Winning Tracker: A New Model for Real-time Winning Prediction in MOBA Games

Chuang Zhao
zhaochuang@tju.edu.cn
College of Management and Economics, Tianjin University
Tianjin, China

Hongke Zhao∗
hongke@tju.edu.cn
College of Management and Economics, Tianjin University
Tianjin, China

Yong Ge
yongge@arizona.edu
The University of Arizona
Arizona, USA

Runze Wu∗
wurunze1@corp.netease.com
Fuxi AI Lab, NetEase Games
Hangzhou, Zhejiang, China

Xudong Shen
hzshenxudong@corp.netease.com
Fuxi AI Lab, NetEase Games
Hangzhou, Zhejiang, China

ABSTRACT

With an increasing popularity, Multiplayer Online Battle Arena (MOBA) games where two opposing teams compete against each other, have played a major role in E-sports tournaments. Among game analysis, real-time winning prediction is an important but challenging problem, which is mainly due to the complicated coupling of the overall Confrontation1, the excessive noise of the player's Movement, and unclear optimization goals. Existing research is difficult to solve this problem in a dynamic, comprehensive and systematic way. In this study, we design a unified framework, namely Winning Tracker (WT), for solving this problem. Specifically, offense and defense extractors are developed to extract the Confrontation of both sides. A well-designed trajectory representation algorithm is applied to extracting individual's Movement information. Moreover, we design a hierarchical attention mechanism to capture team-level strategies and facilitate the interpretability of the framework. To optimize accurately, we adopt a multi-task learning method to design short-term and long-term goals, which are used to represent immediate state and make end-state prediction respectively. Intensive experiments on a real-world data set demonstrate that our proposed method WT outperforms state-of-the-art algorithms. Furthermore, our work has been practically deployed in real MOBA games, and provided case studies reflecting its outstanding commercial value.

CCS CONCEPTS

• Information systems → Data stream mining; • Computing methodologies → Supervised learning by classification.

KEYWORDS

Online Games, Winning Prediction, Multi-task Learning

∗Corresponding Authors
1Confrontation, a hostile or argumentative situation between opposing parties.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

WWW ’22, April 25–29, 2022, Virtual Event, Lyon, France
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9096-5/22/04...$15.00
https://doi.org/10.1145/3485447.3512274

Figure 1: Real-time winning prediction.

1 INTRODUCTION

In the past ten years, MOBA games, as the leading sub-genre in E-sports, have generated billions of currency equivalent [37, 41]. Players work together as a team to achieve the ultimate victory condition, which is to destroy their enemy’s base whilst protecting their own [21]. Figure 1(a) is a typical scene of a MOBA genre game. Huge amounts of data have been generated from MOBA games, which allows us to extract important insights for game analysis.

In the field of game-related research, winning prediction is a critical and challenging task, which is mainly divided into two paradigms [17]: pre-match prediction [5] and real-time prediction [34]. The former mainly mines the pre-match static information to predict the final result, while the latter requires additional tracking of complicated real-time information to draw the win rate curve as shown in Figure 1(b). Our research focuses on real-time winning prediction, which could improve the player experience in many aspects [19]. For example, a dynamic winning tracker can provide valuable information for both battling sides and auto-generate the thrilling highlight moments during each battle.
According to our preliminary research, most of the existing work focus on pre-match winning prediction [4, 11, 26, 35] and only a few studies explore real-time scenarios [34, 43]. Real-time winning prediction is more challenging, which is mainly due to the complicated coupling of the overall Confrontation, the excessive noise of the player’s Movement, and unclear optimization goals. Firstly, the overall Confrontation is hard to be accurately characterized [24]. Different from pre-match prediction, real-time scenarios need to track the overall Confrontation during the game for prediction. However, the coupling of game data, the diversity of game events, and the excessive noise of player behaviors bring great difficulties to characterize the overall confrontation from a comprehensive perspective. Secondly, the individual’s Movement is not well explored [34, 40], while Movement is precisely one of the most important operations in MOBA games. On the one hand, the coordinates at a time point cannot capture the directionality and temporality of the player’s Movement [8]. On the other hand, the irregularity and complexity of the player’s Movement makes it difficult to use fixed patterns for effective characterization [3]. Thirdly, there is no clear and unified optimization goal for real-time winning prediction. It is inappropriate to directly use the final result as the optimization goal, which ignores the possibility of inconsistency between short-term predictions and game results. Take a battle for example, one side, say Team A, may start with a high morale and gain huge advantages while they unfortunately lose the game due to a misoperation of one key midterm combat.

