Basketball Flow: Learning to Synthesize Realistic and Diverse Basketball GamePlays based on Strategy Sketches

Ming-Feng Kuo
National Yang Ming Chiao Tung University
Taiwan
sunnyday90154@gmail.com

Yu-Shuen Wang
National Yang Ming Chiao Tung University
Taiwan
yushuen@cs.nycu.edu.tw

ABSTRACT

In this study, we present BasketballFlow, a system designed to generate diverse basketball gameplays based on a pre-determined strategy sketch. A strategy sketch is a graphical representation that coaches use to outline their planned tactics, encompassing the projected routes of the ball and the offensive players. Despite the visual depiction of the offensive strategy, less experienced players might find it challenging to fully understand these tactics and often falter in their implementation due to interference from defensive players. Our system aims to remedy this by simulating different game scenarios that illustrate potential defensive maneuvers, thereby aiding these less experienced players in improving their success rate of tactical execution. BasketballFlow is composed of a variational generative adversarial network (VAEGAN) and a normalizing flow. The VAEGAN is tasked with producing highly accurate game scenarios, while the normalizing flow ensures a wide diversity in the simulated outcomes. Compared to other existing methods, BasketballFlow demonstrates superior proficiency in simulating a broad spectrum of gameplays while maintaining a lower Frèchet distance to real gameplays. The effectiveness of our BasketballFlow system is validated through our experimental results.

CCS CONCEPTS

• **Information systems** → **Multimedia content creation**; • **Human-centered computing** → **Interactive systems and tools**.

KEYWORDS

Basketball, sketch, generative networks, normalizing flows

ACM Reference Format:
Ming-Feng Kuo and Yu-Shuen Wang. 2023. Basketball Flow: Learning to Synthesize Realistic and Diverse Basketball GamePlays based on Strategy Sketches. In *ACM Multimedia Asia Workshops (MMAsia '23 Workshops), December 6–8, 2023, Tainan, Taiwan.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3611380.3630167

1 INTRODUCTION

Basketball coaches typically sketch out players' moves and pass routes on a strategy board to illustrate how to carry out offensive strategies. Professional players can use these examples to rehearse game situations on the strategy board, following the coach's plans in actual games. However, for players with less experience, their offense can be easily thrown off by defensive players from the other team, making it hard for them to stick to the original game plan. If these strategies could be shown in real time before the games start, players would gain a better understanding of the strategies and anticipate defensive reactions. This could ultimately boost their overall performance in games.

Although using game simulations to help players understand the reactions of offensive players isn't novel [5, 6, 16, 24], these methods typically rely on regression or generative adversarial networks [12], which often can't generate diverse gameplays that meet the requirements of the initial strategic sketch. Specifically, defensive players can disrupt an offensive strategy in many different ways, so simulating only one possible outcome can limit players' understanding of the strategy. This is particularly true for less experienced offensive players, who may be thrown off if the defenders do not react as they were taught in the simulation, potentially reducing the success rate of their offensive strategy.

We introduce BasketballFlow, a system that combines a variational generative adversarial network (VAEGAN) [27] and a normalizing flow [8, 9, 13, 21] to generate a variety of gameplays based on a given strategy sketch. Specifically, VAEGAN comprises a variational auto-encoder and a generative adversarial network, which uses the offensive strategy sketch to generate a full gameplay that includes the defensive player trajectories. To make the game simulation more accurate, we add a discriminator at the end of the decoder and train the network by optimizing the adversarial loss. We also use the reconstruction and fidelity losses to guide the network's training. While this VAEGAN aims to produce realistic gameplays, the task of the normalizing flow is to sample diverse latent representations for gameplay synthesis. Notably, normalizing flow is a bidirectional network composed of invertible layers, capable of transforming the latent distributions of the game data and a standard normal distribution in both directions. It exploits the benefits of sampling from a normal distribution to generate diverse gameplays that adhere to the conditions of the sketch.

To assess the effectiveness of BasketballFlow, we carry out a comparison study with various baseline methods. We specifically measure the Frèchet distance between the synthesized and actual gameplays, and carry out a visual analysis of the players' distributions on the court to evaluate fidelity. We also showcase the wide range of outcomes that our BasketballFlow system can produce. The experimental results confirm the effectiveness of our approach.

MMAsia '23 Workshops, December 6–8, 2023, Tainan, Taiwan

ACM ISBN 979-8-4007-0326-3/23/12…$15.00
https://doi.org/10.1145/3611380.3630167

2 RELATED WORK

Analytics and Synthesis of Basketball Game Data. Basketball analytics has long been a field of interest, with early research focusing on statistical data and mathematical computations to discern which stats directly or indirectly impacted basketball game outcomes [18, 20, 22, 23]. With more teams embracing data analytics techniques in recent years, this field has increasingly drawn research interest. For instance, Franks et al. [11] developed 'Counterpoints', a system that evaluates players' efficiency during both defense and offense. It provides heat maps showcasing zones where defensive players excel or struggle, offering a comprehensive view of player performances that can inform strategic decisions during games. On another note, Beshai et al. [3] introduced a tool that visualizes basketball shooting data, permitting users to compare player data via interactive charts, thereby making the exploration of basketball analytics more engaging and user-friendly.

