MOBA Game Item Recommendation via Relation-aware Graph Attention Network

Lijuan Duan*
Faculty of Information Technology
Beijing University of Technology
China National Engineering Laboratory for Critical Technologies of Information Security Classified Protection
Beijing, China
ljduan@bjut.edu.cn

Shuxin Li
Faculty of Information Technology
Beijing University of Technology
Beijing Key Laboratory of Trusted Computing
Beijing, China
lisx@emails.bjut.edu.cn

Wenbo Zhang
Faculty of Information Technology
Beijing University of Technology
Beijing Key Laboratory of Trusted Computing
Beijing, China
zhangwenbo@bjut.edu.cn

Wenjian Wang
Faculty of Information Technology
Beijing University of Technology
China National Engineering Laboratory for Critical Technologies of Information Security Classified Protection
Beijing, China
wangwj@emails.bjut.edu.cn

***Abstract*—Recommender systems based on graph attention networks have received increasing attention due to their excellent ability to learn various side information. However, previous work usually focused on game character recommendation without paying much attention to items. In addition, as the team of the match changes, the items used by the characters may also change. To overcome these limitations, we propose a relation-aware graph attention item recommendation method. It considers the relationship between characters and items. Furthermore, the graph attention mechanism aggregates the embeddings of items and analyzes the effects of items on related characters while assigning attention weights between characters and items. Extensive experiments on the kaggle public game dataset show that our method significantly outperforms previous methods in terms of Precision, F1 and MAP compared to other existing methods.**

Keywords—MOBA game, deep learning, item recommendation, graph attention network

I. INTRODUCTION

Multiplayer Online Battle Arena (MOBA) games have developed rapidly in recent years and have become an important part of online games [1]. League of Legends (LOL), as a MOBA type competitive online game developed by American Riot Games Company, has formed a unique e-sports culture. Although there are many factors that determine the outcome of the game, according to the research [2], the top two influencing factors are the character selection in the team and the character's item selection. Up to now, the research on character recommendation [3][4] in MOBA type games has been relatively in-depth, and a variety of character team matching systems [5][6] have been launched, but there are still few researches on item matching recommendation for characters in the team. Especially for novice players, how to choose the appropriate item according to the character's team is more complex, and it is also a major factor affecting the victory of the game. Therefore, it is of certain practical significance to study the character's item recommendation in the team.

The LOL has hundreds of personalized characters, as well as characteristic cultivation systems such as ranking system and rune system [7]. Each game is divided into two teams. Each team is composed of characters (also known as "heroes") controlled by five players (also known as "summoners"), and each team is composed of five summoners. Summoners need to strengthen their controlled characters through role upgrade, skill upgrade or upgrade purchase item. Through mutual assistance and cooperation with other teammates in the team, attack the hero, army, crystal tower and other defense structures of the enemy team, and finally push down the opponent's main base to win the game.

Traditional recommendation systems [8] cannot make full use of various forms of data. They usually focus on one or several specific data sources and ignore many valuable context information, resulting in sub optimization of recommendation system performance. The existing item recommendation research [9] uses Transformer [10] for coarse-grained feature extraction, only considers the single attribute relationship between team and character, but ignores the rich heterogeneous interaction hidden behind it.

To solve the above problems, our contributions are summarized as follows:

- We propose a new MOBA item recommendation model that extracts and fuses information by using a relational perceptual graph attention network architecture.
- The relation-aware module (RA) we proposed constructs the relationships between heroes through the interaction of their personalization features and location features. The graph attention fusion module (GAF) learns the attention weights in the graph structure and continuously updates the hidden layer representation between the aggregated hero and item.
- Obtained excellent performance using our approach on two Kaggle public dataset [11][12].