To address these challenges, this paper proposes a new data-driven end-to-end neural network framework, namely Winning Tracker (WT), to solve the problem of real-time winning prediction. For characterizing complicated and coupled competition state comprehensively, we design different modules to conduct dual perception from overall Confrontation and individual’s Movement. Specifically, we reassemble the information of towers and players in the form of images according to the semantics of offense and defense, and construct different extraction modules from their semantics. Then we divide the battlefield into different zones and code them. By representing the player’s trajectory as a sequence of passing zones’ code, the temporality and directionality of the player’s Movement can be effectively characterized. In addition, we have also devised a hierarchical attention mechanism to capture the strategy of both teams in the competition and promote the interpretability of our framework. For optimization goals, we turn to multi-task learning methods [42]. Short-term goal is developed to capture immediate state, while long-term goal is developed to make end-state prediction. Different goals can not only supplement each other’s information, but also play the role of regularization [45]. The main contributions of this work are as follows:

• To our knowledge, this paper makes a significant effort on real-time winning prediction of MOBA games by considering dual perception of overall Confrontation and individual’s Movement. Specifically, we integrate the heterogeneous information of both sides from the offensive and defensive perspectives to capture the overall Confrontation, and use trajectory representation rather than the coordinates at a time point to characterize individuals’ Movement.

• We develop a novel trajectory characterization algorithm to capture the directionality and temporality of play’s Movement. In addition, we design a novel attention mechanism to capture team-level strategies and significantly increases the interpretability of the framework.

• Short-term and long-term goals are optimized in parallel to capture immediate state and make end-state prediction, which is very effective as shown in our experiments.

• Intensive experiments have demonstrated the effectiveness and robustness of our framework. At the same time, several case studies are presented in the paper, reflecting the interpretability and application value of our framework. The sample data set and code can be obtained in github2 after acceptance.

The rest of this paper is organized as follows. Section 2 briefly introduces related work, and then introduces the details of our proposed model. The experimental results and analysis are given in Section 4. Finally, we summarize the paper in the fifth section.

2 RELATED WORK

Our proposed method solves the problem of real-time winning prediction by constructing a new data-driven framework. Therefore, we briefly return to the most relevant work from two aspects: 1) winning prediction and 2) spatio-temporal data mining. Winning Prediction: Winning prediction is a key issue in game analysis, which can be divided into two paradigms, pre-match winning prediction and real-time winning prediction. Most studies put insights on pre-match predictions [5, 11, 12] but few studies focus on real-time prediction scenarios [6, 7]. Specifically, Gong et al. [11] construct a coordination graph and a suppression graph based on the player’s historical battle records to incorporate the team-up effect and predict the match outcomes. From the perspective of cooperation and competition, Gu et al. [12] design a module based on the attention mechanism to capture the hero’s intra-team and extra-team relationship. Li et al. [26] explore a new perspective and predict the winning rate from the comparison of player-interaction effects between the two teams. The above-mentioned research only uses pre-match information and lacks effective perception of dynamic information during the game. Kang and Lee [22] begin to turn attention to real-time win rate prediction, which uses an O-CNN model to identify statistical information in video screenshots to predict matching results. Yu et al. [43] develop a MOBA-Slice framework to explore the impact of the current state on the outcome of the game. However, the above two works just divide the time series into static states of multiple time slices and make predictions, which lead to a coarse-grained competition state extraction, and thus not practical for performance evaluation. Wang et al. [34] construct a framework called Match Tracing, which combines the two tasks of player action evaluation and win prediction. This work is from the perspective of measuring the value of player actions and provides a new insight for exploring real-time winning prediction.

This paper focuses on real-time winning prediction. We make a significant effort to solve this problem by considering the dual perception of the overall Confrontation and the individual’s Movement. Spatio-Temporal Data Mining: With the rapid development of various technologies such as global positioning systems, mobile
3 PROPOSED METHOD

In this section, we introduce our proposed WT framework for real-time winning prediction. We first introduce the problem definition and the overall architecture of the proposed model. Next we will elaborate on the technical details of each part, and then describe the model optimization process.

