Speedup Training Artificial Intelligence for Mahjong via Reward Variance Reduction

Jinqiu Li*†, Shuang Wu‡, Haobo Fu‡, Qiang Fu‡, Enmin Zhao*†, and Junliang Xing§
*Institute of Automation, Chinese Academy of Sciences, Beijing, China
†School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
Email: {lijinqiu2021, zhaoenmin2018}@ia.ac.cn
‡Tencent AI Lab, Tencent, Shenzhen, China
Email: {shawnswu, haobofu, leonfu}@tencent.com
§Department of Computer Science and Technology, Tsinghua University, Beijing, China
Email: jlxing@tsinghua.edu.cn

***Abstract*—Despite significant breakthroughs in developing gaming artificial intelligence (AI), Mahjong remains quite challenging as a popular multi-player imperfect information game. Compared with games such as Go and Texas Hold'em, Mahjong has much more invisible information, unfixed game order, and a complicated scoring system, resulting in high randomness and variance of the rewarding signals during the reinforcement learning process. This paper presents a Mahjong AI by introducing Reward Variance Reduction (RVR) into a new self-play deep reinforcement learning algorithm. RVR handles the invisibility via a relative value network which leverages the global information to guide the model to converge to the optimal strategy under an oracle with perfect information. Moreover, RVR improves the training stability using an expected reward network to adapt to the complex, dynamic, and highly stochastic reward environment. Extensive experimental results show that RVR significantly reduces the variance in Mahjong AI training and improves the model performance. After only three days of self-play training on a single server with 8 GPUs, RVR defeats 62.5% opponents on the Botzone platform.**

***Index Terms*—Imperfect information game, multi-agent learning, reinforcement learning, Mahjong AI**

I. Introduction

As abstractions of the natural world, games provide an efficient testbed for Artificial Intelligence (AI) to learn complex control in both competitive and cooperative environments. Significant achievements have been made in various perfect information games, such as chess [1] and Go [2]–[5]. AlphaZero [4] introduces a general reinforcement learning framework to solve perfect information games without any domain knowledge by self-play, which outperforms human masters in the Go game. However, imperfect information games (IIGs) are still challenging due to the partially visible states.

Recent research mainly chooses Poker [6] as a benchmark to evaluate which has a two-player or multi-player form and a limited or non-limited form. This work aims at designing an intelligent Reinforcement Learning (RL) agent for Mahjong, the popular multi-player IIG in Asia. Mahjong is much larger than games like poker in the average size of information sets because of the game length and tile number. Due to the diversity of Mahjong game rules and the lack of high-quality human data, we choose to use RL to start from scratch. There are two critical problems when training Mahjong AI with RL. The first problem is the inaccurate estimation of the value. AI can only observe its own hands and public hands; however, opponents' hands also affect the value primarily. In Mahjong rules, if any player gets a specific card and chooses to end the game, other players get negative rewards. So the relative value of the global information in Mahjong is more important than the value of private information.The second problem is that there exists randomness in the distribution of rewards. Even at the end of the game, players get positive rewards with a considerable difference due to the different last tile and the different ways of getting the last tile (Dianpao, Zimo). This problem makes it difficult for reinforcement learning algorithms to learn the correct distribution of rewards.

To handle these above problems, we propose reward variance reduction (RVR) for Mahjong AI design. RVR integrates two techniques into the reinforcement learning framework. First, we propose a *relative value network* that considers not only the player's state but also other players' states. This network better gives the value of each state in the training process and helps the policy converge faster. Different from the global information value estimate in Honor of Kings [7], we consider the properties in Mahjong and give a new loss function for multi-player zero-sum games. Then we design an *expected reward network* to give the expected reward in the final state. Unlike giving true rewards with high variance, this network considers complex situations such as several players Shanten, the different types of the last tile, and the different ways of getting the last tile (Dianpao, Zimo) and gives the agents the expected rewards. It effectively reduces the variance of reward distribution in the Mahjong game.

To summarize, we in this work make the following three main contributions:

- We train an expert-level Mahjong AI from scratch with limited computing resources via self-play, defeating 62.5% opponents on the Botzone platform.
- We propose a relative value estimate algorithm that contains other players' information in each decision node

