A Player Behavior Model for Predicting Win-Loss Outcome in MOBA Games

Xuan Lan[1], Lei Duan[1](✉), Wen Chen[1], Ruiqi Qin[1], Timo Nummenmaa[2], and Jyrki Nummenmaa[2]

[1] School of Computer Science, Sichuan University, Chengdu, China
xuan_lan@163.com, richforgood@163.com, {leiduan,wenchen}@scu.edu.cn
[2] Faculty of Natural Sciences, University of Tampere, Tampere, Finland
{timo.nummenmaa,jyrki.nummenmaa}@uta.fi

Abstract. Multiplayer Online Battle Arena (MOBA) game is currently one of the most popular genres of online games. In a MOBA game, players in a team compete against an opposing team. Typically, each MOBA game is a larger battle composed of a series of combat events. During a combat, the behavior of each player varies and the outcome of a game is determined both by the variation of each player's behavior and by the interactions within each instance of combat. However, both the variation and interaction are highly dynamic and difficult to master, making it hard to predict the outcome of a game. In this paper, we present a player behavior model (called pb-model). The model allows us to predict the result of a game once we have collected enough data on the behaviour of the players. We first use convolution to extract the features of player behavior variation in each combat and model them as sequences by time. Then we use a recurrent neural network to process the interaction among these sequences. Finally, we combine these two structures in a network to predict the result of a game. Experiments performed on typical MOBA game dataset verify that our pb-model is effective and achieves as high as 87.85% prediction accuracy.

Keywords: Game outcome prediction · Data mining · Deep learning

1 Introduction

Online gaming is increasingly popular as entertainment for more and more people. It is reported that, in 2015, young people spent 5 h on online gaming per day on average[1]. Among various kinds of online games, such as first-person shooter game, real-time strategy game and massively multiplayer online game, the *multiplayer online battle arena* (MOBA) game attracting over millions of concurrent

This work was supported in part by the National Natural Science Foundation of China under grant No. 1572332, the Fundamental Research Funds for the Central Universities under grant No. 2016SCU04A22, and the China Postdoctoral Science Foundation under grant No. 2016T90850.

[1] http://caas.raptr.com/category/most-played/.

users is a fusion of role-playing game, action game and real-time strategy game. Table 1 lists some representative MOBA games including *League of Legends*, which is the most popular one in the US and Europe during 2017[2].

Table 1. Some popular MOBA games

Game	Developer	Popular rank (see footnote 2)
League of Legends (LoL)	Riot games	1
Defense of the Ancients (DotA) 2	Valve corporation	10
Heroes of the Storm (HotS)	Blizzard entertainment	15

Typically, in a MOBA game, players are divided into two teams. Each player controls a single character in a team who competes versus another team of players. There are various abilities and advantages for each player character to contribute to the overall strategy of the team. The team destroying the opposing team's main structure is the winner. Moreover, there are many resources dispersed in the game map. Therefore, the players also have to fight with the adversaries to acquire these resources to enforce their fighting ability.

It should be noted that most of the games offered by the *World Cyber Games* (WCG) are MOBA. It is reported that a typical online MOBA game should has the following properties: (1) multiplayer: players from all over the world are allowed to challenge each other in the games. (2) competition: the purpose of the players (or teams) is to compete against others to win the games.

Recently, more and more game developers have recognized the popularity of MOBA games in the market. Thus, with its current status in the market, analysis of player behavior is an indispensable design step in the development of a succesfull MOBA game. Artificial intelligence and data mining techniques are highly useful to build a model of players behavior. For example, Erickson *et al.* [1] evaluated players' expertise level in game *DotA2* (a typical MOBA game).

Correspondingly, modeling player behavior to discover the unknown patterns and knowledge of games based on players' interactions or even possible interactions is attracting attention from not only industry but also academic fields. The analysis results can be utilized to predict the outcomes of online matches or to improve the game itself, e.g. by enhancing the intelligence of game AI, to improve defence against Bots, or to improve game balance. Gameplay simulations based on modeling player strategies based on player character interactions have also been used in game rule analysis [2].

By modeling player behaviors, the win-loss outcome of a MOBA game, i.e., the winning team, can be predicted. For example, Erickson *et al.* [3] used player behaviors to predict the outcome of the *Starcraft* game, and Rioult *et al.* [4] also used player behaviors to predict the outcome of game *DotA2*. However, to the

[2] https://newzoo.com/insights/rankings/top-20-core-pc-games/.

best of our knowledge, there is no previous work developing a general prediction model for MOBA games. In other words, a prediction model that can be used for all kinds of MOBA games without apriori knowledge.

