PerfectDou: Dominating DouDizhu with Perfect Information Distillation

Guan Yang [*] [1] Minghuan Liu [*] [2] Weijun Hong [1] Weinan Zhang [2] Fei Fang [3] Guangjun Zeng [1] Yue Lin [1]

Abstract

As a challenging multi-player card game, DouDizhu has recently drawn much attention for analyzing competition and collaboration in imperfect-information games. In this paper, we propose PerfectDou, a state-of-the-art DouDizhu AI system that dominates the game, in an actor-critic framework with a proposed technique named perfect information distillation. In detail, we adopt a perfect-training-imperfect-execution framework that allows the agents to utilize the global information to guide the training of the policies as if it is a perfect information game and the trained policies can be used to play the imperfect information game during the actual gameplay. To this end, we characterize card and game features for DouDizhu to represent the perfect and imperfect information. To train our system, we adopt proximal policy optimization with generalized advantage estimation in a parallel training paradigm. In experiments we show how and why PerfectDou beats all existing AI programs, and achieves state-of-the-art performance.

1. Introduction

With the fast development of Reinforcement Learning (RL), game AI has achieved great success in many types of games, including board games (e.g., Go (Silver et al., 2017b), chess (Silver et al., 2017a)), card games (e.g., Texas Hold'em (Brown & Sandholm, 2018), Mahjong (Li et al., 2020)), and video games (e.g., Starcraft (Vinyals et al., 2019), Dota (Berner et al., 2019)). As one of the most popular card games in China, DouDizhu has not been studied in depth until very recently. In perfect-information games such as Go, agent can observe all the events occurred previously including initial hand of each agent and all agents' actions. In contrast, DouDizhu is an imperfect-information game with special structure, and an agent does not know other agents' initial hands but can observe all agents' actions. One challenge in DouDizhu is that it is a three-player game with both competition and collaboration: the two *Peasant* players need to cooperate as a team to compete with the third *Landlord* player. In addition, DouDizhu has a large action space that is hard to be abstracted for search-based methods (Zha et al., 2021).

Although various methods have been proposed for tackling these challenges (You et al., 2019; Jiang et al., 2019), they are either computationally expensive or far from optimal, and highly rely on abstractions with human knowledge (Zha et al., 2021). Recently, Zha et al. (2021) proposed DouZero, which applies simple Deep Monte-Carlo (DMC) method to learn the value function with pre-designed features and reward function. DouZero is regarded as the state-of-the-art (SoTA) AI system of DouDizhu for its superior performance and training efficiency compared with previous works.

Unfortunately, we find DouZero still has severe limitations in many battle scenarios, which will be characterized in Section 6.6. To establish a stronger and more robust DouDizhu bot, in this paper, we present a new AI system named PerfectDou, and show that it leads to significantly better performance than existing AI systems including DouZero. The name of our program follows the key technique we use – perfect information distillation. The proposed technique utilizes a Perfect-Training-Imperfect-Execution (PTIE) framework, a variant of the popular Centralized-Training-Decentralized-Execution (CTDE) paradigm in multi-agent RL literature (F. et al., 2016; L. et al., 2017). Namely, we feed perfect-information to the agent in the training phase to guide the training of the policy, and only imperfect-information can be used when deploy the learned policy for actual game play. Correspondingly, we further design the card and game features to represent the perfect and imperfect information. To train PerfectDou, we utilize Proximal Policy Optimization (PPO) (Schulman et al., 2017) with Generalized Advantage Estimation (GAE) (Schulman et al., 2015) in a distributed training system.

In experiments, we show PerfectDou beats all the existing DouDizhu AI systems and achieves the SoTA performance in a 10k-decks tournament; moreover, PerfectDou is the

[*] Equal contribution. Yang is responsible for the basic idea, system design and implementation details; Liu mainly contributes to the methodology, writing and experimental design. [1]NetEase Games AI Lab [2]Shanghai Jiao Tong University [3]Carnegie Mellon University. Correspondence to: Yue Lin <gzlinyue@corp.netease.com>.

Preprint.

most training efficient, such that the number of samples required is an order of magnitude lower than the previous SoTA method; for application usage, PerfectDou can be deployed in online game environment due to its low inference time.

2. Preliminaries

Imperfect-Information Extensive-Form Games. An imperfect-information extensive-form (or tree-form) game can be described as a tuple $G = (\mathcal{P}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, \mathcal{T}, \chi, \rho, r, \mathcal{I})$, where $\mathcal{P}$ denotes a finite set of *players*, $\mathcal{A}$ is a finite set of actions, and $\mathcal{H}$ is a finite set of *nodes* at which players can take actions and are similar to states in an RL problem. At a node $h \in \mathcal{H}$, $\chi : \mathcal{H} \to 2^{\mathcal{A}}$ is the action function that assigns to each node $h \in \mathcal{H}$ a set of possible actions, and $\rho : \mathcal{H} \to \mathcal{P}$ represents the unique acting player. An action $a \in A(h)$ that leads from h to h' is denoted by the successor function $\mathcal{T} : \mathcal{H} \times \mathcal{A} \to \mathcal{H}$ as $h' = \mathcal{T}(h, a)$. $\mathcal{Z} \subseteq \mathcal{H}$ are the sets of terminal nodes for which no actions are available. For each player $p \in \mathcal{P}$, there is a reward function $r_p \in r = r_1, r_2, \ldots, r_{|\mathcal{P}|} : \mathcal{Z} \to \mathbb{R}$. Furthermore, $\mathcal{I} = \{\mathcal{I}_p | p \in \mathcal{P}\}$ describes the information sets (infosets) in the game where $\mathcal{I}_p$ is a partition of all the nodes with acting player p. If two nodes h, h' belong to the same infoset I of player p, i.e., $h, h' \in I \in \mathcal{I}_p$, these two nodes are indistinguishable to p and will share the same action set. We use $I(h)$ to denote the infoset of node h. Upon a certain infoset, a policy (or a behavior strategy) π_p for player p describes which action the player would take at each infoset. A policy can be stochastic, and we use $\pi_p(I)$ to denote the probability vector over player p's available actions at infoset I. With a slight abuse of notation, we use $\pi_p(h)$ to denote the stochastic action player p will take at node h. For two nodes h, h' that belong to the same infoset I, it is clear that $\pi_p(h) = \pi_p(h') = \pi_p(I)$. Therefore, the objective for each player p is to maximize its own total expected return at the end of the game: $R_p \triangleq \mathbb{E}_{Z\sim\pi}[r_p(Z)], Z \in \mathcal{Z}$.

The DouDizhu Game. DouDizhu (a.k.a. Fight the Landload) is a three-player card game that is popular in China and is played by hundreds of millions of people. Among the three players, two of them are called the *Peasants*, and they need to cooperate as a team to compete against the other player called the *Landlord*. The standard game consists of two phases, bidding and cardplay. The bidding phase designates the roles to the players and deals leftover cards to the *Landlord*. In the cardplay phase, the three players play cards in turn in clock-wise order. Within a game episode, there are several *rounds*, and each begins with one player showing a legal combination of cards (solo, pair, etc.). The subsequent players must either choose to pass or beat the previous hand by playing a more superior combination of cards, usually in the same category. The round continues until two consecutive players choose to pass and the player who played the last hand initiates to the next round. DouDizhu is in the genre of shedding where the player wins by emptying his's hand, or loses vice versa. Therefore, in this game, the suit does not matter but the rank does. The score of a game is calculated as the base score multiplied by a multiplier determined by specialized categories of cards (Appendix B.2 shows the details). In this paper, we only consider the cardplay phase, which can be formulated as an imperfect-information game. More detailed information about the game can be referred to Zha et al. (2021).