II. RELATED WORK

Recommendation methods based on graph attention networks have become the state-of-the-art in recommender

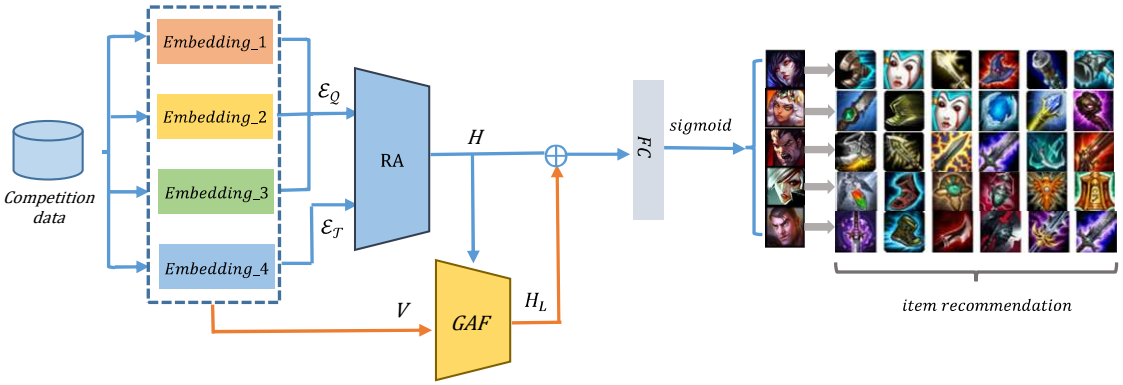

Fig. 1. The framework of Relation-aware Graph Attention Network

systems due to their advantages in processing structured data and mining contextual information.

Wang et al.[13] proposed a Disentangled Heterogeneous Graph Attention Network (DisenHAN) for top-N recommendation, which uses meta-relations to decompose higher-order connectivity between pairs of nodes, and decouples user and item representations to aggregate the corresponding aspect features. Lim et al.[14] proposed a Spatial-Temporal-Preference User Dimensional Graph Attention Network (STP-UDGAT), which exploits personalized user preferences and explores new Point-of-Interest (POI) in the global spatial-temporal-preference (STP) neighborhood. Song et al.[15] proposed a new recommendation method based on dynamic graph attention networks to model dynamic user interests and context-dependent social influences. The method uses a bilinear function to capture feature correlations between items, which can simultaneously learn user dynamic interests and social influence. Liu et al.[16] proposed a user-specific graph attention mechanism to aggregate local contextual information in knowledge graphs for recommendation. Which utilizes contextual information extracted from users' historical behavior data to simulate users' preferences for items. Xie et al.[17] proposed a novel heterogeneous graph neural network framework (GraphDR) for diverse recommendation to improve the accuracy and diversity of recommendations. GraphDR builds a huge heterogeneous preference network to record different types of user preferences, and conducts a field-level heterogeneous graph attention network for node aggregation. Wu et al.[18] proposed dual-graph attention networks to collaboratively learn representations for dual social effects, where one network is modeled by user-specific attention weights and the other by dynamic and context-aware modeled attention weights. At the same time, we investigate the underlying structural feature relationships in the information. Applying a novel relation-aware method and graph attention mechanism to MOBA game item recommendation.

III. METHODOLOGY

In this section, we first introduce the overall framework for MOBA item recommendation based on relational graph attention networks. Then, a relation-aware module for obtaining the representation of the relationship between hero and item nodes is introduced. And we describe the graph attention fusion module, which recommends suitable item according to the different degrees of influence between heroes. Then we introduce the loss function.

A. Overall architecture

In order to reduce the information loss in the multi-attribute feature encoding process, the data attribute information is mapped into dense feature vectors through the embedding layer, and the latent relationship features between the embedded attribute entities are preserved. First, the input data of each game is sent into four types of embeddings, which embed the characteristics of role, type, item, and team. The input data for each match includes four attributes including hero role, hero type, used items and hero team.

Then, splicing the results of the embedding of hero role, hero type and item to get each hero's personality embedding matrix $\mathcal{E}_Q = \{b_{role}, b_{type}, b_{item}\}, \mathcal{E}_Q \in \mathbb{R}^{N\times D}$, and the team embedding $\mathcal{E}_\mathcal{T} = \{b_{team}\}, \mathcal{E}_\mathcal{T} \in \mathbb{R}^{N\times D}$ represents the location characteristics of the hero. These two types of embeddings are used as the input of the relationship perception module, according to the relationship perception. The spatial features and attribute features obtained by the module are used to construct the later relational graph structure. Then, using the relational graph structure through the graph attention network, the representation of each node is refined to obtain team-level relationship-aware features. After the fully connected layer and sigmoid activation layer, the character item vector is output, and then through the softmax layer, the final output is the item recommendation result that best matches each character in the winning team. As shown in Figure 1.