§ Corresponding author

TABLE I
RULES IN CHINESE STANDARD MAHJONG

Category	Type		Description	Number
Card	Ordinal tiles		Dot (1-9) / Bamboo (1-9) / Character (1-9)	each type has 4 tiles
	Honor tiles	Wind tiles	East / South / West / North	each type has 4 tiles
		Dragon tiles	Red / Green / White	each type has 4 tiles
Action	Discard		Discard a tile from your private hands.	34
	Chow		Make a straight of 3 Ordinal tiles like (1,2,3 Dot) by seizing the upper player's latest tile Discard.	63
	Pong		Make a Three-of-a-Kind of 3 identical tiles like (1,1,1 Dot) by seizing other players' latest tile Discard.	34
	Kong	Concealed-Kong	Make a Four-of-a-Kind of 4 identical tiles like (1,1,1,1 Dot) when the 4 tiles are in your private hands. The tiles are not revealed to other players.	34
		Exposed-Kong	Make a Four-of-a-Kind of 4 identical tiles like (1,1,1,1 Dot) by seizing other players' latest tile Discard.	34
		Add-Kong	Make a Four-of-a-Kind of 4 identical tiles like (1,1,1,1 Dot) by adding a tile to your exposed Three-of-a-Kind.	34
	Hu		Compose Legal hands by Draw a tile on your own (Zimo) or seizing other players' latest tile Discard (Dianpao). Zimo will get more Fans than Dianpao. It ends the game.	1
	Pass-Hu		Give up the action of Hu and then you should choose another legal action.	1
	Draw		Give up legal actions of Chow, Pong or Kong, and Draw a tile from the Remaining wall.	1
Terminology	Suit		The type of Ordinal tiles, i.e. Dot, Bamboo, Character.	3
	Fulu		The meld generated by action of Chow, Pong or Kong.	Up to 4 per player
	Discards		The tiles that player Discard.	Up to 24 per player
	Remaining wall		The tiles that player can Draw except initial hands.	Up to 21 per player
	Fan		The point is called Fan in Mahjong.	-
	Official hands		The Official hands is a part of hands that satisfy some special requirements, such as tiles should be same in the suit. Different Official hands come with different scores, 12 in total.	81
	Legal hands		Legal hands are generally in the form of four melds (straight, Three-of-a-Kind or Four-of-a-Kind) and a pair, or in the form of 7 pairs. Legal hands can belong to multiple Official hands, and the score of the hands is the sum of Fans of the corresponding Official hands.	-
	Shanten		When there is only one tile away from Legal hands.	-

to reduce the value's variance and guide the AI to make decisions under imperfect information as close as possible to the optimal strategy under perfect information.

- We design an expected reward network that trains the network during the self-play process and uses the network's output as the reinforcement learning reward signal to reduce the game's randomness and improve the training speed.

Based on the above technical contributions, our experimental results show that the proposed methods significantly reduce the variance and randomness among the Mahjong training and speed up the model training process.

II. PREREQUISITES

A. Chinese Standard Mahjong

There are many variations of Mahjong, usually distinguished by the type of cards and the actions players can do with a card. The rules of Chinese Standard Mahjong were formulated by the General Administration of Sport of China in 1998 and have become one of the two most popular competitive Mahjong in the world. Chinese Standard Mahjong has 144 cards with 36 types, and each type has four same cards. Thirty-six types could be further divided into three classes: Ordinal tiles include 27 types, Honor tiles include seven types, and Bonus tiles include eight types. Bonus tiles are usually ignored in the game. There are four players in each game of Mahjong, one of which is the banker, and each player is dealt with 13 cards at the beginning. Through a series of actions such as Discard, Chow, Pong, Kong, Draw, and Hu, players make their hands compose Legal hands to win. Please see Table I for details about the rules.

Mahjong is still an unsolved benchmark for multi-agent reinforcement learning. Three interesting properties make Mahjong particularly challenging to solve. Firstly, it is a complex game with much hidden information. The number of information sets represents the number of all possible decision nodes in an imperfect information game. The average size of the information sets represents the amount of hidden information behind each situation in the game. Mahjong is 10^{121} and 10^{48} in these two indicators, respectively, which are much larger than other games such as bridge and Texas Hold'em. The player only knows the 13 cards in his hands and the cards that have been played before, but he does not know the cards in other people's hands and the cards remaining on the wall. The visible information is much smaller than invisible information. With more invisible information, there will be more possible situations, making it difficult for AI to connect the final reward signals with the current observations during training. Secondly, Mahjong is not only complicated in terms of cards but also intricate in action rules. In a Mahjong game, players need to consider a variety of decision types: not only deciding which card to Discard in their round but also choosing Chi, Pong, Kong, or Hu in others' rounds. According to the rules, there are many types to finish a game, such as Draw a particular card to Hu, getting a specific card from others to Hu, and Draw all cards on the wall with no one Hu. In the case of Hu, different hand types have different scores, and some even differ by several times. Thirdly, unlike the previous two-player zero-sum game, many special issues in the multi-player game need to be considered. Mahjong players want to maximize the benefits, not only from their perspective but also from the opponent's perspective, the upper, and the next. In addition, any player's Pong and Kong may change

the current playing order, which causes the game tree to be irregular and dynamic.

B. Reinforcement Learning

In self-play, given an opponent whose policy is known and fixed, the original four-player Mahjong game reduces to a single-player RL problem. We consider the standard Markov Decision Process (MDP). An MDP consists of a set of *states* $\mathcal{S} = \{s_0, s_1, s_2, \dots, s_t, \dots\}$, a set of *actions* $\mathcal{A} = \{a_k\}_{k=1}^{K}$, and a *reward function* $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$. After executing an action $a_t \in \mathcal{A}$ at each state $s_t \in \mathcal{S}$, the agent will enter a new state s_{t+1} according to the transition probability model and get a reward $r(s_{t+1}|s_t, a_t)$. The objective of the agent is to maximize the cumulative rewards $R = \sum_{t=0}^{\infty} \gamma^t r(s_{t+1}|s_t, a_t)$, where γ is the *discount factor* to favor more recent rewards. In RVR Mahjong AI, we employ the popular Proximal Policy Optimization (PPO) [8] learning algorithm.

III. Related Work

A. Reinforcement Learning in Imperfect Information Games

Cepheus [9], DeepStack [10], Libratus [11], and Pluribus [12] have made breakthroughs in two-player and six-player Texas Hold'em using Counterfactual Regret Minimization (CFR) [13]–[15]. CFR relies on the game tree and converges to Nash equilibrium after multiple iterations by calculating regrets. Nevertheless, it is difficult to apply in games with large state space because of the complicated game tree. Recently, Rebel [16], AlphaHoldem [17], DeltaDou [18], and DouZero [19] use reinforcement learning and achieve significant progress in two-player Texas Hold'em and Doudizhu. Benefited from the growth of deep neural networks, complex games with multiple units and complex scenarios such as StarCraft II [20], Dota2 [21], and Honor of Kings [7], [22] have achieved outstanding performance by deep reinforcement learning methods.