The key challenges of DouDizhu include how the *Peasants* work as a team to beat the *Landlord* with card number advantage using only imperfect information. For example, one *Peasant* can try to help his teammate to win by always trying the best to beat the *Landlord*'s cards and play cards in a category where the teammate has an advantage. In addition, the action space of DouDizhu is particularly large, and there are 27,472 possible combinations of cards that can be played in total with hundreds of legal actions in a hand. Furthermore, the action space cannot be easily abstracted since improperly playing a card may break other potential card combinations in the following rounds and lead to losing the game.

3. Methodology

In this section, we first introduce how perfect-training-imperfect-execution works for a general imperfect-information game. Then, we formulate the DouDizhu game as an imperfect-information game to solve.

3.1. Perfect Information Distillation

In card games as DouDizhu, the imperfect-information property comes from the fact that players do not show their hand cards to the others. And therefore the critical challenge for each player is to deal with the indistinguishable nodes from the same infoset. For such games, consider we can construct a strategically identical perfect-information game and allow one player to observe distinguishable nodes, then, the decisions at each node can rely on the global information and he may have more chances to win the game, like owning a cheating plug-in. This motivates us to utilize the distinguishable nodes for training the agents of imperfect-information games, and therefore we propose the technique of perfect information distillation.

In general, the perfect information distillation is an actor-critic framework trained in perfect-training-imperfect-execution (PTIE) paradigm, a variant of centralized-training-decentralized-execution (CTDE) (F. et al., 2016; L. et al., 2017), as illustrated in Fig. 1. Particularly, CTDE constructs the value function with all agents' observations and actions for general multi-agent tasks. By comparison, our proposed PTIE is designed for imperfect-information games where additional perfect information is introduced to the value function. Actor-critic (Sutton & Barto, 2018) is a template of policy gradient (PG) methods, proposed in the RL literature towards maximizing the expected reward of the policy function through PG with a value function:

$$\nabla_{\theta_p} J = \mathbb{E}_{\pi}[\nabla_{\theta_p} \log \pi_{\theta_p}(a|s) Q_{\pi}(s,a)] \ , \qquad (1)$$

where s denotes a state in an RL problem, Q is the state-action value function learned by a function approximator, usually called the critic. Notice that the critic is playing the role of evaluating how good an action is taken at a specific situation, but only at the training time. When the agent is deployed into inference, only the policy π can be used to inferring feasible actions. Therefore, for imperfect-information games, we can provide additional information about the exact node the player is in to train the critic with self-play, as long as the actor does not take such information for decision making. Intuitively, we are distilling the perfect information into the imperfect policy.

Formally, for each node h, we construct a distinguishable node $D(h)$ for the strategically identical perfect-information game. Then, we define the value function at $D(h)$, $V_{\pi_p}(D(h)) = \mathbb{E}_{a\sim\pi_p,h^0=h}[Q_{\pi_p}(D(h),a)] = \mathbb{E}_{Z|\pi_p,h^0=h}[r_p(Z)]$ as the expected value of distinguishable nodes. In the sequel, we propose a simple extension of actor-critic policy gradient considering parameterized policy π_{θ_p} for each player p:

$$\nabla_{\theta_p} J = \mathbb{E}_{\pi_p}[\nabla_{\theta_p} \log \pi_{\theta_p}(a|h) Q_{\pi_p}(D(h),a)] \ . \qquad (2)$$

In practice, we use a policy network (actor) to represent the policy π_p for each player, which takes as input a vector describing the representation at an indistinguishable node h that the player can observe during the game. For estimating the critic, a value network is utilized, which takes representation of the global information at the distinguishable node $D(h)$. In other words, the value network takes additional information (such as other players' cards in Poker games) as input, while the policy network does not. During the training, the value function updates the values for all distinguishable nodes; then, it trains the policy on every node on the same infoset from sampled data, which implicitly gives an expected value estimation on each infoset. In practice, the generalization ability of neural networks enables the policy to find a better solution, which is also the advantage for using the proposed PTIE framework.

PTIE is a general way for training imperfect-information game agents, and we expect that with PTIE, players can leverage the perfect information during inference to derive coordination and strategic policies. In experiments, we show that this allows PerfectDou to cooperate with each other (as *Peasants*) or compete against the team (as the *Landlord*).

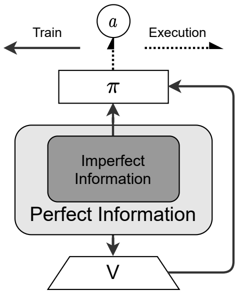
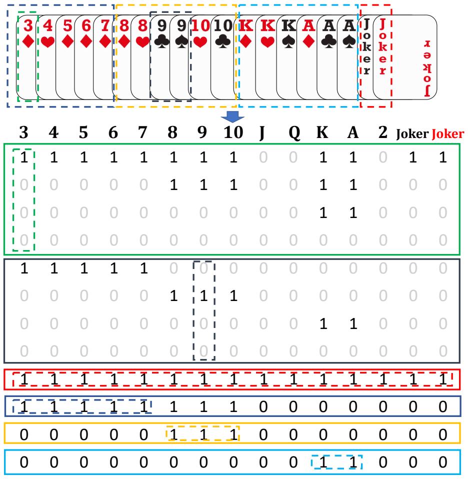


Figure 1: Overview of perfect information distillation within a perfect-training-imperfect-execution framework. The value network takes additional information (such as other players' cards in Poker games) as input, while the policy network does not.

3.2. DouDizhu as An Imperfect-Information Game

As is mentioned in Section 2, the cardplay phase of DouDizhu can be regarded as an imperfect-information game with three players. At each node h, its infoset I_p contains all combinations of the other's invisible handcards. Then at each level of the game tree, the three players take clockwise turns to choose an available action with policies π_p depending on the infoset $I_p(h)$ of the current node h with the reward function r_p. The path from the root of the game tree to a node h contains the initial hand of all players and all the historical moves of all players. The reward functions at leaf nodes are set to be the score the players win or lose at the end of the game.



	3	4	5	6	7	8	9	10	J	Q	K	A	2	Joker	Joker
Card In Hand	1	1	1	1	1	1	1	1	0	0	1	1	0	1	1
	0	0	0	0	0	1	1	1	0	0	1	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Solo	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
Pair	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
Trio	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
Bomb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rocket	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Chain of Solo	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
Chain of Pair	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
Chain of Trio	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

Figure 2: Card representation matrix. Columns stand for 15 different card ranks and rows stand for correspondingly designed features. The first 4 rows are the same as Zha et al. (2021), and the last 8 rows are additional design for encoding the legal combination of cards.

4. PerfectDou System Design

In this section, we explain how we construct our PerfectDou system in detail, with the proposed perfect information distillation technique, and several novel components designed

Table 1: Feature design of perfect-information (distinguishable nodes) and imperfect-information (indistinguishable nodes) for the game.

