PlayBest: Professional Basketball Player Behavior Synthesis via Planning with Diffusion

Xiusi Chen*
University of California, Los Angeles
Los Angeles, CA, USA
xchen@cs.ucla.edu

Wei-Yao Wang*
National Yang Ming Chiao Tung University
Hsinchu, Taiwan
sf1638.cs05@nctu.edu.tw

Ziniu Hu
California Institute of Technology
Pasadena, CA, USA
acgbull@gmail.com

David Reynoso
University of California, Los Angeles
Los Angeles, CA, USA
dmreynos@g.ucla.edu

Kun Jin
University of Michigan, Ann Arbor
Ann Arbor, MI, USA
kunj@umich.edu

Mingyan Liu
University of Michigan, Ann Arbor
Ann Arbor, MI, USA
mingyan@umich.edu

P. Jeffrey Brantingham
University of California, Los Angeles
Los Angeles, CA, USA
branting@g.ucla.edu

Wei Wang
University of California, Los Angeles
Los Angeles, CA, USA
weiwang@cs.ucla.edu

Abstract

Dynamically planning in complex systems has been explored to improve decision-making in various domains. Professional basketball serves as a compelling example of a dynamic spatio-temporal game, encompassing context-dependent decision-making. However, processing the diverse on-court signals and navigating the vast space of potential actions and outcomes make it difficult for existing approaches to swiftly identify optimal strategies in response to evolving circumstances. In this study, we formulate the sequential decision-making process as a conditional trajectory generation process. Based on the formulation, we introduce PlayBest (PLAYer BEhavior SynThesis), a method to improve player decision-making. We extend the diffusion probabilistic model to learn challenging environmental dynamics from historical National Basketball Association (NBA) player motion tracking data. To incorporate data-driven strategies, an auxiliary value function is trained with corresponding rewards. To accomplish reward-guided trajectory generation, we condition the diffusion model on the value function via classifier-guided sampling. We validate the effectiveness of PlayBest through simulation studies, contrasting the generated trajectories with those employed by professional basketball teams. Our results reveal that the model excels at generating reasonable basketball trajectories that produce efficient plays. Moreover, the synthesized play strategies exhibit an alignment with professional tactics, highlighting the model's capacity to capture the intricate dynamics of basketball games.[1]

CCS Concepts

• **Computing methodologies → Planning for deterministic actions**; **Machine learning**.

Keywords

Planning, Diffusion model, Conditional sampling, Sports analytics

ACM Reference Format:
Xiusi Chen*, Wei-Yao Wang*, Ziniu Hu, David Reynoso, Kun Jin, Mingyan Liu, P. Jeffrey Brantingham, and Wei Wang. 2024. PlayBest: Professional Basketball Player Behavior Synthesis via Planning with Diffusion. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3627673.3680092

1 Introduction

The exploration of dynamic systems and their planning has broad applicability across various domains. Whether it involves developing strategies for team sports [44], managing traffic flow [51], coordinating autonomous vehicles [24], or understanding the dynamics of financial markets [29], these scenarios can be effectively framed as dynamic systems characterized by intricate interactions and decision-making processes. The ability to comprehend and plan within these systems becomes crucial to achieving optimal outcomes. Basketball, with its high level of dynamism and complexity as a team sport, serves as a captivating illustration of a real-time dynamic system with intricate tactical elements. A basketball game requires continuous adaptation and strategic decision-making. Coaches and players rely on pertinent environmental and behavioral cues including teammates' and opponents' current positions and trajectories to select play strategies that respond effectively to opponents' actions and adapt to real-time situational changes. Existing methods in sports analytics and trajectory optimization [41, 44, 48] have made progress in modeling and predicting player movements and game outcomes. However, these approaches struggle to capture the intricate dynamics of basketball games and produce flexible, adaptive play strategies that can handle the uncertainties and complexities inherent in the sport. The challenges arise from the following two features of basketball games:

[1] The code is at https://github.com/xiusic/diffuser_bball.

*Equal contribution.

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3680092

Modeling the complex environmental dynamics: Capturing environmental dynamics in basketball games is a very challenging task due to the inherent complexity of the game, for example, rapid changes in game situations and numerous possible actions at any given moment. The spatio-temporal nature of basketball data, including multiple player positions and ball trajectories, further complicates the modeling process. The need for a computationally efficient and scalable approach to handle the massive amounts of data generated during basketball games presents a major challenge in modeling environmental dynamics.

Reward Sparsity: An additional challenge lies in addressing reward sparsity. Unlike other reinforcement learning (RL) environments where immediate feedback is readily available after each action, basketball games often see long sequences of actions leading up to a single reward event (e.g., the scoring of a basket). This results in a sparse reward landscape, as many actions contribute indirectly to the final outcome but are not themselves immediately rewarded. This scenario complicates the learning process as it becomes more challenging for the planning algorithm to accurately attribute the impact of individual actions to the final reward. Designing effective methods to address the reward sparsity challenge remains a significant hurdle in applying typical planning algorithms to basketball and similar sports games.

Recently, powerful trajectory optimizers that leverage learned models often produce plans that resemble adversarial examples rather than optimal trajectories [23, 40]. On the contrary, modern model-based RL algorithms tend to draw more from model-free approaches, such as value functions and policy gradients [46], rather than utilizing the trajectory optimization toolbox. Methods that depend on online planning typically employ straightforward gradient-free trajectory optimization techniques like random shooting [33] or the cross-entropy method [7, 10] to circumvent the above problems.

