ShuttleFlow: Learning the Distribution of Subsequent Badminton Shots using Normalizing Flows

Yun-Hsuan Lien[1*], Chia-Tung Lian[1] and Yu-Shuen Wang[1]

[1*]Department of Computer Science, National Yang Ming Chiao Tung University, 1001 University Road, Hsinchu, 300, Taiwan.

*Corresponding author(s). E-mail(s): sophia.yh.lien@gmail.com;
Contributing authors: doutble@gmail.com; yushuen@cs.nctu.edu.tw;

Abstract

This paper introduces ShuttleFlow, a simple yet effective model designed to forecast badminton shot types and shuttle positions. This tool could be invaluable for coaches, enabling them to identify opponents' weaknesses and devise effective strategies accordingly. Given the inherent unpredictability of player behaviors, our model leverages conditional normalizing flow to generate the distributions of shot types and shuttle positions. This is achieved by considering the players and their preceding shots on the court. To augment the performance of our model, especially in predicting outcomes for players who have not previously competed against each other, we incorporate a novel regularization term. Additionally, we utilize Poisson disk sampling to reduce sample redundancy when generating the distributions. Compared to state-of-the-art techniques, our results underscore ShuttleFlow's effectiveness in forecasting shot types and shuttle positions.

Keywords: Uncertainty, Shot Types, Shuttle Positions, Distribution Modeling

1 Introduction

Badminton is a sport where two competitors aim to strike a shuttle over a net, attempting to land it within the opponent's court to gain points. Esteemed international events, such as the Thomas Cup, Sudirman Cup, and the Olympics, are orchestrated annually by the Badminton World Federation. The ability to accurately anticipate the

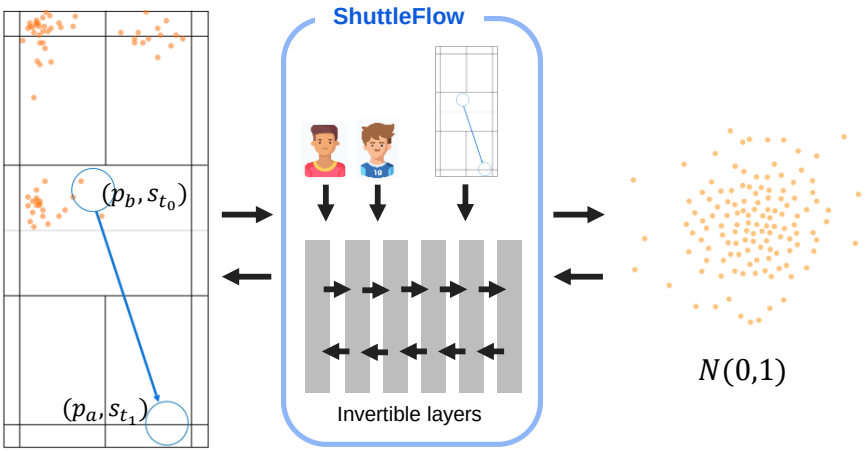

Fig. 1: We introduce ShuttleFlow, an invertible network capable of transforming a normal distribution into the distribution of shot types and shuttle positions, conditioned on the players and their previous shots. Specifically, the blue circles and arrow on the court denote the positions of shots at times t_0 and t_1, respectively, and the orange dots represent potential shuttle positions at shot t_2. In this example, player p_a executes shot t_1, while player p_b performs shots t_0 and t_2.

shuttle's trajectory and shot type is pivotal. Such foresight aids players in positioning themselves optimally and offers coaches critical data for refining game strategies. While predicting exact shot types and shuttle positions remains a significant challenge – owing to players' strategic objectives of making the shuttle descend where their opponent might struggle to reach – it's not insurmountable. Factors like the shuttle's inherent design, the restrictive presence of a net, and the game's swift pace often result in players exhibiting certain patterns. For instance, a player transitioning from the rear to the front of the court is typically more inclined towards defensive plays such as drops or clears, rather than aggressive smashes. Thus, while predicting the exact next move is arduous, calculating its likelihood based on these patterns becomes considerably more manageable.

Our objective is to forecast the distribution of shot types and shuttle locations under specific conditions. The recent ShuttleNet approach (Wang et al, 2021b) uses a transformer network to model shuttle distribution as a bivariate normal distribution, taking data in previous shots as input and generating a mean, variance, and correlation coefficient. Chang et al (2022) expanded on this by incorporating dynamic graphs where each node symbolizes the player's position at a shot. However, the bivariate normal distribution assumption is not consistently valid, as shuttles can land in any position that is challenging for opponents to return (see Figures 1 and 3). Furthermore, both methods predict shot types by minimizing cross-entropy loss, a process that overlooks uncertainty and results in suboptimal generalization (Li et al, 2020). As a result, despite using advanced network architectures and training strategies, these approaches fall short in accurately forecasting shot types and shuttle positions.

The core innovation of our work is that the prediction of shot types and shuttle positions can be significantly simplified when approached as a distribution modeling problem. To support this perspective, we present a simple yet effective forecasting method that surpasses earlier techniques reliant on complex network architectures and training schemes. Our approach utilizes a generative network known as conditional normalizing flows (NF) (Kingma and Dhariwal, 2018; Grathwohl et al, 2018), designed to optimize the distribution of both shot types and shuttle locations. Specifically, an NF takes a random vector z as input and produces a sample x in a target distribution. Compared to generative adversarial networks (GANs) (Goodfellow et al, 2014), NF is free of mode collapse (Winkler et al, 2019), which is a phenomenon where the model fails to generate data as diverse as the actual data distribution. For example, in Figure 1, the shuttle distribution consists of three clusters, and the model may only produce samples belonging to one of these clusters if mode collapse occurs. To mitigate forecasting inaccuracies, we propose a regularization term within the conditional NF to improve its generalization, particularly when predicting matches involving players who have not previously competed against each other. Furthermore, we utilize the Poisson disk sampling (Wei, 2008), a technique known to yield more uniformly distributed samples compared to random sampling, to increase the sampling quality.

Our conditional NF effectively models possible shot types and shuttle locations, leveraging data regarding players and their previous shots. The efficacy of our approach was assessed by comparing it to several baseline models. Experimental results demonstrated the effectiveness of our methodology.

2 Related Work

Normalizing flows

Normalizing flows (NFs) are neural networks that can generate data by transforming a noise vector from a standard normal distribution into more complex samples such as images. They have the benefits of being easy to sample from, stable to train, and accurate at estimating probability densities. However, to be tractable and invertible, normalizing flow layers have certain conditions which may limit their expressiveness compared to feedforward networks. Despite this, several methods have been developed to mitigate these limitations. Discrete normalizing flow methods (Dinh et al, 2014, 2016; Kingma and Dhariwal, 2018; Ho et al, 2019) utilize the change-of-variable theory and involve transforming the data between layers. Continuous normalizing flow methods (Chen et al, 2018; Grathwohl et al, 2018) model data dynamics over time and are formulated as ordinary differential equations. Furthermore, both discrete and continuous normalizing flows can model distributions that are conditioned on given attributes (Lugmayr et al, 2020; Abdal et al, 2021; Yang et al, 2019).