Imperfect Feature Design	Perfect Feature Design	Size	
Current player's hand		$1 \times 12 \times 15$	
Unplayed cards	Previous player's hand cards	$1 \times 12 \times 15$	$1 \times 12 \times 15$
Current player's played cards		$1 \times 12 \times 15$	
Previous player's played cards	Next player's hand cards	$1 \times 12 \times 15$	$1 \times 12 \times 15$
Next player's played cards		$1 \times 12 \times 15$	
3 leftover cards	Minimum play-out steps of all previous player's hand cards	$1 \times 12 \times 15$	1
Last 15 moves		$15 \times 12 \times 15$	
Previous player's last move	Minimum play-out steps of all next player's hand cards	$1 \times 12 \times 15$	1
Next player's last move		$1 \times 12 \times 15$	
Minimum play-out steps of all hand cards		1	
Number of cards in current player's hand		1	
Number of cards in previous player's hand		1	
Number of cards in next player's hand		1	
Number of bombs		1	
Game control of current player		1	

for DouDizhu that help it summit the game. Particularly, PTIE requires different representations as input layer for the policy and the value network by feeding the value function with perfect information (distinguishable nodes) and the policy with imperfect information (indistinguishable nodes).

4.1. Card Representation

In our system, we encode each feasible card combination with a 12×15 matrix, as shown in Fig. 2. Specifically, we first encode different ranks and numbers with a 4×15 matrix, where the columns correspond to the 15 ranks (including jokers) and similar to Zha et al. (2021), the number of ones in the four rows of a single column represents the number of cards of that rank in the player's hand. Different from Zha et al. (2021), we further propose to encode the legal combination of cards with the player's current hand, to help the agent realize the different property of various kinds of cards (see Section 4.2). The feature sizes of each part are shown in Appendix C.1.

4.2. Node Representation

In the game of DouDizhu, the distinguishable node $D(h)$ should cover all players' hand cards at h, along with the game and player status. Therefore, we propose to represent h with imperfect features and $D(h)$ with perfect feature designs, shown in Tab. 1. In detail, the imperfect features include a flatten matrix[1] of $23 \times 12 \times 15$ and a game state array of 6×1. On the contrary, the perfect features consist of a flatten card matrix of $3 \times 12 \times 15$ and a game state array of 8×1. Therefore, they are totally asymmetric. In our practice, we find including the policy feature into the value function can achieve a better performance.

4.3. Network Structure and Action Representation

The PerfectDou system follows the general actor-critic design, and we take PPO (Schulman et al., 2017) with GAE (Schulman et al., 2015) as the learning algorithm. Slightly different from Eq. (2), PPO estimates the advantage $A_p = R_p - V_{\pi_p}$ as the critic instead of Q_{π_p}. For value network, we use a MLP to handle encoded features (the detailed structure is shown in Appendix C.3). As for the policy network, we first utilize an LSTM to encode all designed features; to encourage the agent to pay attention to specific card types, the proposed network structure will encode all the available actions into feature vectors, as depicted in Tab. 7. The output of the legal action probability is then computed with the action and game features, as illustrated in Fig. 3. Formally, we concatenate the node representation e_s with each action representation e_{a^i} separately, and get the legal action distribution:

$$p(a) = \text{softmax}(f([e_s, e_{a^i}]_{i=1}^N) \,, \tag{3}$$

where a^i is the i-th action, $[\cdot]$ denotes the concatenation operation for N available actions, and f are layers of MLPs. This resembles the target attention mechanism in Ye et al. (2020).

4.4. Node Reward Design

If we only care about the result at the end of the game, the reward at leaf nodes is rather sparse; in addition, players can only estimate their advantage of winning the game using imperfect information during the game, which could be inaccurate and fluctuated. To that end, we propose an augmented reward function for DouDizhu at each node. Instead of letting the players estimate the advantage separately, we utilize an oracle for evaluating each player, particularly, the minimum steps needed to play out all cards, which can be treated as a simple estimation of the distance to win. The reward function is then defined as the advantage difference computed by the relative distance to win of the two camps in two consecutive timesteps. Formally, at timestep t, the reward function is:

$$r_t = \begin{cases} -1.0 \times (\text{Adv}_t - \text{Adv}_{t-1}) \times l, & \textit{Landlord} \\ 0.5 \times (\text{Adv}_t - \text{Adv}_{t-1}) \times l, & \textit{Peasant} \end{cases} \tag{4}$$

[1] Short for a matrix flattened to a one-dimensional vector.

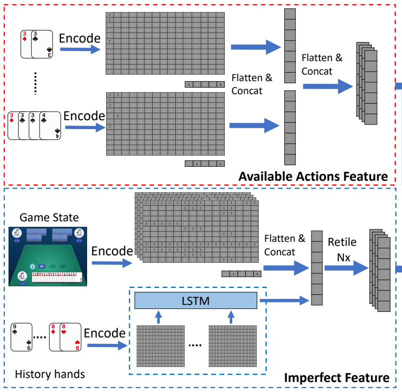


Figure 3: The policy network structure of PerfectDou system. The network predicts the action distribution under current imperfect information of the game, including the state information and the available actions feature.

$$\text{Adv}_t = N_t^{Landlord} - \min\left(N_t^{Peasant1}, N_t^{Peasant2}\right) \ , \tag{5}$$

where l is a scaling factor, and N_t is the minimum steps to play out all cards at timestep t.

For instance, in a round, at timestamp t, the distance of the *Landlord* to win is 5 and the distances of two *Peasants* are 3 and 7, which means *Peasants* have a larger advantage since the relative distance is 2 for *Peasants* and -2 for the *Landlord*. However, if the *Landlord* plays a good hand such that all *Peasants* can not suppress, the *Landlord* will in result get a positive reward due to the decreased relative distance of the *Landlord*, i.e., from 2 to 1. Correspondingly, the *Peasants* would get a negative reward as their relative distances are getting larger. On the contrary, if the *Peasant* with a distance of 3 just suppresses the *Landlord*'s playing hand and the other *Peasant* passes, the reward for both camps will be 0. Such a reward function can encourage the cooperation between *Peasants*, since the winning distance is defined by the minimum steps of both players. In our implementation, the computation of the rewards is carried out after a round of the game, hence to promote training efficiency.

4.5. Distributed Training Details

To further expedite the training procedure, we design a distributed training system represented in Fig. 4. Specifically, the system contains a set of rollout workers for collecting the self-play experience data and sending it to a pool of GPUs; these GPUs asynchronously receive the data and store it into their local buffers. Then, each learner randomly samples mini-batches from its own buffer and compute the gradient separately, which is then synchronously averaged across all GPUs and back propagated to update the neural networks. After each round of updating, new parameters are sent to every rollout worker. And each worker will load the latest model after 24 (8 for each player) steps sampling. Such a decoupled training-sampling structure will allow PerfectDou to be extended to large scale experiments. Our design of the distributed system borrows a lot from IMPALA (Espeholt et al., 2018), which also keeps a set of rollout workers to receive the updated model, interact with the environment and send back rollout trajectories to a learner. The main difference is derived from the learning algorithm where we use PPO with GAE instead of actor-critic with V-trace (Espeholt et al., 2018).

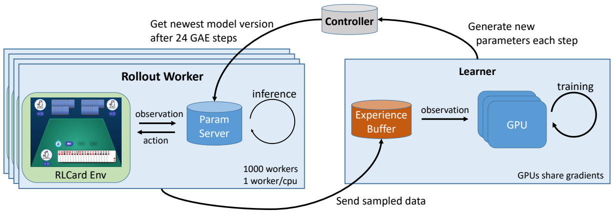


Figure 4: Illustration of the distributed training system.

