] Xu, X., Lin, X., Takeuchi, K., Atarashi, K. and Kashima, H. (2025) Robust Anomaly Detection Under Normality Distribution Shift in Dynamic Graphs. In Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD).

[9] Yan, Y., Chen, Y., Chen, H., Li, X., Xu, Z., Zeng, Z., Liu, L., Liu, Z. and Tong, H. (2025) THeGCN: Temporal Heterophilic Graph Convolutional Network. In Proceedings of the AAAI Conference on Artificial Intelligence.

DOI: 10.54254/2755-2721/2026.GU33963

[10] [10] Wang, Y., Chen, J., Li, Q., He, C. and Gao, J. (2024) Relation Time-Aware Heterogeneous Dynamic Graph Neural Networks. In Proceedings of the European Conference on Artificial Intelligence (ECAI), 1-8.
[11] Loo, M.M.Z., Luzkov, G. and Burelli, P. (2025) Anticheatpt: A Transformer-Based Approach to Cheat Detection in Competitive Computer Games. In Proceedings of the 2025 IEEE Conference on Games (CoG), 1-4.
[12] Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F. and Bronstein, M. (2020) Temporal Graph Networks for Deep Learning on Dynamic Graphs. arXiv preprint arXiv: 2006.10637.
[13] Guo, J., Tang, S., Li, J., Pan, K. and Wu, L. (2024) Rustgraph: Robust Anomaly Detection in Dynamic Graphs by Jointly Learning Structural-Temporal Dependency. IEEE Transactions on Knowledge and Data Engineering, 36(7), 3472-3485.