B. Mahjong AI Design

Efforts in Mahjong mainly focus on the following aspects. The first one is the knowledge-based search algorithm [23], which mainly simplifies the huge state and action space of Mahjong by previous human knowledge and rules. In 2020, [24] builds an effective search tree by abstracting Japanese Mahjong into multiple Markov decision processes (MDPs) according to the hands. However, there are still some shortcomings: the model's ability is minimal if the designer makes unreasonable assumptions or abstractions.

The second one is the supervised learning algorithm. In 2014, the University of Tokyo adopted the supervised learning method to train a Mahjong AI by using many high-level human players' data. Their AI's average score [25] reached six dan in the Tenhou platform[1] and was rated above 96.6% of the Japanese Mahjong players. However, the model's performance based on supervised learning usually fluctuates wildly during the complete Mahjong game training.

[1]Obtained from: https://tenhou.net/

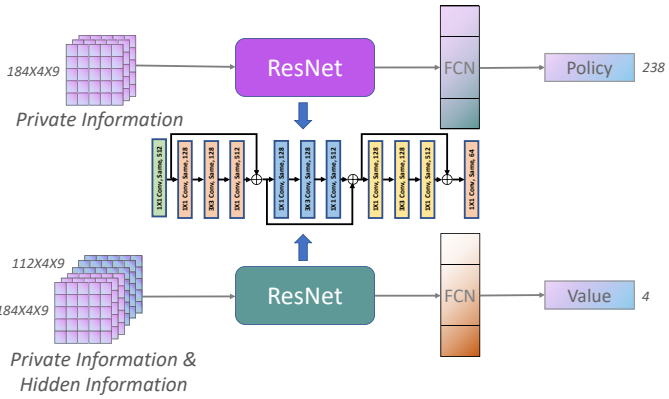

Fig. 1. RVR architecture. RVR network consists of a ResNet [28] to encode cards and 2 layers of FCN with hidden dimension of 1024. The network predicts the action based on the private information, and predicts the value base on both the private information and the hidden information.

The third one is the reinforcement learning algorithm. Microsoft Research Asia develops a Japanese Mahjong AI named Suphx [26], which includes five deep convolution neural network models and a rule-based winning model. It uses a gradient-based reinforcement learning algorithm to update the network. It uses a parametric Monte Carlo strategy to dynamically adjust the offline strategy before the game in real-time to better adapt to the current state. Suphx achieved ten dan in the Tenhou platform, which outperforms 99.99% of officially ranked human players. It is currently the strongest Japanese Mahjong AI. However, this model requires many computing resources and is difficult to apply directly to the Chinese Standard Mahjong due to a large amount of human data. Actor-Critic Hedge [27] extends the actor-critic algorithm framework in deep reinforcement learning and solves 1-on-1 Mahjong.

IV. Methods

To design a Mahjong AI, we provide a deep RVR architecture equipped with the RL schema to learn policy and value functions. We illustrate the RVR architecture in Fig. 1. As shown in Fig. 1, it contains a policy network like the traditional RL network and a global information value network. The input of the policy network is the game state representations of private information, and the input of the value network is the game state representations of private information and hidden information. RVR network gives better value estimates during training and does not use extra information during testing. To better solve the problem of variance in Mahjong, we propose a new loss function that considers the relative value relation among the whole four players for the value network. This design is the reason that we call it a relative value network. Also, we present an expected reward network to reduce the randomness in Mahjong. In the following, we highlight game state representation in Mahjong and these two networks' training methods. We believe these new techniques and underlying principles help develop general learning algorithms for more IIG AIs.

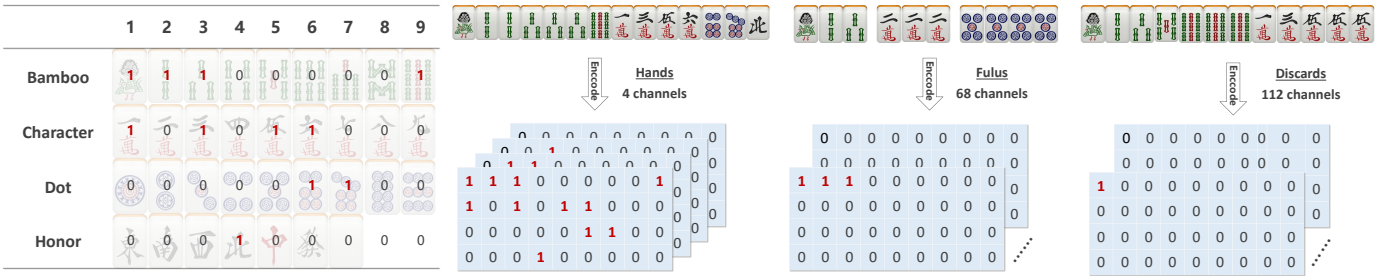

Fig. 2. Encoding of Mahjong state. We encode the Mahjong tiles into a 4×9 binary matrix, with each column corresponding to one number, and each row corresponding to one suit. The private hands, Fulus, and Discards are encoded into 4 channels, 68 channels, and 112 channels, respectively. Taking the private hands as an example, the i-th row in the j-th column of the n-th channel means there are n tiles of the i-j type in the hands.

