Proceedings of the 4th International Conference on Computing Innovation and Applied Physics
DOI: 10.54254/2753-8818/83/2025.19923

# *Feature Analysis and Neural Network Optimization in Bridge Bidding Systems*

**Wenzhi Wang**[1,a,*]

[1]*The High School Affiliated to Renmin University of China, China*
*a. 17310056709@163.com*
**corresponding author*

***Abstract:*** With the successful application of Artificial intelligence (AI) technology in a range of information intelligence projects similar to bridge, a more scientifically rigorous new field has been opened by studying the logic of AI technology. AI has demonstrated the ability to challenge human intelligence projects such as bridge. This study focuses on the bidding system of bridge, obtains real simulated data of bridge bidding, combines basic card features with artificially constructed features, and analyzes the key issues faced by bidding research. A neural network structure based on Deep Q-Network (DQN) is proposed to find a suitable neural network structure for bidding game research. Through the comparison of testing costs and total bidding times, as well as the analysis of feature ablation experiments, it is found that different features have different impacts on the machine learning model performance of the bridge bidding system. Features such as "number of each suit cards," "high-card points," and "honors" have a positive effect on the model performance, while other features have limited impact on the model performance. The application of Deep Learning methods in bridge bidding algorithm research contributes to the advancement of machine gaming and AI industries.

***Keywords:*** Bridge bidding, Feature construction, DQN, Ablation experiment.

## 1. Introduction

Artificial intelligence (AI) technologies, represented by Deep Learning, have swiftly infiltrated various aspects of human work and life. The rapid dissemination of AI-related knowledge through the internet has opened people's limited understanding of AI, allowing them to realize the power of AI and the potential for liberating productivity. After various intellectual competition projects were conquered by AI, the bridge project, which has temporarily held its ground as a human domain, has also made breakthrough progress in recent years in the face of pressure from AI customized products. Several years ago, most bridge players did not take this seriously, but as one after another game of intellectual competition similar to bridge was perfectly conquered by AI technology, another more scientifically rigorous new world unfolded before people. As AI theories and technologies mature, AI has been able to simulate the information processes of human consciousness and thinking, and has fully acquired the ability to challenge bridge, a human-dominant intellectual project.

Bridge is a highly strategic four-player card game where players aim to achieve the optimal contract through bidding and playing phases to win as many points as possible compared to the opponents [1]. Bridge is a highly competitive zero-sum game characterized by phased gameplay and

DOI: 10.54254/2753-8818/83/2025.19923

a mix of cooperation and confrontation. Despite significant progress made in chess and Go through Deep Learning and Reinforcement Learning techniques, bridge still faces significant challenges due to its complexity in strategy. In the bidding phase of bridge, each player can only see their own hand and needs to convey information to their partner and disrupt the opponents through bidding, making existing complete information game algorithms unsuitable for direct application in bridge bidding [2].

Studying bridge bidding strategies holds important theoretical and practical significance. Firstly, bridge bidding strategy is a classical problem in information games, which is a focal point in current AI research. Researching this problem can drive the development of game theory and AI fields. Secondly, studying bridge bidding strategies can provide new perspectives and challenges for man-machine battles. Unlike Go and chess, bridge involves a mix of cooperation and confrontation, posing new requirements for AI systems. Bridge machine gaming shares many similarities with economic and social game scenarios, making it highly valuable for research. Applying Deep Learning to bridge machine gaming not only has exploratory significance but also provides valuable experience for future research and has practical significance in driving economic and social development.