NF is a versatile tool with applications beyond creating content. For instance, they have been utilized to predict uncertain pedestrian trajectories (Zięba et al, 2020) and to capture complex distributions (Sendera et al, 2021). Louizos and Welling (2017) also proposed multiplicative NF for approximating the posterior in a variational setting for Bayesian neural networks. In this work, we use NF to model the distributions of

shot types and shuttle positions, and implement Poisson disk sampling (Wei, 2008) and regularization strategies to enhance forecasting accuracy.

Sports Analytics

Machine learning has been widely used in analyzing sports such as basketball, soccer, and football. These techniques can be employed to discover new tactics (Decroos et al, 2018; Beal et al, 2020), evaluate player actions (Decroos et al, 2019), and predict the outcome of games (Sharma et al, 2021). In addition to team sports, specific methods have also been developed for racket sports like badminton. Since data is crucial in sports analysis, researchers have applied computer vision techniques to identify stroke types, court lines (Chu and Situmeang, 2017), player positions (Ghosh et al, 2018), and reconstruct 3D shuttle trajectories (Shen et al, 2018; Liu and Wang, 2022). While video footage may not always provide sufficient detail about players' fine motor skills (e.g., due to low resolution or occlusion), Song et al (2020) equipped sensors on a smart glove for players to wear and recorded their hand movements during badminton games. Hsu et al (2019) and Wang et al (2020) used both videos and sensors to collect badminton data for tactical analysis.

Sports analytic systems typically depend on visualization to navigate complex tactics and player behaviors (Wu et al, 2021; Chen et al, 2023). To our knowledge, there are few methods for automatically analyzing badminton game data. One of them is ShuttleNet (Wang et al, 2021b), which uses a transformer network to model the shuttle distribution using a bivariate normal distribution. Chang et al (2022) introduced DyMF, which further refines ShuttleNet by incorporating dynamic graphs where each node signifies a player's position at a shot. However, as depicted in Figure 1, the assumption of a bivariate normal distribution may not always hold. Furthermore, the aforementioned methods approach shot type forecasting as a classification task and optimize the network using cross-entropy loss. Considering the inherent uncertainty in subsequent shot types, expecting precise forecast types from the training data could lead to memorization and impaired generalization (Li et al, 2020). In contrast, our method is grounded on generative models and reframes forecasting as a distribution modeling problem, substantially simplifying the task of predicting shot types and shuttle positions.

3 Backgrounds

A normalizing flow (NF) is a type of neural network that can transform data in both the forward and backward directions. Let $f : \mathcal{R}^d \rightarrow \mathcal{R}^d$ represent an NF, and let $\mathbf{x}$ and $\mathbf{z}$ be samples drawn from the distributions $p_x(\mathbf{x})$ and $p_z(\mathbf{z})$, respectively. We express the relationship between $\mathbf{x}$ and $\mathbf{z}$ as: $\mathbf{z} = f^{-1}(\mathbf{x}), \mathbf{x} = f(\mathbf{z})$. In practice, $p_x(\mathbf{x})$ can represent an unknown complex data distribution, while $p_z(\mathbf{z})$ is a standard normal distribution that is easy to sample from and estimate density for. It is also possible to extend an NF to a conditional NF by replacing the distribution $p_x(\mathbf{x})$ with $p_{x|y}(\mathbf{x}|\mathbf{y})$, where $\mathbf{y}$ is a set of attributes that can describe $\mathbf{x}$. Although an NF can be implemented using discrete or continuous methods, we choose the latter due to its proven efficiency (Chen et al, 2018; Grathwohl et al, 2018). In this context, the evolution of data over

time can be expressed as an ordinary differential equation (ODE):

$$\frac{d\mathbf{z}_t}{dt} = f(t, \mathbf{y}, \mathbf{z}_t), \tag{1}$$

where $\mathbf{z}_t$ is the state of $\mathbf{z}$ at time t, $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$ is a random vector, and $\mathbf{y}$ is the condition. The function f satisfies uniform Lipschitz continuity at each $\mathbf{z}_t$, which allows for the invertibility of f. It also parameterizes how a data point $\mathbf{z}_{t_0}(= \mathbf{z})$ evolves to $\mathbf{z}_{t_1}(= \mathbf{x})$ at each time t. The relationship between $\mathbf{x}$ and $\mathbf{z}$ is therefore expressed as follows:

$$\mathbf{x} = \mathbf{z}_{t_0} + \int_{t_0}^{t_1} f(t, \mathbf{y}, \mathbf{z}_t) dt. \tag{2}$$

Since the backward function of NF transforms $p_{x|y}(\mathbf{x}|\mathbf{y})$ to $p_z(\mathbf{z})$, we train the network f^{-1} by minimizing the negative log-likelihood (NLL)

$$\mathcal{L} = -\log p_{x|y}(\mathbf{x}|\mathbf{y}) = -\log p_z(\mathbf{z}_{t_0}) + \int_{t_0}^{t_1} tr\left(\frac{\partial f}{\partial \mathbf{z}_t}\right) dt. \tag{3}$$

Given that the conditional NF is formulated as an ODE, we employ the adjoint method (Pontryagin et al, 1962), an ODE solver, to optimize the network parameters. For the details of these equations and the optimization algorithm, we direct readers to (Chen et al, 2018; Grathwohl et al, 2018).

4 Modeling Shot and Shuttle Location Distributions

The dataset employed in this study contains the shuttle trajectories of professional badminton players during tournament matches. Each trajectory is characterized as a sequence of player positions and shot types, $(\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, ...\mathbf{s}_{n-1})$, where $\mathbf{s}_t = (x_t, y_t, h_t)$ represents a shot at time t, (x, y) indicates a shuttle position, h is a one-hot vector signifying the shot type[1], and n is the total count of shots in a rally.

Since players' decisions on shots are mostly influenced by the recent game conditions, we concentrate on the relationships between consecutive shots t_0, t_1, and t_2. This approach allows us to segment long shuttle trajectories into overlapping sections, each consisting of three shots. Specifically, we extract tuples $(p_a, p_b, \mathbf{s}_{t_0}, \mathbf{s}_{t_1}, \mathbf{s}_{t_2})$ from the dataset, where p_a and p_b are one-hot vectors representing the players. In this context, player p_a executes shot $\mathbf{s}_{t_1}$, while player p_b performs shots $\mathbf{s}_{t_0}$ and $\mathbf{s}_{t_2}$, as depicted in Figure 1. Hence, given a condition $\mathbf{c} = (p_a, p_b, \mathbf{s}_{t_0}, \mathbf{s}_{t_1})$, the goal is to regress this condition to the subsequent shot $\mathbf{s}_{t_2}$.

4.1 Distribution Mapping

Given that player behaviors are inherently unpredictable, even under similar conditions $\mathbf{c}_i$ and $\mathbf{c}_j$, the following shot types and shuttle positions (i.e., $\mathbf{s}_{t_2}(\mathbf{c}_i)$ and $\mathbf{s}_{t_2}(\mathbf{c}_j)$) may vary. Training a neural network to regress similar data to distinct outcomes can

[1]The 10 shot types are net shot, lob, defensive shot, smash, drop, push/rush, short service, clear, drive, and long service.