A. State Representation

The states of Mahjong games can be divided into two categories: visible and invisible information. The visible information that the player can access includes his hands, everyone's Fulus, and everyone's Discards. The invisible information includes the opponent's hands and the remaining wall. Since Mahjong is not naturally encoded in image format like Go, in this paper, we design unique encoding methods for Mahjong based on prior human knowledge to facilitate the neural network training process. As shown in Fig. 2, we use a 4×9 binary matrix to represent the Mahjong state, where columns correspond to the nine Ordinal tiles and Honor tiles, and rows correspond to the four suits. This encoding method unifies the suits and numbers of Ordinal tiles, making learning the relationship more accessible. Winning a Mahjong game is to make the private hands compose Legal hands. The private hands largely determine the subsequent game strategy, so we use four channels to represent it, the i-th row in the j-th column of the n-th channel means there are n tiles of the i-j type in the hands. We use 68 channels to represent Fulus, which can help to infer the rest of the players' hands and playing strategies. The 68 channels contain channels of each player's Chow melds (4×4), the tiles in Chow melds seized from other players (4×4), Pong melds (4×4), Kong (except Concealed-Kong) melds (4×4) and the player's own Concealed-Kong melds (4×1). Based on the currently known information, it is possible to roughly estimate the distribution of the remaining cards to help make better decisions, so we use 112 channels to represent the Discards, including channels of each player's sequence of historical Discards (24×4, one card one channel) and each player's counts of historical Discards (4×4). The private information is encoded into 184 channels in summary. Similarly, we use another 112 channels to represent the hidden information, including the channels of opponent's hands (4×3), each player's sequence of Remaining wall (21×4, one card one channel), and each player's counts of Remaining wall (4×4). We encode the Mahjong actions into 238 dimensions according to Discard, Chi, *etc*. The policy network outputs the probability of 238 actions being selected.

B. Relative Value Network

One key factor in training the deep architecture with the above state representation is the learning paradigm with a suitable value estimator. The value of the current state depends not only on the visible information but also on the hidden information. Without the hidden information, it is difficult for players to determine their relative value, which is one of the fundamental reasons Mahjong games are challenging to solve. [26] uses knowledge distillation and trains a value network with high-performance human data. This value network is trained by supervised learning using global information and then makes the output of the value network in reinforcement learning as close to this network as possible. However, without high-quality data, the Mahjong training process makes it difficult to give an effective value with limited observations.

To solve this problem, we modify the variant value network to get the global state information, while the strategy network still uses the original visible information as input. To make the value network learn the relative relation of the four players' values, the output of the network is all four players' value V_θ. The optimization objective is shown in (1):

$$\arg\min_{\theta} \frac{1}{|D_k|\,T} \sum_{\tau \in D_k} \sum_{t=0}^{T} \sum_{j=1}^{4} \left(V_\theta \left(s_t^j, h_t^j \right) - r_t^j \right)^2, \quad (1)$$

where θ denotes the parameters of the relative value network, $|D_k|$ denotes the number of trajectories of the training data in the k-th iteration, T denotes the total length of the trajectory, s_t^j denotes the state for the j-th player in the t-th round, h_t^j denotes the hidden information for the j-th player in t-th round, $V_\theta(s_t^j, h_t^j)$ denotes the value predicted by the value network, and r_t^j denotes the cumulative return with discount factor of the j-th player in the t-th round. We guarantee that the sum of all four players' rewards is 0 at any round. By Lagrange's theorem, such a loss function can be designed to make the sum of the estimated values 0. In a sense, this is similar to the idea of the popular operation batch normalization [29] in machine learning.

Some previous works [7], [22] also report that better results can be achieved by using global information for guidance. Different from the global value network in [7], we consider the relative value in the Mahjong game and change the loss function for the value network. The difference between variant value network, global value network, and relative value network is shown in Fig. 3. Compared with variant value network and global value network, relative value network

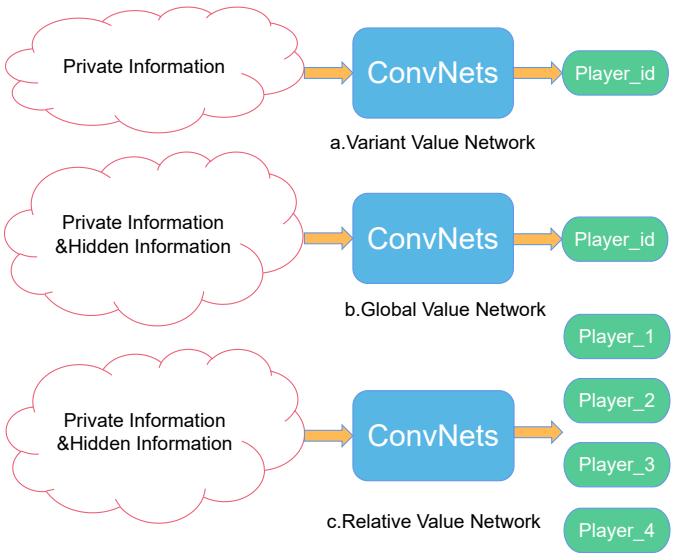

Fig. 3. Comparison diagram among variant value training structure, global value training structure, and relative value training structure. Relative states' value ensures that the sum of the states' values in the training process is 0.

considers the properties of a four-player zero-sum game. It makes the value estimation more accurate by this loss function.