In this work, we first formulate the planning problem in basketball as a multi-player behavior synthesis task, and instantiate the behavior synthesis task as a trajectory generation task. Following the recent success of generative models in applications of single-agent planning [4, 17], we propose a novel application of the diffusion model called PlayBest (PLAYer BEhavior SynThesis), to generate optimal basketball trajectories and synthesize adaptive play strategies. Under most circumstances, the diffusion model serves as a generative model to capture the distribution of the input samples. In our study, we extend it as a powerful technique to enable flexible behavior synthesis in dynamic and uncertain environments since there is no existing online environment for basketball simulations. The diffusion process explores different potential trajectories and adapts to changes in the environment through the iterative sampling process to model basketball court dynamics. To guide the reverse diffusion process with rewards, PlayBest features a value guidance module that guides the diffusion model to generate optimal play trajectories by conditional sampling. This integration naturally forms a conditional generative process, and it allows PlayBest to swiftly adapt to evolving conditions and pinpoint optimal solutions in real-time.

We instantiate PlayBest in a variety of simulation studies and real-world scenarios, demonstrating the effectiveness of PlayBest in generating high-quality basketball trajectories that yield effective plays. Extensive results reveal that our proposed approach outperforms conventional planning methods in terms of adaptability, flexibility, and overall performance, showing a remarkable alignment with professional basketball tactics.

The core contributions of this work are summarized as follows:

- We attempt to formulate the basketball player behavior synthesis problem as a guided sampling/conditional generation of multiple players and ball trajectories from diffusion models.
- We present PlayBest, a framework featuring a diffusion probabilistic model with a value function, to instantiate the conditional generative model. We adapt the model to integrate multi-player behaviors and decisions in basketball and show that a number of desirable properties are obtained.
- We showcase the effectiveness of PlayBest via both quantitative and qualitative studies of the trajectories generated and validate the practicality of adopting PlayBest to investigate real basketball games.

2 Preliminary

2.1 Diffusion Probabilistic Models

Diffusion probabilistic models [16, 38] stand out as a unique approach to learning complex data distributions, symbolized by $q(\boldsymbol{\tau})$, based on a collection of samples, denoted as $\mathcal{D} \coloneqq \{\boldsymbol{x}\}$.

On a high level, two processes are at the core of their operation: a predefined forward noising mechanism $q(\boldsymbol{\tau}^{i+1}|\boldsymbol{\tau}^i) \coloneqq \mathcal{N}(\boldsymbol{\tau}^{i+1}; \sqrt{\alpha_i}\boldsymbol{\tau}^i, (1-\alpha_i)\boldsymbol{I})$ and a trainable reverse or "denoising" process $p_\theta(\boldsymbol{\tau}^{i-1}|\boldsymbol{\tau}^i) \coloneqq \mathcal{N}(\boldsymbol{\tau}^{i-1}|\mu_\theta(\boldsymbol{\tau}^i, i), \Sigma_i)$. Here the Gaussian distribution is represented as $\mathcal{N}(\mu, \Sigma)$, and the variable α_i is pivital in determining the variance schedule. We begin with a sample $\boldsymbol{x}_0 \coloneqq \boldsymbol{x}$, followed by latents $\boldsymbol{\tau}^1, \boldsymbol{\tau}^2, ..., \boldsymbol{\tau}^{N-1}$, and culminate with $\boldsymbol{\tau}^N \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, factoring in specific values for α_i and an adequately extended N.

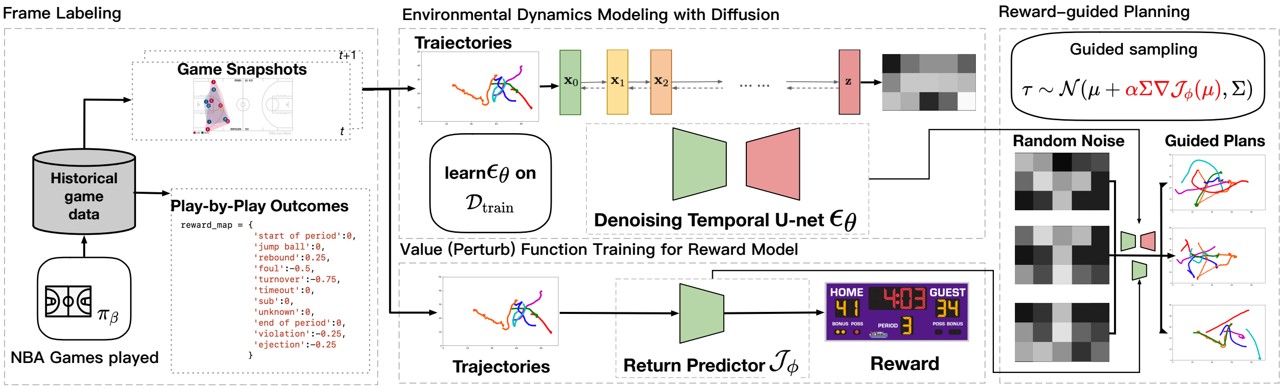
2.2 Trajectory Optimization Problem Setting in Basketball Strategy

In basketball, we can consider the game as a discrete-time system with dynamics $\mathbf{s}_{t+1} = \boldsymbol{f}(\mathbf{s}_t, \mathbf{a}_t)$, where $\mathbf{s}_t$ represents the state of the play, and $\mathbf{a}_t$ denotes the action or basketball maneuver. Trajectory optimization aims to find a sequence of actions $\mathbf{a}^*_{0:T}$ that maximizes an objective $\mathcal{J}$, such as maximizing the score. This can be represented as:

$$\mathbf{a}^*_{0:T} = \arg\max_{\mathbf{a}_{0:T}} \mathcal{J}(\mathbf{s}_0, \mathbf{a}_{0:T}) = \arg\max_{\mathbf{a}_{0:T}} \sum_{t=0}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \tag{1}$$

where T defines the planning horizon. $\boldsymbol{\tau} = (\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)$ is the trajectory of states and actions, and $\mathcal{J}$ becomes the objective value of the play.

This model, when applied to basketball, facilitates the creation of dynamic strategies that adapt to real-time game scenarios. By simulating noise-corrupted play sequences and iteratively denoising them, one can derive actionable insights into players' behaviors, leading to more effective in-game decision-making and planning.