Typically, a MOBA game is composed of a series of combat instances. This means, players interact with each other and the game environment in a combat situation. A player may be victorious or be defeated in combat. Victory or defeat in a combat situation will affect the behavior of the player (e.g., experience, equipments, coins). The variation of player behaviors is highly dynamic and quite different during combats. Therefore, it is necessary to analyze the relationships between player behaviors and combat outcomes.

In this paper, we propose a player behavior model representing the behavior variations, to predict the win-loss outcome of MOBA games. The main contributions include: (1) Developing a convolutional neural network, called pb-CNN, to extract the variation of player behaviors in each combat during a MOBA game; (2) Building a recurrent neural network, called pb-RNN, to model the sequences of player behavior variations in a MOBA game; (3) Constructing a player behavior model, called pb-model, to predict the win-loss outcome of a MOBA game based on both pb-CNN and pb-RNN; (4) Testing the effectiveness of pb-model based on game data set of *DotA2*.

The rest of this paper is organized as follows. Section 2 reviews the related works. Section 3 gives a detailed description of our behavior model, and Sect. 4 verifies our model. We conclude our work in Sect. 5.

2 Related Work

Recently, it has been recognized that the analysis of player behavior is important for the design of MOBA games. Some artificial intelligence and data mining techniques are applied to model player behavior. In general, there are two kinds of works on player behavior analysis.

The first category uses player behavior to analyze their influence on the game. Pobiedina *et al.* [5] ranked the players behaviors to find important factors influencing the outcome of game *DotA2*. Tinnawat [6] analyzed the behavior relationship between the role in the *DotA* and the leadership of the same person in real life. He found that people who have stronger leadership skills in real life tend to act the role of "carry" in games. The "carry" is a role that has large power and leads other members to fight in the game. Drachen *et al.* [7] used player behavior to analyze the spatiotemporal relationship among skills and the actions of the team. Castaneda *et al.* [1] studied different behaviors between the novices and the experts in *DotA2* and spread this study to other domains. Li *et al.* [8] used player behaviors to visualize snowballing and comeback of MOBA games. They developed a visual analytic system which can find key events and game parameters resulting in snowballing or comeback occurrences in MOBA games. Suznjevic *et al.* [9] used player behaviors to evaluate player skills in the game, calculating the player rating based solely on the match outcome.

The second category predicts the outcome of the game based on player behaviors. Erickson *et al.* [3] extracted behaviors from game logs, and constructed a model based on the current game state to predict the outcome of the *Starcraft*. Rioult *et al.* [4] used the topological player behaviors to predict the outcome of game *DotA2*. These behaviors consist of area of polygon described by the players, inertia, diameter, and distance to the base. Dereszynski *et al.* [10] used the hidden markov model, Weber *et al.* [11] used the decision tree and Synnaeve *et al.* [12] used the Bayesian model to predict strategies in Real-Time Strategy Games. Wang *et al.* [13] aimed at lineups using Naïve Bayes classifier to predict the outcome of *DotA2*. Cleghern *et al.* [14] used a value-split model for health of players to predict future states in time series attribute data. Yang *et al.* [15] proposed that combats in the game determined the ultimate outcome, by constructing combat sequences to identify the patterns in graph, and using these patterns to predict the outcome.

Deep learning is a powerful and effective tool with increasingly growing application in many domains. So far, some work has been done in analyzing game player behaviors using deep learning methods. Park *et al.* [16] proposed a deep learning based player evaluation model by combining both quantitative game statistics and the qualitative analysis provided by news articles. This model was based on the player behaviors during certain periods and the news articles in the same period. Leibfried *et al.* [17] used deep learning to predict rewards in Atari Games. Their work demonstrated that it was possible to learn system dynamics and the reward structure jointly in Atari Games. Oh *et al.* [18] also used deep learning to perform action-conditional video prediction in Atari Games. Their work was the first to make and evaluate long-term predictions on high-dimensional videos by controlling inputs.

3 Player Behavior Model for Outcome Prediction

In this section, we introduce our pb-model for MOBA game outcome prediction. The proposed pb-model consists of three parts: (1) player behavior extraction, (2) behavior sequence modeling, and (3) game outcome prediction.

