MVAN: Multi-view Attention Networks for Real Money Trading Detection in Online Games

Jianrong Tao[†], Jianshi Lin[†], Shize Zhang[†], Sha Zhao[‡], Runze Wu[†], Changjie Fan[†], Peng Cui[*]

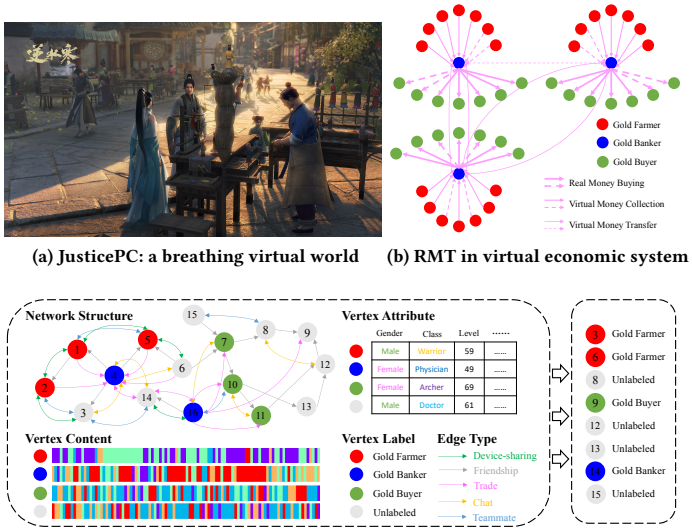
[†]Fuxi AI Lab, NetEase Inc., Hangzhou, China
[‡]College of Computer Science and Technology, Zhejiang University, China
*College of Computer Science and Technology, Tsinghua University, China
{hztaojianrong, linjianshi, zhangshize, wurunze1, fanchangjie}@corp.netease.com,
szhao@zju.edu.cn, cuip@tsinghua.edu.cn

ABSTRACT

Online gaming is a multi-billion dollar industry that entertains a large, global population. However, one unfortunate phenomenon known as *real money trading* harms the competition and the fun. Real money trading is an interesting economic activity used to exchange assets in a virtual world with real world currencies, leading to imbalance of game economy and inequality of wealth and opportunity. Game operation teams have been devoting much efforts on real money trading detection, however, it still remains a challenging task. To overcome the limitation from traditional methods conducted by game operation teams, we propose, MVAN, the first multi-view attention networks for detecting real money trading with multi-view data sources. We present a multi-graph attention network (MGAT) in the graph structure view, a behavior attention network (BAN) in the vertex content view, a portrait attention network (PAN) in the vertex attribute view and a data source attention network (DSAN) in the data source view. Experiments conducted on real-world game logs from a commercial NetEase MMORPG (JusticePC) show that our method consistently performs promising results compared with other competitive methods over time and verifiy the importance and rationality of attention mechanisms. MVAN is deployed to several MMORPGs in NetEase in practice and achieving remarkable performance improvement and acceleration. Our method can easily generalize to other types of related tasks in real world, such as fraud detection, drug tracking and money laundering tracking etc.

CCS CONCEPTS

• **Information systems → Data mining**; **Massively multiplayer online games**; • **Computing methodologies → Anomaly detection**; **Multi-task learning**;



(a) JusticePC: a breathing virtual world (b) RMT in virtual economic system

(c) Illustration of multi-view data sources for RMT detection in online games

Figure 1: (a)-(b) Real money trading in virtual economic system of JusticePC. (c) Illustration of RMT detection from multi-view: network structure view, vertex content view, vertex attribute view and data source view.

KEYWORDS

Real money trading detection; multi-view attention networks; multi-graph attention network; attention mechanism; online games

ACM Reference Format:
Jianrong Tao[†], Jianshi Lin[†], Shize Zhang[†], Sha Zhao[‡], Runze Wu[†], Changjie Fan[†], Peng Cui[*]. 2019. MVAN: Multi-view Attention Networks for, Real Money Trading Detection in Online Games. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3292500.3330687

1 INTRODUCTION

Online gaming is one of the most successful applications having a large number of players interacting in an online persistent virtual world through the Internet. Particularly, a Massively Multi-player Online Role Playing Game (MMORPG) is considered one of the

[‡]NetEase Fuxi AI Lab: named after Fu Xi, the legendary creator in China, and established to enlighten games with artificial intelligence. (https://fuxi.163.com/en/)

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330687

most popular online game genres and the most similar to the real world. In MMORPGs, a player operates several characters, where the character is an avartar and owns its unique features of appearance, gender, name, class, level etc. Each character performs various activities including battles, socializing, trading, communication, cooperation, party play, communities etc. in the virtual world.

The motivation for playing online games is not limited to having fun. Players often pursue wealth in the virtual world, which can be in the form of a character's virtual currency. Such demands have led to the emergence of the exchanging market of virtual assets with real money called real money trading (RMT). In practice, professional cyber criminals are known as a gold farming group (GFG) with a group of gold farmers running multiple client programs for MMORPGs on numerous machines to efficiently collect virtual assets, a group of gold bankers focusing on trading virtual money and managing a large amount of it and a group of gold buyers buying virtual money with real money for quickly achieving a high level in a well organized manner as shown in Figure 1b.

The over production and popularity of RMT could have a negative impact on the virtual economy and cause severe damage to the reputation of the MMORPG. Furthermore, RMT is often associated with other criminal activities, such as money laundering, identity theft and cheating. To keep the virtual world sound and peaceful, game operation teams have been devoting much efforts, however, RMT detection still remains a challenging task.

The challenges we face that hinder the performance of RMT detection in online games are as follow:

(1) ***Excessive human effort.*** Labeling and evaluation efforts are too heavy in traditional real money trading detection settings by the game operation teams. Automatic RMT detection method with negligible human effort is highly demanded.
(2) ***Label uncertainty.*** Rule-generated labels from game operation teams are always full of uncertainties. We are confident at 'confirmed RMTers'. But it is unclear whether the 'banned RMTers' and 'unobservable characters' are RMTers or not.
(3) ***Group Detection.*** Most methods target gold farmers (or game bots) individually. They have less insight of who belongs to the same group. Meanwhile, GFGs fight banning by continuously creating new gold farmers, making current efforts ineffective.
(4) ***Concept drift.*** New types of RMTers evolve over time and get more and more unpredictable for the RMT detection system. Non-stationary behaviors of characters are the main causes of concept drift. Stable RMT detection method is highly demanded.
(5) ***Quick response.*** Anomaly detection in rapidly changing environments is a long-standing problem. We need to detect RMTers as soon as possible when new vertices or edges keep feeding dynamically to avoid economic damage and reduce game loss.

Real money trading can only be carried out in the virtual world by using the MMORPG infrastructure. Therefore in general, the facts and clues of anomalies are recorded on log data in the game server as shown in Figure 1c. The multi-view data source of a character is supposed to be a strong clue in RMT detection. To address those aforementioned challening issues, we propose a **m**ulti-**v**iew **a**ttention **n**etworks (MVAN) for real money trading detection in online games. The contributions of our study are four-fold:

(1) ***Multi-view data sources.*** To the best of our knowledge, this is the first work that introduces an RMT detection method in online games utilizing the strong expressiveness of multi-view data sources to address the aforementioned challenges.
(2) ***The MVAN.*** Multi-view learning is proposed to combine our multi-graph attention network, behavior attention network, portrait attention network and data source attention network.
(3) ***Real evaluation.*** We evaluate our multi-view attention networks based on the real-world dataset. Extensive experiments show the advantages of our method against all baselines.
(4) ***Industrial application.*** Our work has been implemented and deployed in multiple MMORPGs e.g. JusticePC, ghosts, revelation etc. in NetEase Games and received very positive reviews.

2 DATASETS DESCRIPTION

2.1 Game Logs in JusticePC

We use a dataset of JusticePC[1], which is a popular MMORPG released by NetEase Games[2] and creates a breathing virtual world shown in Figure 1a. Various activities performed by the characters or state change events are recorded in the form of structured game logs through the game servers in JusticePC. The game logs consist of the following information.

- Timestamp: when a specific activity or event occurs.
- Character information: the character's role ID, level, class, virtual money, knead face time and VIP, etc.
- Log ID: an ID that identifies the type of activity (hunting monster, trading virtual money, item acquisition, etc.) or state change event type (level up, virtual money increasing or decreasing, etc.).
- Target character information: the other character information if the log is related to the interaction with another character.
- Detailed information: depending on each activity type or event type, detailed information related to the activity type or event type.