Frame Labeling
Game Snapshots
t+1
t
Historical game data
π_β
NBA Games played
Play-by-Play Outcomes
Environmental Dynamics Modeling with Diffusion
Trajectories
$\mathbf{x}_0$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{z}$
learn ϵ_θ on $\mathcal{D}_{\text{train}}$
Denoising Temporal U-net ϵ_θ
Value (Perturb) Function Training for Reward Model
Trajectories
Return Predictor $\mathcal{J}_\phi$
HOME 41
4:03
GUEST 34
PERIOD 3
Reward
Reward–guided Planning
Guided sampling
$\tau \sim \mathcal{N}(\mu + \alpha\Sigma\nabla\mathcal{J}_\phi(\mu), \Sigma)$
Random Noise
Guided Plans

```
reward_map = {
                'start of period':0,
                'jump ball':0,
                'rebound':0.25,
                'foul':-0.5,
                'turnover':-0.75,
                'timeout':0,
                'sub':0,
                'unknown':0,
                'end of period':0,
                'violation':-0.25,
                'ejection':-0.25
            }
```

Figure 1: Overview framework of PlayBest. The overall pipeline can be split into four major components: Frame Labeling, Environmental Dynamics Learning, Value (Perturb) Function Training, and Trajectory Generation Guided by a Reward Function. The diffusion probabilistic model ϵ_θ is trained to model the environmental dynamics. The reward predictor $\mathcal{J}_\phi$ is trained on the same trajectories as the diffusion model. During guided trajectory generation, our model takes both environmental dynamics and rewards as input, performs guided planning via conditional sampling, and generates the trajectories as the guided plan.

2.3 Problem Description

The input for PlayBest consists of a set of basketball game records, denoted as $\mathcal{D}_{raw}$. These game records are composed of distinct elements, described as follows:

Motion Track Data. The motion track data, represented as $\mathcal{D}^{move}$, comprises static snapshots of in-game events, detailing the positions of all players and the ball at a rate of 25 frames per second. A game's progression can be reconstructed and visualized using these snapshots.

Play-by-Play Data. Denoted as $\mathcal{D}^{pbp}$, the play-by-play data offers a game transcript in the form of possessions. This data includes 1) the possession timestamp, 2) the player initiating the possession, 3) the result of the possession (e.g., points scored), and 4) additional unique identifiers employed for possession categorization.

To facilitate learning, we divide $\mathcal{D}_{raw}$ into $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$, representing the training and testing sets, based on gameplay timestamps. We formally define our task as follows:

Given a set of game records $\mathcal{D}_{train} = \mathcal{D}^{move}_{train} \cup \mathcal{D}^{pbp}_{train}$ and a reward function $\mathcal{J}_\phi$, with $\mathcal{J}_\phi$ depending on the reward definition given by the discriminative rules applied to $\mathcal{D}^{pbp}_{train}$, the objective is to generate trajectories $\{\boldsymbol{\tau}\}$ leaning towards the higher-reward regions of the state-action space. In essence, our goal is to develop a policy $\pi_{\theta,\phi}(\mathbf{a} \mid \mathbf{s})$, parameterized by θ and ϕ, that determines the optimal action based on the states associated with each frame in $\mathcal{D}^{move}_{test}$.

3 The PlayBest Framework

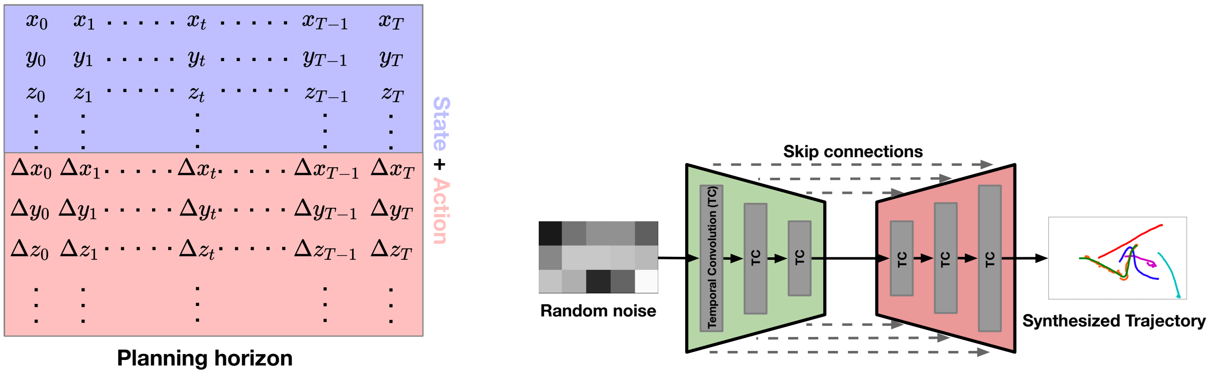
In this section, we describe in detail how our framework is designed. We first give an overview and then present details of the model architecture including the diffusion and value function modules.

3.1 Framework Overview

Figure 1 depicts the PlayBest pipeline. The historical game replay data originates from actual games played during the 2015-2016 NBA regular season. Each team competes per their unknown policies π_β. The raw game data encompasses multiple modalities, and a game is characterized by a series of high-frequency snapshots (e.g., 25 frames per second). At any given time t, a snapshot includes an image displaying all player and ball positions, as well as additional metadata like the results of each possession (shot made/miss, free-throw made/miss, rebound, foul, turnover, etc), shot clock, and game clock at time t.

Out of the historical game replay data, we construct the player trajectories and ball trajectories to create the trajectory dataset $\mathcal{D}^{move}$. We then use the trajectory dataset $\mathcal{D}^{move}_{train}$ to train a diffusion model ϵ_θ that aims at modeling the distribution of the 3-dimensional player and ball movements. The training process of the diffusion model mimics the training procedure of what is usually referred to as offline RL, where there is no online environment to interact with. However, the diffusion model by itself can only generate "like-real" trajectories that do not necessarily lead to a goal-specific outcome. To further generate trajectories that can represent "good plans", we train a value function that maps any possible trajectory to its expected return. During the sampling stage, the mean of the diffusion model is perturbed by the gradient of the value function. In this way, the guided sampling is capable of generating the trajectories biased towards the high-reward region. Incorporating a diffusion model in planning problems not only enhances efficient exploration and resilience in volatile environments, but also addresses the challenge of long-horizon planning, enabling the generation of strategic, noise-reduced trajectories over extended periods.