3.1 Player Behavior Extraction

During each combat in a MOBA game, many factors with respect to player behaviors, such as abilities, equipments, levels, and experiences, are changing during the game. In a combat of a MOBA game, the variation of player behaviors are closely related to the win-loss outcome of the combat. For example, if a team wins a combat, the player behaviors of this team will vary more quickly than the losing one. For example, in game *DotA*2, the winning team will have more golds, more abilities, and more equipment. Clearly, the player behavior variation is the most important factor to the win-loss outcome of a combat.

In a MOBA game, we denote B the set of player behaviors, B_i the i-th player behavior variations between every two consecutive combats, p the feature of behavior variations that we want to extract, and $S(\cdot)$ the relevant function among all behaviors in a behavior set. The feature model is formulated as follows.

$$p = S(B_1, B_2, ..., B_n) \tag{1}$$

A proper relevance function is the key of our feature model. There are two problems should be addressed. The first is that disposing these behaviors jointly may result in exploding parameters. A straightforward solution to this problem is fusing some behaviors together as the best feature. However, the impacts of behaviors not chosen are ignored completely. The second is that extracting dynamic behavior variation is difficult due to their varying over time.

In Deep Learning, the CNN model is adept at extracting features, decreasing the amount and dimension of parameters and saving the runtime of processes. Based on CNN, we develop pb-CNN, which takes all player behaviors in a combat as input, and embeds them into input matrices to get the feature of behavior variations. We apply following convolution to pb-CNN.

$$R_i = g(WB_i + b) \tag{2}$$

where g is a non-linear active function, b is a bias vector, and W is the convolution kernel. In pb-CNN, we convolute each behavior for 100 times. We denote by R_i the feature map of the i-th player behavior. By convoluting all the behaviors in the behavior set, we can get a series of feature maps.

In pooling, pb-CNN uses max-pooling to get the max value from all feature maps, which is the most important feature. Then pb-CNN utilizes various sizes of pooling kernels aiming at different sizes of feature maps in order to get the same size of vector results. Each behavior can get its own feature vector O_i by pooling, and all of them combined is the feature of behavior variations. The relevant function is formulated as follows.

$$p = S(B_1, B_2, ..., B_n) = \sum_{n=1}^{i} O_i \tag{3}$$

In the process of combination, we combine these vectors in order to get the feature of behavior variations. Figure 1 shows the process of combination.

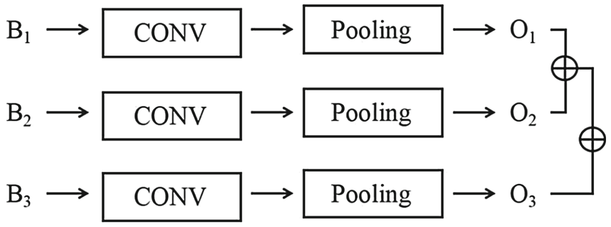


Fig. 1. Structure of pb-CNN

3.2 Behavior Sequence Modeling

As described above, in a MOBA game, a team has to compete for the resources and protect its buildings in order to win a game. There are several combats in a game, each result of the current combat will influence the outcome of next combat, and eventually the outcome of the whole game. Clearly, the game outcome is determined by the results of a series of combats that contained in the game.

We can combine all the combats into a sequence. Let $C = \{C_1, C_2, ..., C_i\}$ be the combat sequence in a game, where C_i is the outcome of the i-th combat. Considering the feature of a previous combat will influence the outcome of later combats, the interactions between different combat features should also be considered, besides for the features considered in pb-CNN. Let p_i be the feature of the variation from the i-th combat to the $(i+1)$-th combat. We combine these features into a sequence in the order of combats. Let $P = \{p_1, p_2, ..., p_i\}$ be the feature sequence of combat variations, H the relevant function between all features in the combat sequence, and W the prediction vector. The feature sequence model is formulated as follows.

$$W = H(p_1, p_2, ..., p_i) \tag{4}$$

Different from traditional networks, RNN utilizes a recurrent structure to dispose interactions between sequences. LSTM is one kind of RNN, inheriting the peculiarity of RNN in disposing sequences, but solves the vanishing and exploding gradients by "gate". We input the feature sequence P to the LSTM, and formulate pb-RNN as follows.

$$f_t = \sigma(W_f p_t + U_f h_{t-1} + b_f) \tag{5}$$

$$i_t = \sigma(W_i p_t + U_i h_{t-1} + b_i) \tag{6}$$

$$\tilde{C} = tanh(W_c p_t + U_c h_{t-1} + b_c) \tag{7}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C} \tag{8}$$

$$O_t = \sigma(W_o p_t + U_o h_{t-1} + b_o) \tag{9}$$

$$h_t = O_t \cdot tanh(C_t) \tag{10}$$

where W, U, b are the parameter matrices, $\sigma(\cdot)$ is the sigmoid function, p_t is the t-th feature, h_t is the network state of t-th feature and C_t is the cell of the t-th feature. Equation 5 is the forget gate, Eq. 6 is the input gate and Eq. 9 is the output gate, these three gates are used to process data. Equations 7 and 8 are the cells, which remove, save and update data from three gates.