C. Expected Reward Network

Due to the great randomness in Mahjong, using the raw reward of the game as a training signal for reinforcement learning may not indicate how good or bad the strategy is, especially for top professional players. For example, in the game's final stage, two players are only one step away from the Legal hands. They can both Draw the card they need from the remaining wall, which means that both players have the possibility of Zimo to win by themselves. Whoever Draw the needed tile first will win, while the other player will lose many Fans, and the final reward may be very different due to the last tile Draw. However, this negative reward is not owing to bad strategy but the randomness of the Mahjong game. Since the number of remaining tiles in the remaining wall may be significant, it is difficult for AI to explore those data randomly, which results in slow convergence of the model.

For the reason above, we need a new reward signal to eliminate irrelevant factors such as randomness for better evaluation of strategies and a good guide of optimization directions. Nowadays, few researchers have researched this area. To solve this problem, we introduce the expected reward network, which can predict each player's expected reward based on the game's final state to help the training process be smoother and improve the training speed. The training data for the expected reward network is derived from the reinforcement learning process via self-play, and the optimization objective is shown in (2):

$$\arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left(f_\theta \left(g_i^{T-1} \right) - r_i \right)^2 \tag{2}$$

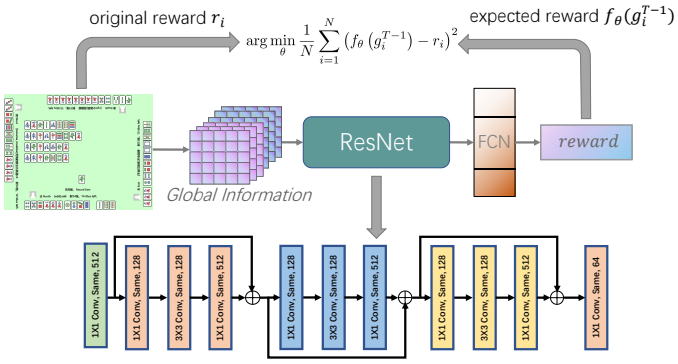

Fig. 4. The training process of the expected reward network. It derives training data from the reinforcement learning process via self-play to predict each player's expected reward based on the game's final state.

TABLE II
MODEL PERFORMANCE AGAINST SEARCH MODELS

Environment	Model	AF	WR
Simplified Mahjong	RL-1	+6.94%	+3.96%
	RL-2	+4.10%	+1.43%
	RL-3	+7.46%	+4.91%
Chinese Standard Mahjong	RL-1	+5.22%	+6.34%
	RL-2	+3.81%	+2.10%

where θ denotes the parameter of the expected reward network, N denotes the number of games in the training data, r_i denotes the i-th game of four players' reward in T-th round (i.e. the last round), g_i^{T-1} denotes the state in i-th game $(T-1)$-th round, which includes the four players' hands, Fulus, and remaining wall. After the training of the expected reward network, we use the output of this network as the new reward signal in $(T-1)$-th round for each player during the training process. The training process is shown in Fig. 4. Similar to the relative value network, the expected reward network considers global information and gives better rewards with less randomness for Mahjong games.

V. EXPERIMENTS

A. Experimental Setup

We run all experiments on a single server with 56 Intel Xeon E5-2690 CPU @2.60GHz and 8 TITAN V GPUs. We set the batch size $M = 2048$, $\lambda = 0.995$, the discount factor $\gamma = 0.95$, and adopt Adam optimizer with a learning rate $\phi = 0.0003$. For parameters of PPO [8], we set the policy clip range to 0.2, the value clip range to 0.2, and the gradient clip range to 0.5.

B. Ablation Experiments

1) Simplified Mahjong: The Chinese Standard Mahjong is very complex and not convenient for an experimental environment. While retaining the basic rules and unique challenges, we designed a simplified Mahjong environment, retaining only 1Dot to 6Dot, 1Bamboo to 6Bamboo, East and Red, with three cards of each kind, a total of 42. The number of players is reduced to three, the actions including Chi, Pong, and Kong are

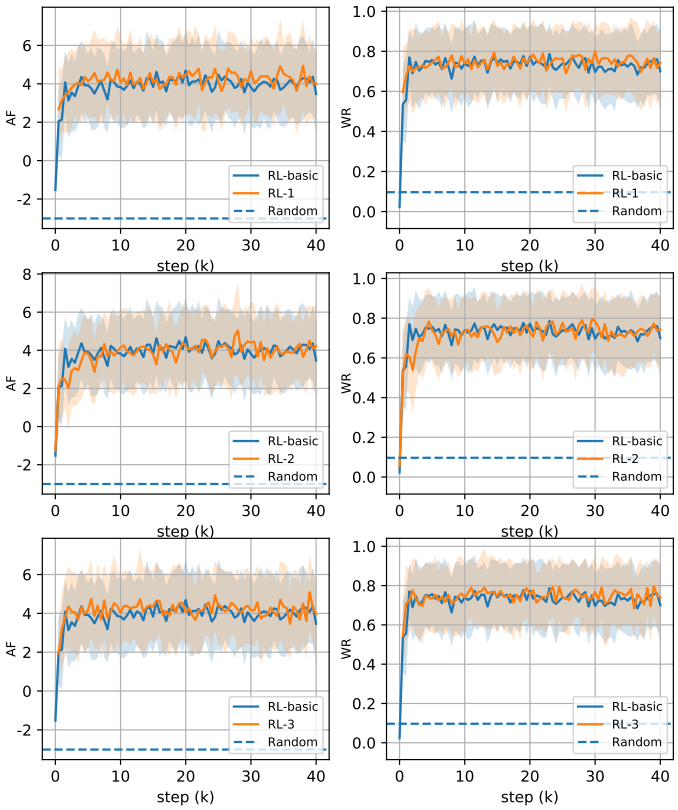

Fig. 5. Result in Simplified Mahjong. The plot shows the AF and WR of the Mahjong agents using different components against Search model w.r.t. the number of training steps in Simplified Mahjong.