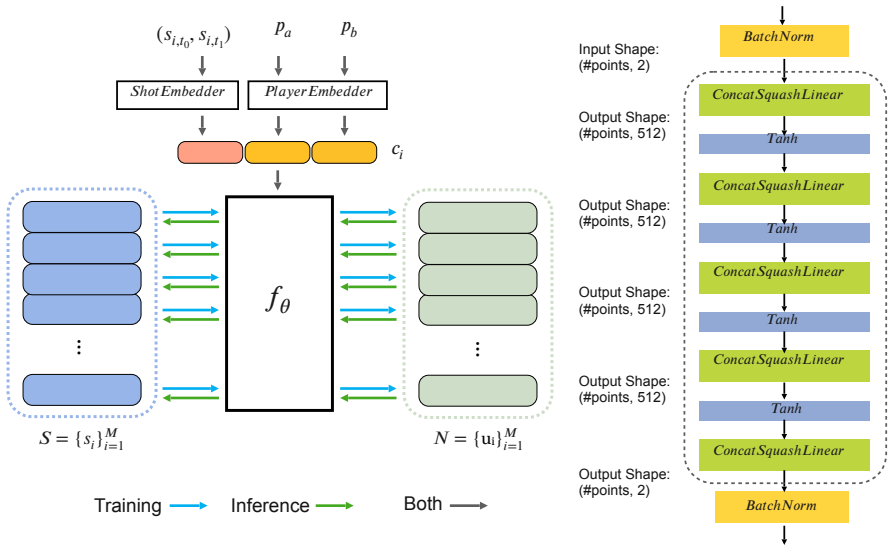

Fig. 2: The diagram on the left illustrates the architecture of our ShuttleFlow. The arrows in different colors depict the direction of data flow during training and inference. The layers and specific functions of f are depicted on the right side.

lead to overfitting (Li et al, 2020). Hence, we choose to model the distribution of possible outcomes rather than the precise shot type and shuttle position. The objective thus becomes to generate the distribution of the subsequent shot $\mathcal{S}_i|\mathbf{c}_i$, where $\mathcal{S}_i$ includes the shots that meet the condition $\mathbf{c}_i$. To achieve this, we train an NF to map a normal distribution to the distribution of shot types and shuttle positions conditioned on the players and their previous shots. The NF is defined as follows:

$$\mathbf{s}_j = \mathbf{u}_{t_0} + \int_{t_0}^{t_1} f_\theta(t, \mathbf{c_i}, \mathbf{u}_t)dt. \tag{4}$$

where $\mathbf{u}_{t_0} \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$, θ is the parameter of f, and $\mathbf{s}_j$ is a shot in $\mathcal{S}_i$ that satisfies $\mathbf{c}_i$. Since an NF is a bi-directional neural network, we train the backward function f_θ^{-1} using the negative log-likelihood:

$$\mathcal{L}_{n\ell\ell} = -\log p_{\mathcal{S}|\mathcal{C}}(\mathbf{s}_j|\mathbf{c}_i) = -\log p_u(\mathbf{u}_{t_0}) + \int_{t_0}^{t_1} tr\left(\frac{\partial f}{\partial \mathbf{u}_t}\right) dt. \tag{5}$$

Upon completion of training, the forward function f_θ is employed to predict upcoming shots $\mathcal{S}_i$, utilizing information about the players and their preceding shots $\mathbf{c}_i$. As NFs are bidirectional neural networks, their ability to transform a data distribution into a normal distribution implies a reciprocal capacity to recreate the complete data distribution when reverting from the normal distribution. This guarantees an accurate modeling of shot type and shuttle position distributions.

4.2 Positional embedding

The NF maps the distribution from $\mathcal{N}$ to $\mathcal{S}_i|\mathbf{c}_i$, which is conditioned on player positions and shot types. Since $\mathcal{S}_i|\mathbf{c}_i$ can vary significantly in terms of $\mathbf{c}_i$, we apply positional embedding (Mildenhall et al, 2020) to $\mathbf{s}$ before inputting it into the network. Let $p \in \{x, y\}$ be a coordinate of $\mathbf{s}$. We expand $\mathbf{s}$ to pos($\mathbf{s}$) by increasing the dimensionality of each p to $2L$ using: $(\sin(2^0\pi p), \cos(2^0\pi p), ..., \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p))$. Namely, the condition $\mathbf{c}_i = (p_a, p_b, \text{pos}(\mathbf{s}_{i,t_0}), \text{pos}(\mathbf{s}_{i,t_1}))$ is obtained.

4.3 Regularization

Coaches may require predictions for matches between specific player pairs without available training data, which could lead to unexpected outcomes. To address this, we introduce a regularization term, assuming a player would exhibit typical behavior when facing a new opponent. Specifically, we define $\mathcal{S}_{p_a p_b}(\mathbf{s}_{i,t_0}, \mathbf{s}_{i,t_1})$ to represent the distribution played by players p_a against p_b when they are in conditions $\mathbf{s}_{i,t_0}$ and $\mathbf{s}_{i,t_1}$. We also define $\mathcal{S}_{p_a}(\mathbf{s}_{i,t_0}, \mathbf{s}_{i,t_1}) = \cup_{p_b}\mathcal{S}_{p_a p_b}(\mathbf{s}_{i,t_0}, \mathbf{s}_{i,t_1})$, where p_b can be an arbitrary player in the dataset. In other words, $\mathcal{S}_{p_a}$ represents the behavior of player p_a in general, regardless of the opposing player. For simplicity, we will use $\mathcal{S}$ as an abbreviation for $\mathcal{S}(\mathbf{s}_{i,t_0}, \mathbf{s}_{i,t_1})$, and assume identical conditions when comparing these terms.

The regularization term is based on the assumption that player p_a would respond in a standard manner when competing against a new opponent p_c. Let $D(\cdot, \cdot)$ denote the distance between two distributions. The regularization term can be expressed as:

$$\mathcal{L}_{reg}(p_a, p_c) = |D(S_{p_a p_c}, S_{p_a})|\,. \tag{6}$$

Since $\mathcal{S}_{p_a}$ can be pre-computed as the union of $\mathcal{S}_{p_a,p_b}$ over all p_b using training data, minimizing Equation 6 is equivalent to transforming the distribution of $\mathcal{S}_{p_a}$ to a standard normal distribution $\mathcal{N}$ subject to $\mathbf{c}_{p_a p_c}$, in which the two players in the condition are p_a and p_c. Therefore, we rewrite the term as follows:

$$\begin{aligned}\mathcal{L}_{reg}(p_a, p_c) &= -\log p_{\mathcal{S}|\mathcal{C}}(\mathbf{s}_j|\mathbf{c}_{p_a p_c}) \\ &= -\log p_u(\mathbf{u}_{t_0}) + \int_{t_0}^{t_1} tr\left(\frac{\partial f}{\partial \mathbf{u}_t}\right) dt. \quad \forall \mathbf{s}_j \in \mathcal{S}_{p_a}\,.\end{aligned} \tag{7}$$

In our implementation, we minimize the regularization term $\mathcal{L}_{reg}(p_a, p_c)$ solely to pairs of players p_a and p_c who have not previously competed against each other.