We record more than 100 billion different types of game logs which contains more than 20 million character creation activity logs up to now. Specifically, we use the game logs in JusticePC of more than 227,148 characters in a selected game server which has severe RMT problems, from 1st November to 31st December, 2018. Among these characters, 12,529 characters are gold farmers, 1,549 characters are gold bankers and 2,737 characters are gold buyers. The RMTers are identified and labeled by NetEase Games' operation teams. Considering the privacy, the characters in our dataset are ensured by anonymizing all personal identifiable information.

2.2 Social Graphs Construction

We construct five different types of graphs from the game logs which include a transaction graph, a device-sharing graph, a friendship graph, a team graph and a chat graph. They are visualized in Figure 2 and build up a multi-relational graph between characters.

[1] https://n.163.com/
[2] http://game.163.com/

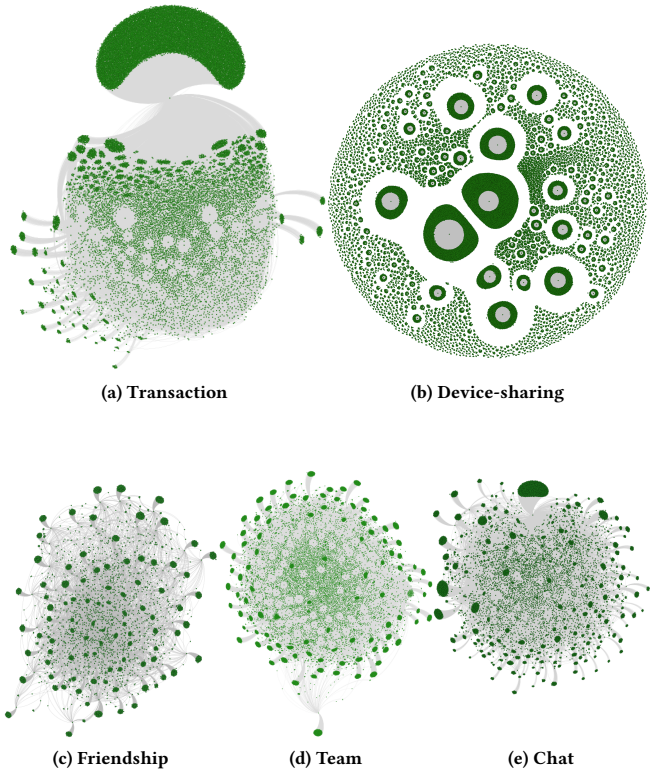


Figure 2: Social graphs in JusticePC

The transaction graph shows assets exchange relations between characters in the virtual world. Edges indicate the virtual currency of established transactions between characters.

The device-sharing graph reveals the relation of characters sharing the same device. Edges exist between characters, indicating log-in activities with same device in the history.

The friendship graph is built upon unidirectional friendship in online games. A character can send an invitation to another character and remove friends from his friendship lists.

The team graph is made up of collaborative relations between characters. A team is temporally formed with the same goal and is disbanded after achieving the goal.

The chat graph expresses the communication relationship between characters. A character can send a private message to other characters for individual communications.

We measure the relative frequency of 1-hop neighbours around RMTers which is plotted in Figure 3. We conclude that the social graphs have the following properties for separating RMTers with regular characters.

- **Distance aggregation:** Characters that are close to each other have similar labels and RMTers tend to form dense relationships with each other.
- **Structural difference:** Graph structures of RMTers are different from regular characters in the number of 1-hop neighbours of different types of characters.
- **Usage diffusion:** The probability of farming gold, banking gold or buying gold by individual characters increases dramatically if neighbors in social graphs participate in the real money trading.
- **Relations correlation:** Social interactions like participating in teams, making friends, exchanging messages or sharing devices are highly correlated with the real money trading activities.

2.3 Behavior Sequences Construction

Each character behavior sequence is composed of lists of events ordered by time stamp which contain three features as followed:

- Event ID: the current event conducted by a character, for example, using a certain skill, obtaining a certain item, etc.
- Interval: the time interval in seconds that has passed between the last and the current game event of a character.
- Level: the current game level for the character. The lowest level of each player is 1 and the highest is raised regularly.

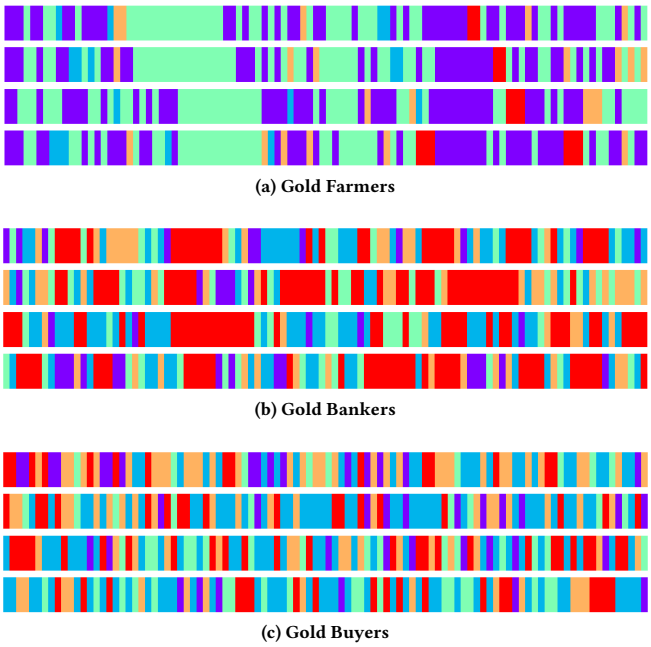
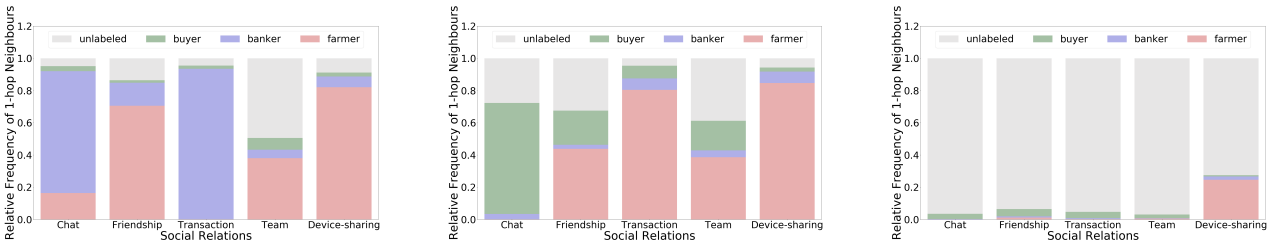


Figure 4: Behavior sequences of gold farmers are similar to each other, behavior sequences of gold buyers show diversity, while behavior sequences of gold bankers show some repeated trading events.

Figure 4 visualizes the behavior sequences for three types of RMTers, which gives us a general idea of how behavior sequences differ from each other. Each slot represents a game event, and different game events are assigned different colors to differentiate them, e.g., red slots correspond trading events. As we can see from the figure, the behavior sequences of gold farmers exhibit a simplex pattern compared to others and show some similarity. Because gold farmers obtain virtual assets by continuing to receive and complete tasks and periodically transfer them to gold bankers. Gold bankers' and gold buyers' behavior sequences are more complicated and



(a) Gold Farmers (b) Gold Bankers (c) Gold Buyers

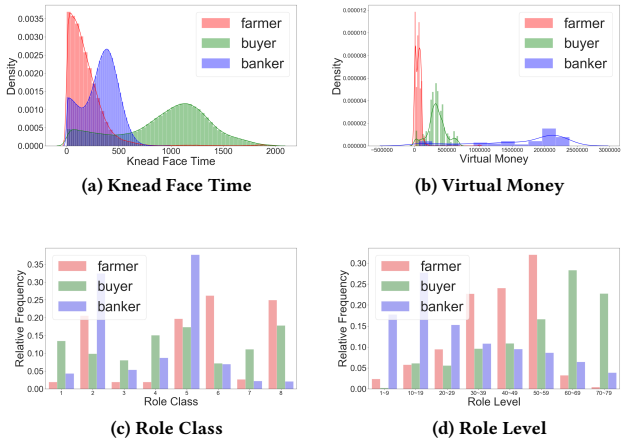
Figure 3: Relative frequency of 1-hop neighbours around RMTers: gold farmers, gold bankers, gold buyers.

unpredictable than gold farmers. Gold bankers show some repeated trading events while gold buyers show some more diversity, which allow our models to correctly classify the characters.

2.4 Character Portraits Construction

We extract some of the current character portraits information from the game logs in real time, such as gender, class, level, knead face time, game scores, and (virtual or real) currencies etc.