B. Relation-aware module (RA)

In the relationship-aware module, the multi-head attention mechanism of the transformer encoder is used to aggregate the representation of related features. At the same time, the relationship-aware module is designed considering the positional relationship of the team to which the hero belongs and the dependencies between heroes. As shown in Figure 2.

The $\mathcal{E}_Q \in \mathbb{R}^{N\times D}$ and $\mathcal{E}_\mathcal{T} \in \mathbb{R}^{N\times D}$ are respectively sent to the transformer encoder in the relationship perception module, and the feature extraction for different attributes is realized by stacking the attention blocks to obtain the self-feature H_Q and position feature $H_\mathcal{T}$ of the N heroes.

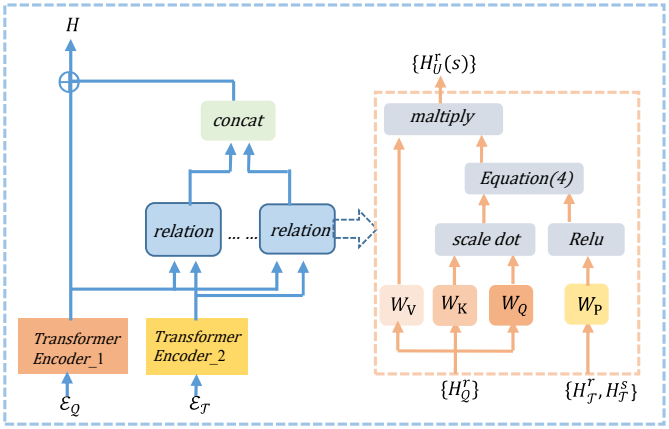

Fig. 2. The framework of RA module

$$H_Q = Transformer\ Encoder(\mathcal{E}_Q; \theta^{(Q)}), H_Q \in \mathbb{R}^{N\times D} \quad (1)$$

$$H_\mathcal{T} = Transformer\ Encoder(\mathcal{E}_\mathcal{T}; \theta^{(\mathcal{T})}), H_\mathcal{T} \in \mathbb{R}^{N\times D} \quad (2)$$

Among them, N is the number of heroes in a team, D is the attribute encoding dimension, and $\theta^{(Q)}$ is the trainable parameter of the transformer encoder for the Q feature and $\theta^{(\mathcal{T})}$ is the trainable parameter of the transformer encoder for the $\mathcal{T}$ feature.

The feature set of N heroes $\{(H_Q^s, H_\mathcal{T}^s)\}_{s=1}^N$ is sent to the relation-aware module for feature extraction based on global relationship. Among them, the calculation formula of the relationship feature $H_U(s)$ relative to the *s-th* hero is:

$$H_U(s) = \sum_r \varphi^{rs}(W_v \cdot H_Q^r) \quad (3)$$

$H_U(s)$ is used to represent the relationship between the *s-th* hero and the rest of the heroes in the match.

The number of relation sub-parts of the relation-aware module is N, and each relation module uses the two-part feature of the relevant hero as input, obtains different relation features, then fused with the hero's original features through the concat operation to get the final relation feature. H_Q^r represents the own characteristics of the hero r in the game, W_v is a linear transformation operation, and the relationship weight φ^{rs} represents the influence from other heroes, which is realized by the following formula:

$$\varphi^{rs} = \frac{\varphi_\mathcal{T}^{rs} \cdot \exp(\varphi_Q^{rs})}{\sum_x \varphi_\mathcal{T}^{xs} \cdot \exp(\varphi_Q^{xs})} \quad (4)$$

The φ^{rs} in the formula is mainly determined by the variables $\varphi_\mathcal{T}^{rs}$ and φ_Q^{rs}. $\Phi_\mathcal{T}^{rs}$ represents the weight of hero r in the position feature of current hero s, Φ_Q^{rs} represents the feature weight between hero r and hero s for their own influence. their formulas are as follows:

$$\varphi_Q^{rs} = \frac{dot(W_K H_Q^r, W_Q H_Q^s)}{\sqrt{d_x}} \quad (5)$$

$$\varphi_\mathcal{T}^{rs} = Relu(W_P \cdot (H_\mathcal{T}^r, H_\mathcal{T}^s)) \quad (6)$$