5. Related Work

Imperfect-Information Games. Many popular card games are imperfect-information games and have attracted much attention. For instance, Li et al. (2020) worked on the four-player game Mahjong and proposed a distributed RL algorithm combined with techniques like global reward prediction, oracle guiding, and run-time policy adaptation to win against most top human players; in addition, Lerer et al. (2020) adopted search-based and imitation methods to learn the playing policy for Hanabi. Iterative algorithms such as Counterfactual Regret Minimization (CFR) and its variants (Zinkevich et al., 2007; Moravčík et al., 2017; Brown & Sandholm, 2018) are also well-used for a fully competitive game, Hold'em Poker, whose action space is generally designed at the scale of tens (fold, call, check and kinds of bet) and the legal actions at each decision point are even less (Moravčík et al., 2017; Zhao et al., 2022). Consequently, they are not appropriate for DouDizhu due to the large action space with difficult abstraction and the mixed game property (both cooperative and competitive).

DouDizhu AI systems. Besides the recent SoTA work DouZero (Zha et al., 2021), many researchers have made efforts on utilizing the power of RL into solving DouDizhu. However, simply applying RL algorithms such as DQN and A3C into the game can hardly make benefits (You et al., 2019). Therefore, You et al. (2019) proposed Combinational Q-Network (CQN) that reduces the action space by heuristics action decoupling; moreover, DeltaDou (Jiang et al., 2019) utilized Monte-Carlo Tree Search (MCTS) for DouDizhu, along with Bayesian inference for the hidden information and a pre-trained kicker network for action abstraction. DeltaDou was also reported as reaching human-level performance.

6. Experiments

We conduct comprehensive experiments to investigate the following research questions.

RQ1 How good is PerfectDou against SoTA DouDizhu AI?

RQ2 What are key ingredients of the PerfectDou system?

RQ3 How is the inference efficiency of PerfectDou?

To answer **RQ1**, we empirically evaluate the performance against existing DouDizhu programs. Regarding **RQ2**, we conduct ablation studies on key components in our design. And for **RQ3**, we calculate the average inference time for all algorithms involved. Finally, we conduct in-depth analysis and provide interesting case studies of PerfectDou. In the Appendix, we report more results including a battle against skilled human players.

6.1. Experimental Setups

Baselines. We evaluate PerfectDou against following algorithms under the open-source RLCard Environment (Zha et al., 2019). For evaluation, we directly take their public (or provided) codes and pre-trained models.

1. **DouZero** (Zha et al., 2021): A recent SoTA baseline method that had beaten every existing DouDizhu AI system using Deep Monte-Carlo algorithm.
2. **DeltaDou** (Jiang et al., 2019): An MCTS-based algorithm with Beyesian inference. It achieved comparable performance as human experts.
3. **Combinational Q-Network (CQN)** (You et al., 2019): Based on card decomposition and Deep Q-Learning.
4. **Rule-Based Algorithms**: Including the open-source heuristic-based program **RHCP-v2** (Jiang et al., 2019; Zha et al., 2021), the rule model in RLCard and a **Random** program with uniform legal moves.

Metrics. The performance of DouDizhu are mainly quantified following the same metrics in previous researches (Jiang et al., 2019; Zha et al., 2021). Specifically, given two algorithms `A` against `B`, we calculate:

- **WP** (Winning Percentage): The proportion of winning by `A` in a number of games.
- **ADP** (Average Difference in Points): The per-game averaged difference of scores between `A` and `B`. The base score is 1 and every bomb doubles the score. This is a more reasonable metric for evaluating DouDizhu AI systems as further discussed in Appendix B.2.

In our experiments, we choose ADP as the basic reward (except the tournament results of the column of WP shown in Tab. 2), which is augmented with the proposed reward signal in Section 4.4 during the early training stage.

6.2. Comparative Evaluations

We conduct a tournament to demonstrate the advantage of our PerfectDou, where each pair of the algorithms plays 10,000 decks, shown in Tab. 2 (**RQ1**). Since the bidding performance in each algorithm varies and poor bidding would affect game results significantly, for fair comparison, we omit the bidding phase and focus on the phase of cardplay. In detail, all games are randomly generated and each game would be played two times, i.e., each competing algorithm is assigned as *Landlord* or *Peasant* once. We use WP and ADP as the basic reward respectively for comparing over these two metrics for all evaluating methods.

Overall, PerfectDou dominates the leaderboard by beating all the existing AI programs, no matter rule-based or learning based algorithms, with significant advantage on both WP and ADP. Specifically, as noted that DouDizhu has a large variance where the initial hand cards can seriously determine the advantage of the game; even though, PerfectDou still consistently outperforms the current SoTA baseline – DouZero. However, we find that PerfectDou is worse than the result published in DouZero paper (Zha et al., 2021) when competing against RHCP. To verify this problem, we test the public model of DouZero (denoted as DouZero (Public) with grey color). To our surprise, its performance can match most of the reported results in their paper except the one against RHCP, where it only takes a WP of 0.452 lower than 0.5, indicating that the public model of DouZero can not beat RHCP as suggested in the original paper, and in fact PerfectDou is the better one.

It is also observed that some competition outcome has a high WP and a negative ADP. A potential reason can be explained as such agents are reckless to play out the bigger cards without considering the left hand, leading to winning many games of low score, but losing high score in the others. From our statistics of online human matches, the WP of winner is usually in a range of $0.52 \sim 0.55$ when the player tries to maximize its ADP.

We further reveal the sample efficiency of PerfectDou by comparing the competing performance w.r.t. different training frames. As shown in Tab. 3, we compare two versions of PerfectDou (1e9 and 2.5e9 frames) against two versions of DouZero (roughly 1e9 and 1e10 frames). From the tournament in Tab. 2, we know the final version of PerfectDou (2.5e9 frames) outperforms the final version of DouZero ($\sim$1e10 frames). However, to our surprise, an early stage of PerfectDou is able to beat DouZero. With the same 1e9 training samples, PerfectDou wins DouZero with a large

Table 2: DouDizhu tournaments for existing AI programs by playing 10k decks. Algorithm A outperforms B if WP is larger than 0.5 or ADP is larger than 0 (highlighted in boldface). The algorithms are ranked according to the number of the other algorithms that they beat. We note that DouZero is the current SoTA DouDizhu bot. Numerical results except marked $*$ are directly borrowed from (Zha et al., 2021).

Rank	A \ B	PerfectDou		DouZero		DeltaDou		RHCP-v2		CQN		Random	
		WP	ADP	WP	ADP	WP	ADP	WP	ADP	WP	ADP	WP	ADP
1	PerfectDou (Ours)	-	-	**0.543***	**0.143***	**0.584***	**0.420***	**0.543***	**0.506***	**0.862***	**2.090***	**0.994***	**3.146***
2	DouZero (Paper)	-	-	-	-	**0.586**	**0.258**	**0.764**	**1.671**	**0.810**	**1.685**	**0.989**	**3.036**
-	DouZero (Public)	0.457*	-0.143*	-	-	**0.585***	**0.253***	0.451*	**0.060***	**0.828***	**1.950***	**0.986***	**3.050***
3	DeltaDou	0.416*	-0.420*	0.414	-0.258	-	-	**0.691***	**1.528***	**0.784**	**1.534**	**0.992**	**3.099**
4	RHCP-v2	0.457*	-0.506*	**0.549***	-0.060*	0.309*	-1.423*	-	-	**0.770***	**1.414***	**0.990***	**2.670***
5	CQN	0.138*	-2.090*	0.190	-1.685	0.216	-1.534	0.230*	-1.414 *	-	-	**0.889**	**1.912**
6	Random	0.006*	-3.146*	0.011	-3.036	0.008	-3.099	0.010*	-2.670*	0.111	-1.912	-	-

gap (WP of 0.732 and ADP of 1.270), which is even better than the 1e10 sample-trained DouZero. This indicates that PerfectDou is not only the best performance but also the most training efficient. The related training curves are shown in Appendix D.3.