Researchers began studying bridge games in the 1970s . Currently developed bridge programs include past World Computer Bridge Championship winners such as Bridge Baron [3], Gro [4], Jack and WBridge5 [5] and Finesse [6]. In recent years, significant progress has been made in the application of Deep Learning in bridge bidding. Training bidding strategies by simulating human expert bidding data through deep neural networks has become a research hotspot. Amit et al. used heuristic search algorithms to address incomplete information issues through Monte Carlo sampling and introduced a collaborative training learning framework that continuously optimizes choice strategies during training, surpassing the state-of-the-art bidding algorithms at that time [7]. Delooz et al. demonstrated that Self-Organizing Maps (SOM) could effectively bid out no-trump contracts [8]. Neural networks was used to solve the double-dummy problem in bridge [9]. Ho and Lin proposed a learning framework based on the Upper Confidence Bound algorithm, enabling programs to achieve a level similar to computer bridge champions without mimicking human bidding systems and in no-pass situations [9]. Yeh et al. used a Reinforcement Learning model based on the Upper Confidence Bound algorithm to develop new bidding systems during the bidding process [10]. Jiang Rong et al. proposed a Deep Learning-based bridge bidding system, significantly improving bidding strategy performance by designing a card estimation neural network. Their research indicates that the effectiveness of cooperative bidding and information exchange is crucial for enhancing bridge bidding strategies [1]. Louet al. established a bridge bidding system model based on Deep Learning, predicting game understanding based on bidding sequences to guide Monte Carlo sampling and generate bidding decisions through search [11]. Zhang et al. modeled the bidding process and built an LSTM neural network based on hand understanding for multitask learning to generate bidding decisions, demonstrating the superiority of the model in understanding the bidding process on a real game dataset [12]. From the review of the above literature, it is not clear how features are constructed in bridge bidding systems. Most are based on basic card information as features in the model, with little use of important manually extracted features such as distribution points and voids. The inclusion of important features is crucial for improving model performance. Furthermore, while common mathematical models and regular neural networks are applied in bidding systems, the use of Reinforcement Learning models is not common in bridge bidding systems, leading to significant differences in bidding prediction analysis results.

This paper will obtain real simulated data for bridge bidding. Then basic card features with manually constructed features will be combined to build and train bidding models based on the DQN model for card play and evaluation models to be applied to the entire bidding strategy for optimal bidding selection methods. By applying AI systems to bridge bidding, significant improvements can be made in AI's performance in handling complex information games, further advancing AI

DOI: 10.54254/2753-8818/83/2025.19923

technology. It is hoped that through the development and understanding of the technology in this paper, AI's performance in bridge bidding can surpass that of ordinary bridge players.

## 2. Dataset

### 2.1. Introduction of Dataset

In bridge, a deck of 52 cards is used, excluding the jokers. The game is divided into four directions: North, South, East, and West, forming two opposing teams. North-South form one team, while East-West form the other. The entire game process is divided into two phases: the bidding phase and the playing phase. The bidding phase is a process of information exchange between the two sides in a shared space, as the types of hands can vary greatly. At the beginning of the bidding phase, cards are dealt in clockwise order by a designated dealer, with each player ultimately receiving thirteen cards and only knowing the information of their own hand. After dealing, the dealer becomes the first player to bid, known as the opener, followed by bidding in clockwise order. During bidding, players can choose an action from 38 bidding conventions, including "PASS, {1♣, 1♦, 1♥, 1♠, ... , 7NT}, X, XX". It is important to note that if selecting an action from the 35 options, it must be after the previously chosen action; otherwise, it will be considered a violation. When three consecutive players choose the PASS action, the bidding phase ends, and the contract is completed. The contract result is the final substantive bid plus X or XX. The purpose of the bidding phase is for a team with strong cards to negotiate higher-level contracts to earn more bonus points, while a team with weaker cards can disrupt the opponents to reach an appropriate contract. The contract is established through a series of bidding actions, with one team proposing a bidding level agreement and the other team accepting it. The goal of the declaring side is to fulfill the contract, while the defending side aims to prevent the opponents from fulfilling the contract. Contracts are divided into two types: suit contracts and no-trump contracts. A suit contract designates one of the four suits as the trump suit, allowing a player to win a trick by playing a card of that suit if they don't have the lead suit; a no-trump contract, denoted as "NT", means there is no trump suit, and regardless of the suit played, a player cannot win the trick if they don't have the lead suit.

