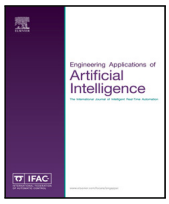
Engineering Applications of Artificial Intelligence 164 (2026) 113395

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Research paper

Offline reinforcement learning for badminton tactical decision-making

Mingjiang Liu, Weizhi Tao, Hailong Huang *

Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, 999077, Hong Kong, China

ARTICLE INFO

Keywords:
Badminton
Tactical decision-making
Offline reinforcement learning
Offline policy evaluation

ABSTRACT

Sports data mining is becoming increasingly vital in modern competitive sports, driven by the need for athletes to continuously enhance their performance. Traditional methods of analyzing sports data rely heavily on expert experience and manual effort, which can be inefficient and unreliable. With advancements in artificial intelligence (AI), sports data is now being processed autonomously, providing more quantitative insights and more comprehensive analysis. This paper focuses on the role of tactics in sports, particularly in badminton, and explores the potential of using AI to enhance badminton tactical decision-making. We investigate the application of offline reinforcement learning (Offline RL) to develop tactical policies from pre-collected datasets, addressing challenges including algorithm design and offline policy evaluation. Specifically, we propose a new variant of conservative Q-learning (CQL), tailored for the hybrid action space to train tactical policies using the integrated offline dataset Shuttle. To evaluate these policies, we develop a preference-based reward model that aligns with tactical preferences, offering an alternative to traditional offline policy evaluation methods. Our computer-based experimental results and analysis demonstrate that the proposed method achieves higher average rewards than all baseline methods and the behavior policy used for data collection. This underscores the potential of the proposed method to enhance badminton tactical decision-making and offer athletes more effective tactical recommendations. Code and data are available at https://github.com/Wenminggong/Offline_RL_for_Badminton.

1. Introduction

Sports data mining is increasingly playing a crucial role in modern competitive sports. With "faster, higher, and stronger" as the slogan of competitive sports, and only one winner in each match, athletes must undergo extensive targeted training to enhance their performance continuously. Traditionally, coaches, analysts, and experts have relied on their professional experience to analyze sports data and design training programs for athletes. This often involves reviewing match videos to identify athletes' weaknesses or adjusting training focus based on physical test data. However, this method is relatively unreliable and inefficient, as it heavily depends on expert experience and requires significant manual effort (Cossich et al., 2023). Nowadays, with the advancement of artificial intelligence (AI) technology, sports data is being collected, processed, and analyzed by AI models (Srilakshmi and Joe, 2023; Wang et al., 2024b; Fernando et al., 2019). This autonomous process generates more quantitative and comprehensive reports (Lin et al., 2024). Consequently, more rigorous training and decision-making recommendations produced by AI models to maximize winning chance is becoming the future trend.

Commonly, tactics and technical skills are two of the most crucial aspects of a match (Kolman et al., 2019). Tactics represent high-level strategies designed to achieve specific goals in a particular sport. Technical skills, on the other hand, refer to the ability to control the body to execute specific movements. Although the effectiveness of tactics varies depending on the opponent, technical skills are generally considered to be more closely tied to an individual athlete. In this paper, we will focus on tactics, investigating how to leverage advanced AI technologies to make tactical decision-making to enhance an athlete's chances of winning.

Particularly, our work focuses on badminton, as it is one of the most widely recognized and representative competitive sports. In a badminton match, two players or two teams (each consisting of two members) compete by alternately hitting a shuttlecock to score points against each other. The process of hitting involves a series of tactical decisions (Wang et al., 2023b). For instance, the athlete must determine the type of shot to execute, such as a net shot or a smash, and decide on the optimal placement of the shuttlecock. Additionally, after returning the shuttlecock, the athlete must decide on his/her subsequent positioning to prepare for the next shot. Currently, in academia, AI models are primarily applied in badminton for match analysis and stroke or tactical prediction (Wang et al., 2022; Chang et al., 2023; Lin

* Correspondence to: The Hong Kong Polytechnic University, 11 Yuk Choi Rd, Hung Hom, Kowloon, 999077, Hong Kong.
E-mail addresses: mingjiangaae.liu@connect.polyu.hk (M. Liu), weizhi.tao@connect.polyu.hk (W. Tao), hailong.huang@polyu.edu.hk (H. Huang).

https://doi.org/10.1016/j.engappai.2025.113395
Received 5 January 2025; Received in revised form 28 October 2025; Accepted 28 November 2025
Available online 30 November 2025

et al., 2024). These approaches commonly employ supervised learning to summarize match situations or predict future shots based on existing data. Works focused on situation summarization and analysis play a supporting role in badminton tactical decision-making, as they still require human experts and coaches to derive effective tactical policies from the summarized information. To address the inherent inefficiency and potential instability of this human-dependent process, this work aims to develop a human-free model that functions as a direct tactical generator. By leveraging such a model, we seek to eliminate human involvement in tactical analysis and mining, thereby reducing human workload and offering more efficient and stable support for badminton tactical decisions. Although studies addressing stroke or tactical prediction can be used to generate future strokes or tactics, they are generally trained to replicate behaviors present in the dataset, which often limits their ability to produce superior tactics for enhancing athlete performance. Therefore, this work attempts to explore a novel approach for generating tactics that outperform those contained in the dataset.

Offline reinforcement learning (Offline RL) as a subset of AI, has recently gained significant attention due to its successes in various domains, such as healthcare applications (Nambiar et al., 2023; Zhang et al., 2023b), chip design (Lai et al., 2023), and robotics (Shah et al., 2023). Offline RL (also known as batch RL) is an approach that learns from a pre-collected static dataset without any online interaction with the environment (Levine et al., 2020). It offers a potential solution to the challenges posed by impractical online interactions in the real world of reinforcement learning (RL). From this point of view, Offline RL is ideally suited for mining badminton tactics, as it does not require online interaction in either real-world settings or badminton simulators. Besides, offline RL has been demonstrated to be capable of achieving a better policy than the behavioral policy used to collect the training data (Fujimoto et al., 2019). However, two significant challenges arise when applying Offline RL to badminton tactics mining: modifying the existing Offline RL algorithm to adapt to the hybrid action space of badminton tactics, and effectively using offline datasets to evaluate the learned policies.

Badminton tactical decision-making is a typical hybrid action-space problem that involves both discrete tactical actions and continuous tactical actions. While classical Offline RL algorithms are designed for either discrete action-space problems or continuous action-space problems, which cannot be applied to badminton tactical decision-making directly. To address this challenge, we select the advanced Offline RL algorithm Conservative Q-Learning (CQL) (Kumar et al., 2020) and modify it to accommodate the hybrid action space. Specifically, we follow the approach in Delalleau et al. (2019) by decomposing the action into discrete and continuous components, while assuming that the continuous action component depends on the selected discrete action and that continuous actions are mutually independent. Consequently, we derive a new variant of CQL termed CQL with Hybrid Action Space.

The challenge of effectively evaluating learned policies using offline datasets is referred to as offline policy evaluation (OPE) (Qin et al., 2022). Despite the proposal of numerous OPE methods, their effectiveness in real-world applications remains unverified. The success of these methods is thought to be influenced by the task, the collected data, and the learned policy (Fu et al., 2021). Evaluating OPE methods involves comparing the estimated return with the true return obtained from a simulator or the real world. Consequently, existing OPE methods lack guaranteed reliability as policy evaluators when only pre-collected offline data is available. Therefore, this paper does not utilize OPE methods for policy evaluation. The objective of the OPE method is to estimate a policy's long-term utility (i.e., $Q_\pi(s,a)$ or $V_\pi(s)$) using a pre-collected offline dataset, which is challenging. Instead, we focus on estimating a policy's short-term reward (i.e., $r(s,a)$), aiming to identify a myopic optimal policy. While this approach does not yield the long-term optimal policy, the myopic optimal policy can still guide athletes' tactical decision-making for the subsequent single step. To develop a reward model, we employ preference-based RL to align with the preferences of tactical actions. Preference-based RL is a framework for learning from pairwise preference feedback (Christiano et al., 2017), and has recently shown success in domains such as robotics (Liu and Chen, 2022) and large language models (Ouyang et al., 2022). The preferences for tactical actions (e.g., preferring the action sequence of the winner over that of the loser in a rally) naturally arise from the offline dataset and are used as an optimization objective to train the reward model.

Specifically, we combine two existing badminton datasets, ShuttleSet (Wang et al., 2023b) and ShuttleSet22 (Wang et al., 2023a), into a larger dataset referred to as Shuttle in this paper. Using Shuttle, we train tactical decision-making policies with the proposed CQL with Hybrid Action Space and four other baselines. Concurrently, we develop a reward model to align with the preferences of tactical actions and use this model to evaluate the learned tactical decision-making policies by estimating their average reward. The experimental results demonstrate that the reward model generalizes well to the test set. According to the reward model, all the learned tactical policies achieve higher average rewards than the behavior policy used to generate the offline data. Particularly, CQL with Hybrid Action Space delivers the best performance, highlighting its superiority in enhancing badminton tactical decision-making. The main contributions of this work are concluded as follows:

- To the best of our knowledge, this work is the first attempt to explore the potential of using Offline RL to enhance badminton tactical decision-making.
- Instead of viewing badminton tactical decision-making as a turn-based sequence decision problem, we formulate it as a player-based Markov Decision Process (MDP), offering a standard framework for RL. Furthermore, to accommodate the hybrid action space inherent in badminton tactical decision-making, we develop a variant of the offline RL algorithm: CQL with Hybrid Action Space. We demonstrate that this variant can be effectively applied to real-world badminton tactical decision-making scenarios.
- Based on offline data, we propose to use a preference-based reward model to evaluate the trained tactical policies. Although this model evaluates only the myopic optimality of a policy, it serves as a viable alternative to existing OPE methods, which lack guaranteed reliability.
- Experimental results and analysis indicate that the tactical policy derived from CQL with Hybrid Action Space achieves significantly higher average rewards than that of the behavior policy used to generate the offline data and other baselines, which shows the potential value of this approach to be applied for badminton athletes' tactical training and recommendation.

The rest of the paper is structured as follows. Firstly, we discuss the related works in Section 2 and present the problem statements and modeling in Section 3. Then, we systematically introduce the proposed method tailored to the hybrid action space of badminton tactics and preference-based offline policy evaluation in Section 4 and Section 5, respectively. In Section 6, the experiments are detailed, with the results and discussions showing Offline RL has the potential to enhance badminton tactical decision-making. Section 7 presents the limitations of this work and the corresponding future research directions. Conclusions are given in Section 8.

2. Related works

In this section, we start by exploring recent applications of AI technologies in badminton, highlighting advancements in game analysis and stroke or tactical prediction, while also noting their limitations in active decision-making. We then discuss the development of Offline RL and its potential for enhancing tactical decision-making in badminton. Following this, we examine methods for OPE, which are crucial for

assessing learned policies when neither online nor simulator interactions are possible. Finally, we delve into preference-based reward learning, focusing on leveraging pairwise preferences to develop a reward function, which has the potential to replace traditional OPE methods and address their reliability issues.

2.1. Badminton data mining

Various AI technologies have been employed for game analysis and stroke or tactical prediction in badminton. Leveraging advanced computer vision, the court, players, and shuttlecock can be autonomously and accurately detected in videos, with the shuttlecock's movement being effectively tracked (Yang et al., 2024; Chu and Situmeang, 2017). These detection and tracking capabilities allow for the automatic segmentation of videos into distinct sets and rallies, enabling the removal of break times or transitional video clips (Huang et al., 2022). Additionally, match data, such as player scores and shot location distributions, can be automatically summarized (Lin et al., 2024). Furthermore, virtual reality (VR) technology facilitates the reconstruction of 3D game views, enhancing the comprehensive analysis of game details (Lin et al., 2024). The advancement and application of these AI technologies offer numerous benefits, including automated large-scale data collection (Wang et al., 2023b,a), improved viewing experience during live broadcasts, and more effective training and match preparation for athletes and coaches. Although current methods can facilitate badminton tactical decision-making, they primarily serve as support tools. A coach or expert is still needed to refine and finalize effective tactical decisions that can help athletes win matches, using the data summarized by the AI models. To minimize reliance on human experts, it is valuable to train AI models to generate effective tactical decisions directly from offline data. To gain insights into tactical usage, AI models have been developed to classify tactical types or predict forthcoming tactics. Specifically, support vector machines (SVM) were utilized for stroke classification (Chu and Situmeang, 2017) and transformer encoder–decoder models were used to predict future strokes and landing positions based on previous rally actions (Wang et al., 2022; Ibh et al., 2024). Additionally, a graph-based forecasting model was proposed to further predict movement positions (Chang et al., 2023). To predict future strokes, landing positions, and movement positions simultaneously, a strong hierarchical imitation learning model was employed (Wang et al., 2024a). While such predictive models are capable of generating tactical decisions, their training is often oriented toward replicating the strategies within the dataset. This makes it difficult for them to produce superior tactics that effectively enhance athlete performance, as they are not inherently optimized for creating novel winning strategies.