6.3. Ablation Studies

We want to further investigate the key to the success of our AI system (**RQ2**). Specifically, we would like to analyse how our design of the feature and the training framework help PerfectDou dominate the tournament of DouDizhu. To this end, we evaluate different variants of PerfectDou and the previous SoTA AI system – DouZero, including:

1. ImperfectDouZero[2]: DouZero with our proposed imperfect-information features.
2. ImperfectDou: PerfectDou with only imperfect-features as inputs for the value function.
3. RewardlessDou: PerfectDou without node reward.
4. Vanilla PPO: Naive actor-critic training with imperfect-features only and without additional reward.

The ablation experiments are designed as competitions among ImperfectDou, RewardlessDou and PerfectDou against DouZero for comparing the effectiveness of perfect information distillation and perfect intermediate reward separately; while the battle of ImperfectDouZero and DouZero against PerfectDou are designed for excluding the benefit from feature engineering. Results for all comparisons are shown in Tab. 4. Even with the imperfect features only, ImpefectDou can still easily beat DouZero with the same training steps; however, DouZero turns the tide with much more training data. Furthermore, our proposed node features seems not appropriate for DouZero to achieve a better results compared with its original design. Additionally, without the node reward, PerfectDou still beats DouZero with higher WP (in spite of sacrificing a lot of ADP), indicating the effectiveness of perfect reward in training, without which it would risk losing points to win one game. Finally, without both node reward and perfect feature design for the value function, vanilla PPO simply can not perform well. Therefore, we can conclude that our actor-critic based algorithm along with the PTIE training provides a high sample efficiency under our feature design, and the node reward benefits the rationality of our AI.

Table 3: Training efficiency comparison over 100k decks.

A \ B	DouZero (∼1e9)		DouZero (∼1e10)	
	WP	ADP	WP	ADP
PerfectDou (2.5e9)	-	-	0.541	0.130
PerfectDou (1e9)	0.732	1.270	0.524	0.014
DouZero (∼1e10)	0.698	1.150	-	-

Table 4: Ablation studies over 100k decks.

A \ B	DouZero (∼1e9)		DouZero (∼1e10)		ImperfectDouZero (∼1e9)	
	WP	ADP	WP	ADP	WP	ADP
PerfectDou (1e9)	0.732	1.270	0.524	0.014	0.731	1.350
ImperfectDou (1e9)	0.717	1.180	0.486	-0.057	0.723	1.320
RewardlessDou (1e9)	0.738	0.490	0.540	-0.201	0.659	0.587
Vanilla PPO(1e9)	0.509	-0.307	0.346	-0.709	0.433	-0.023

6.4. Runtime Analysis

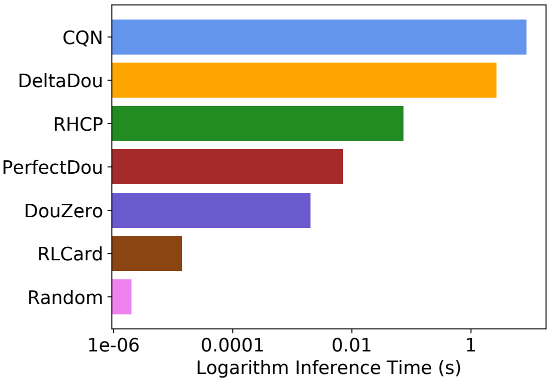


Figure 5: Comparison of the inference time.

We further conduct runtime analysis to show the efficiency of PerfectDou w.r.t. the inference time (**RQ3**), which is reported in Fig. 5. All evaluations are conducted using the same machine. The inference time of each AI could be

[2] Note DouZero cannot require the perfect-information since it will play in a cheating style.

attributed to its solution and implementation in the playing time. CQN uses a large Q network (nearly 10× parameters larger than ours) with a complex card decomposer to derive reasonable hands. As a result, the inference time of CQN is the longest. Besides, both DeltaDou and RHCP-V2 contain lots of times of Monte Carlo simulations, thus slowing down the inference time. As comparisons, DouZero and PerfectDou only require one network forward inference time with a similar number of parameters. For RLCard, only handcraft rules are computed. Therefore, we can notice that PerfectDou is significantly faster than previous programs like DeltaDou, CQN and RHCP, yet is slightly slower than DouZero. To be more accurate, the average inference time of DouZero is 2 milliseconds compared with 6 milliseconds of PerfectDou. And the reason why PerfectDou is a bit slower than DouZero may due to the more complex feature processing procedure. The above analysis suggests that PerfectDou is applicable and affordable to real-world applications such as advanced game AI.

6.5. In-Depth Statistical Analysis

In our experiments, we find that DouZero is leaky and unreasonable in many battle scenarios, while PerfectDou performs better therein. To quantitatively evaluate whether PerfectDou is stronger and more reasonable, we conduct an in-depth analysis and collect the statistics among the games between DouZero and PerfectDou. Particularly, we organize games between PerfectDou and Douzero to play in different roles for 100,000 decks in each setting. Since the roles are assigned randomly instead of opting by agents themselves in our experiments, and the *Landlord* has a higher base score with three extra cards, we observe that playing as a *Landlord* is always harder to win and leads to negative ADPs. From the statistics shown in Tab. 5, we learn many lessons about the rationality of PerfectDou: (i) when playing as the *Landlord*, PerfectDou plays fewer bombs to avoid losing scores and tends to control the game even the *Peasants* play more bombs; (ii) when playing as the *Peasant*, two PerfectDou agents cooperate better with more bombs to reduce the control time of the *Landlord* and its chance to play bombs; (iii) when playing as the *Peasant*, the right-side *Peasant* agent (play after the *Landlord*) of PerfectDou throws more bombs to suppress the Landlord than DouZero, which is more like human strategy.

6.6. Case Study: Behavior of DouZero vs PerfectDou

In this section, we list some of the observations during the games for comparing the behavior of DouZero and PerfectDou to qualitatively support our analysis.

DouZero is more aggressive but less thinking. The first observation is that DouZero is extremely aggressive without considering the left hands. For instance, as shown in Fig. 6, in the beginning DouZero chooses a chain of solo but leaves the pair of 3, which can be dangerous since the pair of 3 is the one of the minimum cards and cannot suppress any card; Fig. 7 illustrates another strong case, where DouZero also chooses a chain of solo to suppress the opponent without considering the consequence of leaving a hand of solos. On the contrary, PerfectDou is more conservative and steady. We believe the proposed perfect information distillation mechanism helps PerfectDou to infer global information in a more reasonable way.

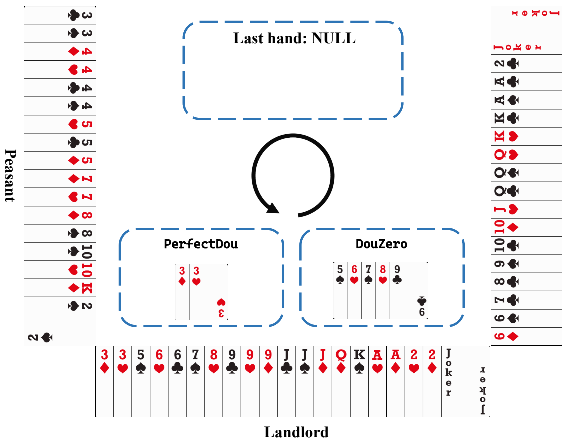


Figure 6: Case study: DouZero is more aggressive by choosing a chain of solo in the beginning but leaves the pair of 3 in the hand.