4.4 Poisson Disk Sampling

To effectively plan tactics, coaches and players need to analyze the distributions of shot types and shuttle positions by their opponents. To do this, they can plot dots on a tactical board that depicts the distribution. One method to achieve this is by randomly drawing samples from a standard normal distribution and feeding them into an NF. However, this approach may result in clumped samples and empty spaces, providing redundant or missing information. To resolve this problem, we utilize Poisson disk

sampling (Wei, 2008) to create more evenly distributed samples. Our method includes drawing samples from a uniform distribution using Poisson disk sampling, converting them to a normal distribution through a cumulative distribution function, and then using the NF to map the samples to the distribution of shot types and shuttle positions.

4.5 Implementation Details

Network architecture

Our network consists of three components: a player encoder, a shot encoder, and an NF. The player encoder converts one-hot vectors p_a and p_b into player embeddings $\mathcal{M}_a$ and $\mathcal{M}_b$, respectively, while the shot encoder takes shots (i.e., 12D, including player positions and shot types) $\mathbf{s}_{t_0}$ and $\mathbf{s}_{t_1}$ as inputs and generates a shot embedding $\mathcal{M}_p$. The three embeddings are then concatenated and used as a condition for the NF. The architecture of the network is shown in the left part of Figure 2. The NF is constructed using a moving batch normalizing layer (Yang et al, 2019), four concatsquash layers (Grathwohl et al, 2018), and another batch normalization layer, as shown in the right part of Figure 2. Additionally, the concatsquash layer is defined as follows:

$$\begin{aligned} CCS(t, \mathbf{c}, \mathbf{u}) &= \sigma_1((W_u\mathbf{u} + b_u) \times gate + bias), \\ \text{where} \quad \text{gate} = \sigma_2(\mathrm{W_{tt}t} + \mathrm{W_{tc}}\mathbf{c} + \mathrm{b_t}), &\text{bias} = (\mathrm{W_{bt}t} + \mathrm{W_{bc}}\mathbf{c} + \mathrm{b_b t}), \end{aligned} \tag{8}$$

$W_u, W_{tt}, W_{tc}, W_{bt}, W_{bc}, b_u, b_t, b_b$ are learnable parameters, and σ_1 and σ_2 are *tanh* and *sigmoid* activation functions, respectively.

Parameter Setting

We used the Adam optimizer to train the network, with a batch size of 64, a learning rate of 0.002, and β_1 and β_2 values of 0.9 and 0.999, respectively. The tolerance for the ODE solver was set to 10^{-5}, and the expanded dimensionality $2L$ for positional embedding was set to 6.

5 Results and Evaluations

We evaluate the efficacy of our algorithm by comparing it against several baselines under two distinct experimental setups. These methods employ the same input as ShuttleFlow to predict shot types and shuttle positions.

5.1 Data and Experiments

We tested our method using the dataset released by ShuttleNet (Wang et al, 2021b), which consists of 78 games played by 28 professional players. The games were divided into 4031 rallies, each with an average of 10 shots. Since each prediction is based on its previous two shots, we excluded rallies with fewer than three shots when evaluating the performance. This resulted in 3553 rallies being used for evaluation. We conducted two experiments to assess the performance of our method.

Method	Setting	Position (RMSE) ↓				Shot type (CE) ↓			
		k=16	k=32	k=64	k=96	k=16	k=32	k=64	k=96
ShuttleNet	Exp 1: by rallies	0.93	0.86	0.81	0.79	2.27	2.27	2.27	2.27
DyMF		0.77	0.75	0.74	0.72	1.98	1.98	1.98	1.98
REG		0.38	0.33	0.30	0.28	1.83	1.70	1.68	1.65
CVAE		0.38	0.33	0.31	0.30	1.82	1.74	1.70	1.68
PGGAN		0.31	0.27	0.26	0.25	1.75	1.68	1.64	1.63
ShuttleFlow		**0.27**	**0.21**	**0.16**	**0.13**	**1.47**	**1.35**	**1.32**	**1.29**
ShuttleNet	Exp 2: by players	0.86	0.79	0.72	0.69	2.29	2.29	2.29	2.29
DyMF		1.05	1.04	1.02	1.01	1.99	1.99	1.99	1.99
REG		0.50	0.44	0.40	0.39	2.12	2.03	1.98	1.96
CVAE		0.51	0.46	0.43	0.42	2.24	2.14	2.11	2.10
PGGAN		0.38	0.34	0.33	0.31	2.21	2.09	2.04	2.01
ShuttleFlow		**0.28**	**0.21**	**0.16**	**0.13**	**1.52**	**1.33**	**1.32**	**1.29**

Table 1: We compared our ShuttleFlow with state-of-the-art methods and several generative models under two experimental settings, with the numbers indicating the forecasting errors associated with different sampling numbers k. It's worth noting that the CE values for both ShuttleNet (Wang et al, 2021b) and DyMF (Chang et al, 2022) remain consistent irrespective of k, as they treat shot type forecasting as a classification problem. In contrast, for the generative models, the CE values exhibit a minor decrease as k increases, attributable to the enhanced accuracy of the estimated categorical distribution. The best results are highlighted in bold fonts.

Experiment 1: data partition by rallies

This experiment, which was conducted according to ShuttleNet's protocol, aimed to forecast games featuring players that the network had learned to recognize. Let p_a and p_b be two players in a testing sample. In this experiment, the rallies played by players (p_a, p_b) must appear in the training set. We randomly partitioned the data into 2728 rallies for training and 725 rallies for testing, while ensuring that the condition mentioned above was met.

Experiment 2: data partition by pairs of players

Due to the limited number of games held in tournaments, there may not be enough rallies played by a specific pair of players for training. Therefore, we evaluated the forecasting results in a scenario where two players compete in a game for the first time. Let p_a and p_b be two players in a testing sample. We assumed that the training set includes rallies played by p_a and p_b, but not by these two players together. To evaluate this scenario, we first grouped the rallies according to the pairs of players and then moved a group to the testing set for evaluation. The remaining groups were used for training. Due to the imbalanced sizes of the data, we chose to evaluate five pairs of players with the most samples.

In both experiments, we trained the NF for 30 epochs. Besides, we randomly selected 10% of the training data and moved it to the validation set for model selection.

5.2 Evaluation Metrics

Our evaluation strategy was consistent with the methodologies employed in ShuttleNet (Wang et al, 2021b) and DyMF (Chang et al, 2022). Although the future shots are

uncertain, the ground truth for each prediction is restricted to one shot type and one court coordinate. To assess prediction accuracy, we drew k random samples from the forecast distribution. For the shuttle's position, the accuracy was evaluated by taking the smallest distance between these samples and the actual result as the error in prediction. In contrast, the shot type's accuracy was determined by creating a categorical distribution from the samples, followed by the computation of the cross-entropy (CE) error. Recall that the CE error is defined by:

$$\mathcal{L}_{CE} = -\sum_{x} p(x) \log q(x), \tag{9}$$

with $p(x)$ and $q(x)$ symbolizing the actual and predicted probabilities for each category x. Given that the CE function aggregates the weighted prediction errors across all classes, the average error across the testing set offers insights into the accuracy of predicted probabilities, even in the face of uncertain future shot types.