The algorithm data studied in this paper is sourced from the dataset generated by modifying the code of Ho-Chun Yen [9]. The dataset used in this paper consists of 100,000 training samples, 20,000 validation samples, and 20,000 testing samples, providing sufficient bidding data to support the research on bridge bidding algorithms. Each sample set contains 104 variables marked with 0 or 1. The first 52 dimensions in the dataset represent the hand of Player 1, while the subsequent 52 dimensions represent the hand of Player 2. The 52 dimensions for each player are encoded using one-hot encoding with 13 bits. This encoding includes all 52 cards in a standard deck, from the 2 of Spades (S2) to the Ace of Spades (SA), followed by the 2 of Hearts (H2) to the Ace of Hearts (HA), then the 2 of Diamonds (D2) to the Ace of Diamonds (DA), and finally the 2 of Clubs (C2) to the Ace of Clubs (CA). In this encoding scheme, a value of 1 indicates that the player has the corresponding card, while a value of 0 indicates that the player does not have that card. Some data structures are shown in Table 1.

Table 1: Player 1's hand is represented using values of 0 or 1.

| Symbol | Value | Suit | Value | Suit | Value | Suit | Value | Suit |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | S2 | 0 | H2 | 0 | D2 | 0 | C2 |
| 2 | 1 | S3 | 0 | H3 | 0 | D3 | 0 | C3 |
| 3 | 1 | S4 | 0 | H4 | 1 | D4 | 0 | C4 |
| 4 | 0 | S5 | 0 | H5 | 0 | D5 | 0 | C5 |

DOI: 10.54254/2753-8818/83/2025.19923

Table 1: (continued).

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 5 | 0 | S6 | 0 | H6 | 1 | D6 | 0 | C6 |
| 6 | 0 | S7 | 0 | H7 | 1 | D7 | 1 | C7 |
| 7 | 1 | S8 | 1 | H8 | 1 | D8 | 0 | C8 |
| 8 | 1 | S9 | 0 | H9 | 0 | D9 | 0 | C9 |
| 9 | 0 | S10 | 0 | H10 | 1 | D10 | 0 | C10 |
| 10 | 0 | SJ | 0 | HJ | 0 | DJ | 0 | CJ |
| 11 | 0 | SQ | 0 | HQ | 1 | DQ | 0 | CQ |
| 12 | 0 | SK | 0 | HK | 0 | DK | 0 | CK |
| 13 | 0 | SA | 0 | HA | 0 | DA | 0 | CA |

This encoding method allows the dataset to clearly represent the hand of each player and provides detailed information about the composition and characteristics of the player's hand, including the distribution of suits and high card points. Based on the presence of 1s and 0s in the encoding, the specific poker cards in a player's hand can be determined. Referring to Table 1 and the provided dataset encoding, we can deduce that Player 1's hand includes: the 2 of Spades (S2), the 3 of Spades (S3), the 4 of Spades (S4), the 8 of Spades (S8), the 9 of Spades (S9), the King of Spades (SK), the 2 of Hearts (H2), the 3 of Hearts (H3), and the Ace of Spades (SA).

## 2.2. Feature Engineering

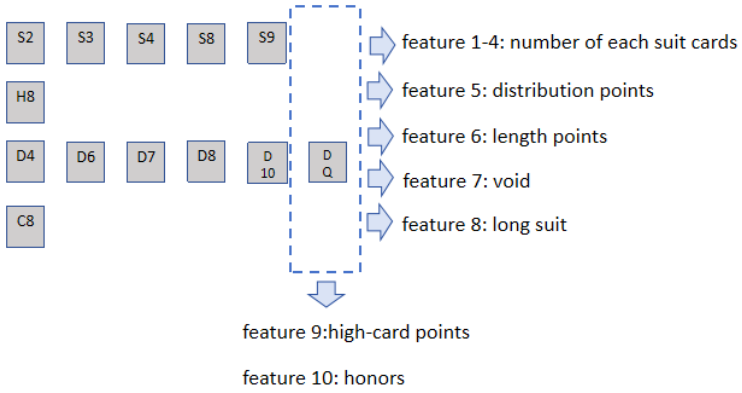


Figure 1: Illustrates the construction of 10 important features.