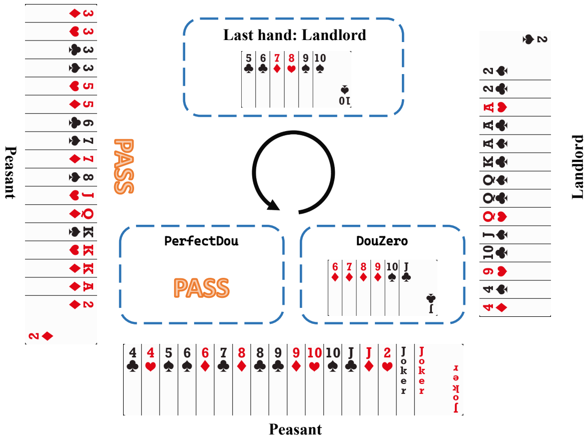


Figure 7: Case study: DouZero is more aggressive by suppressing the *Landlord* but less thinking on the consequence of the left hands of solos.

PerfectDou is better at guessing and suppressing. We observe another fact that the usage of perfect information distillation within the PTIE framework benefits PerfectDou a lot by suppressing the opponents in advance. In Fig. 8 shows a case when the teammate puts a pair of T[3], DouZero chooses to pass; on the contrary, PerfectDou chooses suppressing by a pair of Q – the minimal pair of the *Landlord*.

[3] We denote T(en) as the card 10 for simplicity.

Table 5: Average per game statistics of important behaviors over 100k decks: `Game Len` is the average number of rounds in a game; `% Bomb` represents the average percentage of bombs (a type of card can suppress any categories except the bomb with a higher rank, see Appendix B) played in the game; `Left` and `Right` are the relative position to the *Landlord*; and `Landlord Control Time` measures the number of rounds that the landlord plays an action suppressing all other players.

Landlord Agent	WP	ADP	`Game Len`	`%Bomb of Left` *Peasant*	`%Bomb` of *Landlord*	`%Bomb of Right` *Peasant*	`Landlord Control Time`	*Peasant* Agent
PerfectDou (2.5e9)	0.446	-0.407	33.347	68.05	28.46	74.90	12.993	DouZero (∼1e10)
DouZero (∼1e10)	0.421	-0.461	33.911	66.24	28.73	75.29	9.005	
PerfectDou (2.5e9)	0.387	-0.608	31.157	66.13	26.67	79.68	10.518	PerfectDou (2.5e9)
DouZero (∼1e10)	0.360	-0.686	31.267	64.80	26.72	79.29	7.123	

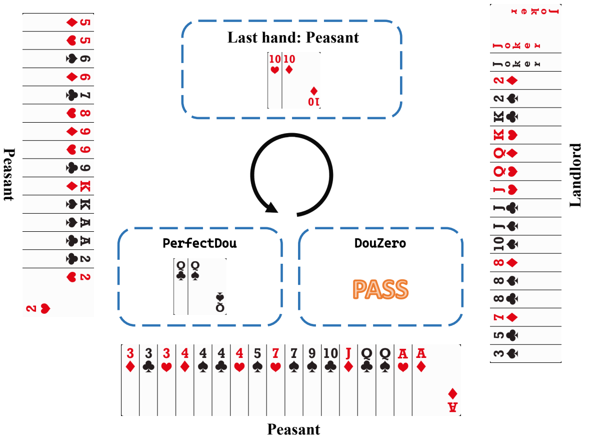


Figure 8: Case study: the teammate shows a pair of T and DouZero chooses to pass; on the contrary, PerfectDou chooses suppressing by a pair of Q – the minimal pair of the opponent.

PerfectDou is better at card combination. In the battle shown in Fig. 9, PerfectDou shows the better ability on the strategy of card combination. Specifically, PerfectDou chooses to split the plane (999, TTT since it considers there is a chain of solo ($9TJQK$) left. However, DouZero only takes the trio, which will be easily suppressed by the opponent. This benefits from the proper design of the card representation and the action feature of PerfectDou.

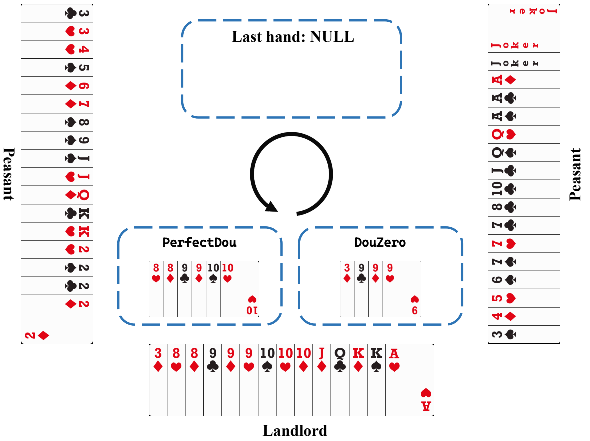


Figure 9: Case study: PerfectDou chooses to split the plane (999, TTT) since it considers there is a chain of solo ($9TJQK$) left.

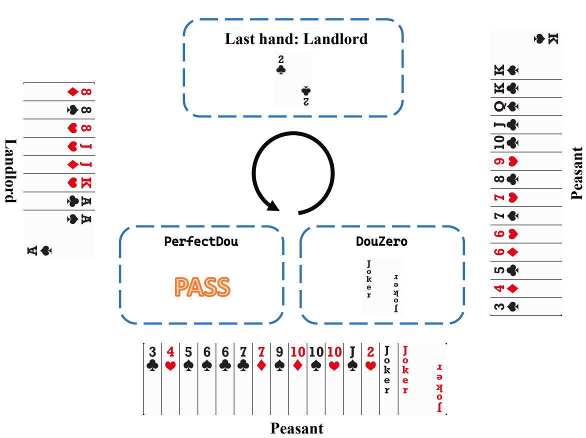


Figure 10: Case study: DouZero splits the rocket bomb while PerfectDou chooses to keep it.

PerfectDou is more calm. Fig. 10 depicts a typical and interesting scenario where PerfectDou shows its calm and careful consideration over the whole. In the game, the last hand is of the *Landlord* with a solo 2, and it only has 8 cards left in the hand. DouZero seems afraid and splits the rocket bomb; however, PerfectDou benefits from the advantage reward design and is calm considering there is a greater chance on winning the game with a higher score by keeping the bomb.

7. Conclusion

In this paper, we propose PerfectDou, a SoTA DouDizhu AI system that dominates the game. PerfectDou takes the advantage of the perfect-training-imperfection-execution training paradigm, and is trained within a distributed training framework. In experiments we extensively investigate how and why PefectDou can achieve the SoTA performance by beating all existing AI programs with reasonable strategic actions.

Acknowledgement

The SJTU team is supported by "New Generation of AI 2030" Major Project (2018AAA0100900), NSFC (62076161). Minghuan Liu is supported by Wu Wen Jun Honorary Doctoral Scholarship, AI Institute, SJTU.

References

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

Brown, N. and Sandholm, T. Superhuman ai for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pp. 1407–1416. PMLR, 2018.

F., J. N., A., Y. M., de F., N., and W., S. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2137–2145, 2016.

Jiang, Q., Li, K., Du, B., Chen, H., and Fang, H. Deltadou: Expert-level doudizhu ai through self-play. In *IJCAI*, pp. 1265–1271, 2019.

L., R., W., Y., T., A., H., J., A., P., and M., I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pp. 6379–6390, 2017.

Lerer, A., Hu, H., Foerster, J., and Brown, N. Improving policies via search in cooperative partially observable games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7187–7194, 2020.