We compare the character portraits information of different RMTers and find some valuable information. From Figure 5 we can find that gold farmers' knead face time is concentrated on smaller values because they don't spend much time on the game plays unrelated to virtual assets. The gold buyers pay more attention to the appearance of the characters and spend a lot of time to pinch the face. Gold farmers' virtual money is relatively small and concentrated, because gold farmers' behaviors are very regular and transfer virtual money to gold bankers periodically. Gold bankers collect a lot of virtual money from dozens, hundreds of gold farmers, so the amount of virtual money is in a large scale. Gold buyers' virtual money is between the two, maintaining the character's survival and growth in the game. Gold farmers are concentrated in four role classes, because these professions are easier to use scripts to hunt monsters and complete missions automatically. Gold bankers focus on two role classes, because these occupations are easier to survive in the game and keep safe from accusations by other characters. Gold farmers are concentrated in the medium level, because they can not get more virtual property in too low levels and tasks become more difficult for automation in too high levels. Gold bankers' role level is relatively low for cost saving and disguising themselves. Golf buyers rank to higher levels because they attach great importance to the growth of characters.



(a) Knead Face Time (b) Virtual Money (c) Role Class (d) Role Level

Figure 5: Statistics of four character portraits from RMTers.

3 PROPOSED MVAN

In this section, we provide details of our proposed **M**ulti-**v**iew **a**ttention **n**etworks (MVAN) for real money trading detection in online games. Figure 6 shows the architecture of our proposed method which has four views: network structure view, vertex content view, vertex attribute view and data source view.

3.1 Multi-graph Attention Network

Graph attention network (GAT)[23] specifies different weights to different nodes in a neighborhood, however, ignores the multiple edge types and edge weights information. Specifically, we conduct a multi-relational graph which leads to a **m**ulti-**g**raph **a**ttention ne**t**work (MGAT) also considering edge types and edge weights information. We will start by describing a single multi-graph attention layer to construct multi-graph attention network through stacking this layer as shown in Figure 6[a].

The input to our layer is a set of nodes (characters), c_i means the i-th character, we embed the nodes to vectors through an embedding matrix $\mathbf{W}^{\mathbf{c}}_{\mathbf{emd}}$. We use a five dimensional vector to represent the edge types and edge weights information between two nodes. The five dimension represents the five social relations between characters and each edge constructs a vector:

$$\begin{aligned} \overrightarrow{h}_i &= \mathbf{W}^{\mathbf{c}}_{\mathbf{emd}} c_i, \ i \in [1, N^t_c], \\ \overrightarrow{e}^t_{ij} &= 5zeros(i = t, v = log(w^t_{ij})), \ t \in \mathcal{N}_t. \end{aligned} \tag{1}$$

where $5zeros(i, v)$ means a vector of 5 zeros and sets the i-th dimension with the value v, w^t_{ij} means the edge weight between character c_i and c_j with edge type t, $\mathcal{N}_t = \{ds, ta, ct, fr, tm\}$.

In order to obtain sufficient expressive power to transform the input features into high-level features, at least one learnable linear transformation is required. To that end, as an initial step, a shared

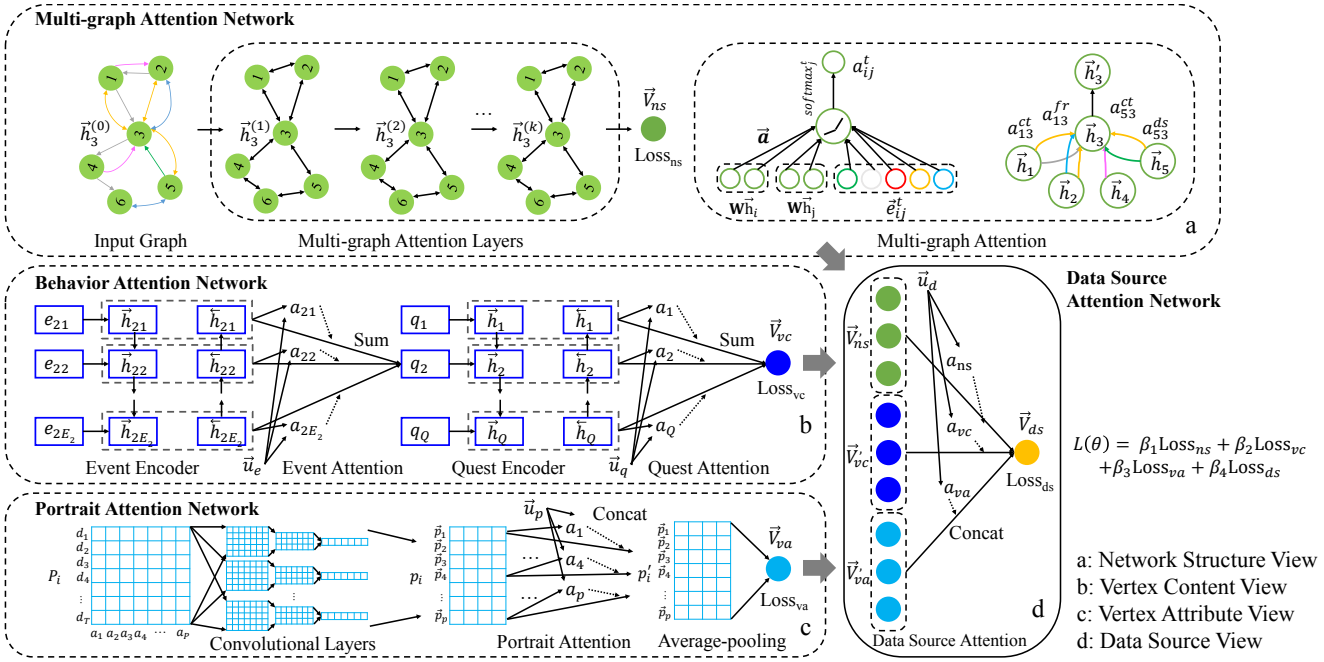


Figure 6: MVAN: Architecture of the proposed Multi-view Attention Networks.

linear transformation, parametrized by a weight matrix $\mathbf{W_g}$, is applied to every node. We then perform a self-attention on the nodes-a shared attention mechanism computes attention coefficients that indicate the importance of node c_j's features to node c_i. We inject the graph structure into the mechanism by performing masked attention-we only use the first-order neighbors $\mathcal{N}_i$ of c_i (including c_i):

$$a_{ij}^t = \frac{exp(\sigma(\overrightarrow{\mathbf{a}}^T[\mathbf{W_g}\overrightarrow{h}_i||\mathbf{W_g}\overrightarrow{h}_j||\overrightarrow{e}^t_{ij}]))}{\sum_{k\in\mathcal{N}_i,s\in\mathcal{N}_t} exp(\sigma(\overrightarrow{\mathbf{a}}^T[\mathbf{W_g}\overrightarrow{h}_i||\mathbf{W_g}\overrightarrow{h}_k||\overrightarrow{e}^s_{ik}]))}, \quad (2)$$

where σ is a nonlinear function like LeakyReLU, $.^T$ represents the transposition operation, $||$ is the concatenation operation and $\overrightarrow{\mathbf{a}}$ is a learnable context vector.

Once obtained, the normalized attention coefficients are used to compute a linear combination of the features corresponding them, to serve as the final output features for every node (after potentially applying a nonlinearity σ like tanh):

$$\overrightarrow{h}'_i = \sigma(\sum_{j\in\mathcal{N}_i,t\in\mathcal{N}_t} a_{ij}^t \mathbf{W_g}\overrightarrow{h}_j). \quad (3)$$

After K multi-graph attention layers, we get $\overrightarrow{V}_{ns} = \overrightarrow{h}'^{(k)}_i$.

3.2 Behavior Attention Network

Character behavior sequences are considered as the vertex content and there are many successful works on mining the sequence-shaped data like long-short term memory networks (LSTM)[7, 24]. We build a hierarchical attention mechanism named **b**ehavior **a**ttention **n**etwork (BAN) to capture the intrinsic hierarchical structure in behaviors as shown in Figure 6[b]. It consists of a quest extractor, a event sequence encoder, a event-level attention layer, a quest sequence encoder and a quest-level attention layer. We describe the details of different components in the following phases.

Quest Extractor: Characters often compete against and/or collaborate with each other to complete given missions (called *quest*) while playing an MMORPG and a quest is composed of continuous and time-dense event sequence. We treat the event interval list as one dimension input and apply DBSCAN[3] to cluster the behavior sequence into segments noted as quest sequences.