Among them, W_K, W_Q and W_P represent the linear transformation matrix, dot(·) represents dot product operation, and d_x is the dimension after dot product. After obtaining a single relation feature $H_U(s)$ and fusing multiple relation features, then fused with the hero's own feature H_Q^s to obtain the final feature representation $H \in \mathbb{R}^{N\times D}$ of each match.

$$H = H_Q^s + concat[H_U^1(s), \ldots, H_U^N(s)] \quad (7)$$

C. *Graph Attention Fusion module (GAF)*

This module uses graph attention network (GAT) for feature fusion of global hero item information. Unlike graph convolutional network, graph attention network allows assigning different importance to related heroes and items nodes, making the model capacity soared. As shown in Figure 3.

The feature expression of a game includes the global comprehensive feature $H = [h_1, h_2 \ldots h_N] \in \mathbb{R}^{N\times D}$ of two teams with N heroes, each hero as a node, the feature vector $h_i \in \mathbb{R}^D$ represents the comprehensive feature of the *i-th* hero node, D is the feature matrix dimension , H as the feature matrix part of the input graph attention network.

Since the feature aggregation is performed by multiple sub-layers in the graph attention network, after inputting H into the graph attention network, the features are aggregated through the *l-th* sublayer inside the network to obtain $H_l = [h_l^1, h_l^2 \ldots h_l^N] \in \mathbb{R}^{N\times D}$. And h_l^i represents the feature aggregated by hero node i at layer l. The relationship between heroes is used as an edge, and the relationship between the edges is calculated by $V = (\mathcal{E}_Q + \mathcal{E}_\mathcal{T}) \cdot (\mathcal{E}_Q + \mathcal{E}_\mathcal{T})^T$, $V \in \mathbb{R}^{N\times N}$, and it is used as the correlation matrix part input to the graph attention network. Then, the influence weight of the hero node i in the *l-th* layer based on the attention mechanism c is calculated by following equation:

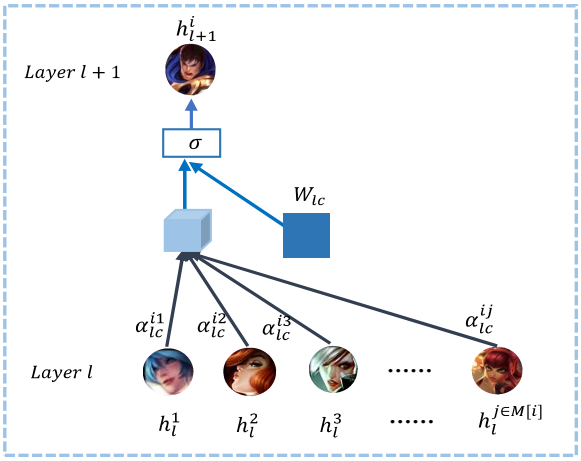

Fig. 3. The schematic diagram of the *l-th* sublayer in the GAF module, which represents the feature aggregation between heroes according to the learned weights α_{lc}^{ij}.

$$\alpha_{lc}^{ij} = \frac{exp(g(a_{lc}^{T}[W_{lc}h_l^i \,||\, W_{lc}h_l^j]))}{\sum_{t\in \mathrm{M[i]}} exp(g(a_{lc}^{T}[W_{lc}h_l^i \,||\, W_{lc}h_l^t]))} \quad (8)$$

Where || represents the operation of connecting multiple vectors, α_{lc}^{ij} represents the attention of hero node *i* on its associated node *j* based on *c*, $c \in (role, type, item, team)$, the result is the influence factor weight value of the *l-th* sublayer. The number of attention heads *C* of graph attention network is set to 4. Indicates that the comprehensive feature expression *H* of each game is divided into four categories: role, type, item and team.