Each sample dataset contains information for two players, and the same approach is used to extract features for each player. As shown in Figure 1, 10 important features were manually constructed in this study. Features 1 to 4 represent the number of Spades, Hearts, Diamonds, and Clubs cards, respectively. Feature 5 represents Distribution Points, which are used to evaluate the distribution of long and short suits in a hand. Long suits (Feature 6) refer to holding more cards in a particular suit, while short suits are the opposite. Distribution Points assign additional points based on the distribution of long and short suits. Length Points refer to the number of long suits in a hand, indicating the quantity of consecutive cards in a specific suit. Typically, hands with longer lengths are considered more promising. Void indicates the absence of cards in a particular suit. Having a void can provide players with additional advantages in certain situations. Long Suit refers to holding consecutive cards in a specific suit, forming a sequence. Holding a long suit can increase the hand's attacking potential and potential doubling value. High-card points are a simple method for evaluating the value of a hand, assisting players in deciding whether to bid or double. High-card points are a method used in bridge

DOI: 10.54254/2753-8818/83/2025.19923

to assess the value of a hand. The calculation of high-card points is as follows: Ace (A) is worth 4 points, King (K) is worth 3 points, Queen (Q) is worth 2 points, and Jack (J) is worth 1 point. For a hand, each card is scored according to the rules above, and the total points obtained are added to determine the hand's high-card points. Honors (Feature 10) refer to holding Aces, Kings, Queens, and Jacks. Holding honors can increase the value of a hand, especially when inside tricks are needed to ensure success in the contract.

In the game of bridge, high-card points are typically used in conjunction with distribution points to comprehensively evaluate the value of a hand. These features (Feature 1-10) can assist players in understanding the potential value and strengths/weaknesses of their hands more comprehensively, enabling them to more accurately evaluate the value of their hands and formulate better bidding strategies. Considering these features in the data processing and feature extraction code related to bridge can enhance the accuracy and effectiveness of the model.

## 3. Model

### 3.1. Bridge Bidding Model Process

Figure 2 illustrates the model training and evaluation process for bridge bidding. While similar to most machine learning model training processes, there are significant differences in the details. Loading data involves loading the training, validation, and testing datasets along with their corresponding cost data. Feature extraction involves extracting features from the data to obtain features for the training and validation sets, following the specific implementation steps as shown in Section 2.2 Feature Engineering. Subsequently, various parameters necessary for model training need to be set, including network structure, learning rate, batch size, etc. When training the model, initialization of the model and RMSprop parameters, including weights, biases, gradients, etc., is crucial. The next step involves training the model based on the data, loading the trained model, and loading pre-trained model parameters based on different total bidding counts (2 to 5). Finally, the training results are evaluated by checking the errors on the testing dataset. The entire process covers the training, validation, and testing phases of the model, along with the initialization and loading of model parameters. Each step aims to evaluate the model's performance on different datasets and make adjustments accordingly.

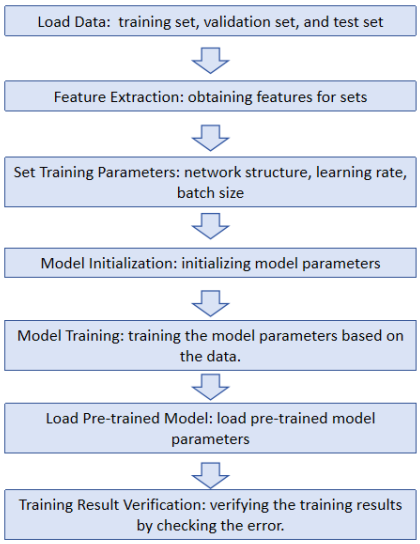


Figure 2: Model Training and Evaluation Process.