2.2. Offline RL

RL offers an online learning paradigm, which encounters significant challenges when online interaction is impractical. To address this, Offline RL was introduced (Levine et al., 2020). Offline RL follows a data-driven learning paradigm, relying solely on previously collected offline data. Although off-policy RL algorithms can learn from offline data naturally, they often struggle to learn effectively from entire offline datasets due to a serious issue: distributional shift (Levine et al., 2020). For instance, although the soft actor-critic (SAC) algorithm (Haarnoja et al., 2018) can utilize data collected by previous policies to update the current policy, it fails to learn on static and offline data (Kumar et al., 2019). To mitigate this, several algorithms have been proposed. Fujimoto et al. introduced a batch-constrained RL algorithm that forces the policy to behave closely to behaviors of the given data by restricting the action space within a subset of the given data (Fujimoto et al., 2019). Kumar et al. devised conservative Q-learning (CQL), which incorporates a simple Q-value regularizer into the standard Bellman error objective to reduce value overestimation caused by distributional shift (Kumar et al., 2020). To balance improving the behavior policy and minimizing deviation from it, implicit Q-learning (IQL) was developed, utilizing a state-conditional upper expectile to estimate the optimal Q-value in that state (Kostrikov et al., 2021). To enhance policy robustness and minimize model bias, dynamics models were incorporated into Offline RL for policy evaluation during training (Swazinna et al., 2021). Another way to address the challenge of learning without online interaction is by modeling the RL problem as a sequence generation problem. This approach leverages high-capacity sequence prediction models, such as Transformer, to generate action sequences that yield high rewards. Notable examples include the decision transformer (Chen et al., 2021) and the trajectory transformer (Janner et al., 2021). The decision transformer conditions an autoregressive model on the expected return, past states, and actions to generate future actions that achieve the expected return, while the trajectory transformer employs beam search to generate future actions. With advancements in effective Offline RL algorithms, Offline RL has been increasingly applied to real-world decision-making problems. Shah et al. introduced the first Offline RL system for robotic navigation, capable of reaching distant goals (Shah et al., 2023). In treatment optimization, where active interaction is restricted, Offline RL has been used to develop effective policies for the treatment of diabetes and sepsis (Nambiar et al., 2023) and the recommendation of ventilator parameters (Zhang et al., 2024). For chip placement, Offline RL has been employed to learn a transferable placement policy that enhances placement quality (Lai et al., 2023). Inspired by these successful applications, this paper explores the potential of using Offline RL in badminton tactical decision-making.

2.3. Offline policy evaluation

Policy evaluation is crucial not only for selecting the optimal learned policy for deployment in online systems but also for assessing the effectiveness of policy learning algorithms, thereby facilitating their development. In the context of Offline RL, policy evaluation must be conducted using solely offline collected data, without any online interaction. To address this, various methods have been devised to estimate the expected return of the learned policy π. Fitted Q-Evaluation (FQE) directly fits a neural network to estimate the expected return by bootstrapping from $Q(s', \pi(s'))$ (Le et al., 2019). A model-based approach considers learning dynamics and reward on transitions, utilizing simulated trajectories generated by the learned policy under the dynamics model to compute the policy's return (Zhang et al., 2021). Additionally, importance sampling conducted by a learned behavior policy can be used to estimate the return. Kostrikov and Nachum utilized self-normalized step-wise importance sampling for this purpose (Kostrikov and Nachum, 2020). To reduce estimation variance, Thomas and Brunskill performed weighted doubly-robust policy evaluation (Thomas and Brunskill, 2016). Importance weights can also be computed without learned behavior policies, using a saddle-point objective (Yang et al., 2020) or a variational power iteration algorithm (Wen et al., 2020). However, existing OPE algorithms do not consistently perform well across a range of simulated tasks (Fu et al., 2021). Furthermore, in a near real-world benchmark (i.e., NeoRL), current OPE methods struggle to select the optimal policy (Qin et al., 2022). Therefore, it is essential to explore alternative methods for offline policy evaluation.

2.4. Preference-based reward learning

Preference-based RL is a framework for learning from pairwise preference feedback without predefined numerical rewards. One representative method involves learning a reward function from pairwise feedback to replace handcrafted numerical rewards, followed by policy optimization based on this learned reward function (Christiano et al., 2017). Typically, pairwise feedback is derived from human preferences,

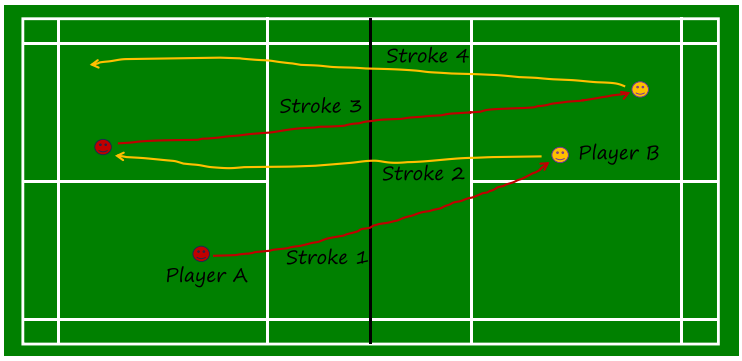


Fig. 1. Illustration shows a badminton rally consisting of four strokes. From a bird's-eye view, Player A and Player B are positioned on opposite sides of the court, each returning two strokes to form the rally. The court is drawn to scale, reflecting the actual dimensions of a badminton court, with white lines indicating the boundaries and a black line representing the net.

requiring at least one human to label preferences between two agent actions or behaviors. Due to the high cost of human feedback in real-world scenarios, poor sample and human feedback efficiency are significant challenges in preference-based RL. To address these issues, an off-policy preference-based RL algorithm was introduced, utilizing relabeled historical experiences and unsupervised pre-training (Lee et al., 2021). Additionally, integrating expert demonstrations and pairwise feedback has also proven effective in enhancing the efficiency of preference-based RL (Ibarz et al., 2018; Palan et al., 2019). A high-efficiency exploration method was developed by considering uncertainty from the reward function (Liang et al., 2022). Recently, to accommodate non-Markovian rewards, a neural architecture using transformers to model human preferences, known as the Preference Transformer, was proposed (Kim et al., 2023). These advancements have facilitated the application of preference-based RL in areas such as Atari games (Christiano et al., 2017; Ibarz et al., 2018) and human–robot interaction (Liu and Chen, 2022). Notably, this approach has been shown successful in fine-tuning large language models to match human preferences and intentions (Ouyang et al., 2022), attracting significant attention with the success of large language models like ChatGPT. In a badminton match, there is a natural preference for the tactical behaviors of the winner over those of the loser. This preference enables the development of a reward function that can serve as an alternative to classical OPE methods for evaluating tactical policies.

3. Problem formulation

In this paper, we focus exclusively on the single-player game of badminton, despite the sport also encompassing doubles matches with two teams of two members each. Typically, a badminton match comprises two or three sets, with victory awarded to the player who first wins two sets. Each set includes a minimum of 21 rallies, where two players alternately return strokes to form a rally. The player who wins a rally earns one point, and the first to reach 21 points claims the set. As shown in Fig. 1, there is an example of a rally consisting of four strokes, where Player A initiates the service and Player B ultimately scores the point after four alternating stroke returns. This alternating stroke process is referred to as a turn-based sequential decision process (Wang et al., 2022). Clearly, a player's decision-making encompasses both tactical decision-making and the execution of technical skills. In this paper, we concentrate on the tactical decision-making aspect and disregard the execution of professional skills necessary to implement the intended tactics.[1] To formally define the tactical decision process, we designate Player A as the serving player and Player B as Player A's opponent, using $\mathcal{M}_{AB}$ to represent a match between Player A and Player B. A rally is denoted as $\mathcal{R}$, assuming the match comprises n rallies, we have $\mathcal{M}_{AB} = \{\mathcal{R}_i\}_{i=1}^{n}$. Furthermore, the ith rally is composed of a sequence of strokes, represented as $\mathcal{R}_i = \{St_1^A, St_2^B, St_3^A, \ldots\}_i$. In essence, a stroke is a decision-making process where the player selects an appropriate action based on his/her current state, thus a stroke can be expressed as a state–action pair, i.e., $St_t^w = (s_t^{turn}, a_t^{turn})^w$, where w represents Player A or Player B, and $\mathcal{R}_i = \{(s_1^{turn}, a_1^{turn})^A, (s_2^{turn}, a_2^{turn})^B, (s_3^{turn}, a_3^{turn})^A, \ldots\}_i$. According to Wang et al. (2024a), the state s^{turn} comprises the current position coordinates of the active player and his/her opponent, represented by $s^{turn} = (pc_p, pc_o)$, where $\cdot_p$ and $\cdot_o$ are used to denote specific attributes of the active player and his/her opponent, respectively. The action a^{turn} includes the shot type executed by the player, the intended landing position of the shuttle, and the player's subsequent movement position, represented by $a^{turn} = (st_p, lp_p, mp_p)$. In this paper, all positions are continuous two-dimensional coordinates represented by (x, y),[2] and 10 shot types are defined by domain experts to distinguish the strokes (Wang et al., 2022), with details provided in Table 1.

However, the turn-based sequential decision-making process involves two agents taking actions alternately. To further formulate this problem as a MDP, we focus on the decision-making of one player in a rally, treating the other player's actions as part of the state of the decision-making player. This MDP is referred to as a player-based MDP in this paper. Specifically, the player-based MDP is defined as a tuple $(S, \mathcal{A}, T, r, \gamma)^w$, where $w \in \{A, B\}$ denotes the player under consideration. Here, S is a set of states. Each state $s^{player} \in S$ includes not only the current position coordinates of the player and his/her opponent but also the opponent's last action, i.e., $s_t^{player} = (s_t^{turn}, a_{t-1}^{turn}) = (pc_{p,t}, pc_{o,t}, st_{o,t-1}, lp_{o,t-1})$.[3] $\mathcal{A}$ is a set of actions. Each action $a^{player} \in \mathcal{A}$ corresponds to a turn-based action, represented as $a_t^{player} = (st_{p,t}, lp_{p,t}, mp_{p,t})$. Typically, $\mathcal{A}$ is a hybrid action space with discrete stroke types and continuous two-dimensional coordinates. To simplify the expression, unless otherwise specified, (s_t, a_t) in the following text refers to $(s_t^{player}, a_t^{player})$. The transition function T describes the probability distribution in the form $T(s_{t+1}|s_t, a_t)$, which transitions current state s_t into next state s_{t+1}. $r : S \times \mathcal{A} \to \mathbb{R}$ defines the reward function. In this paper, we define r as follows:

$$r(s_t, a_t) = \begin{cases} 0 & t \neq terminal \\ \pm 1 & t = terminal \end{cases}. \tag{1}$$

[1] In reality, tactics represent higher-level behaviors that require athletes to control their bodies to perform specific movements associated with these tactics. However, in this paper, we focus solely on tactical decision-making and omit the physical execution process. In other words, we assume that athletes can flawlessly execute any given tactics.

[2] The specific definitions of the continuous two-dimensional coordinates will be given in Section 6.1.

[3] In the turn-based sequential decision process, assuming that the tth stroke is executed by Player A, i.e., $St_t^A = (s_t^{turn}, a_t^{turn})^A$, then the $(t-1)$-th stroke must have been executed by Player B, i.e., $St_{t-1}^B = (s_{t-1}^{turn}, a_{t-1}^{turn})^B$. At time frame t, Player A is the active player, so $(st_{o,t-1}, lp_{o,t-1}, mp_{o,t-1})$ is used to represent the opponent's action of Player A (i.e., the action of Player B) at time frame $t-1$. Additionally, we omit $mp_{o,t-1}$ because $mp_{o,t-1}$ is equivalent to $pc_{o,t}$.

Table 1
Shot types and their meanings.

Shot type	Descriptions
short service	Serve that crosses over the net and lands close to the short service line.
long service	Serve that arcs high and lands deep in the opponent's backcourt.
net shot	Gentle shot that positions the shuttlecock near the net.
clear	Overhead shot where the player hits the shuttlecock from one end of the court to the opposite end.
push/rush	Shot pushed from near the net to reach the backcourt, or a downward shot from near the net designed to land quickly.
smash	Quick, downward-angled shot executed with an overhand motion.
defensive shot	Shot taken when the opponent hits the smash or drive.
drive	Swift and flat shot that travels just above the net, serving both offensive and defensive purposes.
lob	Defensive shot usually executed from the front of the court by pushing the shuttlecock high and deep to the back of the opponent's court.
drop	Shot that positions the shuttlecock near the net, often to force the opponent to move or to set up the next play.

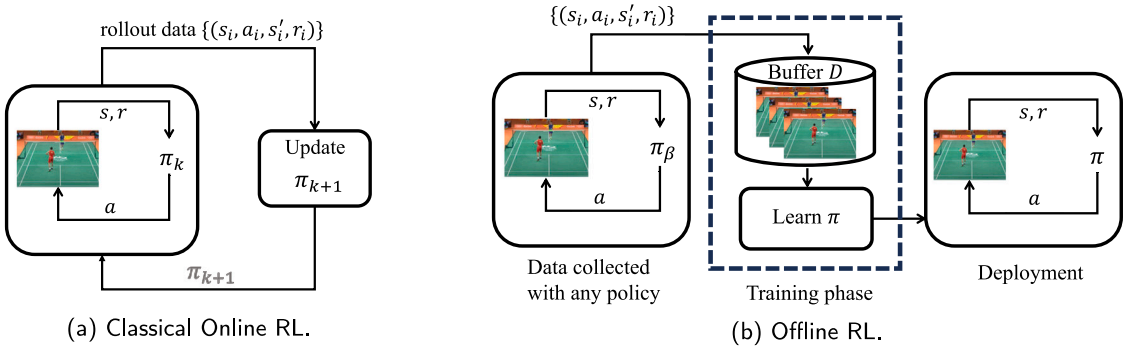


Fig. 2. Pictorial illustration of classic online RL versus offline RL. In online RL, the policy π_k is continuously updated using streaming data collected by π_k itself through ongoing environment interaction. In contrast, offline RL relies solely on a fixed dataset collected in advance by some behavior policy π_β. The training process utilizes only this offline data without any further interaction with the environment, and the trained policy is deployed only after training is fully completed.