Due to the probabilistic nature of the evaluation, we tested several values of k and repeated the experiments 10 times. The root mean square errors (RMSE) and CE errors are reported in Table 1. Intuitively, a higher value of k would lead to a lower distance error because more samples are drawn from the forecast distribution. On the other hand, the CE values are less irrelevant to k since the drawn samples are used to determine the categorical distribution of shot types.

5.3 Comparison to State-of-the-art Methods

We evaluated the performance of ShuttleFlow through a comparative study against two baseline methods: ShuttleNet (Wang et al, 2021b) and DyMF (Chang et al, 2022).

Experiment Results

Table 1 provides the forecasting errors for Experiments 1 and 2 in relation to the number of drawn samples, k. ShuttleFlow outperformed the baseline methods - ShuttleNet (Wang et al, 2021b) and DyMF (Chang et al, 2022) - in both experiments. For shuttle positions, our forecasting errors were approximately three times lower than those of the baselines. These outcomes are consistent with expectations given that ShuttleFlow is adept at modeling complex shuttle positions. The scatter plots showcased in Figure 3 shed light on ShuttleFlow's performance. These plots also highlight the inaccuracy of assuming the shuttle distribution to be a bivariate normal distribution.

Although ShuttleNet and DyMF can be extended by generating multiple Gaussians to fit the shuttle distribution, training such networks is challenging for two main reasons. Firstly, it's not immediately clear which Gaussian is responsible for which part of the data distribution. Secondly, it requires considering not only the distribution of each Gaussian component but also their corresponding weights. By contrast, normalizing flow offers a more straightforward approach, as it directly transforms the data distribution through an invertible function without needing to manage multiple component distributions and their corresponding weights.

In addition to shuttle positions, ShuttleFlow reduced the CE errors in shot type forecasting to about three-quarters of those made by ShuttleNet and DyMF. While

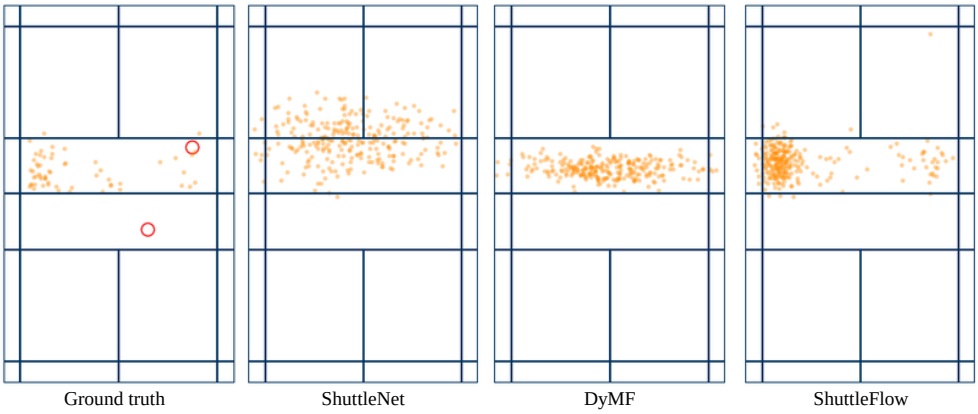

Fig. 3: The predicted shuttle distributions, as derived from ShuttleNet (Wang et al, 2021b), DyMF (Chang et al, 2022), and ShuttleFlow, are presented. The player's positions at shots t_0 and t_1 are denoted by red circles, with the orange dots signifying probable shuttle locations at shot t_2. Both shots t_0 and t_1 are drop shots. The ground truth result is established by collecting shuttle positions from the training data where their conditions closely resemble the one used in this example.

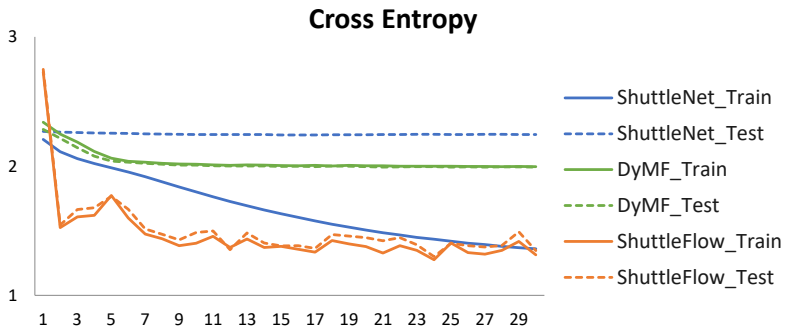

Fig. 4: The CE curves indicate that only ShuttleFlow can model the uncertainty associated with shot types. The experiment was conducted in Experiment 1.

the CE loss function can be used to assess the accuracy of shot type forecasting, it is less ideal for steering network training due to inherent uncertainties. This arises because networks, in their bid to minimize CE loss, often resort to producing sharper class probability vectors during forecasting (Wei et al, 2022; Nguyen et al, 2015). This strategy, however, can result in overfitting and subsequent poor generalization (Li et al, 2020). As illustrated in Figure 4, while ShuttleNet's training loss consistently decreased over time, its testing loss remained high. DyMF, conversely, demonstrated signs of underfitting, with both its training and testing loss no longer decreasing after just 5 training epochs. This may stem from DyMF's robust training strategy that

treats conflicting examples equally (Ma et al, 2020; Wang et al, 2021a). In contrast, our method utilizes a generative network to simultaneously model shot types and shuttle positions, which enhances forecasting accuracy. It's worth noting that while ShuttleFlow's loss values displayed some variations, this is primarily a result of its optimization strategy using the NLL loss instead of the CE loss, along with its method of using sampling to establish the categorical distribution.

5.4 Comparison to Generative Networks

As we have reformulated the forecasting of shot types and shuttle positions as a distribution modeling problem, we also compared ShuttleFlow to several generative networks that can generate a distribution of data points. Detailed descriptions of these methods can be found in Figure 5 and the following paragraphs.

Regression (REG)

We train a regression network that takes a 128D random vector and a specific condition as inputs. It outputs a 256×12 D vector, representing 256 2D positions, and 256 10D vectors, signifying the distribution of upcoming shot types. To train the network, we utilize the earth mover's distance (EMD) (Rubner et al, 2000) to measure the disparity between predicted and true distributions, anticipating similarities between them. Note that training this regression network, as well as the conditional variational autoencoder discussed in the subsequent paragraph, necessitates a specific data collection process since the condition aligns with a particular output distribution. In contrast, training the ShuttleFlow does not require this process because its loss evaluates the likelihood of the transformed distribution being a standard normal distribution. In other words, subsequent shots with different conditions can be optimized simultaneously in the same batch. Further details on the data collection process are provided in appendix.