In essence, our framework utilizes a dataset $\mathcal{D}$ collected by an unknown behavior policy π_β, which can be approximated as the "average" policy for all NBA teams. This dataset is gathered once and remains unaltered during training. The training process relies entirely on the training set $\mathcal{D}_{\text{train}}$ and does not interact with the environment. Upon completion of training, we anticipate that π_θ will exhibit strong generalization on $\mathcal{D}_{\text{test}}$.


x_0 x_1 x_t x_{T-1} x_T
y_0 y_1 y_t y_{T-1} y_T
z_0 z_1 z_t z_{T-1} z_T
Δx_0 Δx_1 Δx_t Δx_{T-1} Δx_T
Δy_0 Δy_1 Δy_t Δy_{T-1} Δy_T
Δz_0 Δz_1 Δz_t Δz_{T-1} Δz_T
State + Action
Planning horizon
Skip connections
Random noise
Temporal Convolution (TC)
TC
TC
TC
TC
TC
Synthesized Trajectory

(a) The shape of the training data. Trajectories are represented by the (x, y, z) coordinates of the ten on-court players across two teams and the ball (11 channels). The action is made up of the momentum of each object at the same timestep.

(b) The general structure of the diffusion model ϵ_θ is implemented by a U-net with temporal convolutional blocks, which have been widely utilized in image-centric diffusion models.

Figure 2: (a, b) The input and diffusion architecture.

3.2 Environmental Dynamics Modeling with Diffusion

Since there is no public basketball environment that is able to provide online simulation, previous studies focus on offline simulations [9]. However, these approaches fall short in providing trajectories with planning strategies and efficiency due to the autoregressive designs, which are also challenging to be extended to incorporate dynamic planning. Therefore, we adopt diffusion models not only to simulate trajectories simultaneously from modeling environmental dynamics but also to be guided by the specific outcomes with conditional sampling.

Model Input and Output. To represent our input that can be consumed by the diffusion model, we represent all the trajectories in the format of a 2-dimensional image as described in Figure 2a. To be specific, we concatenate the state features and action features at each timestep in the game together to form one column of the model input. The features from different timesteps are then stacked together following the temporal order to form the rows. In other words, the rows in the model input correspond to the *planning horizon T* in Section 2.2.

Architecture. As illustrated in Figure 2b, the backbone of the environmental dynamics modeling module is a diffusion probabilistic model ϵ_θ. Diffusion models have been found effective in fitting the distribution of images [16]. Our assumption is that the diffusion models can also learn the underlying distribution of basketball player trajectories by framing as the trajectory optimization problem, thereby modeling the player and ball dynamics. Following image-based diffusion models, we adopt the U-net architecture [35] as the overall architecture. Moreover, to account for the temporal dependencies between different timesteps of the trajectories, we replace two-dimensional spatial convolutions with one-dimensional temporal convolutions.

Diffusion Training. We follow the usual way by parameterizing the Gaussian noise term to make it predict ϵ_t from the input x_t at diffusion step t to learn the parameters θ,:

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\epsilon_t,\tau^0}\left[\|\epsilon_t - \epsilon_\theta(\boldsymbol{\tau}^t, t)\|^2\right], \tag{2}$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ denotes the noise target, t represents the diffusion step, and $\boldsymbol{\tau}^t$ is the trajectory $\boldsymbol{\tau}^0$ corrupted by noise ϵ at diffusion step t.

3.3 Value Function Training for Reward Model

At the heart of the value function is an encoder that takes the trajectory data as input and returns the estimated cumulative reward. The structure of the return predictor $\mathcal{J}_\phi$ takes exactly the first half of the U-Net employed in the diffusion model, and it is followed by a linear layer that generates a single scalar output indicating the reward value.

3.4 Guided Planning as Conditional Sampling

Existing studies [4, 17] have revealed the connections between classifier-guided / classifier-free sampling [12] and reinforcement learning. The sampling routine of PlayBest resembles the classifier-guided sampling. In detail, we condition a diffusion model $p_\theta(\boldsymbol{\tau})$ on the states and actions encompassed within the entirety of the trajectory data. Following this, we develop an isolated model, $\mathcal{J}_\phi$, with the aim of forecasting the aggregated rewards of trajectory instances $\boldsymbol{\tau}^i$. The trajectory sampling operation is directed by the gradients of $\mathcal{J}_\phi$, which adjust the means μ of the reverse process as per the following equations:

$$\begin{aligned} &\mu \leftarrow \mu_\theta\left(\boldsymbol{\tau}^i\right), \\ &\boldsymbol{\tau}^{i-1} \sim \mathcal{N}\left(\mu + \alpha\Sigma\nabla\mathcal{J}_\phi(\mu), \Sigma^i\right), \\ &\boldsymbol{\tau}^{i-1}_{\mathbf{s}0} \leftarrow \mathbf{s}, \end{aligned} \tag{3}$$

where α is the scaling factor to measure the impact of the guidance on the sampling, and

$$\nabla\mathcal{J}(\mu) = \sum_{t=0}^{T} \nabla_{\mathbf{s}_t,\mathbf{a}_t} r\left(\mathbf{s}_t, \mathbf{a}_t\right)\Big|_{(\mathbf{s}_t,\mathbf{a}_t)=\mu_t}. \tag{4}$$

Table 1: NBA 2015 - 16 Regular Season Game Stats. Games are split chronically so that all the games in the test set are after any game in the training set.