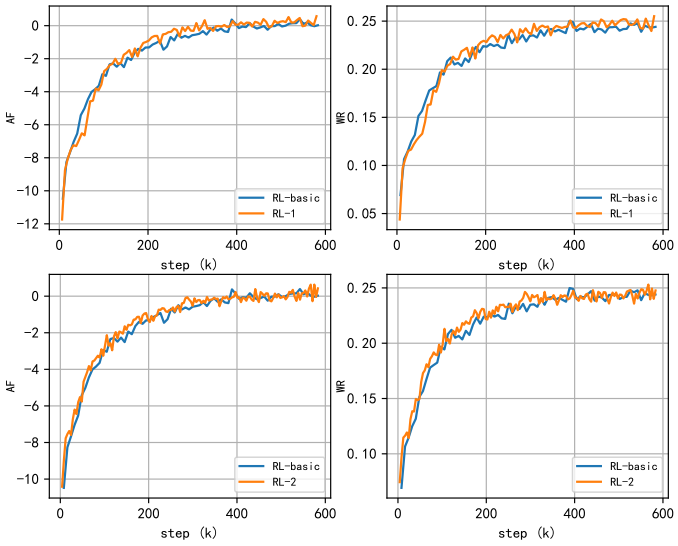

Fig. 6. Result in Chinese Standard Mahjong. The plot shows the AF and WR of the Mahjong agents using different components against Search model w.r.t. the number of training steps in Chinese Standard Mahjong.

reserved, and the number of Official hands types is reduced to four. This simplification reduces the number and average size of information sets and dramatically reduces the necessary training resources and time.

2) Metrics: Inspired by the duplicate bridge, we use the Swiss-system tournament and Duplicate Format, in which each matched three players will play a 100-deck wall, and each wall will play 6 games, so a total of 600 games will be played. Specifically, 6 games per wall mean that the three players are seated in full alignment. The scoring rule is an average of 6 game scores. In this paper, we use two metrics to compare the performance of the model:

- **AF** (Average Winning Fan): The average winning Fan per game. The average score of each game is zero-sum.
- **WR** (Winning Percentage): The number of games won by a player divided by the total number of games.

The different handling of the same hands by different players will lead to various changes in the state. With the same number of games, this evaluation method can reduce randomness and reflect the model's ability more effectively. In this way, the ranking is more persuasive.

3) Model Details: To demonstrate the value of each component, we trained several Mahjong agents via self-play from scratch:

- **RL-basic**: the basic version of the reinforcement learning agent.
- **RL-1**: the RL agent that enhances RL-basic with relative value technique.
- **RL-2**: the RL agent that enhances RL-basic with expected reward technique.
- **RL-3**: the RL agent that enhances RL-basic with both relative value technique and expected reward technique.

To evaluate those models, we also designed a **Search model** using normal search and pruning algorithms based on rules. It first searches for the tiles that need to be replaced between Legal hands and the private hands after performing each legal action, then calculate the probability of obtaining these tiles based on the number of tiles remaining, and uses their cumulative product as the probability to Hu of each legal action. The estimated value of each legal action is obtained by multiplying the probability mentioned above by the score of Legal hands. The search model outputs the action with the highest estimated value.

4) Results: In the Simplified Mahjong, we play the above model against the Search model in the other two positions as fixed opponents. The model with the best performance in the last 2,000 iterations is chosen as the final model. Fig. 5 reports the performance of the model above against Search models in the Simplified Mahjong. Table II summarizes the AF and WR improvements over the RL-basic of each model. Compared with RL-basic, RL-1 improves AF by 6.94% and WR by 3.96%. RL-2 improves AF by 4.1% and WR by 1.43% with no improvement in convergence speed. It shows that the enhancement of RL-1 is more vital than that of RL-2, mainly because the reward signal first affects the estimation of value and then indirectly guides the improvement of the strategy. Hence, the direct improvement of the value loss function makes the estimation more accurate from the global perspective and has a more significant improvement effect.