Conditional Variational Autoencoder (CVAE) (Sohn et al, 2015)

Similar to conditional normalizing flows, CVAEs are frequently employed for data generation based on specific attributes. In our evaluation, the CVAE encodes the upcoming shuttle distributions and shot types (i.e., a 256×12 D vector) into 128D vectors, then decodes these vectors to reconstruct the original data. The given condition serves as an input for both encoding and decoding processes. The network is optimized using Kullback-Leibler divergence and the EMD. After training, the encoder is eliminated, and the decoder processes a 128D random vector and the condition to generate the predicted distribution of shot types and shuttle positions. We utilize the EMD as opposed to Euclidean distance for reconstruction, as it isn't obligatory for each reconstructed point to align closely with its initial position. In fact, employing L1 or L2 reconstruction loss to train this CVAE would lead to failure.

Point Cloud Generative Adversarial Network (PCGAN) (Li et al, 2018)

Since shuttle distributions can be interpreted as 2D point clouds, we employed the PCGAN to predict the distribution of shot types and shuttle positions. The generator accepts 10D random vectors and the condition, subsequently outputting 12D vectors

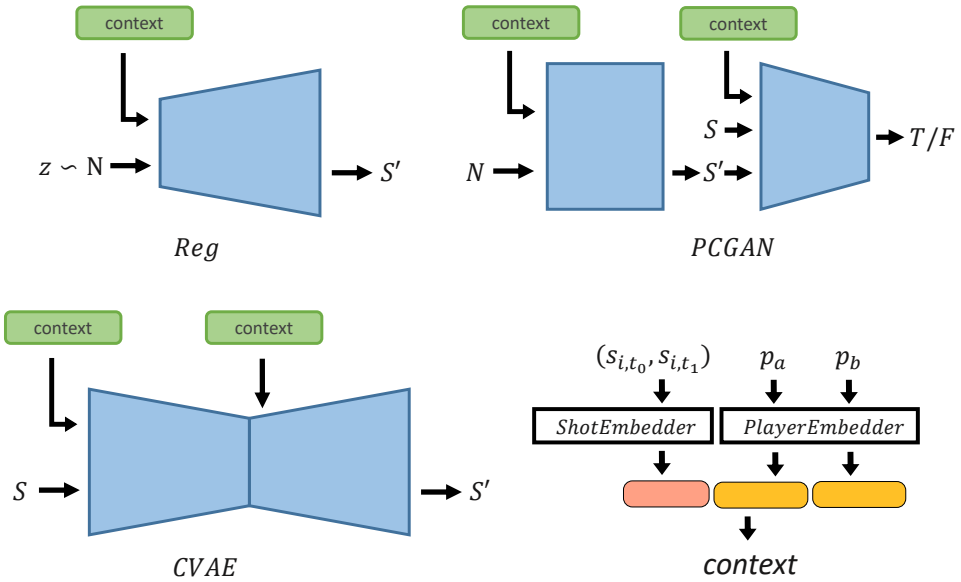

Fig. 5: The network structure of the baseline generative models. The context module, depicted in the bottom right, is identical to our condition embedding network.

that represent shuttle positions and shot types. Following this, the discriminator evaluates whether a shuttle position and its corresponding shot type are authentic or generated by the generator. Both generator and discriminator undergo iterative and alternating training using adversarial loss. To enhance PCGAN's training efficiency, we also incorporated the EMD loss.

Experiment Results

In Table 1, we present the forecasting errors for both shuttle position and shot type. As can be seen, the generative models, including REG, CVAE, PCGAN, surpassed ShuttleNet and DyMF when forecasting shuttle positions. They also outperformed ShuttleNet and DyMF when forecasting shot types in Experiment 1. This advantage can be attributed to the generative models' ability to generate complex data distributions. PGGAN, in particular, displayed remarkable forecasting precision. However, a notable exception was its performance in shot type forecasting during Experiment 2, where PCGAN's performance aligned more closely with that of REG and CVAE because of their comparable CE loss values. This deviation is rooted in insufficient training data. The training approach for these generative models leverages the EMD distance, seeking to narrow the gap between real and generated distributions. Given that there might only be limited upcoming shots that fit a specific condition (i.e., players, positions and shot types in previous shots), the real distributions they're based on might not be representative. Consequently, when training on these distributions, the models struggle to accurately predict shot types in more challenging scenarios.

Method	Exp1: by rallies		Exp2: by players	
	Position	Type	Position	Type
-Reg, -PD	0.38	1.70	0.41	1.79
-PD	0.38	1.68	0.41	1.72
-Reg	**0.34**	**1.58**	0.36	1.65
ShttleFlow	**0.34**	**1.58**	**0.34**	**1.63**

Table 2: We evaluated the effectiveness of the regularization term and the Poisson disk sampling by comparing the forecasting errors of shot types (CE) and shuttle positions (RMSE). For this experiment, we set the number of drawn samples, k, to be 10.

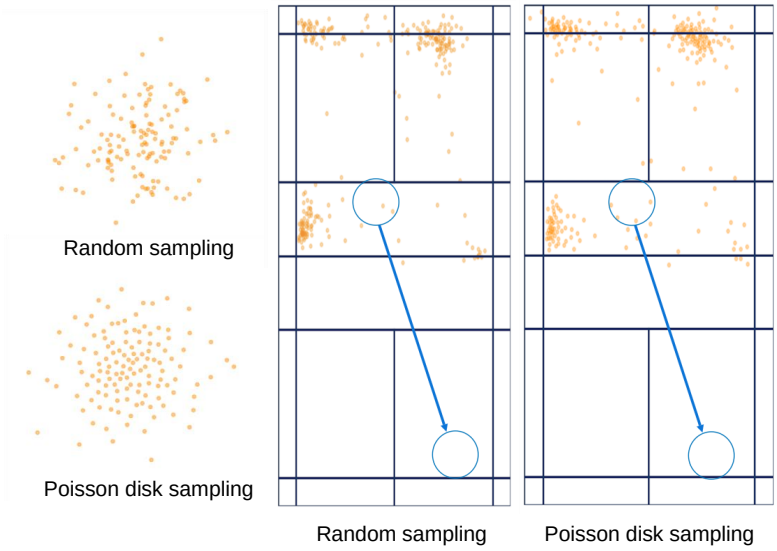

Fig. 6: The left side of the diagram depicts the samples drawn from a normal distribution, with random sampling at the top and Poisson disk sampling at the bottom. The shuttle distributions transformed from the samples drawn by different methods are shown in the middle and on the right. The samples drawn using Poisson disk sampling are clearly less clumped and can represent the distribution better.

This limitation also sheds light on why REG, CVAE, and PCGAN lagged behind our ShuttleFlow in performance.

5.5 Ablation Study

To evaluate the impact of the regularization and sampling strategies, we compared the performance of the methods with and without these features. These variants, which remove regularization and Poisson disk sampling, are referred to as -Reg and -PD, respectively. Table 2 shows that the regularization term is effective for Experiment

Exp2: by players		P1 vs. P2	P1 vs. P3	P1 vs. P4	P5 vs. P6	P5 vs. P7
Position	-Reg	0.39	0.34	0.36	0.35	0.36
	ShuttleFlow	**0.36**	**0.31**	0.36	0.35	**0.33**
Type	-Reg	1.59	1.58	1.57	1.55	1.60
	ShuttleFlow	**1.59**	**1.55**	1.54	**1.54**	**1.56**

Table 3: We examined the outcomes of -Reg and ShuttleFlow in five pairs of players during the evaluation. The CE and RMSE errors suggest that the regularization term was advantageous for specific player pairs.