Which is used to focus on the target aggregation of attention. *M*[*i*] represents the set of hero nodes associated with hero node *i*, and obtains from the correlation matrix *V* whether each hero node is related. $W_{lc} \in \mathbb{R}^{\frac{\mathrm{D}}{\mathrm{C}}\times \mathrm{D}}$ is the linear transformation matrix of the *c-th* attention mechanism. g(·) is the leakyrelu nonlinear function, and $a_{lc} \in \mathbb{R}^{\frac{2\mathrm{D}}{\mathrm{C}}}$ is an attention vector learned during training. The specific process of updating the aggregated features of hero node *i* in the *l+1th* sublayer based on *C* attention mechanisms is shown in the following formula:

$$h_{l+1}^i = ||_{c=1}^{C} \sigma(\sum_{j\in \mathrm{M[i]}} \alpha_{lc}^{ij} W_{lc}\, h_l^j) \quad (9)$$

h_l^j represents the feature of hero node *j* associated with hero *i* in the *l-th* sublayer, σ represents the sigmoid function, W_{lc} is the linear transformation matrix of the *c-th* attention mechanism. h_{l+1}^i is the feature representation of the hero node *i* after the update aggregation in the *l+1th* sublayer of the graph attention network. Therefore, the characteristics of all hero nodes in sub-layer *l+1th* of the graph attention network are aggregated. The graph attention network in a game is expressed as:

$$H_{l+1} = [h_{l+1}^1, h_{l+1}^2 \ldots h_{l+1}^N], H_{l+1} \in \mathbb{R}^{\mathrm{N}\times \mathrm{D}} \quad (10)$$

The input of the graph attention network is H_{l+1} and *V*, and the final output feature contains the comprehensive feature expression H_L between heroes and item, which is expressed as follows:

$$H_L = GAT(H_{l+1}, V; \Theta_l) \quad (11)$$

In this module, the total number of sub-layers of the graph attention network is set to *L=3*. That is, the first sub-layer mainly aggregates the relationship features between heroes within the team. The second sub-layer mainly aggregates the features of the hero relationship between the two teams. The third sub-layer mainly aggregates the relationship between heroes and items. After the update aggregation of the *L* layer, $H_L \in \mathbb{R}^{\mathrm{N}\times \mathrm{D}}$ is obtained from H_{l+1} and *V*. Θ_l is the parameter set of the graph attention network sublayer located in the *l-th* layer.

Then, concatenate *H* and H_L to get the global comprehensive feature expression H' of heroes and items in a game.

$$H \oplus H_L = H', H' \in \mathbb{R}^{\mathrm{N}\times \mathrm{D}} \quad (12)$$

Next, through the fully connected layer FC_1 and the sigmoid function to calculate the prediction score *Y* of whether the team wins or not, which is used to estimate the possibility of team victory. The formula is as follows:

$$Y = sigmoid(FC_1(H^{'})) \quad (13)$$

According to this, through the fully connected layer FC_2 and softmax, finally output the probability value *P* of each item selected by each character in the victorious team. Recommending items to the winning team can optimize the model. The formula is as follows:

$$P = softmax(FC_2(H'')) \quad (14)$$

Among them, H'' means that only the results of the winning team in H' is retained according to *Y*. Therefore, according to the actual needs, a series of items with the highest probability value is recommended for each character in the winning team.

D. Loss functions

The overall loss L_{total} of the model consists of two independent subtask loss functions, including the team winning prediction loss L_{win} and the item recommendation prediction loss L_{items}. Where μ and γ represent the two hyperparameters of the two sub-loss functions respectively.

$$L_{total} = \mu L_{win} + \gamma L_{items} \quad (15)$$

We use the cross entropy loss function to calculate the loss of whether the team wins or not, y_i represents the real label, and $\hat{y}_i$ represents the predicted value of whether the team wins or not. The formula is as follows:

$$L_{win} = -\sum_i [y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)] \quad (16)$$

In the prediction loss of item recommendation, we use the binary cross entropy loss function, which is defined as L_{items}. p_j represents the item label used by the hero in the sample data, and $\hat{p}_j$ represents the recommended item output by our network model.

$$L_{items} = -\frac{1}{N}\sum_{j=1}^{N}\left[p_j \cdot log\left(\hat{p}_j\right) + \left(1 - p_j\right) \cdot log\left(1 - \hat{p}_j\right)\right] \quad (17)$$

IV. EXPERIMENTAL RESULTS

A. Dataset

In this section, we validate the proposed method using two datasets related to MOBA-type games, namely the League of Legends (LOL) Season 7th dataset [11] and the Dota 2 dataset [12] published by Kaggle. The two datasets are divided into training and test sets in a ratio of 4:1. The final LOL dataset contains 157,584 games, the training set contains 126,128 games, and the test set contains 31,456 games. The final Dota 2 dataset has 50,000 games, the training set contains 40,000 games, and the test set contains 10,000 games. After data processing, the data of each game includes multiple attributes of

hero role, hero type, used items, and hero team. In addition, there are also item labels and the label of whether the team wins or not. A preview of the overall dataset is shown in the Table I.