3.1 Problem Definition

Suppose there are $N$ players $\{1, 2, 3, \ldots, n\}$, $M$ observable matches. In each match, there are two opposing teams $T_1$ and $T_2$, and the players on both sides are a subset of $N$. Please note that our method does not limit the number of players on both sides to be the same. Compared with previous studies [11], our method has a wider range of application scenarios. We use a sliding window of size $S$ and step size of 1 for data collection. Assuming that there are $L$ logs from the beginning to the end of a game, there are a total of $L - S + 1$ data for each game. The data of each sliding window will be input into the WT framework to predict the short-term goal $y_t$ and the long-term goal $y_t$ in parallel.

In this paper, we simplify the short-term goal to "whether any tower is destroyed in the next time window". On the one hand, this is in line with the goal of the MOBA game, which is to destroy the opponent’s main building [27]. On the other hand, we turn this question into "If the game ends at the next moment, which team will win". Note that since there is no short-term goal for the last sliding window, we treat it as a multi-class task, i.e., not destroyed, destroyed, and the last window. The long-term goal is the match result. We assume that there is no draw. It is a binary classification task. $T_1$ wins is 0 and $T_2$ wins is 1.

3.2 Overview of WT Framework

Figure 2 illustrates the overall architecture of our proposed WT. Specifically, WT includes three main modules: 1) Confrontation Representation Extractor (CRE), 2) Movement Representation Extractor (MRE), and 3) Multi-task Forecasting Module (MTFM). In CRE, we preprocess log data into images according to the semantics of offense and defense, and constructed extractors from both perspectives to capture overall Confrontation. In MRE, we innovatively divide the game into different zones, and use the zones’ embedding to represent the player’s Movement. In addition, we also design a novel hierarchical attention mechanism in this module to capture team-level strategies. In MTFM, the pre-match information is embedded in a linear form and fused with the embedding of the above two modules to predict short-term and long-term goals in parallel.

3.3 Confrontation Representation Extractor

In this subsection, we elaborate on the details of the offense and defense sub-modules used to extract the overall Confrontation within the time window.

Offense Feature Extractor. Since the MOBA games regard destroying the enemy’s main buildings as a necessary condition of victory, to capture the offensive situation, we preprocess the data of the enemy tower’s remaining Hit Points (HP) and own players’ remaining HP in the form of images, as shown in Figure 2. The higher the HP value, the darker the color. This preprocessing method has three main advantages. Firstly, it can take into account the relative position of the buildings and the players. Secondly, the method of converting data into images has good scalability, and the number of layers can be easily expanded with the increase of data dimensions. Thirdly, the method of defining effective features can avoid directly scanning game screenshots, saving computing resources and reducing noise. Thanks to the high performance of CNN in extracting image information, we adopt LeNet [25] as backbone to extract the offensive situation of both sides, as shown in Figure 3(a).

$$O_{T1}^O = f^O_{T1}(Players_{T1}, Towers_{T1}) + b_0,$$

$$O_{T2}^O = f^O_{T2}(Players_{T2}, Towers_{T2}) + b_0,$$  \hfill (1)

where $O_{T1}^O$ and $O_{T2}^O$ are the offensive representation of $T_1$ and $T_2$ respectively. $f^O_{T1}$ is LeNet model, Players and Towers are the state of Players and Towers respectively and $b_0$ is the offensive bias.

Defense Feature Extractor. In MOBA, defense and offense are relative concepts. While destroying the enemy’s main buildings, we also need to defend against the enemy’s offense. To extract the independent defense of both sides, we also draw the target data into images and use LeNet as our backbone to extract defense representation, as shown in Figure 3(b). The difference from the previous subsection is that we will draw the remaining HP of own tower and the remaining HP of own players as own defense information, and vice versa, as shown in Figure 2.