# Training Games	# Minutes	# Plays	# Frames
480	23, 040	210, 952	34, 560, 000
# Testing Games	**# Minutes**	**# Plays**	**# Frames**
151	7, 248	68, 701	10, 872, 000
# Games	**# Minutes**	**# Plays**	**# Frames**
631	30, 288	279, 653	45, 432, 000

Table 2: Definition of Reward per possession.

Event type	Reward
"start of period"	0
"jump ball"	0
"rebound"	0.25
"foul"	-0.25
"turnover"	-1
"timeout"	0
"substitution"	0
"end of period"	0
"violation"	-0.25
"3 pointer made"	3
"2 pointer made"	2
"free-throw made"	1

where r is the reward function given by the environment. In our case, it comes from the outcome of the possessions derived from $\mathcal{D}^{pbp}$. The detailed algorithm of reward-guided planning is illustrated in Algorithm 1.

Algorithm 1 Reward Guided Planning

```
Require diffusion model $\mu_\theta$, guide $\mathcal{J}_\phi$, scale $\alpha$, covariances $\Sigma^i$
while not done do
    Acquire state $\mathbf{s}$; initialize trajectory $\boldsymbol{\tau}^N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
    //$N$ diffusion steps in total
    for $i = N, \dots, 1$ do
        $\mu \leftarrow \mu_\theta(\boldsymbol{\tau}^i)$
        $\boldsymbol{\tau}^{i-1} \sim \mathcal{N}(\mu + \alpha\Sigma\nabla\mathcal{J}(\mu), \Sigma^i)$
        //conditioned on the initial player positions
        $\boldsymbol{\tau}^{i-1}_{s_0} \leftarrow s$
    end for
    Execute first action of trajectory $\boldsymbol{\tau}^0_{\mathrm{a}_0}$
end while
```

4 Experiments

4.1 Experimental Setup

To quantitatively evaluate the effectiveness of player behavior planning, we focus on measuring the cumulative return given by the learned policy, which serves as an objective evaluation metric to compare the performance of PlayBest with other comparative methods. Evaluating offline RL is inherently difficult as it lacks real-time environment interaction for reward accumulation. Thus the model verification is primarily reliant on utilizing existing replay data. To validate the capacity of our framework in learning efficient tactics, we assess PlayBest's ability to generate efficient plans using diverse data of varying standards.

Dataset. We obtained our data from an open-source repository [1, 2]. The model's input data is a combination of two components: (1) **Player Movement Sensor Data**: This component captures real-time court events, detailing the positions of the players and the ball in Cartesian coordinates. The sampling frequency of this data is 25 frames per second. The statistics are detailed in Table 1. (2) **Play-by-Play**: This segment of information contains the specifics of each possession, such as the termination of the possession (whether through a jump shot, layup, foul, and so forth), the points gained by the offensive team, the location from which the ball was shot, and the player who made the shot, among other details. The data for training and testing is split chronologically: the training set includes games from 2015, amounting to 480 games, while the remaining games from 2016 form the testing set, amounting to 151 games. The statistics are described in Table 1.

Reward Definition. As there is no fine-grained reward design in basketball in previous work, e.g., [9, 49], we define the reward of each possession based on its outcomes, as listed in Table 2. For a certain team that plays the possession, we encourage the possession trajectory if it leads to positive outcomes (e.g., score, rebound) and we punish otherwise (turnover, foul, violation). Note that the same event by the opponent team takes the negative value of the rewards. For example, a 2-point basket made by the team on offense leads to a -2 reward to the training sample of the value function for the team on defense. During our offline evaluation, we employ our value function $\mathcal{J}_\phi$ to gauge the expected return of our policy. By summing all expected rewards from each possession for a team, we can approximate the total points for the team following the learned strategic policies. For each game in the test set, all comparative methods plan trajectories from each possession's actual initial state.

Baselines. As this task has yet to be explored, there are no widely adopted baselines for direct comparison. Therefore, we examine our model with several state-of-the-art offline RL algorithms and a naive baseline to verify its effectiveness:

- Batch-Constrained deep Q-learning (**BCQ**) [14] is an off-policy algorithm for offline RL. It mitigates overestimation bias by constraining the policy to actions similar to the behavior policy, ensuring a more conservative policy.
- Conservative Q-Learning (**CQL**) [26] is an offline RL approach that minimizes an upper bound of the expected policy value to conservatively estimate the action-value function, leading to a more reliable policy.
- Independent Q-Learning (**IQL**) [25] is a multi-agent reinforcement learning approach where each agent learns its own Q-function independently. It offers an efficient solution for multi-agent environments.
- **Random Walk** is the "naive" baseline that can be used to validate the correctness of the value function and to offer an auxiliary

Table 3: Overall performance in return values per possession.

Methods	Random Walk	Ground Truth	BCQ	CQL	IQL	PlayBest
AVG	-9.1172±0.035	0.0448±0.000	0.0964±0.000	0.0986±0.001	0.0992±0.000	**0.4473±1.235**
MAX	-9.0753	0.0448	0.0967	0.0995	0.0992	**2.2707**

Table 4: The effects of the scaling factor α. We repeat our sampling process 5 times and report the mean and variance for the average returns per possession.

α	**0**	**0.01**	**0.1**	**1**	**10**
AVG	0.0859±0.0052	0.0894±1.2263	0.4473±1.2349	3.0870±1.4955	10.8090±2.4050
MAX	0.0932	1.8844	2.2707	5.3534	14.2389

comparative method that corresponds to the case where all the players navigate randomly within the range of the court.

4.2 Implementation Details

We set the planning horizon length to 1,024 so that all trajectories in the training data can be fitted in our diffusion model. The diffusion step is set to 20 in all experiments. The learning rate is 2×10^{-5} without learning rate scheduling. The hidden dimension is set following [17]. The training batch size is set to 512. We train all models for 245K training steps. The value function is optimized with the mean square error loss. All experiments are run on the NVIDIA Tesla V100 Tensor Core GPUs with 16GB memory.