TABLE I. OVERVIEW OF TWO DATASET

	LOL Ranked Matches 7th Season	**Dota 2 Ranked Matches**
Matches	157,584	50,000
Champions	136	113
Roles	5 (Top, Mid, Bot, Juggle, Support)	5 (Top, Mid, Juggle, Carry, Support)
Types	6 (Tank, Warrior, Assassin, Mage, Archer, Sup)	6 (Tank, Warrior, Assassin, Mage, Archer, Auxiliary)
Items	89	101
Train/Test	126,128 / 31,456	40,000 / 10,000

B. Experimental details and effects

We set the embedding dimension D of the embedding layer is 512. In the relation-aware module, the number of multi-heads in the transformer encoder is set to 4, and the number of encoder layers in the encoder is set to 2. In the graph attention fusion module, the number of graph attention sublayer is set to 3, the number of hidden layers is set to 256, and the dropout is set to 0.5. During the overall framework training, the batchsize is set to 100 and the learning rate is 3e-5. Using Precision (Precision@k), F1-Score (F1@k), Mean Average Precision (MAP@k) and Recall (Recall@k) these four evaluation indicators. The recommendation result returned by the model is an ordered item recommendation list. Among them, the length of the recommendation list is represented by k. Since the number of optimal item for each character of MOBA game is 6 at one time, the index results of $k = 6$ can more accurately evaluate the model. At the same time, changing the length of the recommendation list to $k = 1$ and $k = 10$ not only verifies the performance of the model from many aspects, but also provides effective guidance for the character to accurately recommend individual item and expand the optional range of item.

We compared our model with the baseline methods such as Decision Tree (D-Tree), Logistic regression (Logit), Artificial Neural Network (ANN), CNN and TTIR (Team-aware Transformer-based Item Recommendation). In order to ensure the fairness of the experiment, we used the same dataset as other methods for the experiment. The experimental results are shown in Table II, in which some comparative experimental results are refered from the paper [9].

The results in Table II show that compared with the best baseline TTIR[9], our method achieves higher results in Precision@k. When $k=6$, the best improvement effect of Precision is 6.1%. Precision increased by 3.3% and 5.4% respectively when k=1 and k=10. The MAP@k are also improved. When k is 1,6 and 10, it is increased by 3.3%, 2.2% and 1.5% respectively. The Recall@k is close to the best baseline recall value. Due to the contradiction of sometimes negative correlation between Precision and Recall indicators, and when the recommendation list k length is long, the item with weak matching with the character may also appear in the recommendation list, the result of the Recall@k metric will be affected. Therefore, we use F1@k to comprehensively measure Precision and Recall indicators, which are increased by 3.0% and 5.0% respectively on k=6 and k=10, which proves that the comprehensive performance of our model is improved.

In order to verify the generalization performance of the model in other game scenes, the best baseline method TTIR and our method are migrated to Dota 2 dataset for experiments. The experimental settings and evaluation indicators remain unchanged. The experimental results are shown Table III.

As can be seen from Table III, our method achieves the best performance on Precision@k, MAP@k, Recall@k and F1@k (k=1, 6, 10) metrics compared to the best baseline TTIR method. For Precision@k, it improves by 1.2%, 2.4%, and 2.5% when k=1, 6, and 10, respectively. For MAP@k, it improves by 1.2%, 0.6% and 0.9% when k=1, 6, 10, respectively. For Recall@k, the improvement is at most 2.3% when k=6. For F1@k, the improvement is at most 2.4% when k=10. The experimental results on the Dota 2 dataset show that