Event Encoder: Given a event sequence in a quest i with events $e_{ij}, j \in [1, E_i]$, we first embed the events to vectors through an embedding matrix $\mathbf{W^e_{emd}}$. We use a bidirectional LSTM to summarize information from both directions for events. The bidirectional LSTM contains a forward $\overrightarrow{\textbf{LSTM}}$ and a backward $\overleftarrow{\textbf{LSTM}}$. We obtain the state vector $\overrightarrow{h}'_{ij}$ for a given e_{ij} by concatenating the forward hidden state $\overrightarrow{h}_{ij}$ and backward hidden state $\overleftarrow{h}_{ij}$:

$$\begin{aligned}
\overrightarrow{x}_{ij} &= \mathbf{W^e_{emd}} e_{ij},\ j \in [1, E_i],\\
\overrightarrow{h}_{ij} &= \overrightarrow{\textbf{LSTM}}(\overrightarrow{x}_{ij}),\ j \in [1, E_i],\\
\overleftarrow{h}_{ij} &= \overleftarrow{\textbf{LSTM}}(\overrightarrow{x}_{ij}),\ j \in [E_i, 1],\\
\overrightarrow{h}'_{ij} &= [\overrightarrow{h}_{ij}||\overleftarrow{h}_{ij}].
\end{aligned} \quad (4)$$

Event Attention: Not all events contribute equally to the representation of the quest meaning. Hence, we introduce attention mechanism to extract such events that are important to the meaning of the quest and aggregate the representation of those informative events to form a quest vector. Specifically,

$$\begin{aligned}\overrightarrow{u}_{ij} &= \sigma(\mathbf{W_e}\overrightarrow{h}'_{ij} + \mathbf{b_e}),\\ a_{ij} &= \frac{exp(\overrightarrow{u}_{ij}^T\overrightarrow{\mathbf{u}}_\mathbf{e})}{\sum_{j=1}^{E_i} exp(\overrightarrow{u}_{ij}^T\overrightarrow{\mathbf{u}}_\mathbf{e})}\\ \overrightarrow{q}_i &= \sum_{j=1}^{E_i} a_{ij}\overrightarrow{h}'_{ij}.\end{aligned} \tag{5}$$

That is, we first feed the event annotation $\overrightarrow{h}'_{ij}$ through a one-layer MLP to get $\overrightarrow{u}_{ij}$ as a hidden representation of $\overrightarrow{h}'_{ij}$, then we measure the importance of the event as the similarity of $\overrightarrow{u}_{ij}$ with a event level context vector $\overrightarrow{\mathbf{u}}_\mathbf{e}$ and get a normalized importance weight a_{ij} through a softmax function.

Quest Encoder: After event encoder and event attention, we get a quest sequence with quests $\overrightarrow{q}_i, i \in [1, Q]$ in which Q stands for number of quests of samples. We use a bidirectional LSTM to encode the quests same as the event encoder:

$$\begin{aligned}\overrightarrow{h}_i &= \overrightarrow{\mathbf{LSTM}}(\overrightarrow{q}_i),\ i \in [1, Q],\\ \overleftarrow{h}_i &= \overleftarrow{\mathbf{LSTM}}(\overrightarrow{q}_i),\ i \in [Q, 1],\\ \overrightarrow{h}'_i &= [\overrightarrow{h}_i || \overleftarrow{h}_i].\end{aligned} \tag{6}$$

Quest Attention: To reward quests that are clues to correctly classify a character, we again use attention mechanism and introduce a quest level context vcector u_q and use the vector to measure the importance of the quests. This yields:

$$\begin{aligned}\overrightarrow{u}_i &= \sigma(\mathbf{W_q}\overrightarrow{h}'_i + \mathbf{b_q}),\\ a_i &= \frac{exp(\overrightarrow{u}_i^T\overrightarrow{\mathbf{u}}_\mathbf{q})}{\sum_{i=1}^{Q} exp(\overrightarrow{u}_i^T\overrightarrow{\mathbf{u}}_\mathbf{q})},\\ \overrightarrow{V}_{vc} &= \sum_{i=1}^{Q} a_i\overrightarrow{h}'_i.\end{aligned} \tag{7}$$

3.3 Portrait Attention Network

Characters share very different portraits information, so we propose a **p**ortrait **a**ttention **n**etwork (PAN) considering the vertex attribute view. As shown in Figure 6[c], at each timestamp t, we treat one charcater i with its historical portraits as one $T \times P$ image, where the size T controls the period granularity and P means number of portraits. As a result, we have an image as a tensor (having one channel) $P_i \in \mathbb{R}^{T\times P\times 1}$, for each character i and timestamp t. The PAN takes P_i as input $P_i^{(0)}$ and feeds it into K convolutional layers. The transformation at each layer k is defined as follows:

$$P_i^{(k)} = \sigma(P_i^{(k-1)} * \mathbf{W^{(k)}} + \mathbf{b^{(k)}}), \tag{8}$$

where $*$ denotes the convolutional operation and $\mathbf{W^{(k)}}, \mathbf{b^{(k)}}$ are two sets of parameters in the k^{th} convolution layer. Note that the width of all filters is set to P just like the way in natural language processing tasks and the parameters $\mathbf{W^{(0),\ldots,(K)}}$ and $\mathbf{b^{(0),\ldots,(K)}}$ are shared across all characters to make the computation tractable. After K convolution layers with C different filters, we get $p_i \in \mathbb{R}^{P\times C} = (\overrightarrow{p}_1, \overrightarrow{p}_2, \ldots, \overrightarrow{p}_p)$. Then an attention mechanism is applied on vertex attributes same to what we do above. Finanly we use a average-pooling layer to transform the output of portrait attention layer p'_i to a feature vector $\overrightarrow{V}_{va} \in \mathbb{R}^{P\times 1}$:

$$\begin{aligned}\overrightarrow{u}_i &= \sigma(\mathbf{W_p}\overrightarrow{p}_i + \mathbf{b_p}),\\ a_i &= \frac{exp(\overrightarrow{u}_i^T\overrightarrow{\mathbf{u}}_\mathbf{p})}{\sum_{i=1}^{P} exp(\overrightarrow{u}_i^T\overrightarrow{\mathbf{u}}_\mathbf{q})},\\ p'_i &= [a_i\overrightarrow{p}_i]_{||},\ i \in [1, P],\\ \overrightarrow{V}_{va} &= pool_{avg}(p'_i).\end{aligned} \tag{9}$$

where $[]_{||}$ means the concatenation operation for a list and $pool_{avg}$ means the average pooling operation.

3.4 Data Source Attention Network

Different data sources may be complemented to the character classification, but we don't know which data source contributes more. Given $\overrightarrow{V}_{ns}$, $\overrightarrow{V}_{vc}$ and $\overrightarrow{V}_{va}$, we propose a novel **d**ata **s**ource **a**ttention **n**etwork (DSAN) to capture inner relationship among them. Firstly, we use the single layer MLP to transform view vectors into the same dimision:

$$\overrightarrow{V}'_v = \mathbf{W_v}\overrightarrow{V}_v,\ v \in [ns, vc, va]. \tag{10}$$

Then we use an attention mechanism in the data source view and generate a_{ns}, a_{vc}, a_{va} for computing data source view vector $\overrightarrow{V}_{ds}$:

$$\begin{aligned}\overrightarrow{u}_v &= \sigma(\mathbf{W_d}\overrightarrow{V}'_v + \mathbf{b_d}),\ v \in [ns, vc, va],\\ a_{ns} &= \frac{exp(\overrightarrow{u}_{ns}^T\overrightarrow{\mathbf{u}}_\mathbf{d})}{exp(\overrightarrow{u}_{ns}^T\overrightarrow{\mathbf{u}}_\mathbf{d}) + exp(\overrightarrow{u}_{vc}^T\overrightarrow{\mathbf{u}}_\mathbf{d}) + exp(\overrightarrow{u}_{va}^T\overrightarrow{\mathbf{u}}_\mathbf{d})},\\ a_{vc} &= \frac{exp(\overrightarrow{u}_{vc}^T\overrightarrow{\mathbf{u}}_\mathbf{d})}{exp(\overrightarrow{u}_{ns}^T\overrightarrow{\mathbf{u}}_\mathbf{d}) + exp(\overrightarrow{u}_{vc}^T\overrightarrow{\mathbf{u}}_\mathbf{d}) + exp(\overrightarrow{u}_{va}^T\overrightarrow{\mathbf{u}}_\mathbf{d})},\\ a_{va} &= \frac{exp(\overrightarrow{u}_{va}^T\overrightarrow{\mathbf{u}}_\mathbf{d})}{exp(\overrightarrow{u}_{ns}^T\overrightarrow{\mathbf{u}}_\mathbf{d}) + exp(\overrightarrow{u}_{vc}^T\overrightarrow{\mathbf{u}}_\mathbf{d}) + exp(\overrightarrow{u}_{va}^T\overrightarrow{\mathbf{u}}_\mathbf{d})},\\ \overrightarrow{V}_{ds} &= [a_{ns}\overrightarrow{V}'_{ns} || a_{vc}\overrightarrow{V}'_{vc} || a_{va}\overrightarrow{V}'_{va}].\end{aligned} \tag{11}$$