DOI: 10.54254/2753-8818/83/2025.19923

### 3.2. Bidding System Comprising Neural Networks

In the machine learning process of a bridge bidding system constructed using neural networks, variables are initialized, and data structures for storing results are set up initially. Subsequently, data batches are iterated through, processing the validation dataset in batches, one batch at a time. Data for the current batch, including the hands of Player 1 and Player 2, is extracted from the validation set. Using the trained model parameters, predictions are made using the latest DQN function to determine the card-playing decisions for each player. The card-playing decisions are then processed, updating the card-playing history based on these decisions and recording the playing situations, further calculating costs. Costs for each card play are computed based on rules and playing situations, and these costs are accumulated into the total cost. The loop continues until the maximum bidding rounds are reached, calculating the final cost. The output includes the maximum allowed bidding rounds, total validation cost, and the final bidding results. This entire process involves model predictions, cost calculations, and result outputs, aiming to evaluate the model's performance on the validation dataset. Through this process, a deeper understanding of the model's bidding performance on real data is gained, facilitating further optimization and adjustments to the model.

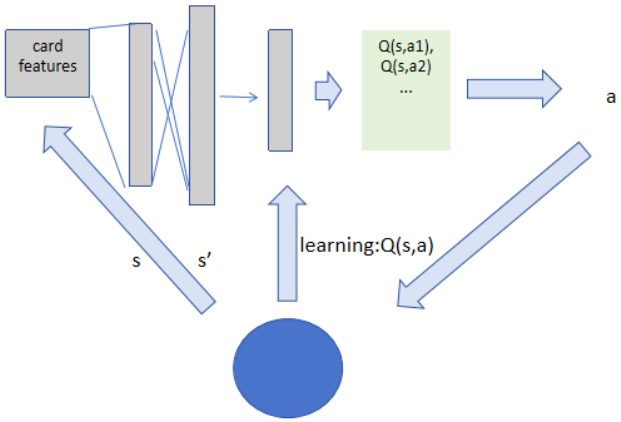


Figure 3: Introduction to DQN.

The core of the bidding system is the DQN algorithm [13]. The DQN algorithm proposed two solutions, both of which have shown remarkable performance in Atari games. Subsequent DQN-related algorithms have been improvements upon it, essentially opening the doors to deep Reinforcement Learning. DQN is essentially Q-Learning and neural networks as shown in Figure 3. Q-Learning algorithm requires maintaining a Q-table and updating it according to the formula mentioned. The learning process involves updating this Q-table. The update formula for the action-value function Q(St,At) is as follows:

$$Q(St,At,w) \leftarrow Q(St,At,w)+\alpha[Rt+1+\gamma maxaq\hat{}(st+1,at,w)-Q(St,At,w)]$$

Here, α is the learning rate, Rt+1 is the reward obtained at time step t+1, γ is the discount factor, St+1 is the next state, at is the action selected in the current state St, maxaq^(st+1,at,w) represents the maximum value after selecting an action in the next state, and w is the parameter. The update formula for the parameter $w$ is:

$$\Delta w=\alpha(Rt+1+\gamma maxaq\hat{}(st+1,at,w)-q\hat{}(st,at,w))\cdot\nabla wq\hat{}(st,at,w)$$

This update method is based on gradient descent, where ∇wq^(st,at,w)represents the gradient with respect to the parameters w of the action-value function. DQN addresses this issue through experience replay. Specifically, it involves using a memory buffer to store data acquired during the exploration

DOI: 10.54254/2753-8818/83/2025.19923

process. This process typically includes the following steps: retrieving a batch of data from the memory buffer and computing target values. By utilizing experience replay, DQN can effectively leverage the advantages of offline policy learning. The behavior policy is responsible for collecting experience data, while the target policy focuses on maximizing value.