2, where the term was specifically designed to enhance the network's generalization performance when the training data does not include any matches between certain pairs of players. Additionally, we provided forecasting details in Table 3. The results show that the regularization term is not detrimental, and it provides advantages for certain pairs of players. We suspect that the pairs without considerable improvement are due to substantial changes in tactics. Note that under such circumstances, no strategies are guaranteed to be effective.

The Poisson disk sampling is surprisingly beneficial in both settings, although its initial goal was to depict shuttle distributions to coaches and players in a better way. However, its effectiveness was reasonable, as redundant samples were prevented, so the errors could be further reduced. Figure 6 visually illustrates this phenomenon. In the comparison, the distribution forecasted using Poisson disk sampling, shown on the right, is noticeably more uniform and spans a broader area compared to the one generated through random sampling. This visual distinction underscores the reduced forecasting error attributed to the Poisson disk sampling method.

5.6 Sampling Efficiency

ShuttleFlow efficiently forecasts upcoming shot types and shuttle positions by generating k examples. Each generated sample is a concise 12D vector indicating a coordinate and a shot type. Moreover, all k samples can be computed simultaneously. In real-world applications, ShuttleFlow takes approximately 0.5 seconds to produce a distribution consisting of 96 samples on an RTX 4090 GPU, demonstrating interactive performance. It's important to note that we chose not to use diffusion models for predicting shuttle distributions due to sampling efficiency. Unlike normalizing flows, which only need one forward pass to map noises to data, diffusion models (Yang et al, 2022) require many sequential steps involving stochastic differential equations to gradually remove noise. Since a distribution needs to have enough samples to be representative, this sequential process can significantly increase computational cost.

5.7 Limitations and Future Works

Despite the promising performance of ShuttleFlow in forecasting shot types and shuttle positions in badminton games, there remains significant room for enhancement. While the regularization term can enhance the network's generalization when certain pairs of players are not present in the training data, this improvement appears to be somewhat limited due to the differences in tactics between players. Furthermore, our

evaluations of ShuttleFlow have been limited to badminton games. Though it holds promise for forecasting shots in other racquet sports like tennis and table tennis, the lack of publicly available datasets impedes such evaluations. Lastly, our prediction approach leans on NF instead of a recently popular generative model – diffusion model (Yang et al, 2022), primarily due to its inference efficiency. Given recent advancements that have reduced the training and inference times of diffusion models, exploring them further is a compelling avenue for future work.

6 Conclusion

We have introduced ShuttleFlow, an innovative and effective method for predicting shot types and shuttle positions in badminton games, taking into account the players and their conditions at previous shots. Unlike existing state-of-the-art methodologies, such as ShuttleNet (Wang et al, 2021b) and DyMF (Chang et al, 2022), which assume Gaussian patterns in shuttle distributions and neglect the inherent uncertainty in shot types, ShuttleFlow breaks this convention when forecasting. By modeling the distribution of shuttle positions and shot types, ShuttleFlow significantly improves forecasting accuracy despite its relatively simple architecture and straightforward network training process. As demonstrated in experimental results, ShuttleFlow has the potential to substantially contribute to the domain of sports analytics by accurately predicting player behaviors in badminton games.

Acknowledgements. We thank the reviewers for their constructive comments. This study is based upon work partially supported by the National Science and Technology Council (NSTC), Taiwan, under Contract No. 111-2221-E-A49 -129 -MY3, Contract No. 113-2425-H-A49 -001 - and Contract No. 113-2221-E-A49 -149 -MY3.

Appendix A

A1. Data Collection for Training Generative Models

Training generative models to forecast future shots necessitates a specific data collection process since the condition aligns with a particular output distribution. To achieve this goal, we divided the badminton court into $2m \times m$ grids, with $m = 8$ in our implementation. This resulted in each grid corresponding to a real-world area of approximately 0.84×0.76 square meters, which is roughly the space a player occupies on the court. We collect the distribution $\mathcal{S}_i$ for each condition $\mathbf{c}_i$ if the previous two shots locate at the same quad and with the same shot types. Due to badminton tactics and data collection issues, the size of the distribution, $\mathcal{S}_i$, varies and may contain only a few samples. To ensure data reliability, we removed the distribution $\mathcal{S}_i|\mathbf{c}_i$ if the size of $\mathcal{S}_i$ was smaller than a threshold of $\gamma = 16$. For sets containing more than γ samples, we randomly duplicated or discarded samples to make the set size equal to 256 and added a small noise vector to each duplicated sample to increase data diversity.

	REG	CVAE	PCGAN	ShuttleNet	DyMF	ShuttleFlow
Time (s)	28,630	28,760	30,703	8,418	2,307	36,575
# Param	655,870	755,198	418,427	108,000	15,694	467,486

Table A1: We compare the training time, measured in seconds, and the number of parameters between the baselines and ShuttleFlow.

RMSE; k=96	No Condition	+Player	+Position	ShuttleFlow
Exp1: by rallies	1.98	0.64	0.18	0.13
Exp 2: by players	1.96	0.44	0.19	0.13

Table A2: We evaluate the forecast errors of shuttle positions when not taking into account the players and their positions in the previous two shots.

k=16	Exp1: by rallies		Exp2: by players	
	RMSE	CE	RMSE	CE
ShuttleNet	0.93 ± 0.06	2.27 ± 0.07	0.86 ± 0.04	2.29 ± 0.06
DyMF	0.77 ± 0.02	1.98 ± 0.04	1.05 ± 0.03	1.99 ± 0.03
REG	0.38 ± 0.04	1.83 ± 0.09	0.50 ± 0.03	2.12 ± 0.07
CVAE	0.38 ± 0.03	1.82 ± 0.06	0.51 ± 0.02	2.24 ± 0.06
PCGAN	0.31 ± 0.03	1.75 ± 0.05	0.38 ± 0.04	2.21 ± 0.08
ShuttleFlow	0.27 ± 0.05	1.47 ± 0.07	0.28 ± 0.02	1.52 ± 0.09

Table A3: The average and standard deviation of the forecast results are presented.

A2. Comparison of Computational and Memory Costs

Table A1 compares the training time and network size of different baseline models and ShuttleFlow. Despite the generative models being larger than ShuttleNet (Wang et al, 2021b) and DyMF (Chang et al, 2022), their forecasting accuracies are indeed higher than these two non-generative models, as shown in Table 1. Note that even though ShuttleNet and DyMF used larger networks, they could only generate bivariate normal distributions when forecasting shuttle distributions, which is not advantageous.