3.5 Prediction Component and Loss Function

The high level representation vector $\overrightarrow{V}_{ns}, \overrightarrow{V}_{vc}, \overrightarrow{V}_{va}, \overrightarrow{V}_{ds}$ learned from different views can be used for RMT detection. We use the softmax function to output the prediction and the negative log likelihood of the correct labels as training loss:

$$\begin{aligned}\overrightarrow{Pre}^v &= softmax(\mathbf{W_v}\overrightarrow{V}_v + \mathbf{b_v}),\ v \in [ns, vc, va, ds],\\ Loss_v &= -\sum_{i=1}^{M} Y_i ln Pre_i^v,\ v \in [ns, vc, va, ds].\end{aligned} \tag{12}$$

where M represents the number of samples and Y_i means the real label of sample i.

We provide details about the loss function used for jointly training our propose MVAN. The loss function we used is defined as:

$$L(\theta) = \beta_1 Loss_{ns} + \beta_2 Loss_{vc} + \beta_3 Loss_{va} + \beta_4 Loss_{ds}, \tag{13}$$

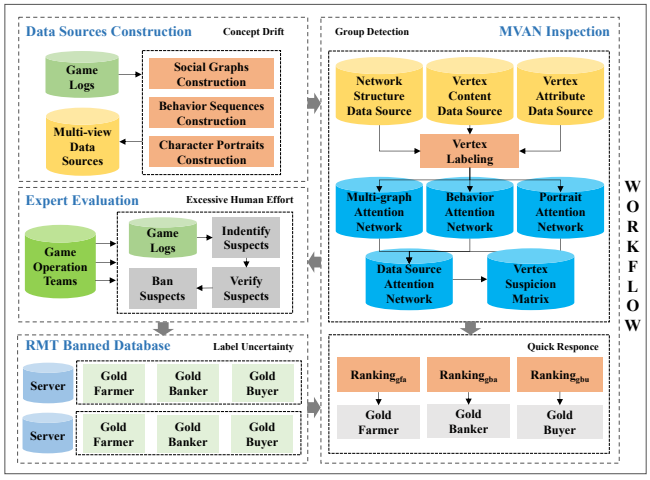


Figure 7: Workflow for RMT detection in online games

where θ are all learnable parameters in the MVAN and $\beta_{1,2,3,4}$ are hyper parameters. We use Tensorflow and Keras to implement our proposed model[3] and use Adam for optimization.

3.6 Workflow for RMT Detection

Our workflow for RMT detection is shown in Figure 7. It contains four modules: data sources construction, MVAN inspection, expert evaluation and RMT banned database.

- **Data sources construction:** It constructs social graphs, behavior sequences and character portraits from game logs *in a week window that updates daily*. The multi-view data sources are then feed into MVAN for generating suspecious RMT characters.
- **MVAN inspection:** We run MVAN with multi-view data sources from data sources construction module and RMTer labels from RMT banned database. Aiming at reducing the human labors in the succeeding manual inspection by MMORPG operators, we regard RMT detection as ranking all characters through estimating their likelihood of being an RMTer.
- **Expert evaluation:** The inspected results from MVAN inspection module are monitored and evaluated periodically by professional game operation teams. Banned once confirmed.
- **RMT banned database:** Once a character is determined to be a RMTer, operators ban all the characters operated by the same account with or without exhortations. Users can appeal against the banning if they are not involved in RMT. These banned and unbanned characters are all updated in the RMT banned database.

4 EXPERIMENTS

4.1 Experimental Settings

We partition the dataset and get the non-overlapped training (11.01-11.23), validation (11.24-11.30) and test set (12.01-12.31) respectively. The samples are constructed in a one-step sliding window of 7 days and labeled as RMTers or unlabeled characters. To counter the data imbalance problem, we upsample the gold banking and gold buying samples and construct regular samples by downsampling to the size of gold farming samples. In validation and test set, half of the labeled samples are selected for evaluation, while the other half are used to train the semi-supervised models. Most models are trained on the training set, and the optimal parameters are selected on the verification set. Some semi-supervised models directly verify the optimal parameters on the verification set, and all the model performances are compared on the test set.

[3]https://github.com/fuxiAIlab/MVAN

4.2 Evaluation Metrics

We measure the performance of RMT detection based on $precision_{avg}$, $recall_{avg}$ and $F1_{avg}$, which are defined as follows:

$$
\begin{aligned}
precision^d &= \frac{\#^d \ of \ correctly \ identified \ RMTers}{\#^d \ of \ characters \ identified \ as \ RMTers}, \\
recall^d &= \frac{\#^d \ of \ correctly \ identified \ RMTers}{\#^d \ of \ RMTers}, \\
F1^d &= \frac{2 \times precision^d \times recall^d}{precision^d + recall^d}, \\
m_{avg} &= avg(m^d), m \in \{precision, recall, F1\}, d \in [D_s, D_e],
\end{aligned}
\tag{14}
$$

Where d represents the specific date, D_s is the starting date and D_e is the ending date in the test set. The metrics are evaluated and compared separately for gold farmers, gold bankers and gold buyers on test dataset with different methods.

4.3 Baselines for comparison

We compare our proposed method with the following baselines. We tune all the parameters and then report the best performance.

- HCR_{opr}: Game operation teams detect real money trading with hand-crafted rules.
- VA_{cnn}: CNN without attention with the vertex attribute view.
- VA_{pan}: PAN with the vertex attribute view.
- VC_{nguard}: NGURAD[20] with the vertex content view.
- VC_{ban}: BAN with the vertex content view.
- NS_{n2v}: Node2vec[6] for learning vertex embeddings, concatenated for a classifier with the network structure view.
- NS_{gat}: GAT[23] for learning vertex embeddings, concatenated for a classifier with the network structure view.
- NS_{mgat}: MGAT for learning vertex embeddings, concatenated for a classifier with the network structure view.
- $ALL_{!dsan}$: Proposed MVAN without the data source view.
- ALL_{dsan}: Proposed MVAN with all four views.

4.4 Evaluation Results

4.4.1 Performance Comparison. Table 1 shows the evaluation metrics ($precision_{avg}|recall_{avg}|F1_{avg}$) of the proposed method as compared to all the other competing methods and the stand deviation is also provided. First of all, all machine learning methods show improvement to HCR_{opr}. Our proporsed ALL_{dsan} (MVAN) achieves the highest $precison_{avg}$, the highest $recall_{avg}$ and the highest $F1_{avg}$ for different kinds of characters among all the methods. It also has much relative improvement over the best performance among baseline methods with a single vertex attribute view (VA_{pan}), a single vertex content view (VC_{ban}) or a single graph

structure view (NS_{mgat}), which absolutely verifies our multi-view learning framework with multi-view data sources. More specifically, we can find that models with the attention mechanism present better performance than the models without the attention in the same view, which strongly indicates the importance and retionality of our attention mechanism based architecture.