## 4. Results

In the DQN model, key aspects related to neural network training are determined based on parameter settings, as shown in Table 2. In this table, decayRate controls the decay of the learning rate, momentum represents the momentum parameter, and alpha is another learning rate parameter. The input denotes the size of the input layer, which includes 52 cards, 36 suits, and 5 features. lsize refers to the size of the hidden layer, layer indicates the number of hidden layers, and output represents the size of the output layer. The configuration of these parameters directly influences the training effectiveness and performance of the neural network.

Table 2: Key Parameters of the DQN Model.

| parameters | explonation | value |
|---|---|---|
| batchsize | The batch size for both updating and training influences the efficiency and stability of training. | 50 |
| decayRate | Controlling the decay rate of the learning rate may help balance exploration and exploitation during training. | 0.98 |
| momentum | Controlling the decay rate of the learning rate may help balance exploration and exploitation during training. | 0.82 |
| alpha | Another learning rate may be used for different weight updates or optimizers. | 0.83 |
| input | The input layer size describes the dimensionality of the input data. | 93 |
| lsize | The size of the hidden layer impacts the network's representational capacity and complexity. | 128 |
| layer | The number of hidden layers affects the depth and complexity of the network. | 4 |
| output | The output layer size describes the dimensionality of the output data. | 36 |

We discussed the computational time required to train our model. The code is written in MATLAB, and the training is conducted on a device equipped with an Intel Xeon Platinum 8368 Processor running at 2.40 GHz and 128GB of RAM. We listed the training time per epoch and total training time for 2-5 rounds of DQN model training as shown in Table 3. The training time increases with the increase in Total bids: From the data in the table, it can be observed that as Total bids increase from 2 to 5, the training time per epoch gradually increases to 1.23 seconds, 2.77 seconds, 4.39 seconds, and 6.11 seconds, respectively. This indicates that an increase in model complexity leads to a significant increase in training time. The convergence time also increases with the increase in Total bids: From the data in the table, it can be seen that as Total bids increase from 2 to 5, the total training time required for the model to converge gradually increases to 1.01 minutes, 2.31 minutes, 6.22 minutes, and 13.83 minutes, respectively. This suggests that as model complexity increases, more training time is needed for the model to converge. Relationship between training time and convergence time: There is a clear correlation between training time and convergence time. As the training time per epoch increases, the total training time also increases. This indicates that an increase in training time directly impacts the model's convergence speed. Impact of model complexity on training time and convergence time: With an increase in Total bids, both training time and convergence time significantly increase.

DOI: 10.54254/2753-8818/83/2025.19923

Table 3: Average Computational Time of the DQN Model.

| Total bids | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| running time per epoch (sec) | 1.23 | 2.77 | 4.39 | 6.11 |
| running time until converge (min) | 1.01 | 2.31 | 6.22 | 13.83 |

In the test set, the cost of Wbridge5 is 3.0039 [5], whereas based on the data results in Figure 3, our cost results are superior. Analyzing the data in Figure 4 allows for a comparison between the test cost and the total number of bids. There is a certain difference between the test cost and the training cost at each total number of bids, indicating that the model's performance on test data may differ from its performance on training data. Impact of total number of bids on cost: As the total number of bids increases from 2 to 5, the overall trends in test cost and training cost do not exhibit a monotonic behavior. In such cases, further data analysis may be necessary to understand the patterns of the total number of bids on cost. Stability of model performance: Observing the test cost data reveals that the variation in test cost is not very significant across different total numbers of bids. This suggests that the model's performance remains relatively stable across different total numbers of bids, indicating a certain level of robustness. Finding the optimal total number of bids: Comparing the data of test cost and training cost can help determine the optimal choice of total number of bids. In this scenario, a balance between model performance and cost is needed to identify the most suitable total number of bids for a specific task.