$$D_{T1}^D = f^D_{T1}(Players_{T1}, Towers_{T1}) + b_D,$$

$$D_{T2}^D = f^D_{T2}(Players_{T2}, Towers_{T2}) + b_D,$$  \hfill (2)

where $D_{T1}^D$ and $D_{T2}^D$ are the defensive representation of $T_1$ and $T_2$ respectively. $f^D_{T1}$ is another LeNet model and $b_D$ is the defensive bias. Note that although defense and offense both extract information...
where $C_1^T$ and $C_2^T$ are the Confrontation of $T_1$ and $T_2$ respectively, \(\odot\) and \(\oplus\) represent Hadamard product and concatenation respectively. Then we use LSTM [16] to capture the overall Confrontation within the time window.

$$h_S = \text{LSTM}(C_1, C_2, ..., C_S), \quad (4)$$

Note that we take the last hidden layer $h_S$ as the overall Confrontation within the time window.

### 3.4 Movement Representation Extractor

In this subsection, we mine spatio-temporal data to take advantage of the temporality and directionality of player’s Movement and design a unique attention mechanism to capture team-level strategies. Please note that the player’s Movement in this paper is represented by a trajectory vector.

**Trajectory Feature Extraction.** We divide the battlefield into different zones according to $N \times M$ grid, as shown in Figure 4(a), and code the zones. Then we use the sequence of zones the player passes through during the time window to represent the player’s trajectory. As shown in Figure 4(b), player A’s trajectory is represented as

Figure 2: The overall framework of WT. Icon color shade represents the remaining Hit Points (HP) of the tower or player.

Figure 3: Confrontation Representation Extractor (CRE). (a) Offense feature extractor. (b) Defense feature extractor.

Figure 4: Movement Representation Extractor (MRE). (a) Zone coding. (b) Individual trajectory embedding. (c) Team-level strategy embedding. See section 3.4 for details.
[02,07,07,08,08,03,03]. Finally, the zones sequence will be extracted by LSTM to form the final player’s trajectory embedding.

\[ p_i^{T_m} = \text{LSTM}(p_i^{T_m}, p_i^{T_m}, \ldots, p_i^{T_S}), \]

where \( p_i \) is the zone id of player \( i \) at time \( t \), and \( T_m \) is \( T_1 \) or \( T_2 \).

**Team-level Strategy Extraction.** When we obtain the Movement embeddings of all players, we develop a novel hierarchical attention mechanism to extract team-level strategies, as shown in Figure 4(c). To prevent noise caused by different teams, we design independent attention mechanisms for both sides of the competition.

\[ u_i^{T_m} = \tanh(W_u p_i^{T_m} + b_u), \]

\[ a_i^{T_m} = \exp(u_i^{T_m} u_i^{T_m}) \]

\[ \sum_{i=1}^{k} \exp(u_i^{T_m} u_i^{T_m}), \]

\[ s_i^{T_m} = \sum_{i=1}^{k} a_i^{T_m} p_i^{T_m}, \]

where \( \uparrow \) represent transpose operation, \( u_i \) is the target optimization vector, \( a_i^{T_m} \) is the attention weight of the player \( i \)'s Movement, \( k \) is the number of players of \( T_m \) and \( s \) is the team-level strategy. Please note that the team-level strategy extraction here provides new insights for measuring player contributions, which we will elaborate on in section 4.4.

### 3.5 Model Optimization

In this section, we first describe the prediction process of two tasks, and then define the loss function of WT for training.

**Multi-task Forecasting Module.** To capture immediate state and make end-state prediction, we adopt multi-task learning method to predict two goals in parallel, as explained in the Problem Definition section. In addition to capturing the real-time competitive status, it is also very important to consider the pre-match strength comparison of the two teams. Therefore, we take a function \( f(x) \) to embed the pre-match information (e.g. average level, team equipment level) \( x \) and then concat it with the overall Confrontation \( h_S \) and team-level strategy \( s_i^{T_m} \) to obtain these knowledge.

\[ o = f(x) \oplus h_S \oplus s_i^{T_m}, \]

Note that the processing of pre-match information \( f(x) \) only uses linear combinations of features. Without loss of generality, other research algorithms that explore the characterization of pre-match features can be applied here. This paper focuses on the extraction of real-time competitive representations, and does not expand.