This implies that only the terminal step has a non-zero reward: the reward is 1 if the decision-making player scores a point, and -1 otherwise. $\gamma \in (0, 1]$ is a reward discount factor. According to this definition, a turn-based rally can be divided into two player-based trajectories, such that $\mathcal{R}_i = \{\tau^A, \tau^B\}_i$, where

$$\tau^w = \{(s_1^{player}, a_1^{player}, r_1)^w, (s_2^{player}, a_2^{player}, r_2)^w \\ , \ldots, (s_T^{player}, a_T^{player}, r_T)^w\}, w \in \{A, B\}. \tag{2}$$

Based on the defined player-based MDP, our goal is to identify an optimal policy, denoted as π^*, that maximizes the expected long-term discounted rewards (Sutton and Barto, 2018):

$$\pi^* = \arg\max_\pi \mathbb{E}_{a_t \sim \pi(a_t|s_t)} \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right]. \tag{3}$$

As illustrated in Fig. 2, the difference between the classical online RL and offline RL is presented. The absence of both online interaction and a simulator prevents classical online RL methods from being directly applied to train tactical policies. Specifically, we only have access to a pre-collected dataset $D = \{(s_t, a_t, s_t', r_t)\}$ derived from match records, where s_t' denotes the state at time $t + 1$. Therefore, Offline RL becomes the suitable approach for training tactical policies. Our objective is to derive an optimal policy π^* and evaluate its performance using this offline dataset. Once obtained, the optimal policy π^* can function as an automated badminton tactic generator, supporting athlete training and performance enhancement.

4. Offline RL with hybrid action space

4.1. RL preliminaries

One representative approach to optimizing the RL objective in Eq. (3) involves estimating either a state-value function or a state–action value function. The state-value function $V_\pi(s_t)$ represents the expected discounted cumulative reward obtained by following policy π starting from state s_t, while the state–action value function $Q_\pi(s_t, a_t)$ represents the expected discounted cumulative reward obtained by following policy π starting from the state–action pair (s_t, a_t). Formally, these value functions are defined as (Sutton and Barto, 2018):

$$V_\pi(s_t) = \mathbb{E}_{\tau \sim p_\pi(\tau|s_t)} \left[\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})\right], \tag{4}$$

$$Q_\pi(s_t, a_t) = \mathbb{E}_{\tau \sim p_\pi(\tau|s_t, a_t)} \left[\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})\right], \tag{5}$$

where $\tau = (s_t, a_t, \ldots, s_T, a_T)$ denotes a trajectory consisting of a sequence of states and actions sampled according to policy π, and $p_\pi(\cdot)$ represents the trajectory distribution. In the Offline RL setting, the dataset D typically does not contain all possible state transitions. We therefore define the empirical Bellman operator as (Kumar et al., 2020):

$$\hat{\mathcal{B}}^\pi Q(s, a) = r(s, a) + \gamma \mathbb{E}_{a' \sim \pi(a'|s'), s' \sim T(s'|s,a)}[Q(s', a')]. \tag{6}$$

Standard RL algorithms generally alternate between two key steps to obtain an optimal policy: policy evaluation and policy improvement (Sutton and Barto, 2018). These steps can be formulated as (Kumar et al., 2020):

$$\hat{Q}^{k+1} \leftarrow \arg\min_Q \mathbb{E}_{s,a,s' \sim D} \left[(\hat{\mathcal{B}}^{\hat{\pi}^k} \hat{Q}^k(s, a) - Q(s, a))^2\right], \tag{7}$$

$$\hat{\pi}^{k+1} \leftarrow \arg\max_\pi \mathbb{E}_{s \sim D, a \sim \pi(a|s)} \left[\hat{Q}^{k+1}(s, a)\right], \tag{8}$$

where $\hat{Q}$ and $\hat{\pi}$ denote the estimated state–action value function and its corresponding optimal policy, respectively.

4.2. Conservative Q-learning with hybrid action space

Badminton tactical decision-making involves both discrete and continuous tactical behaviors, classifying it as a hybrid action-space problem. In this section, we focus on the advanced Offline RL algorithm:

conservative Q-learning (CQL) (Kumar et al., 2020) and develop a variant of CQL specifically adapted for the hybrid action space.

The distributional shift between the learned policy and the policy that collected the data is one of the major challenges of Offline RL. Applying existing value-based off-policy RL algorithms directly in an offline setting often leads to overestimation of the value function because of bootstrapping from out-of-distribution actions and errors in function approximation. To learn a conservative estimate of the value function, CQL was proposed (Kumar et al., 2020) to provide a lower bound on the true values. To establish a lower bound for the state-value function in a policy $\pi(a|s)$, such that $\mathbb{E}_{\pi(a|s)}[\hat{Q}_\pi(s,a)] \leq V_\pi(s)$, we can introduce an additional regularization term during the policy evaluation step (Kumar et al., 2020), as follows:

$$\hat{Q}^{k+1} \leftarrow \arg\min_Q \alpha^{cql}(\mathbb{E}_{s\sim D, a\sim\pi(a|s)}[Q(s,a)] - \mathbb{E}_{s\sim D, a\sim\pi_\beta(a|s)}[Q(s,a)]) + \frac{1}{2}\mathbb{E}_{s,a,s'\sim D}[(\hat{\mathcal{B}}^{\hat{\pi}^k}\hat{Q}^k(s,a) - Q(s,a))^2], \tag{9}$$

where $\pi_\beta(a|s)$ represents the behavior policy used to collect the dataset D, and $\alpha^{cql} \geq 0$ is a trade-off factor. To reduce computational cost, the one-step policy evaluation is further derived as follows (Kumar et al., 2020):

$$\hat{Q}^{k+1} \leftarrow \arg\min_Q \alpha^{cql}\mathbb{E}_{s\sim D}[\log \sum_a \exp(Q(s,a)) - \mathbb{E}_{a\sim\pi_\beta(a|s)}[Q(s,a)]] + \frac{1}{2}\mathbb{E}_{s,a,s'\sim D}[(\hat{\mathcal{B}}^{\hat{\pi}^k}\hat{Q}^k(s,a) - Q(s,a))^2]. \tag{10}$$

CQL is compatible with any off-policy RL algorithms. In this work, we concentrate on the soft actor-critic algorithm (SAC) (Haarnoja et al., 2018). Taking entropy regularization into account during policy evaluation, the policy evaluation step is presented as follows:

$$\hat{Q}^{k+1} \leftarrow \arg\min_Q \alpha^{cql}\mathbb{E}_{s\sim D}[\log \sum_a \exp(Q(s,a)) - \mathbb{E}_{a\sim\pi_\beta(a|s)}[Q(s,a)]] + \frac{1}{2}\mathbb{E}_{s,a,s'\sim D}[(r(s,a) + \gamma(\mathbb{E}_{a'\sim\hat{\pi}^k(a'|s')}[\hat{Q}^k(s',a')] + \alpha^{en}\mathcal{H}(\hat{\pi}^k(a'|s'))) - Q(s,a))^2], \tag{11}$$

where α^{en} is a temperature parameter used to balance the trade-off between policy entropy and reward, and $\mathcal{H}(\pi)$ denotes the policy entropy. Besides, the policy improvement can be derived as follows (Haarnoja et al., 2018):

$$\hat{\pi}^{k+1} \leftarrow \arg\min_\pi \mathbb{E}_{s\sim D}[-\alpha^{en}\mathcal{H}(\pi(a|s)) - \mathbb{E}_{a\sim\pi(a|s)}[\hat{Q}^{k+1}(s,a)]]. \tag{12}$$

As discussed in Section 3, we consider a hybrid action space comprising discrete stroke types and two kinds of continuous two-dimensional coordinates: the intended landing position of the shuttle and the player's subsequent movement position. For simplicity, we denote the discrete and continuous parts of the action as a^d and a^c, respectively. Intuitively, we assume that the continuous action component depends on the chosen discrete action, and that the intended landing position of the shuttle and the player's subsequent movement position are independent. According to Delalleau et al. (2019), the policy can be decomposed as follows:

$$\begin{aligned}\pi(a|s) &= \pi(a^d|s)\pi(a^c|s,a^d) \\ &= \pi(st_p|s)\pi(lp_p|s,st_p)\pi(mp_p|s,st_p).\end{aligned} \tag{13}$$

Furthermore, the policy entropy in Eqs. (11) and (12) can be calculated as follows (Delalleau et al., 2019):

$$\mathcal{H}(\pi(a|s)) = \mathcal{H}(\pi(a^d|s)\pi(a^c|s,a^d)) = \mathcal{H}(\pi(a^d|s)) + \sum_{a^d}\pi(a^d|s)\mathcal{H}(\pi(a^c|s,a^d)). \tag{14}$$

To prevent one of these two entropies from overshadowing the other, we can employ two weighted factors α_d^{en} and α_d^{en}, to promote exploration for both discrete and continuous actions, respectively (Delalleau et al., 2019).

$$\mathcal{H}(\pi(a|s)) = \alpha_d^{en}\mathcal{H}(\pi(a^d|s)) + \alpha_c^{en}\sum_{a^d}\pi(a^d|s)\mathcal{H}(\pi(a^c|s,a^d)). \tag{15}$$

For clarity, CQL with Hybrid Action Space is detailed in Algorithm 1.

Algorithm 1: CQL with Hybrid Action Space

```
Initialize Q-function Q_θ, and policy π_φ.
for each training timestep t do
    //POLICY EVALUATION
    Updating the Q-function Q_θ by taking gradient steps on the
      objective outlined in Equation (11).
    // POLICY IMPROVEMENT
    Enhancing the policy π_φ by taking gradient steps on the
      objective outlined in Equation (12).
end
```

5. Preference-based offline policy evaluation

5.1. Myopic policy evaluation

In Offline RL settings, where online interaction is limited or unavailable, traditional Off-Policy Evaluation (OPE) algorithms such as Fitted Q-Evaluation (FQE) (Le et al., 2019), model-based methods (MB) (Fu et al., 2021), and importance sampling methods (IS) (Fu et al., 2021) are often considered to lack guaranteed reliability (Qin et al., 2022; Fu et al., 2021). This is because evaluating an OPE algorithm typically requires comparing the estimated long-term reward $\hat{V}_\pi(s_0)$ with the true long-term reward $V_\pi(s_0)$ through online interaction, either in the real world or a realistic simulator. Furthermore, estimating the long-term reward of a policy involves bootstrapping the Q-function, as in FQE, learning a transition model to simulate environment dynamics, as in MB, or first learning a behavior policy, as in IS. These approaches involve multi-step estimations to compute a single value, which may lead to error accumulation. As a result, estimating long-term rewards using solely from an offline dataset poses a considerable challenge. To tackle this, we propose an alternative approach: myopic policy evaluation, which focuses on estimating the one-step reward for the learned policy. This method avoids the error accumulation problem. We will next present the formulation of the myopic policy evaluation and explain why the myopic policy remains valuable in the context of badminton tactical decision-making.

In contrast to the optimal long-term policy derived from Eq. (3), the optimal myopic policy focuses on maximizing the immediate, one-step reward, defined as follows:

$$\pi^*_{myopic}(a|s) = \arg\max_a r(s,a), \forall s \in \mathcal{S}. \tag{16}$$

Correspondingly, evaluating the myopic policy in the offline settings just needs to estimate the average reward:

$$\hat{r}_\pi^{AVG} = \mathbb{E}_{s\sim D, a\sim\pi}[\hat{r}(s,a)], \tag{17}$$

where $\hat{r}(s,a)$ represents a generalized estimated reward. Once the optimal myopic policy π^*_{myopic} is learned, it can be used to determine the best tactical action based on the current or historical situations. This approach is particularly useful in scenarios such as analyzing game video recordings of a player. The myopic policy can generate a sequence of one-step optimal actions, which serve as reference actions for the player to enhance their performance. By focusing on immediate rewards, the myopic policy provides actionable insights that can be directly applied to improve decision-making in real time, offering a practical tool for tactical refinement and strategic planning.

5.2. Preference-based reward learning

According to Eq. (17), the evaluation of the myopic policy depends on a generalized estimated reward function. In the context of the reward function defined in Eq. (1), only the terminal step of a rally yields a non-zero reward. However, in autonomous decision-making, it is challenging to predict whether a given action will conclude a

rally due to the absence of environmental dynamics. Consequently, Eq. (1) cannot be directly used as the evaluation reward. Inspired by the observation that there are some preferences between different players' tactical decisions, for example, we favor the tactical decisions of the winner, as they are more likely to maximize the chances of winning. We explore preference-based RL, which substitutes numerical rewards with preferences between two tactical decision-making segments (Christiano et al., 2017). Formally, a decision-making segment is a sequence of states and actions $\{(s_k, a_k), (s_{k+1}, a_{k+1}), \dots, (s_{k+H}, a_{k+H})\}$. Given that rallies in badminton are relatively short (with an average length of 10), we propose using the entire player-based rally $\tau^w = \{(s_1^{player}, a_1^{player})^w, (s_2^{player}, a_2^{player})^w, \dots, (s_T^{player}, a_T^{player})^w\}, w \in \{A, B\}$ to establish preferences.

Assuming we have a pair of rallies from Player A_0 and Player B_0: (τ^{A_0}, τ^{B_0}), the preference can be expressed as $pr^0 = (\tau^{A_0} \succ \tau^{B_0})$ to indicate that rally τ^{A_0} is preferred over τ^{B_0}, or $pr^0 = (\tau^{B_0} \succ \tau^{A_0})$ to indicate that rally τ^{B_0} is preferred over τ^{A_0}. Intuitively, rallies employing more effective tactics should yield higher cumulative rewards. The learned reward function must adhere to this principle. Following the approaches of Christiano et al. (2017) and Liu and Chen (2022), we use the learned reward function $\hat{r}$ to model the preference predictor for a pair of rallies according to the Bradley–Terry model (Bradley and Terry, 1952):

$$P_\psi[\tau^{A_i} \succ \tau^{B_i}] = \frac{\exp \sum_{t=0}^{T} \hat{r}_\psi(s_t^{A_i}, a_t^{A_i})}{\sum_{w \in \{A_i, B_i\}} \exp \sum_{t=0}^{T} \hat{r}_\psi(s_t^{w}, a_t^{w})}. \tag{18}$$

To align the preference predictor with the provided preference feedback, we treat the reward learning process as a binary classification problem. Specifically, the reward function $\hat{r}_\psi$, parametrized by ψ, is updated to minimize the following cross-entropy loss (Liu and Chen, 2022):

$$\begin{aligned} J_r(\psi) = -\mathbb{E}_{(\tau^{A_i}, \tau^{B_i}, pr^i) \sim D} \big[& \mathbb{I}\{pr^i = (\tau^{A_i} \succ \tau^{B_i})\} \log P_\psi[\tau^{A_i} \succ \tau^{B_i}] \\ & + \mathbb{I}\{pr^i = (\tau^{B_i} \succ \tau^{A_i})\} \log P_\psi[\tau^{B_i} \succ \tau^{A_i}] \big]. \end{aligned} \tag{19}$$

6. Experiments

To assess whether Offline RL can develop a superior badminton tactical decision policy compared to the behavior policy[4] used to generate the offline data, we present the results of a series of experiments in this section. First, we detail the data processing steps in Section 6.1, which aim to produce a larger and high-quality player-based badminton dataset for policy training and evaluation. Using this offline dataset, we trained a generalized reward model and several tactical policies. We then employed the reward model to evaluate the myopic average rewards of the learned tactical policies. The details and main results are presented in Section 6.2. Finally, we present deeper analyses and discussions in Section 6.3 to provide clear insights about what tactical behaviors contribute to the improvement of tactical decision-making.