Table 1: Performance comparison

Methods	Gold Farmer	Gold Banker	Gold Buyer
HCR_{opr}	39.43% ± 8.46%|30.17% ± 5.34%|34.02% ± 6.40%	31.09% ± 2.29%|12.63% ± 0.88%|17.96% ± 1.22%	18.52% ± 1.19%|17.90% ± 1.24%|18.20% ± 1.19%
VA_{cnn}	61.79% ± 4.58%|42.37% ± 3.85%|50.24% ± 4.10%	37.19% ± 3.04%|18.51% ± 4.96%|24.30% ± 4.51%	35.15% ± 2.51%|33.82% ± 1.94%|34.44% ± 2.00%
VA_{pan}	63.37% ± 5.23%|42.72% ± 4.11%|51.01% ± 4.48%	41.85% ± 2.31%|23.53% ± 4.12%|29.94% ± 3.77%	36.11% ± 2.19%|34.56% ± 1.65%|35.29% ± 1.66%
VC_{nguard}	76.75% ± 2.83%|64.45% ± 3.95%|70.03% ± 3.32%	55.13% ± 3.65%|41.72% ± 5.78%|47.31% ± 4.48%	22.88% ± 0.71%|21.20% ± 0.81%|22.01% ± 0.68%
VC_{pan}	78.57% ± 3.24%|67.43% ± 2.90%|72.55% ± 2.81%	55.83% ± 4.23%|43.13% ± 4.24%|48.58% ± 3.87%	24.39% ± 1.30%|24.78% ± 1.59%|24.58% ± 1.38%
NS_{n2v}	73.03% ± 1.08%|53.43% ± 2.20%|61.69% ± 1.59%	65.16% ± 3.03%|50.27% ± 4.80%|56.62% ± 3.49%	62.80% ± 0.95%|61.42% ± 1.31%|62.09% ± 0.91%
NS_{gat}	74.46% ± 1.02%|60.51% ± 1.45%|66.75% ± 1.14%	66.46% ± 3.98%|50.61% ± 3.60%|57.38% ± 3.06%	64.31% ± 0.97%|63.40% ± 1.21%|63.84% ± 0.77%
NS_{mgat}	76.31% ± 1.39%|60.76% ± 2.14%|67.63% ± 1.68%	70.59% ± 2.76%|55.30% ± 3.42%|61.94% ± 2.50%	66.71% ± 1.07%|65.73% ± 1.07%|66.21% ± 0.76%
ALL_{dsan}	87.50% ± 1.24%|76.71% ± 1.81%|81.74% ± 1.27%	74.69% ± 3.18%|60.07% ± 4.82%|66.45% ± 3.25%	74.82% ± 0.77%|67.46% ± 0.79%|70.94% ± 0.53%
ALL_{dsan}	**89.76% ± 0.80%|79.67% ± 1.30%|84.41% ± 0.78%**	**78.04% ± 4.30%|65.67% ± 3.78%|71.26% ± 3.41%**	**78.12% ± 1.00%|70.05% ± 0.95%|73.86% ± 0.74%**

4.4.2 Attention Weight Explanation. In order to illustrate the contexts dependency and importance captured by our proposed method, we extract the four-view average attention weights in ALL_{dsan} from different types of characters separately. The attention weights vary from 0 to 1 and are visualized in a matrix plot.

Vertex attribute view: In Figure 8(a), we can see that virtual money plays the most important role for detecting RMTers while gold farmers focus more on finishing game tasks, gold bankers ignore the gameplays and gold buyers charge real money for more game persuit. Regular characters are enjoying the gameplays and persuit the competion and fun in the virtual world.

Vertex content view: We label the events and quests into 8 categories separately in our crowdsourcing platform[4]. In Figure 8(b), RMTers perform more events related to money while gold farmers obtain more items and kill more monsters to obtain virtual money and gold bankers perform the world-wide activities to sell virtual money. We also find that different quests show different importance on classifying characters as shown in Figure 8(c), e.g., PvE quests for gold farmers, social quests for gold bankers, PvP quests for gold buyers and other enjoying quests for regular characters.

Network Structure view: In Figure 9(a), we find that the trade relation plays the most important role for detecting RMTers while the other relations play important roles for determining regular characters. Excluding trading, RMTers detection show some difference on relatively important social relations, e.g., the device-sharing relation for gold farmers, the chat relation for gold bankers and the friendship relation for gold buyers.

Data source view: In Figure 9(b), we conclude that different characters share very different data sources importance to classify them, e.g., VC for gold farmer detection, NS for gold banker and gold buyer detection, VA for confirming regular characters.

4.4.3 Industrial Evaluation. Our proposed method has been applied to JusticePC, Ghost II, Ghost Mobile and Revelation Online in NetEase. The game operation teams evaluated our identified suspects and the most recent reports show that we have achieved much better precision performance while covering much more suspicious characters compared with the former rule-based methods. It is confirmed that our method performs stably and detects RMTers as soon as possible which contributes to large amounts of RMB saving. The result is promising and yet to be further refined.

[4]https://zb.163.com/about

5 RELATED WORK

5.1 RMT Detection in Online Games

Real money trading detection methods have evolved over the years, and the literature on the problem can be classified into three generations. The first generation of such methods is signature-based, and utilizes client-side detection such as antivirus programs or CAPTCHA-based techniques[4, 5]. The second generation of methods focus on data mining techniques, and use server-side detection systems[8, 9, 13, 20, 21, 21], which focus mainly on distinguishing between anomalous characters and benign characters by analyzing server-side log files. Such techniques are widely used commercially and are coupled with logging techniques and various data mining algorithms for highly accurate detection. The third generation methods are a surgical strike policy[2, 10, 11]. They detect all industrialized GFGs by group assuming that members in a group have frequent interaction and abnormal patterns.

5.2 Attention Mechanism

Recently, attention mechanisms become popular and one of the benefits is that they allow for dealing with variable sized inputs, focusing on the most relevant parts to make decisions. Bahdanau *etal*. [1] first introduced a general attention model that did not assume a monotonic alignment. Later, researchers developed a number of multi-level attention-based models to select the relevant features and encoder hidden states in different applications, such as neural machine translation[1], recommendation systems[16], document classification[24], sentiment analysis[12] and others. When an attention mechanism is used to compute a representation of a single sequence, it is commonly refferd to as self-attention. Self-attention has proven to be useful together with Recurrent Neural Networks or convolutions and is also sufficient for constructing a powerful model obtaining state-of-art performance[22].

5.3 Multi-View Learning

Multi-view learning has been proposed to leverage the information from diverse domains or from various feature extractors, and combining the heterogeneous properties from different views can better characterize objects and achieve promising performance. There are three main kinds of solutions: early integration (Co-training), intermediate integration (subspace learning) and late integration(multiple kernel training). Early integration concatenates information from different views together and treats it as a single-view learning task. Co-EM[15], Bayesian Co-Training[25], and Co-Regression[26] belong to this category. Intermediate integration combines the information from different views at the feature level to form a joint feature space. MSNL[17], SM^2L[18] and aM^2L[14] fall into this category. Late integration method firstly builds individual models based on separate views and then combines these models[19].

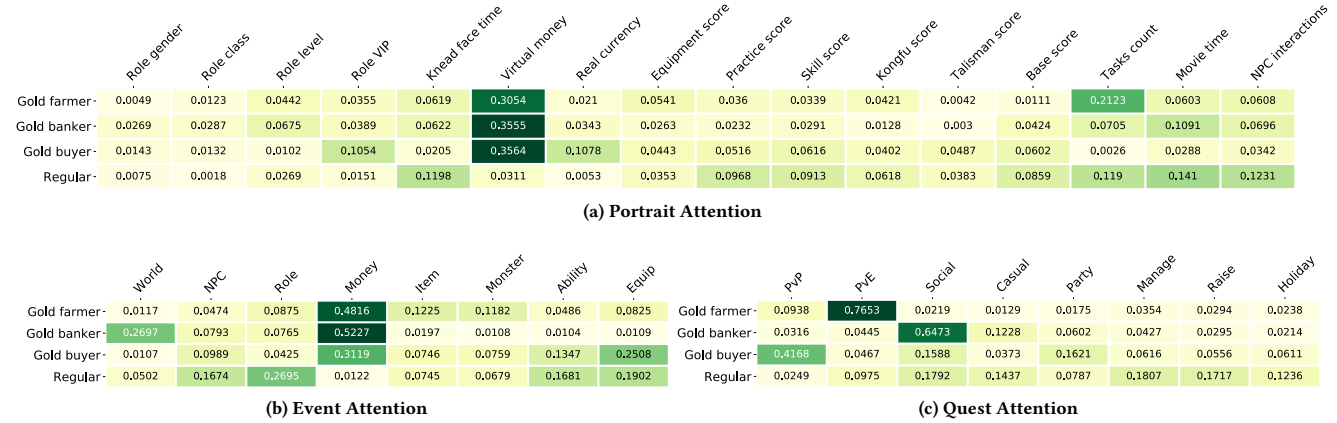


Figure 8: Attention weights visualization in the vertex attribute view and the vertex content view

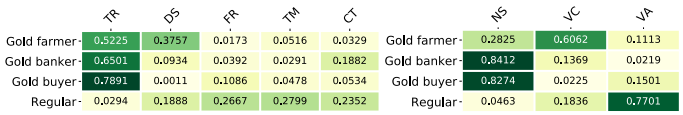


Figure 9: Attention weights visualization(NS and DS)

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a novel multi-view attention networks (MVAN) for real money trading detection in online games, which is the first to utilize the strong expressiveness of multi-view data sources. Specifically, MVAN consists of four network components with attention mechanisms from four different views. We evaluate MVAN on a dataset of JusticePC and the experimental results show the improvement of detecting RMTers significantly and the rationality of the attention mechanism architecture. What's more, MVAN has been implemented and deployed in multiple MMORPG productions in NetEase Games and received very positive reviews. Our promising future work is to deploy MVAN in other types of game productions in NetEase Games and generalize the multi-view attention networks to the other detection tasks.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv:1409.0473* (2014).