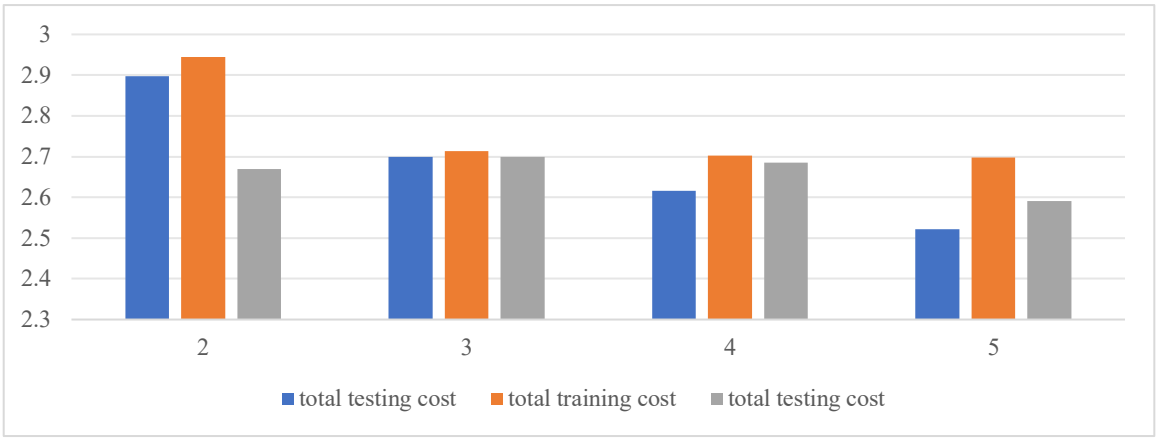


Figure 4: Evaluation of Prediction Results.

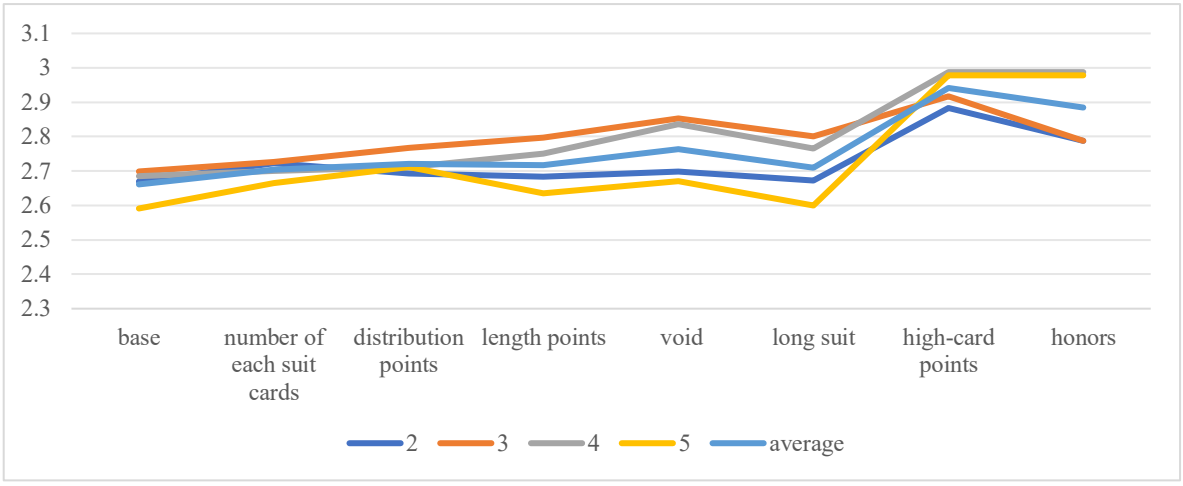


Figure 5: Ablation Experiments of features in bridge bidding system.

DOI: 10.54254/2753-8818/83/2025.19923

Through the analysis of the results of the ablation experiments in Figure 5, we conducted an in-depth exploration of the feature importance of machine learning applied to the bridge bidding system. The feature "number of each suit cards" has a positive impact on model performance, and its removal leads to a slight decrease in average performance. The feature "distribution points" also has a certain impact on model performance, and its removal results in a slight decrease in average performance. The removal of the feature "length points" has a small impact on model performance, with a not very noticeable decrease in average performance. Removing the "void" feature leads to a slight decrease in average performance. The removal of the "long suit" feature has a minor impact on model performance. The feature "high-card points" significantly positively impacts model performance, and its removal leads to a significant decrease in average performance. The feature "honors" also significantly positively impacts model performance, and its removal results in a noticeable decrease in average performance. In summary, the features "number of each suit cards," "high-card points," and "honors" play a positive role in model performance, while the influence of other features is relatively minor. In conclusion, different features play different roles in the machine learning model of the bridge bidding system, with complex and diverse impacts. Optimizing model performance requires a comprehensive consideration of the roles of each feature to achieve better predictive performance and cost-effectiveness.