6.1. Data preprocessing

ShuttleSet is a publicly accessible turn-based singles badminton dataset containing stroke-level records, aimed at encouraging research in badminton stroke prediction (Wang et al., 2023b). It includes data from 44 international badminton matches conducted between 2018 to 2021, featuring 27 top-ranking men's and women's singles players. Unlike datasets generated autonomously, ShuttleSet relies on annotations from domain experts, providing detailed descriptions of badminton matches and ground truth features such as stroke types and shuttle landing coordinates. To further expand the dataset, ShuttleSet22 (Wang et al., 2023a) was collected using the same labeling tools and similar stroke-level data formats as ShuttleSet. ShuttleSet22 includes 58 international badminton matches from 2022, involving 35 top-ranking singles players. In this paper, we integrated these two datasets into a larger dataset referred to as Shuttle. Consequently, Shuttle encompasses a total of 94 international badminton matches from 2018 to 2022 and includes 43 top-ranking singles players.[5]

Due to the presence of flawed data in the dataset, it is necessary to perform data screening and filtering to ensure that only high-quality data is retained for analysis and model training. We applied a series of criteria to filter the data, with each criterion and the resulting data quantities detailed in Table 2. This process resulted in 5416 valid rallies for training and evaluation. We split the dataset into a training set, validation set, and test set with a ratio of 7:2:1. This division ensures that the validation and test sets include data from players not present in the training set, allowing us to assess the model's generalization performance. The statistical results of these three datasets are shown in Table 3.

Both ShuttleSet and ShuttleSet22 are based on a turn-based sequential decision process, where a rally is formulated with two players alternately returning strokes. Considering the player-based MDP, we need to convert these turn-based rallies into player-based rallies. As mentioned in Section 3, each turn-based rally can be converted into two player-based rallies, so the number of player-based rallies in the Shuttle dataset will double. Specifically, the training set, validation set, and test set contain $3,930 \times 2 = 7,860$, $904 \times 2 = 1,808$, and $582 \times 2 = 1,164$ rallies, respectively. However, during a match, the two competing players stand on opposite sides of the court, resulting in different coordinate distributions for each player.[6] Using these original coordinates directly for model training can result in poor performance due to inconsistencies in coordinate distributions. To address this issue, we utilized the symmetry of the court to normalize the coordinates. As shown in Fig. 3, when a coordinate should be located on the right side of the court, we use o_r as the coordinate origin and normalize this coordinate with $\bar{x} = \frac{x_{o_r} - x}{\Delta x}, \bar{y} = \frac{y - y_{o_r}}{\Delta y}$. Conversely, when a coordinate should be located on the left side, we use o_l as the origin and normalize it with $\bar{x} = \frac{x - x_{o_l}}{\Delta x}, \bar{y} = \frac{y_{o_l} - y}{\Delta y}$. After normalization, all coordinates will follow the same distribution. In this case, the normalized coordinates $\bar{x} \in [0, 1]$ and $\bar{y} \in [0, 1]$ indicate in-bounds areas; otherwise, they are out of bounds.

6.2. Experimental settings and main results

6.2.1. Preference-based reward learning

In addition to the standard Bradley–Terry model shown in Eq. (18), we also examined a variant referred to as the Normalized Bradley–Terry model, formulated as follows:

$$P_\psi[\tau^{A_i} \succ \tau^{B_i}] = \frac{\exp \frac{1}{T_{A_i}} \sum_{t=0}^{T_{A_i}} \hat{r}_\psi(s_t^{A_i}, a_t^{A_i})}{\sum_{w \in \{A_i, B_i\}} \exp \frac{1}{T_w} \sum_{t=0}^{T_w} \hat{r}_\psi(s_t^{w}, a_t^{w})}. \tag{20}$$

In this model, T_{A_i} represents the length of a player-based rally τ^{A_i}. The length normalization is employed to mitigate the adverse effects caused by differences in rally lengths.

[4] The offline dataset was collected from a series of international matches performed by multiple top players, simplify, we use the term "the behavior policy" throughout the paper to denote the tactical policies employed by all the players in the collected matches.

[5] There are 8 duplicate matches and 19 duplicate players between ShuttleSet and ShuttleSet22. When combining these datasets, duplicates were removed, resulting in a final match count in Shuttle that is less than the sum of the two datasets (i.e., $94 < 44 + 58$), and a final player count that is also less than the total number of players in the two datasets (i.e., $43 < 27 + 35$).

[6] Both ShuttleSet and ShuttleSet22 provide positions with two-dimensional coordinates on the image plane. Furthermore, they provide homography matrices to transform these image-plane coordinates into court-plane coordinates.

Table 2
Data filtering procedures.

Process descriptions	Rally num	Action num
Original dataset.	7,888	81,939
Removing the rally contains the action marked as *flaw*.	6,051	65,167
Removing the rally contains the action without *getpoint_player* label.	6,040	65,021
Removing the rally contains the action without *shot_type* label or *shot_type=unknown*.	6,033	64,987
Removing the rally contains the action without *location* information.	5,973	64,829
Removing the rally contains the action with *lose_reason=unknown*.	5,966	64,749
Removing the rally contains the action with *lose_reason=misjudged*.	5,790	63,047
Removing the rally contains the action marked as *out* while *landing_coordinate* belongs to the inside area.	5,722	62,304
Removing the rally contains the action marked as *in* while *landing_coordinate* belongs to the outside area.	5,701	62,128
Removing the rally contains the action marked as *no_passing_half_court* while *landing_coordinate* belongs to the half count of the opponent.	5,429	59,377
Removing the rally contains the action where the distance between the *hitting_coordinate* and the *player_coordinate* is larger than the range of half court.	5,429	59,377
Removing the rally contains non-monotonic time frames.	5,416	59,472

Table 3
Statistical results of the training set, validation set, and test set.

Items	Training set	Validation set	Test set
Match num	67	18	8
Rally num	3,930	904	582
Player IDs not present in the training set	–	{1, 4, 32, 38, 42}	{2, 3, 21, 26}

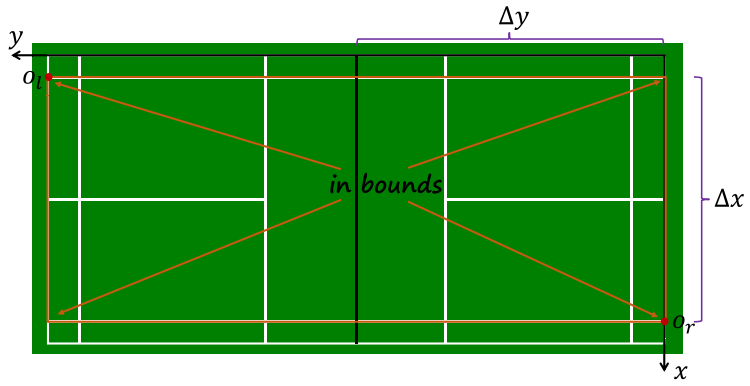


Fig. 3. Illustration of the coordinate normalization.

As discussed in Section 5.2, we utilize the entire player-based rally to establish preferences. Intuitively, the winner's tactical decisions are favored over the loser's in a rally. We pair the player-based rallies from the same turn-based rally into a preference pair, labeling the preference based on which player wins the rally point. Specifically, if Player A wins the current rally, the preference will be labeled as $(\tau^A \succ \tau^B)$; otherwise, it is $(\tau^B \succ \tau^A)$. This is referred to as rally preference, which is the primary preference used to train the reward model. Besides, we consider another preference called non-terminal rally preference. This preference excludes the terminal action in a turn-based rally and then is constructed similarly to the rally preference. The goal of establishing the non-terminal rally preference is to enable the learned reward function to identify the superior player-based rally even without considering the terminal action in the turn-based rally.

To evaluate which preference predictor and which preference design is better, we compared various settings. All the settings are listed as follows:

- **Bradley–Terry model with rally preference (BT_Rally)**: Using the Bradley–Terry model to predict preferences and training the reward model solely with rally preferences.
- **Bradley–Terry model with rally preference and non-terminal rally preference (BT_Rally2)**: Utilizing the Bradley–Terry model to model the preference predictor and using both rally preferences and non-terminal rally preferences to train the reward model. For a batch of training data, we derive rally preferences and non-terminal rally preferences simultaneously and combine the loss functions from these two preferences by a scalar factor α_{pref}, i.e., $\mathcal{L}_r(\psi) = \mathcal{L}_{rally}(\psi) + \alpha_{pref}\mathcal{L}_{non\text{-}terminal\_rally}(\psi)$.
- **Normalized Bradley–Terry model with rally preference (NBT_Rally)**: Utilizing the normalized Bradley–Terry variant to model the preference predictor and only using rally preferences to train the reward model.
- **Normalized Bradley–Terry model with rally preference and non-terminal rally preference (NBT_Rally2)**: Utilizing the normalized Bradley–Terry variant to model the preference predictor and using both rally preferences and non-terminal rally preferences to train the reward model. The loss functions from these two preferences are combined in the same way as BT_Rally2.

We trained the reward models to minimize the objective specified in Eq. (19) using the training set. The training process was conducted with three random seeds, and the network architecture and hyperparameters of these models are detailed in Appendix A.1. Upon completing the training phase, we evaluated the performance of the learned reward models on the test set. As detailed in Table 3, the test set includes both players who appeared in the training data and those who did not. To examine potential performance variations between in-distribution

Table 4
Evaluation results of reward models.

Settings	In-distribution Scenarios			Out-of-distribution Scenarios		
	RP Acc	NTRP Acc	AP Acc	RP Acc	NTRP Acc	AP Acc
BT_Rally	**0.9692 ± 0.0016**	0.5908 ± 0.0054	0.9227 ± 0.0022	**0.9539 ± 0.0038**	**0.6843 ± 0.0069**	0.9246 ± 0.0020
BT_Rally2	0.9538 ± 0.0000	**0.5919 ± 0.0078**	0.9106 ± 0.0031	0.9458 ± 0.0038	0.6694 ± 0.0157	0.9121 ± 0.0049
NBT_Rally	0.8471 ± 0.0082	0.5138 ± 0.0224	0.9162 ± 0.0098	0.8631 ± 0.0084	0.5894 ± 0.0115	0.9294 ± 0.0116
NBT_Rally2	0.8339 ± 0.0068	0.5490 ± 0.0138	**0.9284 ± 0.0048**	0.8550 ± 0.0157	0.5894 ± 0.0100	**0.9379 ± 0.0055**

and out-of-distribution scenarios, we categorized the test instances into two groups: in-distribution scenarios and out-of-distribution scenarios. This classification is based on whether at least one player in the rally was absent from the training set. The evaluation metrics include rally-preference accuracy (RP Acc), non-terminal rally-preference accuracy (NTRP Acc), and action-preference accuracy (AP Acc). These metrics measure how well the learned reward models align with rally preferences, non-terminal rally preferences, and action preferences,[7] respectively. Table 4 presents the evaluation results, which include the mean and standard deviation over three runs.

As illustrated in Table 4, the Bradley–Terry models (i.e., BT_Rally and BT_Rally2) demonstrate higher rally-preference accuracy compared to the Normalized Bradley–Terry models (i.e., NBT_Rally and NBT_Rally2). This result suggests that the difference in length between paired rallies does not significantly impact the performances of the Bradley–Terry models. As rally preferences are viewed as the primary preferences in this paper, BT_Rally emerges as the optimal configuration for the reward model. Obviously, the non-terminal rally-preference accuracy is notably lower, indicating the challenge of aligning non-terminal rally preferences. Nevertheless, BT_Rally achieves high accuracy in rally-preference and action-preference under both in-distribution (with averaged scores of 0.9692 and 0.9227, respectively) and out-of-distribution scenarios (with averaged scores of 0.9539 and 0.9246, respectively), demonstrating its excellent generalization of rally-preference alignment and action-preference alignment in the test set.

As discussed in Section 1, traditional OPE methods aim to estimate long-term returns. However, these methods lack guaranteed reliability when neither real-world interaction nor simulator interaction is available. Similarly, the preference-based reward model serves as an estimator of the true reward. Nevertheless, its superior preference generalization, as demonstrated in Table 4, provides a confidence coefficient that supports our trust in this evaluation model. Therefore, we use the reward function derived from the BT_Rally model to assess the Offline RL policies in the following section. Additionally, the evaluation results of a typical OPE algorithm (i.e., FQE (Le et al., 2019)) are provided as reference results in Appendix B.2.

6.2.2. Offline RL

To evaluate the performance of the proposed method (CQL with Hybrid Action Space), we compare it against several baselines adapted for the hybrid action space. The selected baselines are as follows:

- **Decision Transformer (DT) with Original Goal**: Decision Transformer (DT) (Chen et al., 2021) formulates Offline RL as a sequence modeling task by leveraging the powerful distribution modeling capacity of transformer architectures (Vaswani et al., 2017). It models the joint distribution of state, action, and reward sequences and then conditions on historical states, actions, and returns-to-go to generate subsequent actions. This baseline specifically uses the original accumulated long-term rewards from the dataset as the returns-to-go.
- **Decision Transformer (DT) with Winning Goal**: This variant replaces the original accumulated returns from the dataset with a constant winning signal (a return of 1) as the expected returns-to-go, representing the goal of winning the game.
- **Behavior Cloning (BC)**: Behavior Cloning (Pomerleau, 1988) is a classical imitation learning method that learns a policy by minimizing the discrepancy between predicted and demonstrated actions in a supervised manner.
- **Sequence-based Behavior Cloning (BC)**: As an extension of BC, this baseline utilizes a transformer architecture for sequence modeling. Its formulation is similar to that of DT but does not incorporate returns-to-go conditioning.