Once the comprehensive representation \( o \) is obtained, we first pass it through a shared parameter layer, and then make predictions for short-term and long-term tasks separately.

\[ y_s = W_s o + b_s, \]

\[ y_i = W_i o + b_i, \]

where \( W_s \) is the shared parameter layer, and \( W_s \) and \( W_i \) are taken to optimize two goals respectively.

**Total Loss.** Both short-term and long-term goals are classification tasks, and we use Cross Entropy(CE) loss for optimization. To make a trade-off between short-term forecast and end-state prediction, we set \( \lambda \) to balance.

\[ L = - \sum_{i=1}^{N} \lambda_i \sum_{i=1}^{N} \left( y_i \log \tilde{y_i} + (1 - y_i) \log (1 - \tilde{y_i}) \right), \]

where \( \lambda_i \) is a parameter that determines the weight of \( t \)-th task, \( y_i \) is the ground-truth label, and \( N \) is the number of categories. Note that the sum of \( \lambda_i \) is 1 and its hyperparameter search will be shown in subsequent experiments.

Note that \( S \) is the length of the sliding window, and we will perform a parameter sensitivity test on it in section 4.3. Here \( S = 7 \) is the optimal result.

### 4 EXPERIMENTS

In this section, we conduct experiments on a real-world large scale game dataset to evaluate the performance of WT on real-time winning prediction. Particularly, we aim to answer the following research questions (RQs):

- **RQ1:** How does WT perform in the real-time winning prediction task compared with current state-of-art methods?
- **RQ2:** How is the parameter sensitivity of the WT model?
- **RQ3:** How effective is our innovative design for improving the depiction of immediate state and end-state prediction?

#### 4.1 Experimental Settings

**Datasets.** We use a dataset from a popular MOBA game, released by a well-known gaming company. In total, we have recorded more than 10 million game logs which include tens of thousands of matches. The detailed log fields are shown in Table 1.

<table>
<thead>
<tr>
<th>Object</th>
<th>Field</th>
<th>Description</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Player</td>
<td>id</td>
<td>player’s id</td>
<td>hash code</td>
</tr>
<tr>
<td></td>
<td>side</td>
<td>player’s team</td>
<td>category</td>
</tr>
<tr>
<td></td>
<td>class</td>
<td>player’s role</td>
<td>category</td>
</tr>
<tr>
<td></td>
<td>grade</td>
<td>player’s level</td>
<td>category</td>
</tr>
<tr>
<td></td>
<td>score</td>
<td>player’s six attribute values</td>
<td>List-float</td>
</tr>
<tr>
<td></td>
<td>coord</td>
<td>player’s coordinate with (x, y)</td>
<td>Tuple-float</td>
</tr>
<tr>
<td></td>
<td>hp</td>
<td>player’s health points</td>
<td>int</td>
</tr>
<tr>
<td>Team</td>
<td>guild_id</td>
<td>team’s guild id</td>
<td>hash code</td>
</tr>
<tr>
<td></td>
<td>morale</td>
<td>team’s morale level</td>
<td>category</td>
</tr>
<tr>
<td></td>
<td>resource</td>
<td>team’s number of resource</td>
<td>int</td>
</tr>
<tr>
<td></td>
<td>num</td>
<td>team’s number of props</td>
<td>int</td>
</tr>
<tr>
<td></td>
<td>kill</td>
<td>team’s total kills</td>
<td>int</td>
</tr>
<tr>
<td>Building</td>
<td>build_id</td>
<td>building’s id</td>
<td>hash code</td>
</tr>
<tr>
<td></td>
<td>type</td>
<td>types of defense tower</td>
<td>category</td>
</tr>
<tr>
<td></td>
<td>hp</td>
<td>tower’s health points</td>
<td>float</td>
</tr>
<tr>
<td></td>
<td>coord</td>
<td>tower’s coordinate with (x, y)</td>
<td>List-float</td>
</tr>
<tr>
<td>Other</td>
<td>ts</td>
<td>logging time</td>
<td>timestamp</td>
</tr>
<tr>
<td></td>
<td>during</td>
<td>current game duration</td>
<td>float</td>
</tr>
<tr>
<td></td>
<td>finish</td>
<td>whether the game is over</td>
<td>category</td>
</tr>
</tbody>
</table>
Several data preprocessing methods have been applied, such as one-hot encoding of categorical variables, normalization of numerical variables, and filtering matches with abnormal state. Specifically, there are the following filtering rules: 1) Abnormal matches that do not meet the rules of the game. 2) Matches with a large amount of missing data. 3) Matches with abnormal log records. We will release the source code and sample dataset after acceptance.