[2] Selin Chun, Deajin Choi, Jinyoung Han, Huy Kang Kim, and Taekyoung Kwon. 2018. Unveiling a socio-economic system in a virtual world: a case study of an MMORPG. In *Proceedings of the World Wide Web Conference on World Wide Web*.

[3] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. Density-based spatial clustering of applications with noise. In *KDD1996*, Vol. 240.

[4] Steven Gianvecchio, Zhenyu Wu, Mengjun Xie, and Haining Wang. 2009. Battle of botcraft: fighting bots in online games with human observational proofs. In *Proceedings of the 16th ACM conference on Computer and communications security*.

[5] Philippe Golle and Nicolas Ducheneaut. 2005. Preventing bots from playing online games. *Computers in Entertainment (CIE)* 3, 3 (2005), 3–3.

[6] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD2016*. ACM, 855–864.

[7] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[8] Ah Reum Kang, Huy Kang Kim, and Jiyoung Woo. 2012. Chatting pattern based game BOT detection: do they talk like us? *KSII Transactions on Internet & Information Systems* 6, 11 (2012).

[9] Ah Reum Kang, Jiyoung Woo, Juyong Park, and Huy Kang Kim. 2013. Online game bot detection based on party-play log analysis. *Computers & Mathematics with Applications* 65, 9 (2013), 1384–1395.

[10] Hyukmin Kwon, Aziz Mohaisen, Jiyoung Woo, Yongdae Kim, Eunjo Lee, and Huy Kang Kim. 2017. Crime Scene Reconstruction: Online Gold Farming Network Analysis. *IEEE Trans. Information Forensics and Security* 12, 3 (2017), 544–556.

[11] Eunjo Lee, Jiyoung Woo, Hyoungshick Kim, and Huy Kang Kim. 2018. No Silk Road for Online Gamers!: Using Social Network Analysis to Unveil Black Markets in Online Games. *arXiv preprint arXiv:1801.06368* (2018).

[12] Yunfei Long, Lu Qin, Rong Xiang, Minglei Li, and Chu-Ren Huang. 2017. A cognition based attention model for sentiment analysis. In *EMLP2017*. 462–471.

[13] Stefan Mitterhofer, Christopher Kruegel, Engin Kirda, and Christian Platzer. 2009. Server-side bot detection in massively multiplayer online games. *IEEE Security & Privacy* 7, 3 (2009).

[14] Liqiang Nie, Luming Zhang, Yi Yang, Meng Wang, Richang Hong, and Tat-Seng Chua. 2015. Beyond doctors: Future health prediction from multimedia and multimodal observations. In *MMM2015*. ACM, 591–600.

[15] Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *CIKM2000*. ACM, 86–93.

[16] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[17] Xuemeng Song, Liqiang Nie, Luming Zhang, Mohammad Akbari, and Tat-Seng Chua. 2015. Multiple social network learning and its application in volunteerism tendency prediction. In *SIGIR2015*. ACM, 213–222.

[18] Xuemeng Song, Liqiang Nie, Luming Zhang, Maofu Liu, and Tat-Seng Chua. 2015. Interest Inference via Structure-Constrained Multi-Source Multi-Task Learning.. In *IJCAI*. 2371–2377.

[19] Sören Sonnenburg, Gunnar Rätsch, and Christin Schäfer. 2006. A general and efficient multiple kernel learning algorithm. In *Advances in neural information processing systems*. 1273–1280.

[20] Jianrong Tao, Jiarong Xu, Linxia Gong, Yifu Li, Changjie Fan, and Zhou Zhao. 2018. NGUARD: A Game Bot Detection Framework for NetEase MMORPGs. In *KDD2018*. ACM, 811–820.

[21] Ruck Thawonmas, Yoshitaka Kashifuji, and Kuan-Ta Chen. 2008. Detection of MMORPG bots based on behavior analysis. In *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*. ACM, 91–94.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[23] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv:1710.10903* (2017).

[24] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *ACL2016*.

[25] Shipeng Yu, Balaji Krishnapuram, Rómer Rosales, and R Bharat Rao. 2011. Bayesian co-training. *Journal of Machine Learning Research* 12, Sep (2011).

[26] Zhi-Hua Zhou and Ming Li. 2005. Semi-Supervised Regression with Co-Training.. In *IJCAI*, Vol. 5. 908–913.

A NOTATION

Notation	Description
c_i, N_c^t	the i-th character, the number of characters at timestamp t
$\mathcal{N}_t, \mathcal{N}_i$	the edge types, the first-order neighbors of c_i including c_i
w_{ij}^t	the edge weight between character c_i and c_j with edge type t
$\overrightarrow{e}_{ij}^t$	the edge vector between character c_i and c_j with edge type t
e, q, E, Q	the event, the quest, the number of events, the number of quests
d, D_s, D_e	the specific date, the starting date, the ending date
$\mathbf{W}_{\mathbf{emb}}, (\mathbf{W}, \mathbf{b})$	the embedding matrix, the learnable transformation parameters
θ, β	the all learnable parameters, the hyper parameters
$a, \overrightarrow{\mathbf{a}} \overrightarrow{\mathbf{u}}$	the computed attention weight, the learnable context vector
$\sigma, .^T$	the nonlinear function, the transposition operation
$\|\|, []_{\|\|}$	the concatenation operation, the concatenation operation for a list
$*, pool_{avg}(.)$	the convolution operation, the average pooling operation
$5zeros(i, v)$	the vector of 5 zeros setting the i-th dimension with the value v
$avg(.), var(.)$	the average function, the variance function
$\overrightarrow{\mathbf{LSTM}}, \overleftarrow{\mathbf{LSTM}}, M, Y$	the bidirectional LSTM layer, the number of samples, the label of samples
VA, VC, NS, DS	vertex attribute, vertex content, network structure, data source
TR, DS, FR, TM, CT	transaction, device-sharing, friendship, team, chat

B MORE ABOUT RMT IN NETEASE GAMES

B.1 NetEase Games

Living up to the company's motto of "good games have no borders", NetEase Games has made great popularity and traction in domestic and overseas markets. In 2017, revenues of NetEase's online gaming segment reached $8.315 billion, up 29.67% year over year. NetEase Games continues to maintain high growth rate and deliver new hit games. To commit to the pursuit of the highest quality games and player experience, NetEase Games has developed and published dozens of popular games especially MMORPGs on pc and mobile, including JusticePC, Ghost II Online, Ghost Mobile, Fantasy Westward Journey Mobile, Fantasy Westward Journey Online etc.

B.2 Real Money Trading

Gold Farmers work full time playing online games in order to harvest virtual goods by using automated programs or by hiring low-cost laborers. Gold farmers work together as a team to accomplish challenging tasks such as finishing a sophisticated quest.
Gold Bankers gather virtual goods from the gold farmers in their own gold farming group and sell them to gold buyers. The banking characters possess most of the game money in the gold farming group, and focus on selling the game money for real money.
Gold Buyers who want to have high level characters easily purchase virtual money with real money from the gold bankers. Those players who do not follow typical steps cause rapid consumption of the game content, which shortens the game lifecycle.

B.3 RMT in NetEase MMORPGs

RMT in NetEase MMORPGs causes serious problems:

- **Imbalance of the virtual economy:** Dealing with a huge amount of virtual currency for RMT causes inflation in the virtual world and hampers the ability of regular players to perform ordinary economic activities.
- **Direct harm to regular players:** RMTers often directly harm other regular players by, for example, occupying specific locations for obtaining currency and items and attacking other players to rob their properties.
- **Encouragement of unauthorized deeds:** RMTers also perform dishonest actions, such as cheating, using bots, and even taking over characters by fraud.
- **Discouragement of honest players:** The unfair advantages of RMTers discourage honest players and prompts them to quit the game. It also prevents new players from joining the game.

Operators inspect characters in MMORPGs by the following steps:

- **Step 1. Identify suspects:** Operators identify characters suspected of involvement in RMT on the basis of, for example, tip-offs from general players or self-advertisements of RMTers.
- **Step 2. Verify each suspect:** Operators verify whether each suspect is an RMTer or not by referring to its previous actions and utterances in logs that are accumulated in the game server.
- **Step 3. Ban the account:** Once a character is determined to be an RMTer, operators ban the account with/without exhortations. Players are allowed to use multiple characters in MMORPGs, so all the characters operated by the same account are disabled.