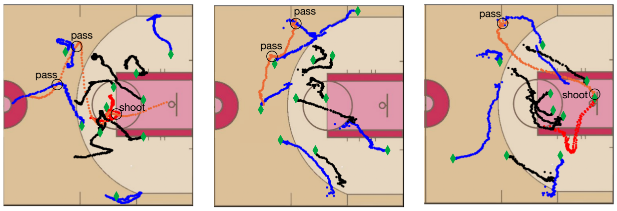
4.3 Overall Performance

Table 3 shows the cumulative scores of the generated trajectories of the compared methods. For all the models, we run each 5 times and report the average performance with the corresponding variance. We observe that: (1) PlayBest consistently and significantly outperforms the baselines and the historical gameplay in generating trajectories with higher rewards. (2) The dedicated offline RL baselines CQL and IQL are also able to learn from historical replays with mixed rewards. However, they perform noticeably worse than PlayBest, indicating that the diffusion model in PlayBest better captures the intrinsic dynamics of basketball gameplay. (3) As expected, the random walk baseline performs poorly, further highlighting the effectiveness of the value function in distinguishing between superior and inferior planning trajectories. These observations suggest that the diffusion model is a powerful tool of modeling complex environmental dynamics and, when combined with guided sampling, becomes a strong planning tool.

4.4 Analysis

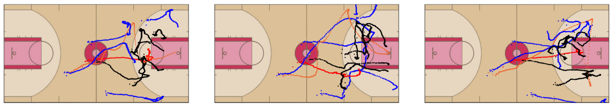
Table 4 demonstrates the overall return evaluated on all the trajectories generated by PlayBest with α being $\{0, 0.01, 0.1, 1.0, 10.0\}$. It is noted that $\alpha = 0$ indicates PlayBest performing unconditional sampling without the perturbation of the gradient of the value function.

4.4.1 Hyperparameter Study. When the diffusion model performs conditional sampling for trajectories, the scaling factor α serves as a balance between quantitative scores and interpretability. With the increase of α, the value guidance generally has a larger impact and improves the overall cumulative rewards on the test games. Then the question becomes, *why not keep increasing the value of α?* To provide a deeper insight into this, we conduct a comparative study demonstrated in Figure 4. We consider trajectories initiated from the same state but with different scaling factors, specifically α values of 0.1, 1.0, and 10.0. By visualizing these trajectories, we aim to demonstrate how variations in the scaling factor can significantly influence the progression and outcomes of the game, further emphasizing the crucial role of this parameter in our model. When $\alpha = 1.0$, there seems to be a mysterious force that pulls the ball to the basket. In the $\alpha = 10.0$ case, the synthesized trajectory becomes even less interpretable since the ball never goes through the basket. In both $\alpha = 1.0$ and $\alpha = 10.0$ cases, the ball exhibits behaviors that defy the laws of physics, seemingly being propelled towards the basket as if being controlled by an invisible player.



(a) Reward: 2.194 **(b) Reward: 0.864** **(c) Reward: 1.541**

Figure 3: (a, b, c): Sampled cases of possessions generated by PlayBest. PlayBest learns strategies deviating from existing data yet still aligning with subjective expectations for effective basketball play. The blue team is on offense and moves towards the right basket, while the black team is on defense. The ball is marked in orange. The player who scores for the blue team is highlighted in Red (no shot attempt in (b)). Diamonds(♦) are final positions of the players. More details are in Section 4.5.



(a) $\alpha = 0.1$ **(b)** $\alpha = 1.0$ **(c)** $\alpha = 10.0$

Figure 4: (a, b, c): Possessions generated by PlayBest with different α.

4.4.2 Ablation Study. The full PlayBest model with sufficient value guidance outperforms the ablation version (i.e., $\alpha = 0$), indicating the necessity of the value guidance. By mere unconditional sampling, the ablation version is already able to generate on average better plans than the ground truth plays in the test set. These observations confirm our two claims: The value-based guided sampling directs the diffusion model to generate trajectories leaning towards the higher-reward regions of the state-action space; and

Table 5: Return values competing against defense.

length m	25	50	75	100
man-to-man	1.410 ± 0.368	1.750 ± 0.059	2.526 ± 0.039	2.814 ± 0.008
2-3 zone	1.424 ± 0.284	1.558 ± 0.309	2.229 ± 0.011	2.327 ± 0.029

the diffusion model on its own can generate coherent and realistic trajectories representing a competent game plan.

4.4.3 The adversary of the game. Notably, basketball games and many other team sports are adversarial. We implemented additional defensive strategies including man-to-man and 2-3 zone defense, and ran the learned policy against these strategies *iteratively* to add adversaries. In each iteration, PlayBest samples a trajectory of length m, and we replace the trajectories corresponding to *defensive* players (5 channels) with those generated with man-to-man or 2-3 zone defense. The trajectories on the defensive side act as adversarial agents competing against the diffusion policy. The results are in Table 5. We observe that: (1) The offensive strategy encoded in PlayBest outplays the man-to-man defense and 2-3 zone defense. (2) When increasing the length of the segment of the trajectory, PlayBest is more likely to generate more coherent trajectories, leading to better returns when faced with the same defense.

4.5 Case Study

We now perform a case study to qualitatively demonstrate the practicability of value-guided conditional generation. Figure 3 shows three cases, all of which are sampled from the trajectories generated by PlayBest. In Figure 3a, we visualize a possession generated with a high reward. The players in the blue team share the ball well and managed to find the red player near the free-throw line. At the time the red player shoots the ball, no defender is between him and the basket. The outcome of this simulated play is a 2-point basket. In Figures 3b and 3c, two different plans with the same horizon are generated by PlayBest given the same initial player and ball positions. In Figure 3b, we observe a more conservative strategy where the ball is repeatedly passed between the blue players near the perimeter, which is also valued with a lower reward. In spite of the same initial conditions, PlayBest generates a more aggressive strategy in Figure 3c in that the ball is passed directly to the low post that leads to a 2-point basket, suggesting an aggressive tactic execution. These cases illustrate that PlayBest is able to not only synthesize realistic trajectories but also output high-reward and diverse trajectories for planning tactics as well as for enhancing decision-making.