In practice, we discover that in the process of most combats, there is a situation called "kill", where the player who is "killed" will lose the golds and cause

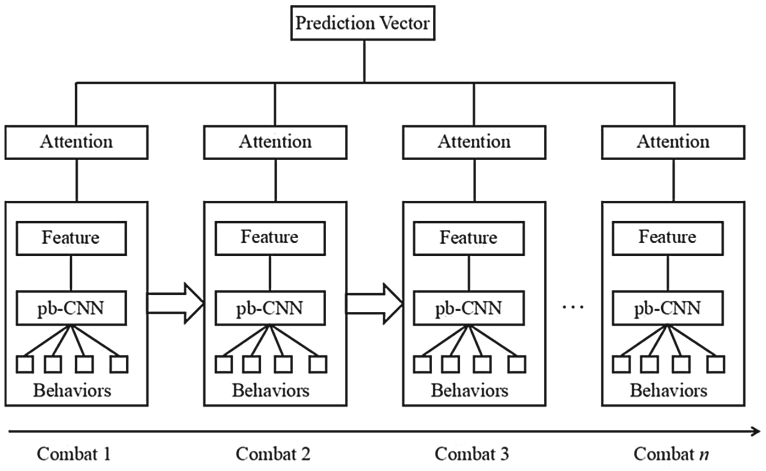


Fig. 2. Framework of pb-RNN

slight effects on the teams, and also influence the outcome of the game. Therefore, we try to add this behavior to our model. Different from the other behaviors, "kill" only happens during a combat and is not time-varying, so it cannot be added to pb-CNN to extract features. We add it to pb-RNN as attention, and the attention is formulated as follows.

$$a_t = \frac{\sum d_t}{D_t} \tag{11}$$

where d_t is the sum of players "killed" in a team at the t-th combat and D_t is the sum of players who are "killed" at the t-th combat. If no "kill" happens in a combat, the value of the attention is 0.5.

For the output of pb-RNN, we apply attention to them to get the state h_t, and finally we can get vector h for prediction by the weighted average of h_t. Vector h contains the relationship of each behavior and interactions among all features in the sequence. Then, we have:

$$W = H(p_1, p_2, ..., p_t) = h = \sum \overline{a_t h_t} \tag{12}$$

By pb-RNN, whenever the game is going on, we can use behaviors and "kill" of each combat to predict the game outcome. We can also update pb-RNN at any time as the match goes on to improve the prediction accuracy. Figure 2 shows the framework of pb-RNN.

3.3 Game Outcome Prediction

We extract features in pb-CNN, and put the feature sequence in pb-RNN. Each of the two teams can get the prediction vectors h_1 and h_2, respectively. Then,

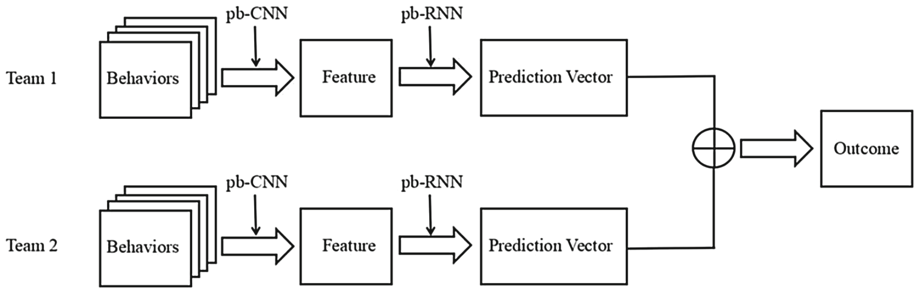


Fig. 3. Framework of the behavior model predicting MOBA game outcome

we combine these vectors and use softmax function to forecast the outcomes. Figure 3 shows the prediction process.