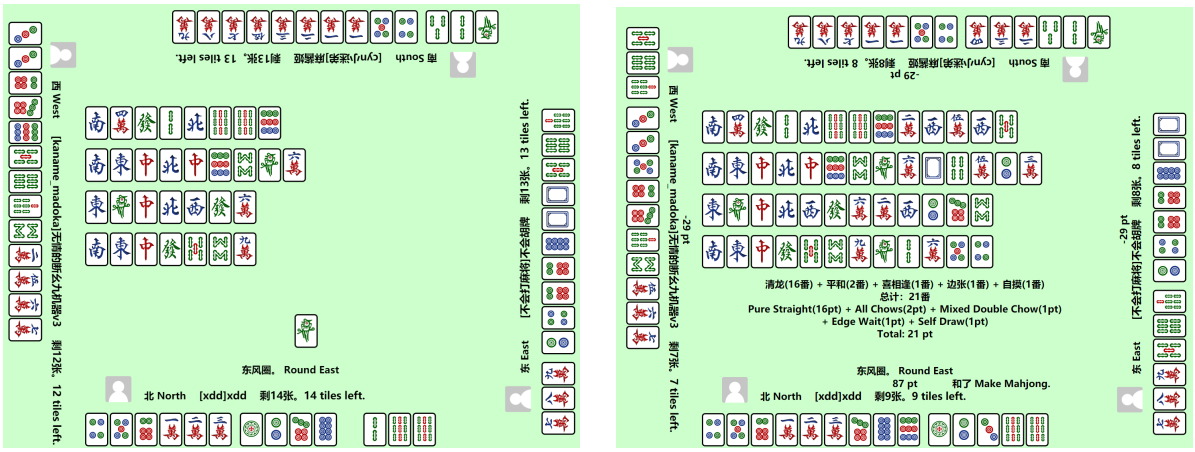

Fig. 7. A case study obtained from Botzone. The agent learns to draw 1Bamboo to compose a Pure Straight (a type of Official hands) and win more Fans. Note that the opponent's cards are hidden in the real game and shown here for better visualization

RL-3 improves AF by 7.46% and WR by 4.91%, which is more effective than RL-1 and RL-2.

C. Experiments in Chinese Standard Mahjong

1) Offline Experiments: In the Chinese Standard Mahjong, we play the RL-basic, RL-1, and RL-2 model against the Search model in the other three positions as fixed opponents. Fig. 6 report the performance curve of the model above against Search models in the Chinese Standard Mahjong. Table II summarizes the AF and WR improvements over the RL-basic of each model. Like the experimental results in the Simplified Mahjong, RL-1 improves AF by 5.22% and WR by 6.34%. As shown in Table II, RL-2 improves AF by 3.81% and WR by 2.1% with faster convergence of the model. The difference in convergence effect between the Simplified Mahjong and the Chinese Standard Mahjong is mainly because the number of Official hands types is less in the former than in the latter. Most Legal hands are normal hands rather than Official hands, so that the final score difference between players is not apparent.

2) Online Experiments: Because of the extensive resources required for model training, we do not test the ability of the RL-3 model on the Chinese Standard Mahjong environment in this paper, but through the online platform Botzone [30]. Botzone is a general AI competition platform founded by Peking University. It aims to measure the level of AI programs through game competition and evaluate their ability by ELO scoring system. The platform is currently open to various game modes such as Gluttony, Go, Landlord, and Chinese Standard Mahjong. In 2020, the Botzone platform and the International Joint Conference on Artificial Intelligence (IJCAI) held the IJCAI 2020 Mahjong AI Competition, whose finalists are basically at the level of advanced players. However, there is still a big gap between them and the top players. In this paper, we compare our Mahjong AI's ELO score in Tianti Leaderboard with the top 16 finalists (among which the top 3 models are not added to the Tianti Leaderboard). Our Mahjong AI ranked 4th and beat 62.5% of the top 16 players.

3) Verification of variance reduction on offline Mahjong data: To verify whether the relative value network reduces the variance of value, we selected three models, including RL-basic, RL-global, and RL-1, and calculated their variance on offline Mahjong data. The **RL-global** mentioned above is the RL agent that enhances RL-basic with the global information, including both private information and hidden information [7]. The only difference between RL-global and RL-1 is that the value network of the former has 1 output for his own, and the latter has 4 outputs for all players. We selected 10,000 rounds of match data and calculated the variance as (3).

$$variance = \frac{1}{N}\sum_{i=1}^{N}(v_i - R_i)^2 \tag{3}$$

The variance of the three models RL-basic, RL-global, and RL-1 is 6.36, 5.59, and 4.44, respectively. It shows that the variance of RL-global is smaller than that of RL-basic, and the variance of RL-1 is further reduced than that of RL-global. This result verifies that the relative value network reduces the value variance compared with global information.

4) Case Study: We conduct case studies to understand the decisions made by our Mahjong AI. As shown in Fig. 7, since the number of 3Dot and 3Bamboo remaining in the wall is the same, the probabilities of choosing 1Dot, 2Dot, 3Dot, and 1Bamboo, 2Bamboo, 3Bamboo to Draw are equal. However, 2Dot and 2Bamboo can compose more straights than 1Dot and 1Bamboo, so 1Dot and 1Bamboo should be drawn in this round. Our AI chooses to draw 1Bamboo because it can compose a Pure Straight (a type of Official hands containing tiles of the same suit from 1 to 9 sequences and is worth 16 Fans), while 1Bamboo cannot compose any Official hands.

D. Discussions

To solve the randomness problem in Mahjong games, we introduce the expected reward network. However, the Mahjong game's randomness does not only exist in the last round but also in every round of the game, especially in the first round when each player is dealt with 13 tiles. It affects the

direction of the game to a great extent. Therefore, better reward signals should be designed to remove the effect of these randomnesses. We will leave this problem to future research.

VI. Conclusions and Future Work

This work has presented a training-efficiency Mahjong AI by introducing reward variance reduction into a new self-play deep reinforcement learning algorithm. It is realized in a deep architecture with relative value network and expected reward network to reduce variance and randomness in the gaming process of Mahjong. Extensive experimental results have verified that the proposed RVR model significantly reduces the variance in Mahjong AI training and improves the model performance over competing baselines. Compared with the state-of-the-art Mahjong AIs, RVR consumes much fewer training resources with satisfactory gaming performances. In future work, we plan to perform deep analyses of the model policy evolutionary process and further improve the training efficiency by incorporating model parallel learning via distributed parallel computing.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China under Grant No. 2020AAA0103401, in part by the Natural Science Foundation of China under Grant No. 62076238 and 61902402, in part by the CCF-Tencent Open Fund, and in part by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant No. XDA27000000.