**Parameter Settings.** Our model is implemented using PyTorch [29]. Except for the necessary concat operation, the embedding size is 64. We adopt kaiming method [14] for parameter initialization. For gradient descent, we take Adam [23] to set the initial learning rate 1e-4 for model optimization. In our proposed model, we set batch to 512 and the loss function is Cross Entropy(CE). In the hyperparameter tuning, we search for window size [5, 7, 8, 9], CNN kernel size [3, 5, 6, 7], weight between short-term and long-term goals [0.4, 0.5, 0.6, 0.7, 0.8], which will be given in our paper. In addition, we randomly select 80% of the data as the training set and 20% of the data as the test set.

**Baselines Methods.** We compare our WT model with the following state-of-art baselines for winning prediction. We have fine-tuned all of them to achieve the best performance.

- **CNN** [13]: CNN is also widely used to process sequence data. In our experiment, we design convolution kernels of different sizes and concatenate them to extract more information.
- **TCN** [2]: TCN uses the structure of 1-D FCN and causal convolution to be more capable of retaining long-term historical information.
- **LSTM** [16]: This is a classical RNNs network, which uses gating mechanism to capture the long-term sequence dependence.
- **Transformer** [32]: Transformers uses the attention mechanism to replace the traditional RNN structure, which has outstanding performance in capturing temporal information.
- **ConvLSTM** [38]: ConvLSTM is one of the most popular algorithms in the field of spatiotemporal data mining, which can directly input images into the model for prediction.

**Evaluation Metrics.** To evaluate the performance of the real-time winning prediction, we use the popular measurement standards. For short-term goal, we use CE, Macro-F1 Score, Micro-F1 Score, Convergence Epoch for evaluation, and for long-term goal, use CE, F1-Score, AUC, Convergence Epoch for evaluation. Except for CE, all indicators are the higher the better, and vice versa. These metrics have been widely used in previous studies [12, 34].

### 4.2 Comparisons with Baseline Methods (RQ1)

Table 2 shows the performance of our model compared to the baselines. Obviously, the proposed WT is superior to all baselines in short-term and long-term goals, demonstrating the effectiveness of our framework. Specifically, for short-term goal, WT is basically the same as other baselines in most indicators, but the convergence speed (Epoch) has improved significantly. We guess that the reason is short-term goal is easier to predict than predicting the final result due to the short interval. For long-term goal, commonly used time series prediction algorithms such as CNN, LSTM, Transformer do not work well. This is mainly due to the fact that they only represent competition state from a single perspective. The effect of TCN is indeed slightly stronger than that of traditional algorithms, which fully demonstrates the superiority of the causal convolution mechanism. Although ConvLSTM performs spatio-temporal data mining, its performance is slightly weaker than LSTM. We guess that the coupling of Confrontation information may cause the noise.

![Figure 5: The impact of λ.](image)

![Figure 6: The impact of S.](image)

**4.3 Robustness Evaluations (RQ2, RQ3)**

To test the robustness of WT, we launch the parameter sensitivity analysis and the sub-module ablation experiments.

#### 4.3.1 Hyperparameter Tuning

In this subsection, we conduct a set of experiments with varied hyperparameters.

**λ-Weight of Short-term and Long-term goals.** According to Figure 5, we can get the following observations. Our model is robust to the short-term goal, different parameters will only impact the speed of convergence, and will not change the final prediction effect. For long-term goal, our algorithm achieves the best results when \( \lambda = 0.6 \). We guess that long-term goal is more difficult than short-term goal and requires higher weights for optimization. If \( \lambda \) is too high or too low, the model tends to optimize a single target and loses the advantage of multi-task learning.

**S-Sliding Window Size.** As shown in Figure 6, it is easy to conclude that our model is robust to window size for short-term goal, and obtains the best performance on long-term goal when the window size is 7. For long-term goal, when the window size is short, the captured immediate state is insufficient, which will weaken the prediction performance. The reduction of training data and the increase of noise in the window will also affect the characterization of the competition state if the window size is too long.