We construct the game outcome prediction model, called pb-model, by combining the output vectors of pb-RNN, h_1 and h_2, as the input of the full connect layer and softmax layer. In full connect layer, pb-model combines the features and maps them to the label. In softmax layer, pb-model gets the classification probability and chooses the max probability as the outcome of the game.

$$r = W(h_1, h_2) + b \tag{13}$$

$$p = softmax(r) \tag{14}$$

where W is a parameter matrix, and b is the bias vector. r is the result of the full connect layer. The objective of pb-model is to minimize the difference between the outcome p we calculate and the real outcome y. In pb-model, we use cross entropy function as our loss function:

$$J(\theta) = -\frac{1}{m}\sum_{i=0}(y^i \log p^i + (1-y^i)\log(1-p^i)) \tag{15}$$

4 Experiment

4.1 Setup

Considering that the game *DotA2* is the most popular MOBA game, we collected *DotA2* game data from Kaggle[3] to evaluate the effectiveness of our proposed pb-model. We chose data that contains temporal information, organized it by time and classified it into each game. In total, 20,000 game instances for model training were collected from 998 *DotA2* games. For each game, we chose the first 20 combats as training data, since the win-loss outcomes of most *DotA2* games are fixed in the first 20 combats.

[3] https://www.kaggle.com/.

In our empirical study, we considered 4 behaviors provided by the game data set, which are "ability", "item", "gold", and "subtype". In a game, the "ability" represents the skills that player character used or got, the "item" represents equipments that player character bought, the "gold" refers to the money that player character awarded and the "subtype" stands for information that broadcasted in the public channel, e.g., barracks were destroyed.

All experiments were performed on Ubuntu 16.04, Intel Xeon E5-2683 2.00 GHz CPU, 64G memory, and GTX1080 GPU. We used tensorflow to encode our program. In our pb-model, the parameters were randomly initialized. In pb-CNN, we chose RELU function as the active function, initialized embedding nominal randomly and set dimension to 200. The size of convolutional kernels was set as 3, 4, and 5, respectively while the sum of kernels was set as 100. In the training step, we set the min-batch as 64 and the learning rate as 0.001. We used Adam as the optimizer and SGD to optimize the loss function. We set the epoch of training to 500 and used early-stop for training.

4.2 Test on Prediction Accuracy

We evaluated the prediction accuracy of pb-model by 10 fold-cross-validation using *DotA2* game data set. In addition, to verify pb-model is insensitive to the game process, we define 3 kinds of game process as follows.

- "First 3": Refers to the first 3 combats in the beginning of the game, by which we test whether or not the outcomes of the game can be determined.
- "First 10": Refers to the first 10 combats of the game which comes to the middle of the game. Two teams have already had several combats so far, so the variation of behaviors has become obvious to some degree. At this time, the result of prediction is of great actual value.
- "First 20": Refers to the first 20 combats of the game, i.e., the end of the game, where the variation of behaviors is completely obvious. We get the prediction result of pb-model, and check if it is correct.

Table 2. Prediction accuracy w.r.t. the number of behaviors and the number of combats in *DotA2* game data

Player behaviors	Number of combats						
	1	3	10	14	16	18	20
Ability	0.5234	0.5468	0.5625	0.58	0.611	0.6032	0.6054
Ability, item	0.5273	0.6875	0.7265	0.7281	0.7226	0.7281	0.7375
Ability, item, gold	0.5078	0.5468	0.7312	0.71875	0.75	0.75	0.7549
Ability, item, gold, subtype	0.5062	0.6436	0.8056	0.8125	0.8556	0.875	0.8785

Table 2 lists the prediction accuracy of pb-model with respect to the number of behaviors and the number of combats. We can see that the accuracy got by the first 10 combats are very close to the accuracy got by all combats. In addition, it is clear to see that the accuracy of prediction has increased when considering more behaviors.

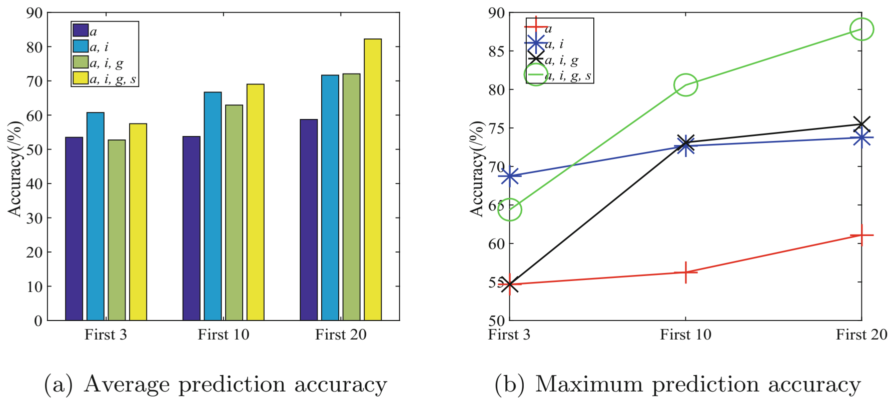


Fig. 4. Performance of pb-model (a: “ability”, i: “item”, g: “gold”, s: “subtype”)