Detailed descriptions, network architectures, and hyperparameters for the proposed method and all baselines are provided in Appendix A.2, Appendix A.3, and Appendix A.4, respectively.

As discussed in Section 5.1, rather than estimating the long-term return for policy evaluation, we employed the learned reward function to assess the myopic optimality of the policy. Given that the BT_Rally model achieves the highest rally-preference accuracy, it is used to estimate the average rewards of the learned policies. We also estimated the average reward of the behavior policy to compare its performance with that of the learned policies.

The test set includes both winning and losing rallies. To demonstrate the performance of the policies in different scenarios, we estimated the average rewards of the policies in winning rallies, losing rallies, and integrated rallies, respectively. Furthermore, to assess policy generalization on out-of-distribution data, we categorized the test set into in-distribution and out-of-distribution scenarios based on whether the players had appeared in the training set. The results are presented in Table 5.

As shown in Table 5, both CQL with Hybrid Action Space and the baseline methods achieve higher average rewards than the behavior policy in both integrated rallies and losing rallies. This indicates that training AI models with winning rallies can effectively help generate improved tactics for losing rallies. Notably, CQL with Hybrid Action Space demonstrates optimal performance, underscoring its superiority in enhancing the myopic optimality of tactical decisions. In winning rallies, however, the behavior policy itself performs well, making it difficult to surpass. Only CQL with Hybrid Action Space succeeds noticeably in this regard, further demonstrating its ability to generate tactics that exceed those present in the dataset. Additionally, given that BC's objective is to replicate dataset actions, its marginal performance gain over the behavior policy merely confirms its effectiveness as a supervised learning method for imitating winning behaviors. When comparing in-distribution and out-of-distribution results, only a slight performance degradation is observed, indicating strong generalization capability on out-of-distribution data. Furthermore, in the comparison between DT with Winning Goal and DT with Original Goal, the former achieves a slightly higher average reward, highlighting the value of winning condition modeling, though it does not perform as well as CQL with Hybrid Action Space.

6.3. Discussions

6.3.1. Which types of tactical actions contribute to the rewards?

In this section, we aim to identify which types of tactical behaviors contribute to the rewards. To investigate this, we examine the

[7] In a turn-based rally, if the terminal action results in scoring a point, it is considered the most crucial decision in the rally. Consequently, action preferences are established as $a_T \succ a_i$ for $i \in [1, T-1]$; otherwise, the preferences are $a_i \succ a_T$ for $i \in [1, T-1]$. We introduced the action-preference accuracy to evaluate the reward model more comprehensively.

Table 5
Average rewards of the learned policies.

Policies	In-distribution Scenarios			Out-of-distribution Scenarios		
	Integrated rallies	Winning rallies	Losing rallies	Integrated rallies	Winning rallies	Losing rallies
Behavior Policy	0.7254	0.8460	0.6047	0.7273	0.8462	0.6439
CQL with Hybrid Action Space	**0.8703 ± 0.0786**	**0.8698 ± 0.0757**	**0.8709 ± 0.0815**	**0.8679 ± 0.0873**	**0.8852 ± 0.0685**	0.8557 ± 0.1006
DT with Winning Goal	0.8027 ± 0.0046	0.7998 ± 0.0042	0.8056 ± 0.0050	0.7939 ± 0.0046	0.7794 ± 0.0090	0.8041 ± 0.0030
DT with Original Goal	0.7821 ± 0.0049	0.7998 ± 0.0042	0.7643 ± 0.0060	0.7706 ± 0.0054	0.7794 ± 0.0090	0.7644 ± 0.0046
BC	0.8482 ± 0.0042	0.8473 ± 0.0043	0.8492 ± 0.0041	0.8581 ± 0.0055	0.8498 ± 0.0066	**0.8638 ± 0.0049**
Sequence-based BC	0.7807 ± 0.0042	0.7789 ± 0.0052	0.7824 ± 0.0033	0.7740 ± 0.0037	0.7638 ± 0.0089	0.7812 ± 0.0024

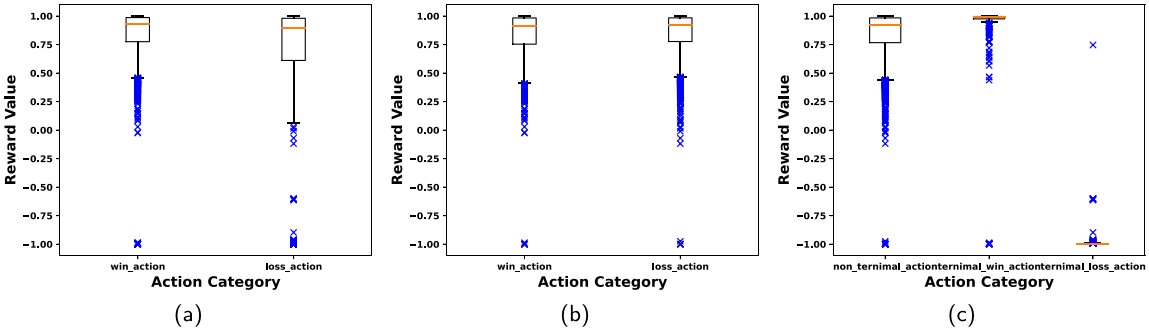


Fig. 4. Reward distributions across different tactical action categories. (a) Rally-preference case. (b) Non-terminal rally-preference case. (c) Action-preference case.

distribution of reward values across three scenarios: rally preference, non-terminal rally preference, and action preference. For the rally-preference scenario, we classify all tactical actions into two categories: win and loss, based on whether the actions originate from winning or losing rallies, and we compare the reward distributions between these two categories. Similarly, in the non-terminal rally-preference scenario, actions are also divided into win and loss, with the distinction that terminal actions in turn-based rallies are excluded. In the action-preference scenario, tactical actions are divided into three categories: non-terminal actions, terminal actions in winning rallies, and terminal actions in losing rallies. We then compare the reward distributions across these three categories. The results are presented in Fig. 4.

From Fig. 4(c), we observe that the terminal actions in winning rallies receive reward values close to 1, while the terminal actions in losing rallies receive reward values near −1. The reward distribution for non-terminal actions falls between these two extremes. This result aligns with the intuition that when a terminal action results in scoring, it is considered the most crucial tactical decision in the rally and should receive the highest reward value. Conversely, when a terminal action results in losing a point, it is deemed a poor tactical decision and should receive a lower reward value. This type of reward distribution enables the reward model to achieve high action-preference accuracy, as shown in Table 4.

As illustrated in Fig. 4(b), the reward distributions for winning and losing actions show no significant difference when terminal actions are excluded. This contradicts the intuition that non-terminal actions from a winning rally contribute more to the winning outcome than those from a losing rally, even if they do not directly lead to victory, they at least help the athlete gain initiative or advantages. However, this result is consistent with the non-terminal rally-preference accuracy, which is shown slightly above 0.6.

Finally, considering the rally-preference scenario, which includes both non-terminal and terminal actions, the reward distributions of winning actions and losing actions can be viewed as integrations of the reward distribution of winning actions in the non-terminal rally-preference scenario with the reward distribution of terminal-win actions in the action-preference scenario and the reward distribution of losing actions in the non-terminal rally-preference scenario with the reward distribution of terminal-loss actions in the action-preference scenario, respectively. Consequently, the average reward for winning actions is slightly higher than that for losing actions, as depicted in Fig. 4(a).

6.3.2. Does the learned policies outperform the behavior policy when excluding the terminal actions?

Terminal actions are assigned extreme reward values, as illustrated in Fig. 4(c). We are curious to see if the learned policies will continue to achieve higher average rewards than the behavior policy when terminal actions are excluded. To investigate this, we remove the terminal actions and compare the average rewards across all policies. The results are shown in Table 6.

Compared to the results in Table 5, the performance of the behavior policy improves significantly in both integrated rallies and losing rallies when terminal actions are excluded. However, no clear improvement is observed in the performance of the learned policies. When terminal actions are excluded, only CQL with Hybrid Action Space shows a distinctly superior performance compared to the behavior policy. This suggests that the policies derived from CQL with Hybrid Action Space have the potential to enhance tactical decision-making earlier in the rallies, beyond just the terminal decisions.

6.3.3. Does the learned policies outperform the behavior policy when excluding the out-of-bounds actions?

In the terminal step, only out-of-bounds actions result in losing a point. As the terminal actions in losing rallies are assigned a negative reward close to −1, out-of-bounds actions may also receive negative rewards. We are curious to see if the learned policies will still achieve higher average rewards than the behavior policy when out-of-bounds actions are excluded. The average rewards, excluding out-of-bounds actions, are reported in Table 7.

Compared to the results in Table 5, the performance of the behavior policy improves significantly in losing rallies, which in turn enhances performance in integrated rallies. The average rewards of all the learned policies, except for CQL with Hybrid Action Space, do not show significant improvement. This suggests that these models are not affected by out-of-bounds actions, as they have learned to land the shuttle within the boundaries. Notably, the performance of CQL with Hybrid Action Space shows a marked increase. Although this

Table 6
Average rewards of the learned policies without terminal actions.

Policies	Integrated rallies	Winning rallies	Losing rallies
Behavior Policy	0.8402	0.8434	0.8369
CQL with Hybrid Action Space	**0.8669 ± 0.1001**	**0.8668 ± 0.0970**	**0.8671 ± 0.1033**
DT with Winning Goal	0.8019 ± 0.0059	0.7979 ± 0.0054	0.8059 ± 0.0064
DT with Original Goal	0.7820 ± 0.0060	0.7979 ± 0.0054	0.7655 ± 0.0073
BC	0.8480 ± 0.0050	0.8453 ± 0.0051	0.8507 ± 0.0049
Sequence-based BC	0.7803 ± 0.0046	0.7780 ± 0.0055	0.7826 ± 0.0037

Table 7
Average rewards of the learned policies without out-of-bounds actions.

Policies	Integrated rallies	Winning rallies	Losing rallies
Behavior Policy	0.8470	0.8532	0.8400
CQL with Hybrid Action Space	**0.9312 ± 0.0025**	**0.9300 ± 0.0025**	**0.9324 ± 0.0024**
DT with Winning Goal	0.8024 ± 0.0054	0.7992 ± 0.0055	0.8056 ± 0.0054
DT with Original Goal	0.7816 ± 0.0059	0.7992 ± 0.0055	0.7645 ± 0.0068
BC	0.8490 ± 0.0052	0.8475 ± 0.0054	0.8506 ± 0.0051
Sequence-based BC	0.7803 ± 0.0049	0.7783 ± 0.0060	0.7823 ± 0.0039

Table 8
Policy evaluation using domain metrics (↑: expecting high value; ↓: expecting low value).

Domain Metrics	CQL with Hybrid Action Space	DT with Winning Goal	DT with Original Goal	BC	Sequence-based BC
Action Difference Rate↑	**0.6647 ± 0.0071**	0.4886 ± 0.0011	0.4861 ± 0.0012	0.5006 ± 0.0022	0.4876 ± 0.0025
Rally Difference Rate↑	**0.8949 ± 0.0094**	0.8519 ± 0.0022	0.8555 ± 0.0042	0.8580 ± 0.0020	0.8559 ± 0.0020
Irrational Shot Type Rate↓	0.0080 ± 0.0138	**0.0 ± 0.0**	**0.0 ± 0.0**	**0.0 ± 0.0**	$3.6e-5 \pm 6.2e-5$
Active Shot Type Rate↑	**0.3526 ± 0.0150**	0.3015 ± 0.0050	0.2766 ± 0.0045	0.3484 ± 0.0089	0.2727 ± 0.0037
Out-of-bounds Action Rate↓	0.0317 ± 0.0496	$0.0002 \pm 6.2e-5$	$0.0003 \pm 6.2e-5$	**0.0 ± 0.0**	0.0001 ± 0.0001
Average Distance of OLP↑	**0.3898 ± 0.0176**	0.2185 ± 0.0018	0.2199 ± 0.0018	0.2139 ± 0.0013	0.2208 ± 0.0029
Average Distance of PMP↓	0.3467 ± 0.0103	**0.1666 ± 0.0020**	0.1722 ± 0.0023	0.4093 ± 0.0003	0.1717 ± 0.0015
Average Distance of CMP↓	0.3430 ± 0.0172	0.1678 ± 0.0017	0.1699 ± 0.0014	**0.1512 ± 0.0013**	0.1673 ± 0.0013

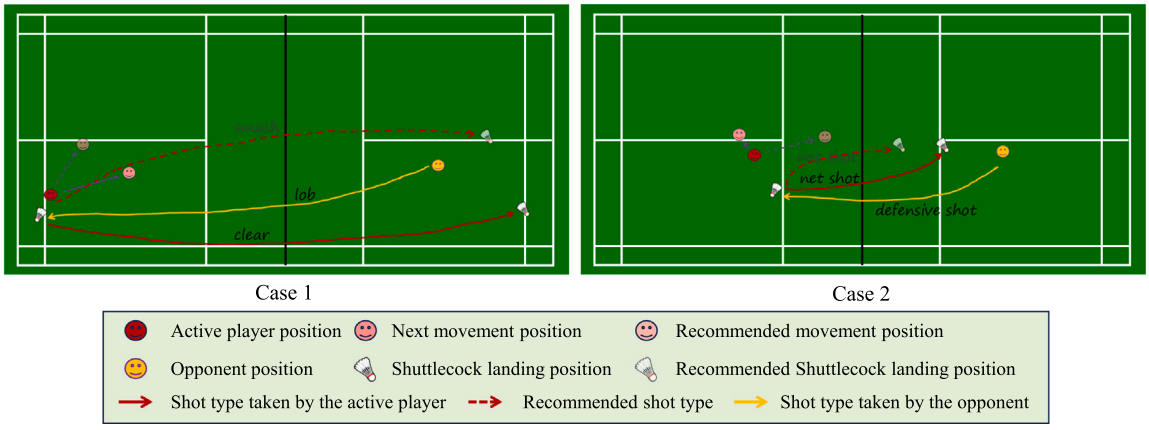


Fig. 5. Two illustrative examples of the tactics generated by CQL with Hybrid Action Space.

result indicates that the policy generated by this model may not strictly adhere to the no-out-of-bounds constraint, it suggests that the policy could further enhance tactical decision-making if we strictly enforce the constraint to keep hits within bounds.