References

[1] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu, "Deep blue," *Artificial Intelligence*, vol. 134, no. 1-2, pp. 57–83, 2002.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[4] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[5] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[6] J. Rubin and I. Watson, "Computer poker: A review," *Artificial Intelligence*, vol. 175, no. 5-6, pp. 958–987, 2011.

[7] D. Ye, Z. Liu, M. Sun, B. Shi, P. Zhao, H. Wu, H. Yu, S. Yang, X. Wu, Q. Guo *et al.*, "Mastering complex control in moba games with deep reinforcement learning," in *AAAI Conference on Artificial Intelligence*, 2020, pp. 6672–6679.

[8] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[9] M. Bowling, N. Burch, M. Johanson, and O. Tammelin, "Heads-up limit hold'em poker is solved," *Science*, vol. 347, no. 6218, pp. 145–149, 2015.

[10] M. Moravcik, M. Schmid, N. Burch, V. Lisy, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "DeepStack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[11] N. Brown and T. Sandholm, "Superhuman AI for heads-up no-limit poker: Libratus beats top professionals," *Science*, vol. 359, no. 6374, pp. 418–424, 2018.

[12] N. Brown and T. Sandholm, "Superhuman AI for multiplayer poker," *Science*, vol. 365, no. 6456, pp. 885–890, 2019.

[13] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, "Regret minimization in games with incomplete information," in *Advances in Neural Information Processing Systems*, 2007, pp. 1729–1736.

[14] M. Lanctot, K. Waugh, M. Zinkevich, and M. Bowling, "Monte Carlo sampling for regret minimization in extensive games," in *Advances in Neural Information Processing Systems*, 2009, pp. 1078–1086.

[15] E. G. Jackson, "Slumbot NL: Solving large games with counterfactual regret minimization using sampling and distributed processing," in *AAAI Conference on Artificial Intelligence Workshops*, 2013, pp. 35–38.

[16] N. Brown, A. Bakhtin, A. Lerer, and Q. Gong, "Combining deep reinforcement learning and search for imperfect-information games," in *Advances in Neural Information Processing Systems*, 2020, pp. 17 057–17 069.

[17] E. Zhao, R. Yan, J. Li, K. Li, and J. Xing, "Alphaholdem: High-performance artificial intelligence for heads-up no-limit texas hold'em from end-to-end reinforcement learning," in *AAAI Conference on Artificial Intelligence*, 2022, pp. 1–9.

[18] Q. Jiang, K. Li, B. Du, H. Chen, and H. Fang, "Deltadou: Expert-level Doudizhu AI through self-play," in *International Joint Conferences on Artificial Intelligence*, 2019, pp. 1265–1271.

[19] D. Zha, J. Xie, W. Ma, S. Zhang, X. Lian, X. Hu, and J. Liu, "DouZero: Mastering Doudizhu with self-play deep reinforcement learning," in *International Conference on Machine Learning*, 2021, pp. 12 333–12 344.

[20] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[21] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[22] D. Ye, G. Chen, W. Zhang, S. Chen, B. Yuan, B. Liu, J. Chen, Z. Liu, F. Qiu, H. Yu, Y. Yin, B. Shi, L. Wang, T. Shi, Q. Fu, W. Yang, L. Huang, and W. Liu, "Towards playing full MOBA games with deep reinforcement learning," in *Advances in Neural Information Processing Systems*, 2020, pp. 621–632.

[23] N. Mizukami and Y. Tsuruoka, "Building a computer mahjong player based on Monte Carlo simulation and opponent models," in *IEEE Conference on Computational Intelligence and Games*, 2015, pp. 275–283.

[24] M. Kurita and K. Hoki, "Method for constructing artificial intelligence player with abstractions to Markov decision processes in multiplayer game of mahjong," *IEEE Transactions on Games*, vol. 13, no. 1, pp. 99–110, 2021.

[25] N. Mizukami, R. Nakahari, A. Ura, M. Makoto, Y. Tsuruoka, and T. Chikayama, "Realizing a four-player computer Mahjong program by supervised learning with isolated multi-player aspects," *Transactions of Information Processing Society of Japan*, vol. 55, no. 11, pp. 2410–2420, 2014.

[26] J. Li, S. Koyamada, Q. Ye, G. Liu, C. Wang, R. Yang, L. Zhao, T. Qin, T.-Y. Liu, and H.-W. Hon, "Suphx: Mastering Mahjong with deep reinforcement learning," *arXiv preprint arXiv:2003.13590*, 2020.

[27] H. Fu, W. Liu, S. Wu, Y. Wang, T. Yang, K. Li, J. Xing, B. Li, B. Ma, Q. Fu, and W. Yang, "Actor-critic policy optimization in a large-scale imperfect-information game," in *International Conference on Learning Representations*, 2022, pp. 1–12.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[29] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How does batch normalization help optimization?" in *Advances in Neural Information Processing Systems*, pp. 7515–7523, 2018.

[30] H. Zhou, H. Zhang, Y. Zhou, X. Wang, and W. Li, "Botzone: an online multi-agent competitive platform for ai education," in *ACM Conference on Innovation and Technology in Computer Science Education*, 2018, pp. 33–38.