**K-Kernel Size.** As mentioned in [15, 39], the CNN kernel size determines the granularity of the representation, which will have a great impact on the convergence and performance of the model. We adjust the kernel size, as shown in Figure 7. Obviously, for short-term goals, the kernel size affects the convergence speed, but it does not affect the final accuracy. We argue that this is because the short-term goal is more concerned with the change process of the
confrontation state, rather than the representation granularity of the confrontation at the time point. The long-term goal performance will fluctuate due to the kernel size, and each indicator will get the best effect when kernel = 5. Compared with short-term goal, the prediction of long-term goal not only needs to capture the changing process of the confrontation, but also needs to extract a better representation of the global confrontation information. The appropriate kernel size can extract the optimal granularity level.

4.3.2 Ablations of Submodules. To evaluate the effect of our design, we eliminate or replace the sub-modules in WT. For fairness, except for the specified ablation module, other settings are kept the same.

- WT-NC: Confrontation Representation Module is removed and the other modules remain the same.
- WT-NT: We remove the Movement Representation Extraction Module and the other modules remain the same.
- WT-NS: In this variant, the pre-match information is not fused.
- WT-NOD: We do not distinguish between offense and defense, i.e., only use one extractor to capture the overall Confrontation.
- WT-NTT: We use one attention mechanism to capture team-level strategies instead of providing independent for both sides.
- WT-NA: When capturing team-level strategies, we use average pooling instead of attention mechanism.

We can conclude from Figure 8(a) that for short-term task, the ablation of the sub-modules does not have a great impact. On the one hand, we guess that the predicted target and the extracted representation are relatively close in terms of time, which makes it easy to predict. On the other hand, the short-term target and the ablation module have a low correlation, resulting in little change. Besides, the short-term goal is to make the final representation be more capable of capturing immediate state, and its robustness has a positive effect on our model. It can be seen from Figure 8(b) and Table 3 that different sub-modules play an irreplaceable role in end-state prediction. In general, WT-NS performs worst, which shows that pre-match strength gap has an significant impact on the match result. Offense and defense are two natural aspects of Confrontation, and the lack of them leads to poor performance of WT-NOD. It is not perfect to capture the confrontation state from a single perspective, i.e., overall Confrontation or individual’s Movement, which is why WT-NT and WT-NC are weaker than WT. Intuitively, the win rate is a comparison of the strengths of two teams. Ignoring team information for characterization will not only cause information loss, but also noise due to the fusion of information between the two teams, resulting in poor accuracy of WT-NTT. We believe that the contribution of different players should be different. It is inappropriate to treat all players equally and ignore the prominent role of key player’s Movement in match result. The performance of WT-NA is consistent with our conclusion, which is slightly lower than our model WT. Our model has achieved the best results, which fully demonstrates the effectiveness of our model design.

Table 3: Comparison with ablation models.

<table>
<thead>
<tr>
<th>Model</th>
<th>Short-Term Goal</th>
<th>Long-Term Goal</th>
<th>AvgRank</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE</td>
<td>F1-Score</td>
<td>AUC</td>
<td></td>
</tr>
<tr>
<td>WT-NA</td>
<td>0.871(3)</td>
<td>0.945(2)</td>
<td>3.0</td>
</tr>
<tr>
<td>WT-NOD</td>
<td>0.779(6)</td>
<td>0.891(5)</td>
<td>5.7</td>
</tr>
<tr>
<td>WT-NS</td>
<td>0.722(7)</td>
<td>0.854(7)</td>
<td>7.0</td>
</tr>
<tr>
<td>WT-NT</td>
<td>0.874(2)</td>
<td>0.938(4)</td>
<td>3.0</td>
</tr>
<tr>
<td>WT-NTT</td>
<td>0.868(4)</td>
<td>0.939(3)</td>
<td>3.0</td>
</tr>
<tr>
<td>WT-NC</td>
<td>0.791(5)</td>
<td>0.877(6)</td>
<td>5.3</td>
</tr>
<tr>
<td>WT</td>
<td>0.901(1)</td>
<td>0.960(1)</td>
<td>1.0</td>
</tr>
</tbody>
</table>

We eliminate or replace the sub-modules in WT. For fairness, except for the specified ablation module, other settings are kept the same.