6.3.4. What types of tactical behaviors improve the performance of the learned policy?

To find out which kinds of tactical elements contribute to the enhanced performance, we defined a series of domain metrics and used these metrics to assess the learned policies. The definitions of the domain metrics are explained in Appendix B.1, and the results are shown in Table 8.

As illustrated in Table 8, CQL with Hybrid Action Space emerges as the optimal model shown in Table 5, achieving the highest value in the Active Shot Type Rate. This finding supports the intuitive understanding that players can seize the initiative by frequently employing active shot types, such as smash and rush. Furthermore, in terms of the Average Distance of OLP, this model also performs best, suggesting that the policy derived by this model effectively mobilizes the opponent, thereby maximizing the opponent's energy expenditure. We believe these two strengths are the key reasons why CQL with Hybrid Action Space outperforms all other models in terms of average reward. This underscores the importance of opponent mobilization and initiative establishment in badminton games. Additionally, CQL with Hybrid Action Space achieves the highest Action Difference Rate and Rally Difference Rate, indicating a greater potential to enhance the performance of the behavior policy through the use of shot types that are different from the behavior policy.

It is crucial to ensure that the tactical policy avoids employing irrational actions, as these may not be viable in real-world scenarios. Although the policy derived from CQL with Hybrid Action Space might occasionally resort to such irrational shot types, the frequency of these occurrences is fortunately low. Similarly, out-of-bounds actions are not desirable in badminton tactical decision-making, as it is viewed as a fundamental strategic principle to be adhered. Nonetheless, CQL with Hybrid Action Space still exhibits a slight probability of generating out-of-bounds actions. Compared to other models that utilize supervised learning,[8] CQL with Hybrid Action Space performs poorly in terms of both the Irrational Shot Type Rate and the Out-of-bounds Action Rate. This highlights the fact that supervised learning can more readily develop policies that comply with strict constraints, while learning through Bellman Bootstrapping poses greater challenges.

Additionally, the policy derived from CQL with Hybrid Action Space leads to players covering greater average movement distances compared to most other policies. Furthermore, the recommended movement positions are significantly farther from the center of the court. These behaviors could potentially increase players' physical exertion during real-world badminton matches. However, these drawbacks may not be identified in an offline policy evaluation setting, as the offline policy evaluation typically does not account for the physical execution involved in playing badminton. As a result, CQL with Hybrid Action Space can still achieve higher average rewards.

6.3.5. Case study

To illustrate the nature of the generated tactics, we present two action rounds from the test set as case studies. Fig. 5 compares the tactics generated by our CQL with Hybrid Action Space against the original tactics observed in the dataset. In Case 1, the opponent plays a "lob" to the left-bottom corner, and the original tactic of the active player is to return a "clear" to the opponent's back court. In contrast, the Offline-RL model recommends a more aggressive "smash" in this situation, a shot that would help the player seize and maintain the initiative. In Case 2, while the Offline-RL model also recommends a "net shot" in response to the opponent's "defensive shot," it specifies a distinct landing position closer to the net. This tactical adjustment is designed to force the opponent to move and enhance the threat level of the net shot. Additionally, the model suggests the player move closer to the net in preparation for a potential return of the net shot.

7. Limitations and future works

Although this work demonstrates the potential of Offline RL as a tactical generator for badminton by adapting CQL for the hybrid action space and utilizing a preference-based reward model, it remains a preliminary exploration of AI-driven badminton tactical decision-making. Its limitations and targeted future research directions are discussed as follows:

- **Myopic Policy Evaluation:** In the absence of online interaction, we employ the estimated average reward rather than the V-value or Q-value for policy evaluation, which is inherently myopic. This approach can only demonstrate that the policy generates a one-step optimal tactical action, but not that it can produce a long-term optimal tactical sequence. Although Section 5.1 discusses the practical value of the myopic optimal policy, our ultimate objective remains to find a policy capable of recommending a tactical sequence that helps athletes win the rally. Therefore, designing effective long-term policy evaluation methods is a key focus for future work. We plan to explore novel OPE algorithms capable of reliably assessing long-term policy performance, or alternatively, to develop a physically simulated badminton environment, similar to Zhang et al. (2023a), for online policy evaluation.
- **Preference Assumption:** We assume that the winner's tactical decisions are superior to the loser's within the same rally, based on the general belief that better tactics lead to winning a point. However, this assumption remains intuitive, and exceptions exist. For instance, a player might be in a dominant position to win the rally but suddenly makes a mistake, resulting in a loss. In such cases, we cannot simply assign rally preference based solely on the rally's outcome.
- **Limited Training Data:** The dataset used in this paper comprises 94 international matches, containing only the tactical decisions of professional badminton athletes. Although our experimental results show that the learned policy can generalize to unseen professional players, its applicability to other scenarios, such as amateur matches or players, remains unvalidated. Additionally, the performance of Offline RL is highly dependent on the diversity of the training data. While our dataset includes both winning and losing rallies, these represent the tactical behaviors of professional athletes, which can be considered expert or near-expert level. Whether incorporating amateur match data would improve policy performance requires further exploration. To address these limitations, we plan to expand the dataset by incorporating a wider variety of matches, particularly from amateur levels, and to retrain the policy using the enriched data to analyze its impact. In addition to offline training, we will also investigate the potential of combining offline learning with online fine-tuning in a badminton tactical simulator.
- **Lack of Exploration in Algorithm Advancement:** The primary contribution of our work lies in developing a pipeline that spans from offline data processing to policy training and evaluation, demonstrating Offline RL's potential as a direct badminton tactical generator to enhance decision-making. Consequently, we have compared CQL with Hybrid Action Space against only two other types of algorithms, rather than all advanced Offline RL methods. Additionally, we have not explored the performance ceiling that this class of algorithms can achieve in this specific domain. To address these limitations, future work will include comprehensive comparisons with state-of-the-art Offline RL algorithms (e.g., the hindsight self-supervision approach (Yu et al., 2023)) and investigate novel algorithmic improvements to further enhance tactical policy performance.
- **Lack of Real-world Validation:** While the learned policy has been quantitatively evaluated on the offline dataset, it lacks validation in real-world scenarios. Such validation requires building a real-time situation processing module to digitize current match states for policy input and a visualization interface to convert tactical recommendations into interpretable graphics for coaches and athletes. Finally, integrating these components with the learned policy for testing. We plan to develop this complete pipeline to generate real-time tactical decisions, and subsequently invite professional badminton coaches and players to evaluate our policy's practical utility through this system.

8. Conclusions

In this study, we investigated the use of Offline RL to enhance tactical decision-making in badminton. A key innovation of our work is the development of a preference-based reward model, which offers an alternative to traditional offline policy evaluation methods. This model emphasizes short-term utility, aligning with tactical preferences to assess the effectiveness of learned tactical policies. Additionally, we addressed the challenges of adapting existing Offline RL algorithms to the hybrid action space inherent in badminton tactics. Our approach

[8] As discussed in Section 4, both DT and BC are trained under the paradigm of supervised learning, whereas CQL follows the reinforcement learning format, updating its policy via Bellman Bootstrapping.

involved using the advanced method: CQL with Hybrid Action Space. This method was tailored to meet the unique demands of badminton. The experimental results demonstrate that all learned policies outperform the behavior policy, with CQL with Hybrid Action Space achieving the highest average rewards. Although we only established the short-term optimality of the policies derived from Offline RL, these results highlight the potential to revolutionize sports strategy by leveraging pre-collected datasets to develop effective tactical policies. This offers promising insights for enhancing athletes' tactical training and recommendations in badminton and beyond.

CRediT authorship contribution statement

Mingjiang Liu: Writing – original draft, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Weizhi Tao:** Writing – review & editing, Investigation, Conceptualization. **Hailong Huang:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the Research Institute for Sports Science and Technology (RISports), The Hong Kong Polytechnic University.

We would like to thank the editors and anonymous reviewers for their valuable comments and suggestions, which have significantly contributed to refining this paper. We also thank Dr. Chao Huang from Adelaide University for her helpful advice in revising the original manuscript.

Appendix A. Algorithm details

A.1. Preference-based reward model

The estimated reward function $\hat{r}_\psi$ is modeled using a multilayer perceptron (MLP) parametrized by ψ. Before the inputs s_t and a_t are fed into the MLP network, we apply a linear embedding layer to process these input features. Specifically, since the input features consist of two types: shot type and position coordinates, we use two separate shared embedding modules to embed the shot type and position coordinates, respectively.

As discussed in Liu and Chen (2022), preference-based RL typically involves three alternating steps: reward learning, policy optimization, and query selection. In the query selection step, ensemble-based sampling is a common and effective strategy to solicit preferences and maximize the information gained. Although our focus is solely on the reward learning step in an offline setting, we maintain the ensemble strategy in this study. Specifically, we fit an ensemble of q reward functions $\{\hat{r}_{\psi_1}, \dots, \hat{r}_{\psi_q}\}$, with each reward function trained on $|D|$ preference pairs sampled from the dataset D with replacement. We then average these functions to obtain the final estimated reward.

We conducted a grid search to identify the optimal hyperparameters. The complete list of hyperparameters explored can be found in Table A.1. The code implementation of the reward model refers to https://github.com/Wenminggong/PbRL_for_PHRI.

A.2. Conservative Q-learning with hybrid action space

We implemented CQL on top of the SAC algorithm. Basically, this algorithm involves two kinds of important scalar factors: α^{cql} for adjusting the conservative regulation term and α^{en} for balancing the trade-off between policy entropy and the reward. Both fixed and self-adjusting scalar factors can be utilized, but we chose the automatic adjustment scheme due to its flexibility. In the context of a hybrid action space, α_d^{en} and α_c^{en} represent the entropy scalar factors for the discrete and continuous action components, respectively. We only need to specify the initial values (i.e., α_0^{cql}, $\alpha_{d_0}^{en}$, and $\alpha_{c_0}^{en}$) and the budget parameters (i.e., b^{cql}, b_d^{en} and b_c^{en}), after which the scalar factors automatically adjust via dual gradient descent (Kumar et al., 2020; Haarnoja et al., 2018).

Particularly, CQL employs $\log \sum_a \exp(Q(s,a))$ in the objective, as shown in Eq. (11), to train the Q-function. According to Kumar et al. (2020), this expression can be computed exactly in discrete action domains. However, in continuous action domains, it requires importance sampling. Following this approach, we sample N action samples from a uniform-at-random $Unif(a)$ and N action samples from the current policy $\pi(a|s)$ at each state to perform importance sampling in continuous action domains. The detailed computation process is outlined below:

$$\begin{aligned}
\log \sum_a \exp(Q(s,a)) &= \log \int_{a^d} \int_{a^c} \exp(Q(s,a)) \\
&= \log \sum_{a^d} \left(\int_{a^c} \exp(Q(s,a)) \right) \\
&= \log \sum_{a^d} \left(\frac{1}{2} \mathbb{E}_{a^c \sim Unif(a^c)} \left[\frac{\exp(Q(s,a))}{Unif(a^c)} \right] \right. \\
&\quad \left. + \frac{1}{2} \mathbb{E}_{a^c \sim \pi(a^c|s,a^d)} \left[\frac{\exp(Q(s,a))}{\pi(a^c|s,a^d)} \right] \right) \\
&\approx \log \sum_{a^d} \left(\frac{1}{2N} \sum_{a_i^c \sim Unif(a^c)}^{N} \left[\frac{\exp(Q(s,a_i))}{Unif(a^c)} \right] \right. \\
&\quad \left. + \frac{1}{2N} \sum_{a_i^c \sim \pi(a^c|s,a^d)}^{N} \left[\frac{\exp(Q(s,a_i))}{\pi(a^c|s,a^d)} \right] \right).
\end{aligned} \tag{A.1}$$

Additionally, the full list of CQL with Hybrid Action Space hyperparameters is provided in Table A.2. The implementation of CQL with Hybrid Action Space refers to https://github.com/corl-team/CORL/tree/main and https://github.com/nisheeth-golakiya/hybrid-sac/tree/main.

A.3. Decision transformer

Unlike conventional RL algorithms such as temporal difference (TD) learning, DT can perform credit assignment directly through self-attention, enhancing its effectiveness in sparse-reward scenarios. To enable transformers to learn meaningful patterns and conditionally generate actions during testing, DT represents the trajectory as follows:

$$\tau = (G_1, s_1, a_1, G_2, s_2, a_2, \dots, G_T, s_T, a_T), \tag{A.2}$$

where $G_t = \sum_{t'=t}^{T} r_{t'}$ denotes the returns-to-go. This trajectory representation allows us to train the model and generate new actions through autoregression. The network architecture of the DT model is illustrated in Fig. A.1. The inputs consist of the entire trajectory, while the outputs are the subsequent actions. In this setting, the policy can be denoted as $\pi(a_t|s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1)$. During training, the objective is to minimize the difference between the action output by the network and the action from the dataset, given the historical returns-to-go, states, actions, and the current returns-to-go and state. Considering the hybrid action space containing discrete stroke types and two kinds

Table A.1
Hyperparameters of reward models.