C DATASET DETAILS

The dataset of JusticePC is now available for downloading[5].
We compare the five different social graphs shown in Table 2.

Table 2: Social Graphs Details

Graph	|V|	|E|	directionality	edge
transaction	49,704	826,106	directed	assets exchange
device-sharing	78,783	10,518,776,798	undirected	device usage
friendship	125,187	81,465,443	directed	friendship
team	114,861	5,729,270	undirected	teamwork
chat	39,999	1,786,275	directed	message delivery
total	227,148	10,608,583,892	both	multi-relations

Table 3 shows the detailed event and quest categories:

Table 3: Event Categories and Quest Categories Details

Category	Typical event examples	Category	Typical quest examples
world	login,logout,speak(world channel)	PvP	comparative conference
NPC	receive tasks,submit tasks,navigate	PvE	challenging missions
role	get married,get divorced,team	Social	marriage,master and apprentice
money	charge real money,trade virtual money	Party	party meeting, party missions
item	obtain items, consume items, sell items	Casual	fishing, imperial examination
monster	attack,attacked,kill,killed	Manage	home building
ability	level up,score increased	Raise	pet fostering, babe rearing
equip	forging weapons,washing attributes	Holiday	guess riddle, dragon boating

Table 4 shows the volume of corresponding records in the log data and distribution of manually identified RMTers.

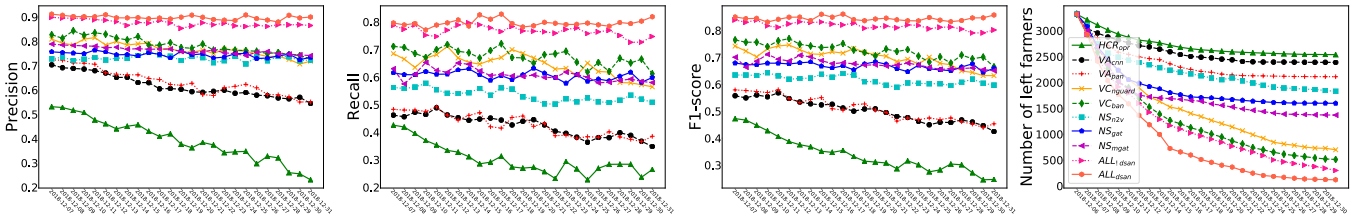
D DETAILED EXPERIMENTAL SETTINGS

In this section, we show the detailed experimental settings:
$\mathbf{VA_{cnn}}$: the number of convolutional layers is 3, all the filters' parameters are (16*3*P), one dense layer (8 neurons) with softmax.
$\mathbf{VA_{pan}}$: the same settings as $\mathbf{VA_{cnn}}$, with an additional self-attention layer set $\overrightarrow{u}_p \in \mathbb{R}^{1\times 16}$.

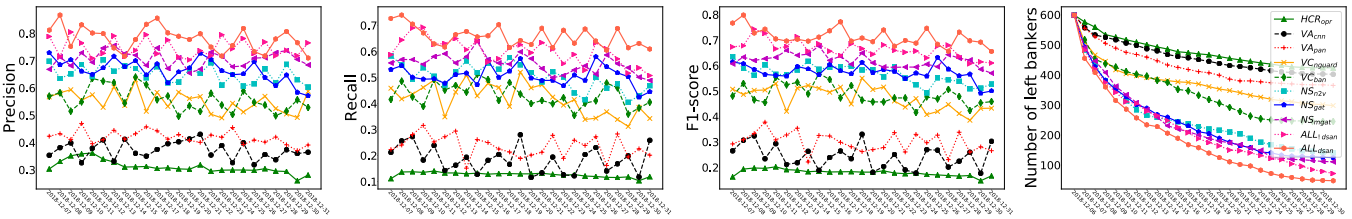
[5]https://github.com/fuxiAIlab/JusticePC

Table 4: Statistics of the dataset segmentation.

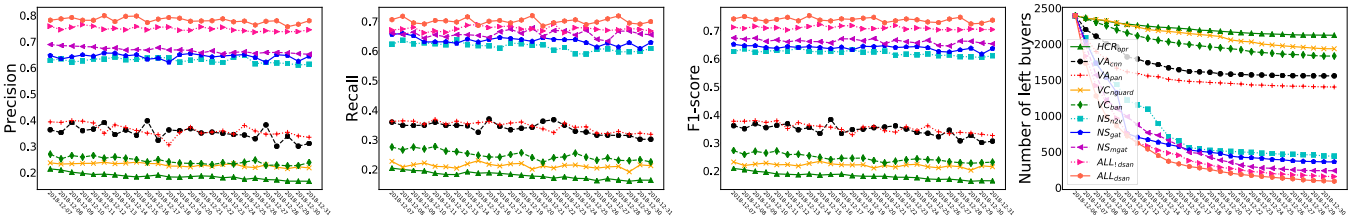
Segmentation	Period	Characters	Identified RMTers				Transaction		Device-sharing		Friendship		Team		Chat	
			Total	Gold Farmer	Gold Banker	Gold Buyer	|V|	|E|	|V|	|E|	|V|	|E|	|V|	|E|	|V|	|E|
Training	11.01 - 11.23 (23 days)	175,296	10,699	7,800	642	2369	28,195	367,291	45,110	882,943,165	114,880	29,588,199	70,038	2,159,478	24,385	678,729
Validation	11.24 - 11.30 (7 days)	143,943	6,830	3,972	675	2,296	19,884	116,520	51,628	737,829,127	115,588	9,260,096	25,740	627,650	16,318	205,638
Test	12.01 - 12.31 (31 days)	195,022	6,205	3,322	599	2,395	29,685	416,129	78,348	8,898,004,506	125,104	42,617,148	76,192	2,942,142	32,392	901,908



(a) Gold Farmer



(b) Gold Banker



(c) Gold Buyer

Figure 10: RMT detection performance comparison over dates

$\mathbf{VC_{nguard}}$: one bidirectional LSTM layer (32 neurons), one self-attention layer, one dense layer (32 neurons) with softmax.
$\mathbf{VC_{ban}}$: the same setting as $\mathbf{VC_{ngurad}}$, except the hierarchical self-attention setting $\overrightarrow{u}_e \in \mathbb{R}^{1\times 64}$, $\overrightarrow{u}_q \in \mathbb{R}^{1\times 64}$.
$\mathbf{NS_{n2v}}$: the characters embedding size is 32, p is 1, q is 1/4, one dense layer (16 neurons) with softmax.
$\mathbf{NS_{gat}}$: the characters embedding size is 32, four graph attention layers, no multi-heads, one dense layer (64 neurons) with softmax.
$\mathbf{NS_{mgat}}$: the same settings as $\mathbf{NS_{mgat}}$, except the multi-graphs attention layer settings.
$\mathbf{ALL_{!dsan}}$: the same settings as $\mathbf{VA_{pan}}$, $\mathbf{VC_{ban}}$ and $\mathbf{NS_{mgat}}$, the size of $\overrightarrow{V}'$ is 32, one dense layer (64 neurons) with softmax.
$\mathbf{ALL_{dsan}}$: the same settings as $\mathbf{ALL_{!dsan}}$, with an additional self-attention layer set $\overrightarrow{u}_d \in \mathbb{R}^{1\times 32}$.
For all models, dropout with ratio 0.2 is used. The models are trained with batch size 64 and early stop is performed by monitoring the validation error.

E ADDITIONAL EXPERIMENTAL RESULTS

Concept drift solution: From Figure 10 we can find that the performance of most baseline methods gradually decreases over time, especially on detecting gold farmers. However, our proposed method has always maintained the best results, while maintaining the relative stability of the performance.
Quick response solution: From Figure 10 we can see that our method can find RMTers much quicker and more comprehensive for game operation teams to take operating measures as soon as possible. This is really important for reducing game loss.

Game	Pre_{gf}	Gain_{gf}	Pre_{gb}	Gain_{gb}	Pre_{gb}	Gain_{gb}
JusticePC	90.17%	35.98%	80.24%	50.43%	80.66%	54.89%
Ghost II	91.56%	38.89%	81.58%	52.69%	83.90%	57.87%
Ghost Mobile	88.76%	31.94%	76.85%	48.74%	76.08%	50.15%
Revelation Online	90.65%	36.58%	82.38%	53.31%	81.49%	57.82%