(a) Acc of short-term goal. (b) Acc of long-term goal. (c) Metrics of long-term goal.
4.4 Case Study

In the deployment phase, we develop two commercial applications, key event detection and player value evaluation, from the perspective of Confrontation and individual’s Movement respectively. To comply with the blind review, we take screenshots of similar scenes from DOTA2 instead of the game corresponding to the data set. As shown in Figure 9(a), we consider the event that causes a large fluctuation in the win rate to be a key event. Around 70 seconds, the enemy has attacked from 3-way when Team 1 has not yet fully defended, and the winning rate drops from 0.68 to 0.41. Around 370 seconds, Team 1 loses the team fight, and the win rate drops from 0.7 to 0.45. Around 610 seconds, Teams 1 destroys three defense towers of the enemy, and the curve increases from 0.42 to 0.68. Around 910 seconds, Team 1 sneaks an attack on the enemy’s crystal, which reverses the battle and leads to the final victory.

Figure 9: Real-world application. Note that they can be used for live commentary or post-game analysis of professional teams.

Inspired by the use of passing value to measure the contribution of athletes in football match [3, 9], for player value evaluation, we measure it from the perspective of Movement value. It can be concluded from Figure 9(b) that the influence of players’ Movement on match results is significantly different. Note that the thickness of the line in Figure 9(b) represents the importance of the trajectory. Intuitively speaking, player A’s trajectory threatens the opponent’s crystal, forcing the enemy to return to defense on the one hand, and encouraging own fighting spirit on the other. Player B’s trajectory is always near the defensive tower or other places where there are no enemies, which is a negative game behaviors. Furthermore, we show the contribution of all players in each slide window of the match in the form of a heat map, as shown in Figure 10. From the perspective of columns, we can analyze the participation of all players in a sliding window. For example, the darker color in the last column means that players are more competitive and move more frequently, which is consistent with our common sense. From the perspective of rows, we can analyze the contribution of a player to the team in the entire game. We sum the importance of the player’s trajectory in each period to obtain the player’s overall contribution,

\[ \text{Score}_i = \sum_{t=1}^{L} a_{i,t}. \]

Figure 10: Contributions of players in a match. Note that there is no intersection between the sliding windows here.

Table 4: Top 5 scoring players in Team 1

<table>
<thead>
<tr>
<th>Player</th>
<th>Actual</th>
<th>Rank</th>
<th>Movement</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100</td>
<td>1</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>90</td>
<td>2</td>
<td>75</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>84</td>
<td>3</td>
<td>81</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>77</td>
<td>4</td>
<td>59</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>76</td>
<td>5</td>
<td>41</td>
<td>5</td>
</tr>
</tbody>
</table>

score, as shown in Table 4. Obviously the rankings of B and C have changed. This is mainly because the traditional method focuses on behaviors that are directly related to the match result, such as killing or destroying towers, while our method provides the possibility to measure player value from another objective perspective.

5 CONCLUSION

In this work, we make a significant effort on real-time winning prediction of MOBA games by considering dual perception from the overall Confrontation and the individual’s Movement. Specifically, we innovatively preprocess the information of players and towers in the form of images, and design offense and defense extractors to capture the Confrontation situation. This paper also fills the gap in the exploration of individual’s Movement and develops a hierarchical attention mechanism to capture team-level strategies. Short-term and long-term goals are optimized in parallel to represent immediate state and make end-state prediction respectively. In the end, intensive experiments on a real data set demonstrate the superiority of our framework in prediction accuracy and convergence speed, and further ablation experiments show the effectiveness and robustness of our sub-module design. Besides, two intuitive and interesting business applications are discussed in case study. This paper has two limitations. First, we did not explore the impact of the pre-match strength gap on the game process but directly integrated it as a feature. Second, in section 3.4, we did not explore more ways of meshing, such as rhombus, hexagon, etc. We will leave them for future research. We also plan to extend our model to other real-time competitions other than MOBA games, such as real sports.

ACKNOWLEDGMENTS

This study was partially funded by the supports of National Natural Science Foundation of China (72101176).