Hyperparameters	BT_Rally	BT_Rally2	NBT_Rally	NBT_Rally2
Batch size	512	512	512	256
Learning rate	$1e-4$	$1e-4$	$1e-4$	$1e-4$
Max epochs	500	500	500	500
Random seed	{0, 1,024, 2,024}	{0, 1,024, 2,024}	{0, 1,024, 2,024}	{0, 1,024, 2,024}
Embedding dimension for shot type	15	15	15	15
Embedding dimension for position coordinates	10	10	10	10
Hidden layer dimension	512	256	256	256
Number of layers	5	3	3	3
Activate function of output layer	tanh	tanh	tanh	tanh
Reward ensemble size q	5	5	1	5
Preference scalar factor α_{pref}	0.5	0.5	0.5	0.5

Table A.2
Hyperparameters of CQL with Hybrid Action Space.

Hyperparameters	Values
Batch size	256
Discount factor	0.99
Learning rate for the policy	$5e-7$
Learning rate for the Q-function	$1e-6$
Random seed	{0, 1,024, 2,024}
Hidden layer dimensions of the policy network and the Q-network	256
Number of layers of the policy network and the Q-network	3
Activate function	Relu
Soft target Q-network update rate	$5e-3$
Learning rate for the scalar factor α^{cql}	$1e-5$
The initial scalar factor α_0^{cql}	e^{-2}
The budget parameter b^{cql}	5.0
Learning rate for the scalar factor α^{en}	$3e-5$
The initial scalar factor $\alpha_{d_0}^{en}$	0.3
The budget parameter b_d^{en}	$0.1 \times \log(10)$
The initial scalar factor $\alpha_{c_0}^{en}$	0.3
The budget parameter b_c^{en}	-2
Max training step	1e+5
Importance sampling number N	10

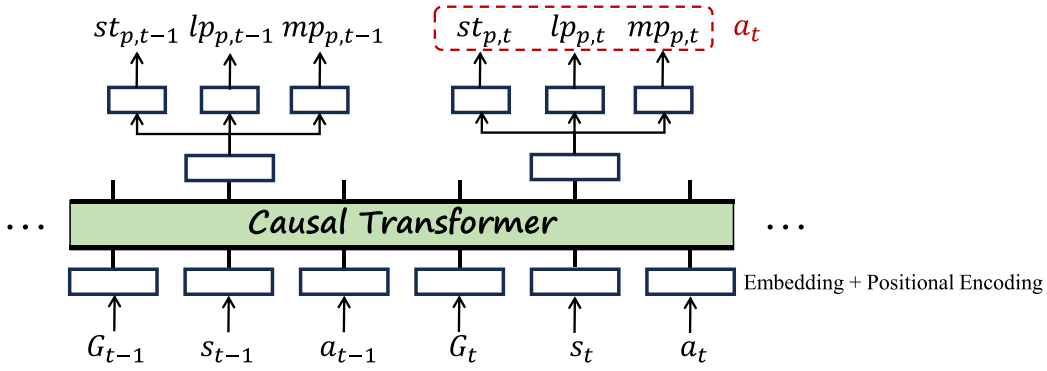


Fig. A.1. The architecture of DT.

of continuous two-dimensional coordinates, we decomposed the policy similarly to Eq. (13):

$$\begin{aligned}&\pi(a_t|s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1) = \\ &\pi(a_t^d|s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1) \\ &\quad \times \pi(a_t^c|s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1, a_t^d).\end{aligned} \tag{A.3}$$

We use a multi-head neural network as the output layer of the model to separately output discrete and continuous actions. The outputs of the discrete actions are the probabilities of 10 stroke types $\pi(a_t^{d_i}|s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1) \in [0, 1]$, where $\sum_i^{10} \pi(a_t^{d_i}|\cdots) = 1$. We use a cross-entropy loss as the objective function of the discrete actions:

$$J(\phi^d) = \mathbb{E}_{s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1 \sim D}[-\sum_{i=1}^{10} y_i \log \pi_{\phi^d}(a_t^{d_i}|\cdots)], \tag{A.4}$$

where $y_i \in \{0, 1\}$ is the label of discrete action. Additionally, two continuous coordinate distributions on two-dimensional planes are used to output the continuous landing and movement positions, with the network trained using log-likelihood.

$$J(\phi^{lp_p}) = \mathbb{E}_{lp_{p,t}, s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1 \sim D}[-\log \pi_{\phi^{lp_p}}(lp_{p,t}|\cdots)]; \tag{A.5}$$

$$J(\phi^{mp_p}) = \mathbb{E}_{mp_{p,t}, s_t, G_t, a_{t-1}, s_{t-1}, G_{t-1}, \dots, a_1, s_1, G_1 \sim D}[-\log \pi_{\phi^{mp_p}}(mp_{p,t}|\cdots)]. \tag{A.6}$$

The final objective is the sum of the above three objectives:

$$J(\phi) = J(\phi^d) + J(\phi^{lp_p}) + J(\phi^{mp_p}). \tag{A.7}$$

The DT model is based on the Transformer architecture (Vaswani et al., 2017), which is highly effective for modeling sequential data. In our work, we utilized the GPT model, specifically GPT-2 (Radford et al., 2018), as the backbone for the DT network. GPT-2 adapts

Table A.3
Hyperparameters of DT.

Hyperparameters	Values
Batch size	128
Learning rate	$1e-6$
Weight decay rate	$1e-3$
Random seed	{0, 1,024, 2,024}
Hidden layer dimension	512
Embedding dimension	64
Number of hidden layers in the Transformer encoder	3
Number of attention heads for each attention layer in the Transformer encoder	2
Activate function	Relu
Dropout	0.1
Warm-up steps	$1e$+4
Max iteration steps	200
Training steps of each interaction	500

the Transformer architecture by incorporating a causal self-attention mask, allowing for auto-regressive generation. Additionally, we performed a grid search to identify the optimal hyperparameters. A comprehensive list of these hyperparameters is provided in Table A.3. The implementation of DT refers to https://github.com/kzl/decision-transformer.

A.4. Behavior cloning

Typically, BC is applied to decision-making problems where the goal is to generate effective actions, whereas supervised learning is generally used for prediction tasks to align outputs with given labels. Specifically, given a training dataset with expected action labels $D = \{(s, a)\}$, the objective of BC is to minimize the following function:

$$J = \mathbb{E}_{(s,a)\sim D}[\frac{1}{2}(a - \hat{a})^2], \tag{A.8}$$

where $\hat{a}$ denotes the action generated by the network (e.g., $\hat{a} = \arg\max_{a'} \pi_\phi(a'|s)$). As discussed in Section 4, when dealing with a hybrid action space, the policy can be decomposed into a discrete component and two continuous components. The objectives for these components are constructed using cross-entropy loss and log-likelihood, respectively. Consequently, the practical objective optimized by BC is as follows:

$$\begin{aligned} J(\phi) = \mathbb{E}_{s_t\sim D}[-\sum_{i=1}^{10} y_i \log \pi_{\phi^d}(a_t^{d_i}|s_t)] + \mathbb{E}_{s_t,lp_{p,t}\sim D}[-\log \pi_{\phi^{lp_p}}(lp_{p,t}|s_t)] \\ + \mathbb{E}_{s_t,mp_{p,t}\sim D}[-\log \pi_{\phi^{mp_p}}(mp_{p,t}|s_t)]. \end{aligned} \tag{A.9}$$

In addition, we introduce a variant of BC in this paper, called Sequence-based BC, which leverages the powerful sequence modeling capabilities of the transformer model. The objective of Sequence-based BC is similar to that of DT, with the exception that it does not include returns-to-go conditioning, as follows:

$$\begin{aligned} J(\phi) =& \mathbb{E}_{s_t,a_{t-1},\ldots,s_1,a_1\sim D}[-\sum_{i=1}^{10} y_i \log \pi_{\phi^d}(a_t^{d_i}|\cdots)] + \\ & \mathbb{E}_{lp_{p,t},s_t,\ldots,s_1,a_1\sim D}[-\log \pi_{\phi^{lp_p}}(lp_{p,t}|\cdots)] \\ & + \mathbb{E}_{mp_{p,t},s_t,\ldots,s_1,a_1\sim D}[-\log \pi_{\phi^{mp_p}}(mp_{p,t}|\cdots)]. \end{aligned} \tag{A.10}$$

The network architectures for both BC and Sequence-based BC are illustrated in Fig. A.2.

The BC model and the Sequence-based BC model are built upon an MLP network and a GPT-2 model, respectively. The hyperparameters for these models are detailed in Tables A.4 and A.5.

Table A.4
Hyperparameters of BC.

Hyperparameters	Values
Batch size	256
Learning rate	$1e-6$
Random seed	{0, 1,024, 2,024}
Hidden layer dimension	256
Number of layers	3
Activate function	Relu
Max training step	$1e$+5

Appendix B. Additional experimental settings and results

B.1. Domain metrics description

All the used domain metrics are defined as follows:

- **Action Difference Rate**: To quantify the difference between the shot types used in the learned policies and those used in the behavior policy, we calculate the proportion of differing shot types. This is expressed as:

$$\text{Action Difference Rate} = \frac{\sum_i^{|D|} \mathbb{I}\{st_i^{\pi'} = st_i^{\pi_\beta}\}}{|D|}, \tag{B.1}$$

  where $|D|$ represents the total number of actions, and π' and π_β denote the learned policy and the behavior policy, respectively. $\mathbb{I}\{st_i^{\pi'} = st_i^{\pi_\beta}\} = 1$ if the shot types used are same (i.e., $st_i^{\pi'} = st_i^{\pi_\beta}$).
- **Rally Difference Rate**: In addition to the Action Difference Rate, we calculate the proportion of rallies that differ. A rally consists of multiple actions, and if there is any difference in shot type within a rally, the entire rally is considered different. This is expressed as:

$$\text{Rally Difference Rate} = \frac{\sum_i^{|D|_r} \mathbb{I}\{\mathcal{R}_i^{\pi'} = \mathcal{R}_i^{\pi_\beta}\}}{|D|_r}, \tag{B.2}$$

  where $|D|_r$ represents the total number of rallies.
- **Irrational Shot Type Rate**: In badminton tactical decision-making, a player needs to choose an appropriate action based on their opponent's action. Only certain shot types are considered reasonable responses for a given shot type executed by the opponent. For instance, if the opponent executes a smash, it is generally not reasonable for the player to return with a smash. To identify irrational shot actions, we measured the frequency of shot type pairs, consisting of the opponent's shot type and the player's response, within the dataset. The results are presented in Fig. B.1. We use a threshold frequency k to mitigate the impact of errors. If the frequency of a shot-type pair exceeds k, it is

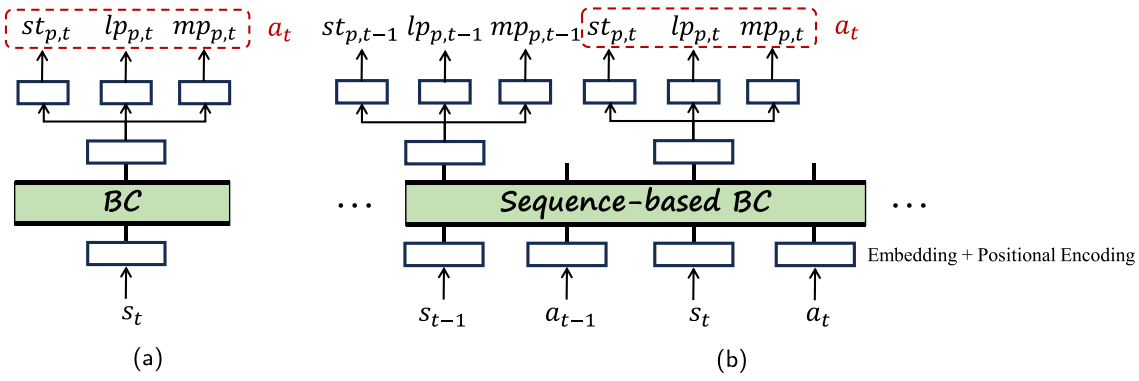


Fig. A.2. (a) The architecture of BC. (b) The architecture of Sequence-based BC.

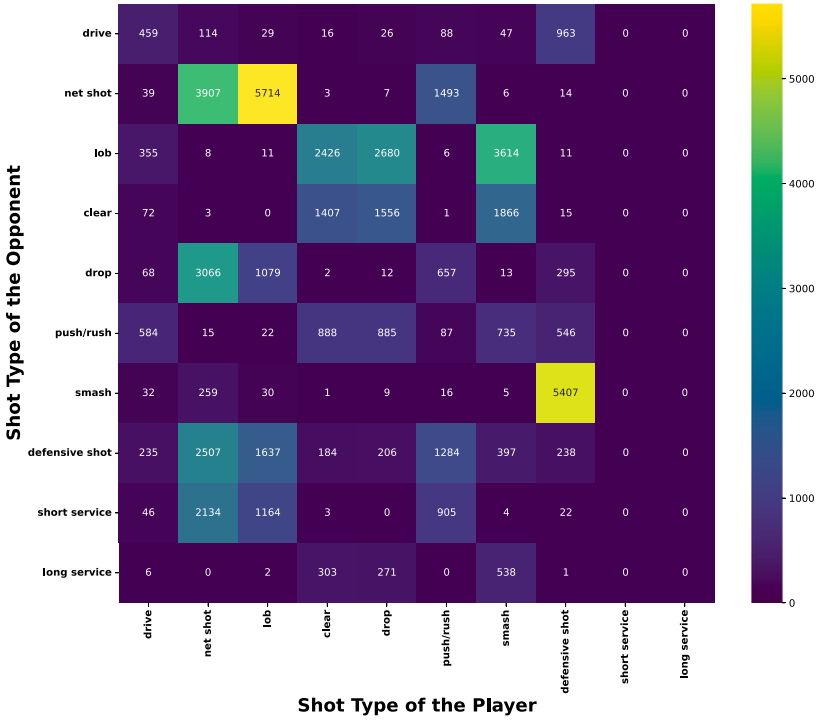


Fig. B.1. The frequency of shot type pairs.

Table A.5
Hyperparameters of Sequence-based BC.