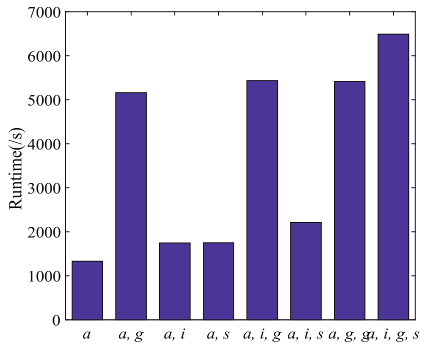
Figure 4 illustrates the average and maximum prediction accuracies of pb-model at different game process stages. From Fig. 4, we can see that:

(i) The win-loss outcome of the first 3 combats will not influence the outcome of a game. It is determined by the interaction of all combats rather than by any sole combat.
(ii) We used 4 behaviors in “First 20” and got an accuracy of prediction of 87.85%, which shows the high accuracy of our pb-model. Besides, the pb-model can be universally applied to all MOBA games, because the behaviors and combats produced in all MOBA games are easy to get.
(iii) Compared with the “First 20”, the result of the “First 10” is of greater value despite its accuracy is lower. This is because the outcome is expected to be predicted as early as possible. The result of the “First 20” has higher accuracy but comes too late, since the final outcome can be directly observed in the end of a game without any need for prediction. The result of the “First 10” is what expected with an accuracy of 80.56%. Its D-value is acceptable, and it demonstrates that pb-model is of value in application and prediction.

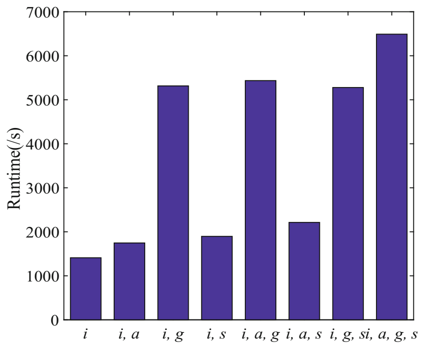
Figure 5 shows the training of pb-model with respect to the behaviors. We can see that the runtime increases greatly when involving behavior “gold”, since behavior “gold” is numeric, while the other three behaviors are nominal.

4.3 Test on Behaviors

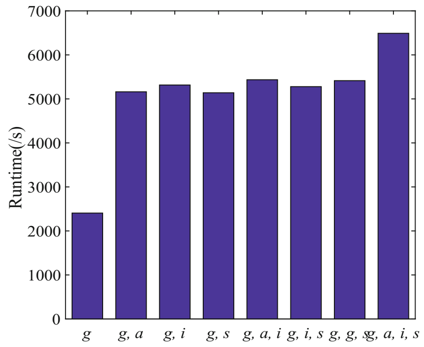
In Sect. 3.1, we found that all behaviors have a different contribution to the game outcome. In this experiment, we test the contributions of behaviors in the result of “First 20”.



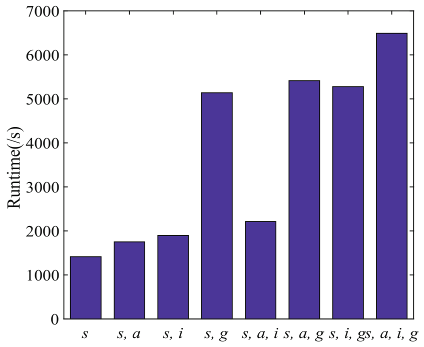
(a) Runtime of behaviors containing a



(b) Runtime of behaviors containing i



(c) Runtime of behaviors containing g

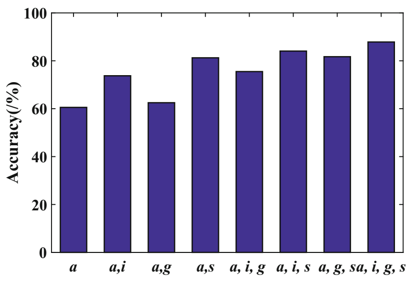


(d) Runtime of behaviors containing s

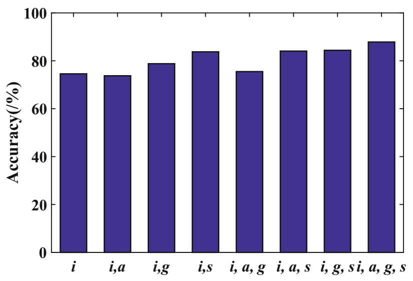
Fig. 5. Training time of pb-model w.r.t. behaviors (a: "ability", i: "item", g: "gold", s: "subtype")