5 Related Work

Reinforcement Learning for Planning. Reinforcement learning is a learning-based control approach. A wide range of application domains have seen remarkable achievements through the use of reinforcement learning algorithms, such as robotics [21], autonomous vehicles [6], industrial regulation [15], financial sectors [32], healthcare [50], gaming [37], and marketing [19]. Despite its wide use, many RL applications depend on an online environment that facilitates interactions. In numerous circumstances, acquiring data online is either expensive, unethical, or dangerous, making it a luxury. Consequently, it is preferable to learn effective behavior strategies using only pre-existing data. Offline RL has been suggested to fully utilize previously gathered data without the need for environmental interaction [3, 13, 14, 26, 27], which has found applications in areas such as dialogue systems [18], robotic manipulation techniques [22], and navigation [20].

Sports & Machine Learning. Machine learning and AI have recently been employed in sports analytics to comprehend and advise human decision-making [5, 11, 34, 36, 39, 43, 47]. [30] suggested a player ranking technique that combines inverse RL and Q-learning. [47] proposed a deep-learning model composed of a novel short-term extractor and a long-term encoder for capturing a shot-by-shot sequence. [48] developed a position-aware fusion framework for objectively forecasting stroke returns based on rally progress and player style. [8] predicted returning strokes and player movements based on previous strokes using a dynamic graph and hierarchical fusion approach. While these methods are effective for producing simulations, they may not fully address the goal of maximizing specific objectives (e.g., winning games). Previous basketball analytics mainly focused on employing recurrent neural networks to analyze player-tracking data for offensive tactics identification and player movement prediction [31, 41, 42, 45]. However, these methods lack labeled interactions between the learning agent and the environment, limiting their ability to uncover optimal decision sequences. Wang et al. [44] explored the use of RL to improve defensive team decisions, especially the execution of a "double team" strategy. Liu et al. [28] designed a method using motion capture data to learn robust basketball dribbling maneuvers by training on both locomotion and arm control, achieving robust performance in various scenarios.

6 Conclusion

In this paper, we introduce PlayBest, the diffusion model with conditional sampling in planning high-rewarded basketball trajectories and synthesizing adaptive play strategies. With the extension of environmental dynamics into the diffusion model and fine-grained rewards for the value function, PlayBest has shown impressive capabilities in capturing the intricate dynamics of basketball games and generating play strategies that are consistent with or even surpass professional tactics. Its adaptive nature has allowed for swift adjustments to evolving conditions and facilitated real-time identification of optimal solutions. Extensive simulation studies and analysis of real-world NBA data have confirmed the advantages of PlayBest over traditional planning methods. The generated trajectories and play strategies not only outperform conventional techniques but also exhibit a high level of alignment with professional basketball tactics. Future work will explore the integration of additional sources of information, such as player fatigue and skill levels, into our framework to further enhance its performance. In addition, we plan to develop an open environment and a set of benchmarks to not only facilitate research on machine learning for sports but also extend to other real-time dynamic systems.

Acknowledgements

This work was partially supported by ARO under contract W911NF 1810208; NIH U24DK097771, U54HG012517; NSF 1829071, 2106859, 2119643, 2200274, 2202693, and 2312501; DARPA HR00112490370, and Optum Lab.

References

[1] 2016. Play-by-Play Data. Available at https://www.bigdataball.com/datasets/nba/. https://www.bigdataball.com/datasets/nba/

[2] 2016. SportVU Data. Available at https://github.com/rajshah4/BasketballData/tree/master/2016.NBA.Raw.SportVU.Game.Logs. https://github.com/rajshah4/BasketballData/tree/master/2016.NBA.Raw.SportVU.Game.Logs

[3] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. 2020. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*. PMLR, 104–114.

[4] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. 2022. Is Conditional Generative Modeling all you need for Decision-Making? *arXiv preprint arXiv:2211.15657* (2022).

[5] Raquel YS Aoki, Renato M Assuncao, and Pedro OS Vaz de Melo. 2017. Luck is hard to beat: The difficulty of sports prediction. In *KDD*. 1367–1376.

[6] Bharathan Balaji, Sunil Mallya, Sahika Genc, Saurabh Gupta, Leo Dirac, Vineet Khare, Gourav Roy, Tao Sun, Yunzhe Tao, Brian Townsend, et al. 2019. Deepracer: Educational autonomous racing platform for experimentation with sim2real reinforcement learning. *arXiv preprint arXiv:1911.01562* (2019).

[7] Zdravko I Botev, Dirk P Kroese, Reuven Y Rubinstein, and Pierre L'Ecuyer. 2013. The cross-entropy method for optimization. In *Handbook of statistics*. Vol. 31. Elsevier, 35–59.

[8] Kai-Shiang Chang, Wei-Yao Wang, and Wen-Chih Peng. 2022. Where Will Players Move Next? Dynamic Graphs and Hierarchical Fusion for Movement Forecasting in Badminton. *arXiv preprint arXiv:2211.12217* (2022).

[9] Xiusi Chen, Jyun-Yu Jiang, Kun Jin, Yichao Zhou, Mingyan Liu, P Jeffrey Brantingham, and Wei Wang. 2022. ReLiable: Offline Reinforcement Learning for Tactical Strategies in Professional Basketball Games. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 3023–3032.

[10] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. 2018. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems* 31 (2018).

[11] Tom Decroos, Jan Van Haaren, and Jesse Davis. 2018. Automatic discovery of tactics in spatio-temporal soccer match data. In *KDD*. 223–232.

[12] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems* 34 (2021), 8780–8794.

[13] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219* (2020).

[14] Scott Fujimoto, David Meger, and Doina Precup. 2019. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*. PMLR, 2052–2062.