TABLE II. RESULTS FOR TOP @K RECOMMENDATION OF SIX METHODS ON LOL DATASET

Method	**Precision@k**			**MAP@k**			**Recall@k**			**F1@k**		
	@1	***@6***	***@10***	***@1***	***@6***	***@10***	***@1***	***@6***	***@10***	***@1***	***@6***	***@10***
D-Tree	0.516	0.319	0.204	0.516	0.648	0.636	0.135	0.491	0.520	0.210	0.379	0.289
Logit	0.672	0.393	0.285	0.672	0.714	0.672	0.178	0.607	0.726	0.277	0.468	0.403
ANN	0.771	0.476	0.341	0.771	0.785	0.743	0.205	0.732	0.864	0.318	0.566	0.481
CNN	0.790	0.484	0.348	0.790	0.795	0.754	0.209	0.744	0.882	0.331	0.586	0.499
TTIR	0.803	0.492	0.351	0.803	0.805	0.764	**0.214**	**0.756**	**0.889**	**0.338**	0.596	0.503
Ours	**0.836**	**0.553**	**0.405**	**0.836**	**0.827**	**0.779**	0.185	0.721	0.873	0.302	**0.626**	**0.553**

TABLE III. RESULTS FOR TOP @K RECOMMENDATION ON DOTA 2 DATASET

Method	**Precision@k**			**MAP@k**			**Recall@k**			**F1@k**		
	@1	***@6***	***@10***	***@1***	***@6***	***@10***	***@1***	***@6***	***@10***	***@1***	***@6***	***@10***
TTIR	0.579	0.310	0.224	0.579	0.647	0.613	0.106	0.335	0.405	0.178	0.321	0.288
Ours	**0.598**	**0.377**	**0.286**	**0.598**	**0.671**	**0.632**	**0.112**	**0.408**	**0.491**	**0.188**	**0.391**	**0.361**

our method enhances the potential relationship between characters and item, improves the accuracy of item recommendation, and proves that the model still has strong generalization in other games of the MOBA genre. Since there are differences in the characters and item used by the opponent in each game, the model will recommend item based on the opponent's character and the opponent's item.

C. Ablation study

Here we show an ablation study in Table IV to display the contribution of each part of our model.

TABLE IV. THE ABLATION EXPERIMENT COMPARES THE EFFECTS OF EACH MODULE.

RA	GAF	Precision@6	Recall@6	MAP@6	F1@6
✗	✗	0.492	**0.756**	0.805	0.596
✓	✗	0.540	0.729	0.823	0.620
✗	✓	0.548	0.715	0.823	0.621
✓	✓	**0.553**	0.721	**0.827**	**0.626**

We used TTIR as the baseline and respectively added RA module and GAF module. The results show that, when only the RA module is added, the models are improved by 4.8%, 1.8%, and 2.4% on Precision@6, MAP@6, and F1@6 respectively, proving that the RA module can effectively extract attribute features and structures between characters and item feature. When only the GAF module is added, the model improves by 5.6% on Precision@6, and the improvement effect on MAP@6 and F1@6 is almost the same as that of adding only the RA module. Adding two modules at the same time improves the model by 6.1% on Precision@6, 2.2% on MAP@6, and 3.0% on F1@6, indicating that the combination of RA module and GAF module maximizes the improvement of model performance.

As shown in Table V, based on the addition of RA module and GAF module, the number of multi-head attention mechanisms *C* in the graph attention network in GAF is changed.

TABLE V. EFFECT OF THE NUMBER OF GRAPH ATTENTION LAYERS IN GAF ON EXPERIMENTAL RESULTS

GMA	Precision@6	Recall@6	MAP@6	F1@6
C=1	0.543	0.708	0.818	0.615
C=2	0.546	0.711	0.821	0.618
C=3	0.550	0716	0.825	0.622
C=4	**0.553**	**0.721**	**0.827**	**0.626**

The experimental results show that the overall performance of the framework improves with the increase of the number of multi-head attention mechanism *C* in GAF module. Since each character can use up to six items at one time, the experimental analysis is mainly carried out on the results of four indicators with k=6. The number of multi-head attention mechanisms selected by the model is *C*=4, which means that different attention weights are integrated for four aspects: role, type, item, and team.

The model achieves the best in various indicators when *C*=4. Compared with *C*=1, the Precision@6 increases by 1.0%, the Recall@6 increases by 1.3%, the MAP@6 increases by 0.9% and the F1@6 increases by 1.1%. Recall@6 and F1@6 increased by 0.5% and 0.4% respectively compared with *C*=3, there is still a certain range of improvement, and the amount of parameters does not increase significantly. Therefore, the number of multi-head attention mechanisms selected by the model is *C*=4, which shows the effectiveness of using multi-head attention mechanism in graph attention network and the importance of paying attention to the attribute characteristics of different roles in the fusion stage.