Table 3. Prediction accuracy with respect to behavior

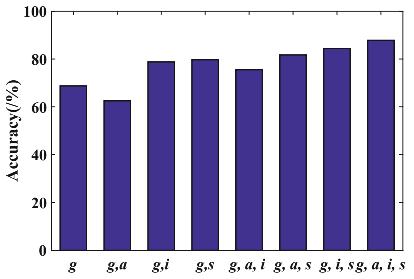
Player behavior	Accuracy (%)	Player behavior	Accuracy (%)
Ability	60.55	Item, subtype	83.75
Item	74.56	Gold, subtype	79.68
Gold	68.75	Ability, gold, item	75.5
Subtype	78.13	Ability, item, subtype	84.06
Ability, item	73.75	Ability, gold, subtype	81.71
Ability, gold	62.5	Item, gold, subtype	84.38
Ability, subtype	81.25	Ability, item, gold, subtype	87.85
Item, gold	78.79		



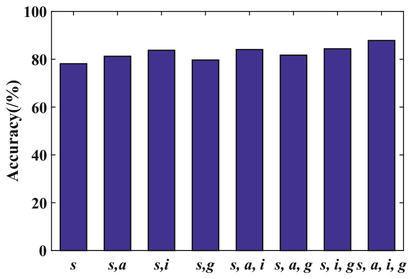
(a) Behaviors containing "ability"



(b) Behaviors containing "item"



(c) Behaviors containing "gold"



(d) Behaviors containing "subtype"

Fig. 6. Prediction accuracies w.r.t. behaviors (a: "ability", i: "item", g: "gold", s: "subtype")

We used all orders of behaviors as the input and gave the prediction accuracy in Table 3. As shown in the table, we can find that different behaviors have different contribution to the result, due to the same number of different behaviors as input but get the different result. Figure 6 illustrates the prediction accuracy contribution by different behaviors. We can see that the prediction accuracy increased slightly when considering "ability". That is, "ability" has the least accuracy contribution compared with other behavior. From Fig. 6, we can see that the rank of behavior contribution: "subtype", "item", "gold", "ability".

4.4 Comparison with CNN and LSTM

In our pb-model, we used pb-CNN to extract behaviors features and pb-RNN to process feature sequence. In this experiment, we will verify the effectiveness of our pb-model and compare it with the ones only using CNN or LSTM. When only using CNN, we only extracted features and combined these features to predict outcome directly. When only using LSTM, we only processed sequences and combined all behaviors without extracting the features from them. We compared our pb-model with CNN and LSTM in different number of behaviors and listed the prediction accuracy in Table 4.

Table 4. Comparation prediction accuracy with CNN and LSTM

Player behavior	CNN(%)	LSTM(%)	pb-model(%)
Ability	46.87	53.90	**60.54**
Ability, item	48.44	61.09	**73.75**
Ability, item, gold	52.18	70.31	**75.49**
Ability, item, gold, subtype	56.25	80.47	**87.85**

As listed in Table 4, CNN and LSTM disregard the influence on each other, so their accuracies are not high. Compared with them, our pb-model is the best one, since it can capture more information of the game than the other two.

5 Conclusions

This paper proposed a player behavior model to predict the win-loss outcome of MOBA games. Firstly, we constructed pb-CNN to extract features of all behaviors in combats. In order to model this combat-behavior structure, we combined these features into a feature sequence, and process it with a pb-RNN model. pb-RNN takes prediction vector as output, and uses softmax function to predict the win-loss outcome of the game. We performed experiments in DotA2 game data set and verified the proposed pb-model is effective. Our player behavior model can get acceptable prediction accuracy using the middle stage of game process. Thus, the pb-model is of great value in application.