[15] A Gasparik, C Gamble, and J Gao. 2018. Safety-first ai for autonomous data centre cooling and industrial control. *DeepMind Blog* (2018).

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.

[17] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. 2022. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991* (2022).

[18] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. 2019. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456* (2019).

[19] Junqi Jin, Chengru Song, Han Li, Kun Gai, Jun Wang, and Weinan Zhang. 2018. Real-time bidding with multi-agent reinforcement learning in display advertising. In *CIKM*. 2193–2201.

[20] Gregory Kahn, Pieter Abbeel, and Sergey Levine. 2021. Badgr: An autonomous self-supervised learning-based navigation system. *IEEE Robotics and Automation Letters* 6, 2 (2021), 1312–1319.

[21] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. 2018. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293* (2018).

[22] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. 2018. Scalable deep reinforcement learning for vision-based robotic manipulation. In *ICRL*. 651–673.

[23] Nan Rosemary Ke, Amanpreet Singh, Ahmed Touati, Anirudh Goyal, Yoshua Bengio, Devi Parikh, and Dhruv Batra. 2019. Modeling the long term future in model-based reinforcement learning. In *International Conference on Learning Representations*.

[24] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. 2021. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems* 23, 6 (2021), 4909–4926.

[25] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. 2021. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169* (2021).

[26] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems* 33 (2020), 1179–1191.

[27] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643* (2020).

[28] Libin Liu and Jessica Hodgins. 2018. Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning. *TOG* 37, 4 (2018), 1–14.

[29] Xiao-Yang Liu, Ziyi Xia, Jingyang Rui, Jiechao Gao, Hongyang Yang, Ming Zhu, Christina Wang, Zhaoran Wang, and Jian Guo. 2022. FinRL-Meta: Market environments and benchmarks for data-driven financial reinforcement learning. *Advances in Neural Information Processing Systems* 35 (2022), 1835–1849.

[30] Yudong Luo, Oliver Schulte, and Pascal Poupart. 2021. Inverse reinforcement learning for team sports: valuing actions and players. In *IJCAI*. 3356–3363.

[31] Avery McIntyre, Joel Brooks, John Guttag, and Jenna Wiens. 2016. Recognizing and analyzing ball screen defense in the nba. In *Proc. of the MIT Sloan Sports Analytics Conference*. 11–12.

[32] Terry Lingze Meng and Matloob Khushi. 2019. Reinforcement learning in financial markets. *Data* 4, 3 (2019), 110.

[33] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 7559–7566.

[34] Pieter Robberechts, Jan Van Haaren, and Jesse Davis. 2021. A Bayesian Approach to In-Game Win Probability in Soccer. In *KDD*. 3512–3521.

[35] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241.

[36] Hector Ruiz, Paul Power, Xinyu Wei, and Patrick Lucey. 2017. " The Leicester City Fairytale?" Utilizing New Soccer Analytics Tools to Compare Performance in the 15/16 & 16/17 EPL Seasons. In *KDD*. 1991–2000.

[37] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* 550, 7676 (2017), 354–359.

[38] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*. PMLR, 2256–2265.

[39] Xiangyu Sun, Jack Davis, Oliver Schulte, and Guiliang Liu. 2020. Cracking the black box: Distilling deep sports analytics. In *KDD*. 3154–3162.

[40] Erik Talvitie. 2014. Model Regularization for Stable Sample Rollouts.. In *UAI*. 780–789.

[41] Zachary Terner and Alexander Franks. 2020. Modeling player and team performance in basketball. *Annual Review of Statistics and Its Application* 8 (2020).

[42] Changjia Tian, Varuna De Silva, Michael Caine, and Steve Swanson. 2020. Use of machine learning to automate the identification of basketball strategies using whole team player tracking data. *Applied Sciences* 10, 1 (2020), 24.

[43] Karl Tuyls, Shayegan Omidshafiei, Paul Muller, Zhe Wang, Jerome Connor, Daniel Hennes, Ian Graham, William Spearman, Tim Waskett, Dafydd Steel, et al. 2021. Game Plan: What AI can do for Football, and What Football can do for AI. *JAIR* 71 (2021), 41–88.

[44] Jiaxuan Wang, Ian Fox, Jonathan Skaza, Nick Linck, Satinder Singh, and Jenna Wiens. 2018. The advantage of doubling: A deep reinforcement learning approach to studying the double team in the NBA. *arXiv preprint arXiv:1803.02940* (2018).

[45] Kuan-Chieh Wang and Richard Zemel. 2016. Classifying NBA offensive plays using neural networks. In *Proc. of MIT Sloan Sports Analytics Conference*, Vol. 4.

[46] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. 2019. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057* (2019).

[47] Wei-Yao Wang, Teng-Fong Chan, Wen-Chih Peng, Hui-Kuo Yang, Chih-Chuan Wang, and Yao-Chung Fan. 2022. How Is the Stroke? Inferring Shot Influence in Badminton Matches via Long Short-Term Dependencies. *ACM Transactions on Intelligent Systems and Technology* 14, 1 (2022), 1–22.

[48] Wei-Yao Wang, Hong-Han Shuai, Kai-Shiang Chang, and Wen-Chih Peng. 2022. Shuttlenet: Position-aware fusion of rally progress and player styles for stroke forecasting in badminton. In *AAAI*, Vol. 36. 4219–4227.

[49] Chen Yanai, Adir Solomon, Gilad Katz, Bracha Shapira, and Lior Rokach. 2022. Q-Ball: Modeling Basketball Games Using Deep Reinforcement Learning. In *AAAI*. AAAI Press, 8806–8813.

[50] Chao Yu, Jiming Liu, and Shamim Nemati. 2019. Reinforcement learning in healthcare: A survey. *arXiv preprint arXiv:1908.08796* (2019).

[51] Xinshi Zang, Huaxiu Yao, Guanjie Zheng, Nan Xu, Kai Xu, and Zhenhui Li. 2020. Metalight: Value-based meta-reinforcement learning for traffic signal control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 1153–1160.