V. CONCLUSIONS

We propose a new MOBA item recommendation model that extracts and fuses information by using a relational perceptual graph attention network architecture. The potential relationship perception of characters and item is constructed through the relation-aware module, and the complementary relationship between different characters is established. The graph attention fusion module is used to distinguish the influence strength between characters, and the feature fusion is more inclined by assigning attention weights. Compared with others, our method captures the potential dependencies between characters and item more adequately, and is particularly effective for diverse item recommendations in MOBA games. In the future, the serialized recommendation of game item can be studied, and how to fully mine the time dimension information and dynamically recommend item is the work that needs further research.

ACKNOWLEDGMENT

The research is partially supported by National Natural Science Foundation of P. R. China (No.62176009) and Key Project of Beijing Municipal Education Commission (No.KZ201910005008).

REFERENCES

[1] Silva, V. D. N., Chaimowicz, L. (2017). Moba: a new arena for game ai. arXiv preprint arXiv:1705.10443.

[2] Looi, W., Dhaliwal, M., Alhajj, R., & Rokne, J. (2018). Recommender system for items in dota 2. IEEE Transactions on Games, 11(4), 396-404.

[3] Summerville, A., Cook, M., Steenhuisen, B. (2016, September). Draft-analysis of the ancients: predicting draft picks in dota 2 using machine learning. In Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference.

[4] Cheuque, G., Guzmán, J., Parra, D. (2019, May). Recommender systems for Online video game platforms: The case of STEAM. In Companion Proceedings of The 2019 World Wide Web Conference (pp. 763-771).

[5] Pérez-Marcos, J., Martín-Gómez, L., Jiménez-Bravo, D. M., López, V. F., & Moreno-García, M. N. (2020). Hybrid system for video game recommendation based on implicit ratings and social networks. Journal of Ambient Intelligence and Humanized Computing, 11(11), 4525-4535.

[6] Wang, Y., Feng, D., Li, D., Chen, X., Zhao, Y., & Niu, X. (2016, July). A mobile recommendation system based on logistic regression and gradient boosting decision trees. In 2016 International Joint Conference on Neural Networks (IJCNN) (pp. 1896-1902). IEEE.

[7] Tassi, P. (2014). Riot's' League of Legends' reveals astonishing 27 million daily players, 67 million monthly. Retrieved October, 30, 2014.

[8] Araujo, V., Rios, F., & Parra, D. (2019, September). Data mining for item recommendation in MOBA games. In Proceedings of the 13th ACM Conference on Recommender Systems (pp. 393-397).

[9] Villa, A., Araujo, V., Cattan, F., & Parra, D. (2020, September). Interpretable Contextual Team-aware Item Recommendation: Application in Multiplayer Online Battle Arena Games. In Fourteenth ACM Conference on Recommender Systems (pp. 503-508).

[10] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems. 2017.

[11] Paolo Campanelli. (2017) League of Legends Ranked Matches. https://www.kaggle.com/paololol/league-of-legends-ranked-matches.

[12] Devin Anzelmo. (2020) Dota 2 Matches Dataset. https://www.kaggle.com/devinanzelmo/dota-2-matches.

[13] Wang, Y., Tang, S., Lei, Y., Song, W., Wang, S., & Zhang, M. (2020, October). Disenhan: Disentangled heterogeneous graph attention network for recommendation. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management (pp. 1605-1614).

[14] Lim, Nicholas , et al. "STP-UDGAT: Spatial-Temporal-Preference User Dimensional Graph Attention Network for Next POI Recommendation." (2020).

[15] Song, W., Xiao, Z., Wang, Y., Charlin, L., Zhang, M., & Tang, J. (2019, January). Session-based social recommendation via dynamic graph attention networks. In Proceedings of the Twelfth ACM international conference on web search and data mining (pp. 555-563).

[16] Liu, Yong, et al. "Contextualized graph attention network for chararecommendation with item knowledge graph."IEEE Transactions on Knowledge and Data Engineering (2021).

[17] Xie, R. , et al. "Improving Accuracy and Diversity in Matching of Recommendation with Diversified Preference Network." (2021).

[18] Wu, Qitian , et al. "Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems." (2019).