The proposed behaviour model can be applied to at least three different purposes, with further research improving its usefulness. The first purpose is predicting the outcome of a match during a live eSports event. Here, the prediction accuracy grows throughout the game, because more combat instances are available for analysis as the game progresses. However, this would require the analysis to be faster than what is currently possible. The second purpose is making changes to the game design based on the model. An example of this need would be the finding that some player choices can provide an unwanted edge in the game, resulting in an unbalanced gameplay experience. The third purpose is to use the data to achieve more realistic gameplay simulations. Modelling player strategies based on the probability of players to engage in certain game features can be used to evaluate and improve a game rules and its design [2]. This model provides data that can be used in such simulations.

There are several interesting issues that deserve further research. First, having both numeric and nominal values together in the current setup in pd-CNN increases the time needed for the computations. We should consider a way to rearrange this differently to improve the efficiency. Second, in pd-CNN, the nominal behaviors were randomly initialized in embedding. We plan to adapt another embedding methods to get the relationship among such behaviors before training, to get a better starting point in the gradient descent, which can further contribute to making a prediction with higher accuracy.

References

1. Castaneda, L.M., Sidhu, M.K., Azose, J.J., Swanson, T.: Game play differences by expertise level in dota 2, a complex multiplayer video game. IJGCMS **8**(4), 1–24 (2016)
2. Nummenmaa, T.: Executable formal specifications in game development: design, validation and evolution. University of Tampere, Tampere University Press (2013)
3. Erickson, G.K.S., Buro, M.: Global state evaluation in StarCraft. In: Digital Entertainment (2014)
4. Rioult, F., Mtivier, J.P., Helleu, B., Scelles, N., Durand, C.: Mining tracks of competitive video games. AASRI Procedia **8**, 82–87 (2014)
5. Pobiedina, N., Neidhardt, J., del Carmen Calatrava Moreno, M., Werthner, H.: Ranking factors of team success. In: Proceedings of 22nd International World Wide Web Conference, pp. 1185–1194 (2013)
6. Nuangjumnong, T.: The influences of online gaming on leadership development. Trans. Comput. Sci. **26**, 142–160 (2016)
7. Drachen, A., et al.: Skill-based differences in spatio-temporal team behaviour in defence of the ancients 2 (dota 2). In: Proceedings of 2014 IEEE Games Media Entertainment, pp. 1–8 (2014)
8. Li, Q., et al.: A visual analytics approach for understanding reasons behind snowballing and comeback in MOBA games. IEEE Trans. Vis. Comput. Graph. **23**(1), 211–220 (2017)
9. Suznjevic, M., Matijasevic, M., Konfic, J.: Application context based algorithm for player skill evaluation in MOBA games. In: Proceedings of 2015 International Workshop on Network and Systems Support for Games, pp. 1–6 (2015)
10. Dereszynski, E.W., Hostetler, J., Fern, A., Dietterich, T.G., Hoang, T., Udarbe, M.: Learning probabilistic behavior models in real-time strategy games. In: Digital Entertainment (2011)
11. Weber, B.G., Mateas, M.: A data mining approach to strategy prediction. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Games, pp. 140–147 (2009)
12. Synnaeve, G., Bessière, P.: A Bayesian model for opening prediction in RTS games with application to StarCraft. In: Proceedings of 2011 IEEE Conference on Computational Intelligence and Games, pp. 281–288 (2011)
13. Wang, K., Shang, W.: Outcome prediction of DOTA2 based on Navïe Bayes classifier. In: Proceedings of 16th IEEE/ACIS International Conference on Computer and Information Science, pp. 591–593 (2017)
14. Cleghern, Z., Lahiri, S., Özaltin, O.Y., Roberts, D.L.: Predicting future states in DOTA 2 using value-split models of time series attribute data. In: Proceedings of the International Conference on the Foundations of Digital Games, pp. 5:1–5:10 (2017)
15. Yang, P., Harrison, B.E., Roberts, D.L.: Identifying patterns in combat that are predictive of success in MOBA games. In: Proceedings of the 9th International Conference on the Foundations of Digital Games (2014)
16. Park, Y.J., Kim, H.S., Kim, D., Lee, H., Kim, S.B., Kang, P.: A deep learning-based sports player evaluation model based on game statistics and news articles. Knowl.-Based Syst. **138**, 15–26 (2017)

17. Leibfried, F., Kushman, N., Hofmann, K.: A deep learning approach for joint video frame and reward prediction in Atari games. CoRR abs/1611.07078 (2016)
18. Oh, J., Guo, X., Lee, H., Lewis, R.L., Singh, S.P.: Action-conditional video prediction using deep networks in Atari games. In: Proceedings of Annual Conference on Neural Information Processing Systems 2015, pp. 2863–2871 (2015)