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

Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

Sergeev, A. and Del Balso, M. Horovod: fast and easy distributed deep learning in tensorflow. *arXiv preprint arXiv:1802.05799*, 2018.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017a.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017b.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Ye, D., Liu, Z., Sun, M., Shi, B., Zhao, P., Wu, H., Yu, H., Yang, S., Wu, X., Guo, Q., et al. Mastering complex control in moba games with deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6672–6679, 2020.

You, Y., Li, L., Guo, B., Wang, W., and Lu, C. Combinational q-learning for dou di zhu. *arXiv preprint arXiv:1901.08925*, 2019.

Zha, D., Lai, K.-H., Cao, Y., Huang, S., Wei, R., Guo, J., and Hu, X. Rlcard: A toolkit for reinforcement learning in card games. *arXiv preprint arXiv:1910.04376*, 2019.

Zha, D., Xie, J., Ma, W., Zhang, S., Lian, X., Hu, X., and Liu, J. Douzero: Mastering doudizhu with self-play deep reinforcement learning. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 12333–12344. PMLR, 2021.

Zhao, E., Yan, R., Li, J., Li, K., and Xing, J. Alphaholdem: High-performance artificial intelligence for heads-up no-limit texas hold'em from end-to-end reinforcement learning. 2022.

Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20:1729–1736, 2007.

Appendix

A. Additional Related Work

Utilizing global information to reduce the complexity of imperfect-information games has also been investigated in some works. For example, AlphaStar (Vinyals et al., 2019), a grandmaster level AI system for StarCraft II. In their implementation, the value network of the agent can observe the full information about the game state, including those that are hidden from the policy. They argue that such a training style improves training performance. In our work, we formulate the idea as Perfect-Training-Imperfect-Execution (PTIE) or perfect information distillation technique for imperfect-information games, and show the effectiveness on complicated card games like DouDizhu. Moreover, in Suphx (Li et al., 2020), a strong Mahjong AI system, they used a similar method namely oracle guiding. Particularly, in the beginning of the training stage, all global information is utilized; then, as the training goes, the additional information would be dropped out slowly to none, and only the information that the agent is allowed to observe is reserved in the subsequent training stage. However, there are obvious difference between Suphx and PerfectDou. In Suphx, the perfect information is used by the actor and thus has to be dropped before the inference stage; on the contrary, PerfectDou feeds the critic with additional observations and distill the global information to the actor.

B. More About DouDizhu

B.1. Term of Categories

In the work of (Zha et al., 2021), they had shown a comprehensive introduction of DouDizhu game, so we think it may be wordy to repeat the stereotyped rules. However, for better understanding the cases shown in this paper, we introduce the typical term of categories in DouDizhu that are commonly used as follows. Note that all cards can suppress the cards in the same category with a higher rank, yet bomb can suppress any categories except the bomb with a higher rank. Rocket is the highest-rank bomb.

1. **Solo** : Any single card.
2. **Pair** : Two matching cards of equal rank.
3. **Trio** : Three individual cards of equal rank.
4. **Trio with Solo** : Three individual cards of equal rank with a Solo as the kicker.
5. **Trio with Pair** : Three individual cards of equal rank with a Pair as the kicker.
6. **Chain of Solo** : Five or more consecutive individual cards.
7. **Chain of Pair** : Three or more consecutive Pairs.
8. **Chain of Trio** : Two or more consecutive Trios.
9. **Plane with Solo**: Two or more consecutive Trios with each has a distinct individual kicker card.
10. **Quad with Pair** : Four-of-a-kind with two sets of Pair as the kicker.
11. **Bomb** : Four-of-a-kind.
12. **Rocket** : Red and black jokers.

B.2. Scoring Rules

In (Zha et al., 2021), they pay more attention to the win/lose result of the game but care less about the score. However, in real competitions, players must play for numbers of games and are ranked by the score they win. And that is why we think ADP is a better metric for evaluating DouDizhu AI systems because a bad AI player can win a game with few scores but lose with much more scores.

Specifically, in each game, the *Landlord* and the *Peasants* have base scores of 2 and 1 respectively. When there is a bomb shown in a game, the score of each player doubles. For example, a *Peasant* player first shows a bomb of 4 and then the *Landlord* player suppresses it with a rocket, then the base score of each *Peasant* becomes 4 and the *Landlord* becomes 8. A player will win all his scores after winning the game, or loses all of them vice versa.

C. Additional System Design Details

C.1. Card Representation Details

In the system of PerfectDou, we augment the basic card in hand matrix with explicitly encoded card types as additional features, in order to allow the agent realizing the different properties of different kind of cards. The size details are shown in Tab. 6.

Table 6: Card representation design.

CARD MATRIX FEATURE	SIZE
CARD IN HAND	4×15
SOLO	1×15
PAIR	1×15
TRIO	1×15
BOMB	1×15
ROCKET	1×15
CHAIN OF SOLO	1×15
CHAIN OF PAIR	1×15
CHAIN OF TRIO	1×15

C.2. Action Feature Details

Table 7: Action feature design.

FEATURE DESIGN	SIZE
CARD MATRIX OF ACTION	12×15
IF THIS ACTION IS BOMB	1
IF THIS ACTION IS THE LARGEST ONE	1
IF THIS ACTION EQUALS THE NUMBER OF LEFT PLAYER'S CARDS IN HAND	1
IF THIS ACTION EQUALS THE NUMBER OF RIGHT PLAYER'S CARDS IN HAND	1
THE MINIMUM STEPS TO PLAY-OUT ALL LEFT CARDS AFTER THIS ACTION PLAYED	1

The action features are a flatten matrix from 12×15 action card matrix plus 1×6 extra dimensions describing the property of the cards as shown in Tab. 7. Since the number of actions in each game state varies, which can lead to different lengths of action features, a fixed length matrix is flattened to store all action features where the non-available ones are marked as zero.

C.3. Value Network Structure

The value network of PerfectDou is designed to evaluate the current situation of players, and we expect that the value function can utilize the global information, in other words, know the exact node the player is in. Therefore, we should feed additional information that the policy is not allowed to see in our design. Specifically, as shown in Fig. 11, the imperfect feature for indistinguishable nodes is encoded using the shared network as in the policy network; besides, we also encode the perfect feature of distinguishable nodes that the policy cannot observe during its game playing. The encoded vector are then concatenated to a simple MLP to get the scalar value output.

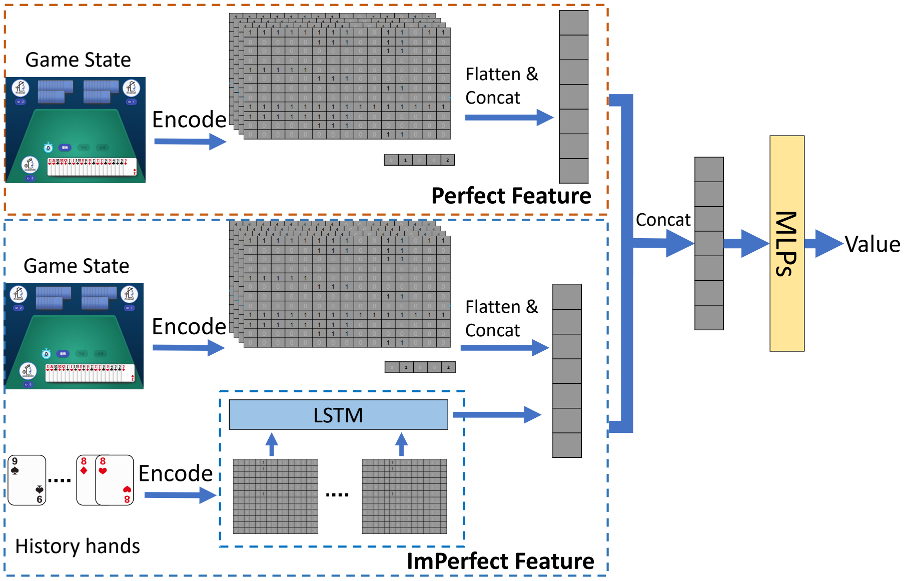


Figure 11: The value network structure of the proposed PerfectDou system. The network predicts values using both the imperfect feature and the perfect feature and distill the knowledge into the policy in the training.