## 5. Conclusion

This study revolves around the research of the bridge bidding system, analyzing the key issues faced by bidding research and proposing the research path of artificial feature construction structure and DQN model. This paper presents the neural network structure of DQN for finding a suitable neural network structure for bidding game research. Through the comparison of test costs with the total number of bids and the analysis of the results of feature ablation experiments, it is found that different features have different impacts on the machine learning model performance of the bridge bidding system. Some features such as "number of each suit cards" and "high-card points" have a positive impact on model performance, while other features such as "length points" may have a minimal impact on model performance. The degree of feature impact may vary under different total bidding numbers, requiring a comprehensive consideration of feature roles to optimize model performance and achieve better predictive performance and cost-effectiveness.

Although this paper has conducted a lot of research and exploration, there are still many aspects that are not fully considered due to time constraints. Based on the potential issues of the model, future research directions may include: designing neural network structures that better fit bridge features to address the issue of a too large state space. Trying more reward functions to find a reward calculation scheme suitable for bridge. Designing a hand strength calculation function because judging the strength of the hand is the basis of bidding, and a function or algorithm specifically for hand strength calculation could be considered.

## References

[1] *Rong, J., Qin, T., & An, B. (2019). Competitive bridge bidding with deep neural networks. arXiv preprint arXiv: 1903.00900.*

[2] *Yegnanarayana B, Khemani D, Sarkar M. Neural networks for contract bridge bidding[J]. Sadhana, 1996, 21: 395-413.*

[3] *Smith S J J, Nau D S, Throop T A. Success in spades: Using AI planning techniques to win the world championship of computer bridge[J]. AAAI/IAAI, 1998, 98: 1079-1086.*

[4] *Ginsberg M L. GIB: Imperfect information in a computationally challenging game[J]. Journal of Artificial Intelligence Research, 2001, 14: 303-358.*

[5] *Gong Q, Jiang Y, Tian Y. Simple is better: Training an end-to-end contract bridge bidding agent without human knowledge[C]//Real-world Sequential Decision Making Workshop in ICML. 2019.*

DOI: 10.54254/2753-8818/83/2025.19923

[6] *Frank I, Basin D, Bundy A. Combining knowledge and search to solve single-suit bridge[C]//AAAI/IAAI. 2000: 195-200.*
[7] *Amit A, Markovitch S. Learning to bid in bridge[J]. Machine Learning, 2006, 63: 287-327.*
[8] *Lucas S M. Computational intelligence and AI in games: a new IEEE transactions[J]. IEEE Transactions on Computational Intelligence and AI in Games, 2009, 1(1): 1-3.*
[9] *Ho C Y, Lin H T. Contract bridge bidding by learning[C]//Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence. 2015.*
[10] *Yeh C K, Hsieh C Y, Lin H T. Automatic bridge bidding using deep Reinforcement Learning[J]. IEEE Transactions on Games, 2018, 10(4): 365-377.*
[11] *Zhang X, Liu W, Lou L, et al. Ai enabled bridge bidding supporting interactive visualization[J]. Sensors, 2022, 22(5): 1877.*
[12] *Zhang X, Liu W, Yang F. A neural model for automatic bidding of contract bridge[C]//2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). IEEE, 2020: 999-1005.*
[13] *Fan J, Wang Z, Xie Y, et al. A theoretical analysis of deep Q-learning[C]//Learning for dynamics and control. PMLR, 2020: 486-489.*