A3. More Evaluations

The performance of ShuttleFlow depends significantly on player indices and their positions from the previous two shots. Player indices are helpful in separating player data during training because player behaviors can differ. The positions from the previous two shots provide context for the player's next move. We assess the forecasting results when these two conditions are not taken into account in Table A2. Additionally, to assess performance stability, we included standard deviation values from five different runs for Experiments 1 and 2 (Table A3). We specifically focus on the results of $k = 16$ because both the prediction errors and variations decrease as we draw more samples (i.e., larger k) from the generated distribution. It's important to note that the standard deviations are low when using randomly selected seeds.

References

Abdal R, Zhu P, Mitra NJ, et al (2021) Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. ACM Transactions on Graphics 40(3):1–21

Beal R, Chalkiadakis G, Norman TJ, et al (2020) Optimising game tactics for football. arXiv preprint arXiv:200310294

Chang KS, Wang WY, Peng WC (2022) Where will players move next? dynamic graphs and hierarchical fusion for movement forecasting in badminton. arXiv preprint arXiv:221112217

Chen RT, Rubanova Y, Bettencourt J, et al (2018) Neural ordinary differential equations. arXiv preprint arXiv:180607366

Chen WT, Wu HY, Shih YA, et al (2023) Exploration of player behaviours from broadcast badminton videos. In: Computer Graphics Forum, Wiley Online Library

Chu WT, Situmeang S (2017) Badminton video analysis based on spatiotemporal and stroke features. In: ACM on International Conference on Multimedia Retrieval, p 448–451, https://doi.org/10.1145/3078971.3079032

Decroos T, Van Haaren J, Davis J (2018) Automatic discovery of tactics in spatio-temporal soccer match data. In: ACM SIGKDD international conference on knowledge discovery & data mining, pp 223–232

Decroos T, Bransen L, Van Haaren J, et al (2019) Actions speak louder than goals: Valuing player actions in soccer. In: ACM SIGKDD international conference on knowledge discovery & data mining, pp 1851–1861

Dinh L, Krueger D, Bengio Y (2014) Nice: Non-linear independent components estimation. arXiv preprint arXiv:14108516

Dinh L, Sohl-Dickstein J, Bengio S (2016) Density estimation using real nvp. arXiv preprint arXiv:160508803

Ghosh A, Singh S, Jawahar CV (2018) Towards structured analysis of broadcast badminton videos. In: IEEE Winter Conference on Applications of Computer Vision, pp 296–304

Goodfellow I, Pouget-Abadie J, Mirza M, et al (2014) Generative adversarial nets. Advances in neural information processing systems 27

Grathwohl W, Chen RT, Bettencourt J, et al (2018) Ffjord: Free-form continuous dynamics for scalable reversible generative models. arXiv preprint arXiv:181001367

Ho J, Chen X, Srinivas A, et al (2019) Flow++: Improving flow-based generative models with variational dequantization and architecture design. In: International Conference on Machine Learning, pp 2722–2730

Hsu T, Chen C, Jut NP, et al (2019) Coachai: A project for microscopic badminton match data collection and tactical analysis. In: Asia-Pacific Network Operations and Management Symposium, pp 1–4, https://doi.org/10.23919/APNOMS.2019.8893039

Kingma DP, Dhariwal P (2018) Glow: Generative flow with invertible 1x1 convolutions. arXiv preprint arXiv:180703039

Li CL, Zaheer M, Zhang Y, et al (2018) Point cloud gan. arXiv preprint arXiv:181005795

Li M, Soltanolkotabi M, Oymak S (2020) Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In: International conference on artificial intelligence and statistics, pp 4313–4324

Liu P, Wang JH (2022) Monotrack: Shuttle trajectory reconstruction from monocular badminton video. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3513–3522

Louizos C, Welling M (2017) Multiplicative normalizing flows for variational bayesian neural networks. In: International Conference on Machine Learning, pp 2218–2227

Lugmayr A, Danelljan M, Van Gool L, et al (2020) Srflow: Learning the super-resolution space with normalizing flow. In: European Conference on Computer Vision, pp 715–732

Ma X, Huang H, Wang Y, et al (2020) Normalized loss functions for deep learning with noisy labels. In: International conference on machine learning, PMLR, pp 6543–6553

Mildenhall B, Srinivasan PP, Tancik M, et al (2020) Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision, pp 405–421

Nguyen A, Yosinski J, Clune J (2015) Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 427–436

Pontryagin LS, Mishchenko E, Boltyanskii V, et al (1962) The mathematical theory of optimal processes

Rubner Y, Tomasi C, Guibas LJ (2000) The earth mover's distance as a metric for image retrieval. International journal of computer vision 40(2):99–121

Sendera M, Tabor J, Nowak A, et al (2021) Non-gaussian gaussian processes for few-shot regression. Advances in Neural Information Processing Systems 34:10285–10298

Sharma M, Lamba M, Kumar N, et al (2021) Badminton match outcome prediction model using naïve bayes and feature weighting technique. Journal of Ambient Intelligence and Humanized Computing 12(8):8441–8455

Shen L, Liu Q, Li L, et al (2018) Reconstruction of 3d ball/shuttle position by two image points from a single view. In: Lames M, Saupe D, Wiemeyer J (eds) International Symposium on Computer Science in Sport, pp 89–96

Sohn K, Lee H, Yan X (2015) Learning structured output representation using deep conditional generative models. Advances in neural information processing systems 28

Song X, Peng Y, Hu B, et al (2020) Characterization of the fine hand movement in badminton by a smart glove. Instrumentation Science & Technology 48(4):443–458. https://doi.org/10.1080/10739149.2020.1737814

Wang DB, Wen Y, Pan L, et al (2021a) Learning from noisy labels with complementary loss functions. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp 10111–10119

Wang W, Chang K, Chen T, et al (2020) Badminton coach ai: A badminton match data analysis platform based on deep learning. Physical Education Journal 53(2):201–213

Wang WY, Shuai HH, Chang KS, et al (2021b) Shuttlenet: Position-aware fusion of rally progress and player styles for stroke forecasting in badminton. arXiv preprint arXiv:211201044

Wei H, Xie R, Cheng H, et al (2022) Mitigating neural network overconfidence with logit normalization. In: International Conference on Machine Learning, PMLR, pp 23631–23644

Wei LY (2008) Parallel poisson disk sampling. Acm Transactions On Graphics (tog) 27(3):1–9

Winkler C, Worrall D, Hoogeboom E, et al (2019) Learning likelihoods with conditional normalizing flows. arXiv preprint arXiv:191200042

Wu J, Liu D, Guo Z, et al (2021) Tacticflow: Visual analytics of ever-changing tactics in racket sports. IEEE transactions on visualization and computer graphics 28(1):835–845

Yang G, Huang X, Hao Z, et al (2019) Pointflow: 3d point cloud generation with continuous normalizing flows. In: IEEE/CVF International Conference on Computer Vision, pp 4541–4550

Yang L, Zhang Z, Song Y, et al (2022) Diffusion models: A comprehensive survey of methods and applications. arXiv preprint arXiv:220900796

Zięba M, Przewięźlikowski M, Śmieja M, et al (2020) Regflow: Probabilistic flow-based regression for future prediction. arXiv preprint arXiv:201114620