D. Experiments

D.1. Implementation Details

In our implementation, a small distributed training cluster is built using 880 CPUs cores and 8 GPUs. Horovod (Sergeev & Del Balso, 2018) is used to synchronize gradients between GPUs, the total batch size is 1024, 128 for each GPU. In the early training stage, the total reward function will be a basic reward (either WP or ADP) augmented with the designed oracle reward as shown in Section 4.4 to help convergence. In the later stage, the augmented reward will be removed. The most important hyperparameters in our experiment are shown in Tab. 8.

Table 8: Hyperparameters. * refers to the maximum version gap allowed between the models used for sampling and training.

Learning rate	3e-4
Optimizer	Adam
Discount factor γ	1.0
λ of GAE	0.95
Step of GAE	24 (8 for each player)
Batch size	1024
Entropy weight of PPO	0.1
Length of LSTM	15 (5 for each player)
Max model lag*	1
Intermediate reward scale	50
Policy MLP hidden sizes	[256, 256, 256, 512, 621]
Value MLP hidden sizes	[256, 256, 256, 256, 1]

D.2. Battle Results Against Skilled Human Players

We further invite some skilled human players to play against PerfectDou. Particularly, each human player plays with two AI players. In other words, each game is involved with either two AI *Peasants* against one human *Landlord*, or one AI *Peasant* cooperating with one human *Peasant* against one human *Landlord*. The results are shown in Tab. 9. One can easily observe that PerfectDou takes evident advantage during the game.

Table 9: Battle results against skilled human players for 1260 episodes of game.

B / A	Skilled human	
	WP	ADP
PerfectDou (2.5e9)	0.625	0.590

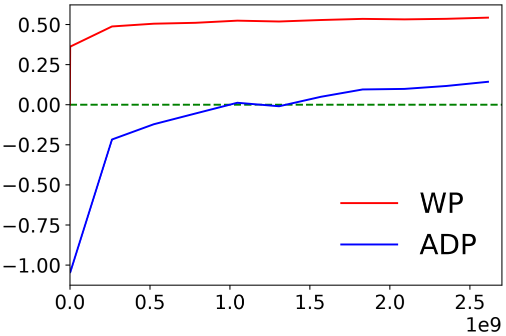


Figure 12: Learning curves of WP and ADP against the final model of DouZero w.r.t. timesteps for PerfectDou. Every evaluation contains 10000 decks. PerfectDou is able to beat DouZero without considering the scores at the beginning of the training, around $1.5e6$ steps.

D.3. Additional Training Results

Fig. 12 shows the learning curves of WP and ADP against DouZero for PerfectDou with a single run, and every evaluation contains 10000 decks. As shown in the figure, PerfectDou can easily beat DouZero (on WP) without considering the scores (ADP) at the beginning of the training; but after 1.5e9 steps of training, PerfectDou is able to fully beat DouZero (both WP and ADP).

D.4. Complete Tournament Results of ADP for *Landlord* and *Peasants*

We report the complete tournament results of ADP and WP for *Landlord* and *Peasants* in Tab. 10 and Tab. 11. PerfectDou tends to have more advantage of *Peasants* than that of *Landlord*, especially when compete against stronger baselines. We believe that the proposed perfect informtation distillation technique allows for better cooperation between two *Peasants*. In addition, since the roles are assigned instead of opting according to hand in our competition, and the *Landlord* has extra three cards and can lose a higher base score, the *Peasants* seems having more chance to win the game. Therefore, almost all methods can play better results as a *Peasant* than that as a *Landlord*.

Table 10: ADP results of DouDizhu tournaments for existing AI programs by playing 10k decks. L: ADP of A as Landlord; P: ADP of A as Peasants. Algorithm A outperforms B if the ADP of L or P is larger than 0 (highlighted in boldface). We note that DouZero is the current SoTA DouDizhu bot. Numerical results except marked $*$ are directly borrowed from (Zha et al., 2021).

Rank	B / A	PerfectDou		DouZero		DeltaDou		RHCP-v2		CQN		Random	
		P	L	P	L	P	L	P	L	P	L	P	L
1	PerfectDou (Ours)	0.656*	-0.656*	**0.686***	-0.407*	**0.980***	-0.145*	**0.872***	**0.138***	**2.020***	**2.160***	**3.008***	**3.283***
2	DouZero (Public)	0.407*	-0.686*	0.435*	-0.435*	**0.858***	-0.342*	**0.166***	-0.046*	**2.001***	**1.368***	**2.818***	**3.254***
3	DeltaDou	**0.145***	-0.980*	**0.342***	-0.858*	**0.476**	-0.476	**1.878***	**0.974***	**1.849**	**1.218**	**2.930**	**3.268**
4	RHCP-v2	-0.138*	-0.872*	**0.046***	-0.166*	-0.974*	-1.878*	**0.182***	-0.182*	**1.069***	**1.758***	**2.560***	**2.780***
5	CQN	-2.160*	-2.020*	-1.368*	-2.001*	-1.218	-1.849	-1.758*	-1.069*	**0.056**	-0.056	**1.992**	**1.832**
6	Random	-3.283*	-3.008*	-3.254*	-2.818*	-3.268	-2.930	-2.780	-2.560*	-1.832	-1.991	**0.883**	-0.883

Table 11: WP results of DouDizhu tournaments for existing AI programs by playing 10k decks. L: WP of A as Landlord; P: WP of A as Peasants. Algorithm `A` outperforms `B` if the WP of L or P is larger than 0.5 (highlighted in boldface). Numerical results except marked $*$ are directly borrowed from (Zha et al., 2021).

Rank	B A	PerfectDou		DouZero		DeltaDou		RHCP-v2		CQN		Random	
		P	L	P	L	P	L	P	L	P	L	P	L
1	PerfectDou (Ours)	**0.622***	0.378*	**0.640***	0.446*	**0.693***	0.474*	**0.609***	0.478*	**0.894***	**0.830***	**0.998***	**0.990***
2	DouZero (Public)	**0.554***	0.360*	**0.584***	0.416*	**0.684***	0.487*	0.427*	0.475*	**0.851***	**0.769***	**0.992***	**0.986***
3	DeltaDou	**0.526***	0.307*	**0.513***	0.317*	**0.588**	0.412	**0.768***	**0.614***	**0.835**	**0.733**	**0.996**	**0.987**
4	RHCP-v2	**0.522***	0.391*	**0.525***	**0.573***	0.386*	0.232*	**0.536***	0.434*	**0.687***	**0.853***	**0.994***	**0.985***
5	CQN	0.170*	0.106*	0.231*	0.149*	0.267	0.165	0.147*	0.313*	0.476	**0.524**	**0.921**	**0.857**
6	Random	0.010*	0.002*	0.014*	0.008*	0.013	0.004	0.015*	0.006*	0.143	0.080	**0.654**	0.346