Hyperparameters	Values
Batch size	128
Learning rate	$1e-6$
Weight decay rate	$1e-3$
Random seed	{0, 1,024, 2,024}
Hidden layer dimension	512
Embedding dimension	32
Number of hidden layers in the Transformer encoder	3
Number of attention heads for each attention layer in the Transformer encoder	2
Activate function	Relu
Dropout	0.1
Warm-up steps	$1e$+4
Max iteration steps	200
Training steps of each interaction	500

Table B.1
Average Q-values of the learned policies estimated by FQE.

Policies	Average Q-values[↑]
CQL with Hybrid Action Space	**−0.2206 ± 0.0192**
DT with Winning Goal	−1.0415 ± 0.0442
DT with Original Goal	−1.0467 ± 0.0190
BC	−0.7355 ± 0.0570
Sequence-based BC	−0.8367 ± 0.0094

considered a reasonable situation. The irrational shot type rate is then calculated as follows:

$$\text{Irrational Shot Type Rate} = \frac{\sum_i^{|D|} \mathbb{I}\{st_i^{\pi'} \notin \mathbb{T}_{rea}\}}{|D|}, \tag{B.3}$$

where $\mathbb{T}_{rea}$ denotes the set of reasonable shot types. In this study, we empirically set $k = 10$.

- **Active Shot Type Rate**: Among all the shot types listed in Table 1, we identify push/rush, smash, and drop as active shot types that can help athletes gain the initiative. We calculate the rate at which these active shot types are used by the learned policies using the following formula:

$$\text{Active Shot Type Rate} = \frac{\sum_i^{|D|} \mathbb{I}\{st_i^{\pi'} \in \mathbb{T}_{active}\}}{|D|}, \tag{B.4}$$

where $\mathbb{T}_{active} = \{\text{push/rush}, \text{smash}, \text{drop}\}$ represents the set of active shot types.

- **Out-of-bounds Action Rate**: Avoiding hitting the shuttle out of bounds is a fundamental tactical strategy that must be adhered to. We calculated the percentage of instances where the learned policies resulted in out-of-bounds shots using the following formula:

$$\text{Out-of-bounds Action Rate} = \frac{\sum_i^{|D|} \mathbb{I}\{lp_i^{\pi'} \in \mathbb{L}_o\}}{|D|}, \tag{B.5}$$

where $\mathbb{L}_o$ denotes the set of out-of-bounds actions.

- **Average Distance between Opponent's Position and Landing Position (Average Distance of OLP)**: Commonly, effectively mobilizing the opponent is a key strategy in badminton. This mobilization can be partially measured by the distance between the opponent's position and the shuttle's landing position. Therefore, we calculate the average distance between these two points.
- **Average Distance between Player's Position and Movement Position (Average Distance of PMP)**: As the distance traveled increases, so does the athlete's physical exertion. Intuitively, minimizing the distance traveled during a game helps conserve energy. Therefore, we calculate the average distance between the player's current position and their next movement position.
- **Average Distance between Court Center and Movement Position (Average Distance of CMP)**: Positioning oneself near the center of a badminton half-court aids players in covering the entire court effectively. To evaluate how close the next movement position determined by the learned policy is to the center of the half-court, we calculated the average distance between the court center and the movement position.

B.2. Offline policy evaluation via FQE

We utilized FQE (Le et al., 2019), a widely adopted offline policy evaluation method, to estimate the average Q-value for each learned policy. The results are presented in Table B.1. It is evident that CQL with Hybrid Action Space outperforms the other models, aligning with the findings displayed in Table 5.

Data availability

Data will be made available on request.

References

Bradley, R.A., Terry, M.E., 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. Biometrika 39 (3/4), 324–345.

Chang, K.S., Wang, W.Y., Peng, W.C., 2023. Where will players move next? dynamic graphs and hierarchical fusion for movement forecasting in badminton. In: Proceedings of the 37th AAAI Conference on Artificial Intelligence, vol. 37, (6), AAAI, pp. 6998–7005.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., Mordatch, I., 2021. Decision transformer: Reinforcement learning via sequence modeling. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. pp. 15084–15097.

Christiano, P.F., Leike, J., Brown, T.B., Martic, M., Legg, S., Amodei, D., 2017. Deep reinforcement learning from human preferences. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. pp. 4302–4310.

Chu, W.T., Situmeang, S., 2017. Badminton video analysis based on spatiotemporal and stroke features. In: Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval. ACM, pp. 448–451.

Cossich, V.R., Carlgren, D., Holash, R.J., Katz, L., 2023. Technological breakthroughs in sport: Current practice and future potential of artificial intelligence, virtual reality, augmented reality, and modern data visualization in performance analysis. Appl. Sci. 13 (23), 12965.

Delalleau, O., Peter, M., Alonso, E., Logut, A., 2019. Discrete and continuous action representation for practical RL in video games. arXiv preprint arXiv:1912.11077.

Fernando, T., Denman, S., Sridharan, S., Fookes, C., 2019. Memory augmented deep generative models for forecasting the next shot location in tennis. IEEE Trans. Knowl. Data Eng. 32 (9), 1785–1797.

Fu, J., Norouzi, M., Nachum, O., Tucker, G., Wang, Z., Novikov, A., Yang, M., Zhang, M.R., Chen, Y., Kumar, A., et al., 2021. Benchmarks for deep off-policy evaluation. In: Proceedings of the 9th International Conference on Learning Representations. pp. 1–27.

Fujimoto, S., Meger, D., Precup, D., 2019. Off-policy deep reinforcement learning without exploration. In: Proceedings of the 36th International Conference on Machine Learning. PMLR, pp. 2052–2062.

Haarnoja, T., Zhou, A., Abbeel, P., Levine, S., 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the 35th International Conference on Machine Learning. PMLR, pp. 1861–1870.

Huang, Y.H., Huang, Y.C., Lee, H.S., İk, T.U., Wang, C.C., 2022. S2-Labeling: Shot-by-shot microscopic badminton singles tactical dataset. In: 23rd Asia-Pacific Network Operations and Management Symposium. IEEE, pp. 1–6.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., Amodei, D., 2018. Reward learning from human preferences and demonstrations in atari. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. pp. 8022–8034.

Ibh, M., Graßhof, S., Hansen, D.W., 2024. A Stroke of Genius: Predicting the next move in badminton. In: Proceedings of the 41st Conference on Computer Vision and Pattern Recognition. IEEE/CVF, pp. 3376–3385.

Janner, M., Li, Q., Levine, S., 2021. Offline reinforcement learning as one big sequence modeling problem. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. pp. 1273–1286.

Kim, C., Park, J., Shin, J., Lee, H., Abbeel, P., Lee, K., 2023. Preference transformer: Modeling human preferences using transformers for RL. In: Proceedings of the 11th International Conference on Learning Representations. pp. 1–19.

Kolman, N.S., Kramer, T., Elferink-Gemser, M.T., Huijgen, B.C., Visscher, C., 2019. Technical and tactical skills related to performance levels in tennis: A systematic review. J. Sports Sci. 37 (1), 108–121.

Kostrikov, I., Nachum, O., 2020. Statistical bootstrapping for uncertainty estimation in off-policy evaluation. arXiv preprint arXiv:2007.13609.

Kostrikov, I., Nair, A., Levine, S., 2021. Offline reinforcement learning with implicit Q-learning. In: Proceedings of the 10th International Conference on Learning Representations. pp. 1–11.

Kumar, A., Fu, J., Soh, M., Tucker, G., Levine, S., 2019. Stabilizing off-policy Q-learning via bootstrapping error reduction. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. pp. 11761–11771.

Kumar, A., Zhou, A., Tucker, G., Levine, S., 2020. Conservative Q-learning for offline reinforcement learning. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. pp. 1179–1191.

Lai, Y., Liu, J., Tang, Z., Wang, B., Hao, J., Luo, P., 2023. Chipformer: Transferable chip placement via offline decision transformer. In: Proceedings of the 40th International Conference on Machine Learning. PMLR, pp. 18346–18364.

Le, H., Voloshin, C., Yue, Y., 2019. Batch policy learning under constraints. In: Proceedings of the 36th International Conference on Machine Learning. PMLR, pp. 3703–3712.

Lee, K., Smith, L.M., Abbeel, P., 2021. PEBBLE: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In: Proceedings of the 38th International Conference on Machine Learning. PMLR, pp. 6152–6163.

Levine, S., Kumar, A., Tucker, G., Fu, J., 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643.

Liang, X., Shu, K., Lee, K., Abbeel, P., 2022. Reward uncertainty for exploration in preference-based reinforcement learning. In: Proceedings of the 10th International Conference on Learning Representations. pp. 1–17.

Lin, T., Aouididi, A., Chen, Z., Beyer, J., Pfister, H., Wang, J.H., 2024. VIRD: Immersive match video analysis for high-performance badminton coaching. IEEE Trans. Vis. Comput. Graphics 30 (1), 458–468.

Liu, M., Chen, C., 2022. Task decoupling in preference-based reinforcement learning for personalized human-robot interaction. In: 2022 International Conference on Intelligent Robots and Systems. IEEE/RSJ, pp. 848–855.

Nambiar, M., Ghosh, S., Ong, P., Chan, Y.E., Bee, Y.M., Krishnaswamy, P., 2023. Deep offline reinforcement learning for real-world treatment optimization applications. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, pp. 4673–4684.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al., 2022. Training language models to follow instructions with human feedback. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. pp. 27730–27744.

Palan, M., Landolfi, N.C., Shevchuk, G., Sadigh, D., 2019. Learning reward functions by integrating human demonstrations and preferences. In: Robotics: Science and Systems XV. pp. 1–11.

Pomerleau, D.A., 1988. ALVINN: an autonomous land vehicle in a neural network. In: Proceedings of the 1st International Conference on Neural Information Processing Systems. pp. 305–313.

Qin, R.J., Zhang, X., Gao, S., Chen, X.H., Li, Z., Zhang, W., Yu, Y., 2022. NeoRL: A near real-world benchmark for offline reinforcement learning. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. pp. 24753–24765.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., 2018. Improving language understanding by generative pre-training. OpenAI Blog 1–12.

Shah, D., Bhorkar, A., Leen, H., Kostrikov, I., Rhinehart, N., Levine, S., 2023. Offline reinforcement learning for visual navigation. In: 2023 Conference on Robot Learning. PMLR, pp. 44–54.

Srilakshmi, G., Joe, I.P., 2023. Sports video retrieval and classification using focus u-net based squeeze excitation and residual mapping deep learning model. Eng. Appl. Artif. Intell. 126, 107176.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.

Swazinna, P., Udluft, S., Runkler, T., 2021. Overcoming model bias for robust offline deep reinforcement learning. Eng. Appl. Artif. Intell. 104, 104366.

Thomas, P., Brunskill, E., 2016. Data-efficient off-policy policy evaluation for reinforcement learning. In: Proceedings of the 33rd International Conference on Machine Learning. PMLR, pp. 2139–2148.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 6000–6010.

Wang, W.Y., Du, W.W., Peng, W.C., 2023a. Shuttleset22: Benchmarking stroke forecasting with stroke-level badminton dataset. arXiv preprint arXiv:2306.15664.

Wang, W.Y., Huang, Y.C., Ik, T.U., Peng, W.C., 2023b. Shuttleset: A human-annotated stroke-level singles dataset for badminton tactical analysis. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. ACM, pp. 5126–5136.

Wang, W.Y., Shuai, H.H., Chang, K.S., Peng, W.C., 2022. Shuttlenet: Position-aware fusion of rally progress and player styles for stroke forecasting in badminton. In: Proceedings of the 36th AAAI Conference on Artificial Intelligence, vol. 36, (4), AAAI, pp. 4219–4227.

Wang, Z., Veličković, P., Hennes, D., Tomašev, N., Prince, L., Kaisers, M., Bachrach, Y., Elie, R., Wenliang, L.K., Piccinini, F., et al., 2024b. TacticAI: an AI assistant for football tactics. Nat. Commun. 15 (1), 1906.

Wang, K.D., Wang, W.Y., Hsieh, P.C., Peng, W.C., 2024a. Offline imitation of badminton player behavior via experiential contexts and Brownian motion. In: 2024 Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, pp. 348–364.

Wen, J., Dai, B., Li, L., Schuurmans, D., 2020. Batch stationary distribution estimation. In: Proceedings of the 37th International Conference on Machine Learning. PMLR, pp. 10203–10213.

Yang, W., Jiang, M., Fang, X., Shi, X., Guo, Y., Al-qaness, M.A., 2024. A high-precision and efficient method for badminton action detection in sports using you only look once with hourglass network. Eng. Appl. Artif. Intell. 137, 109177.

Yang, M., Nachum, O., Dai, B., Li, L., Schuurmans, D., 2020. Off-policy evaluation via the regularized lagrangian. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. pp. 6551–6561.

Yu, X., Bai, C., Wang, C., Yu, D., Chen, C.P., Wang, Z., 2023. Self-supervised imitation for offline reinforcement learning with hindsight relabeling. IEEE Trans. Syst. Man, Cybern.: Syst. 53 (12), 7732–7743.

Zhang, Z., Mei, H., Xu, Y., 2023b. Continuous-time decision transformer for healthcare applications. In: The 26th International Conference on Artificial Intelligence and Statistics. PMLR, pp. 6245–6262.

Zhang, M.R., Paine, T.L., Nachum, O., Paduraru, C., Tucker, G., Wang, Z., Norouzi, M., 2021. Autoregressive dynamics models for offline policy evaluation and optimization. In: Proceedings of the 9th International Conference on Learning Representations. pp. 1–17.

Zhang, B., Qiu, X., Tan, X., 2024. Balancing therapeutic effect and safety in ventilator parameter recommendation: An offline reinforcement learning approach. Eng. Appl. Artif. Intell. 131, 107784.

Zhang, H., Yuan, Y., Makoviychuk, V., Guo, Y., Fidler, S., Peng, X.B., Fatahalian, K., 2023a. Learning physically simulated tennis skills from broadcast videos. ACM Trans. Graph. 42 (4), 1–14.