The emergence of publicly accessible data from STATS SportVU, which records player positions and movements on the court as trajectories, has led to numerous applications. Seidl et al. [24] proposed an interactive offensive tactic diagram playback system that can simulate defensive behavior based on offensive tactic diagrams. In parallel, Chen et al. [6] and Hsieh et al. [16] used a generative adversarial network (GAN) [12] to model defender trajectories based on offensive data, effectively simulating complete basketball games. While the generated data were highly realistic, there was a lack of diversity in the game simulations. Building on their work, our study maintains the authenticity of basketball game simulations while addressing the insufficient diversity in simulation outcomes.

Generative Networks are a subset of machine learning models specifically designed to generate new data instances that closely mimic the original training data. Their applications are broad, extending from the synthesis of novel images to the creation of textual content. The spectrum of these models includes various types, such as generative adversarial networks (GANs) [2, 12, 15], variational autoencoders (VAEs) [10], Autoregressive models [14], normalizing flows [1, 7–9, 17], and diffusions [26]. Each of these types presents unique advantages and limitations. While the goal of this study is not to construct a universal generative model, we refer readers to [4] for the details of these networks.

3 BACKGROUNDS

Normalizing flows [8, 9, 13, 17] are a type of neural network designed to transform basic probability distributions into more intricate ones for generative modeling purposes. Starting with a random variable z_0 that has a defined probability density function $p_{z_0}(z)$, a normalizing flow employs a series of differentiable and invertible transformations f, such that $z_k = f_k \circ \ldots \circ f_1(z_0)$, with k denoting the number of applied transformations. This results in producing a sample z_k from the desired target distribution. Whereas earlier techniques in normalizing flow depended on the change of variables formula to ensure invertibility and differentiability, our approach integrates the neural ordinary differential equations (Neural ODE) [7] to connect the two distributions. In essence, a Neural ODE reframes the transformation from one layer to the next into a continuous trajectory. To achieve this, an ODE solver determines the continuous transformation mapping the inputs to the outputs. When z_t symbolizes the state of a data point at time t, the transformation follows the equation:

$$\frac{d_z(t)}{d_t} = f_\theta(z(t), t), \tag{1}$$

where f represents a neural network with parameters θ determining the state's derivative.

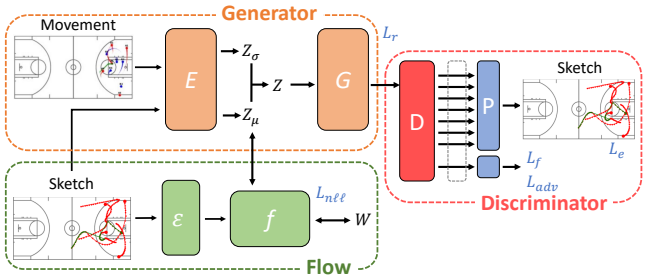

Figure 1: This figure illustrates the architecture of our basketball flow.

4 SYNTHESIS OF BASKETBALL GAMEPLAYS

We present a novel network architecture named BasketballFlow, designed to create realistic simulations of basketball gameplays based on strategy sketches. The architecture of this network consists of a VAEGAN paired with a normalizing flow, as depicted in Figure 1. The encoder's role is to convert gameplays x and the corresponding strategy sketches s into latent representations, denoted as $\mathbf{z}$, while the decoder's job is to reconstruct the full game data x, encompassing the trajectories of both offensive and defensive players. Due to the bottleneck of the encoded representation, where relevant information might be discarded, we employ a discriminator to guide the synthesis of gameplays. Alongside the VAEGAN, a normalizing flow is trained to comprehend the latent distribution $\mathbf{Z}$. This allows it to sample from a variety of representations that comply with an offensive strategy sketch.

4.1 Encoder

The BasketballFlow encoder E comprises three transformer layers, responsible for encoding the concatenated real gameplay x and the associated strategy sketch s into a latent representation z, where $z = E(x, s)$, as detailed in Figure 1. Given the time-series nature of both the trajectories and the sketch, we temporally segment the concatenated data into n patches, with each patch encapsulating 1 second of game time. Acknowledging the 24-second shot clock restriction inherent in basketball games, we set $n = 24$ in our model implementation, utilizing the 'zero' token in the patch to fill in the gaps if the offensive play does not span the full 24 seconds. For training the encoder, we attach a positional encoding to each patch before inputting it into the transformer. We also introduce a 'CLS' patch to the transformer to acquire comprehensive information about the offensive strategy.

We incorporate the variational autoencoder framework within BasketballFlow due to the limited expressiveness of the normalizing flows. Specifically, by moderating the irregularity of the encoded

distribution, we secure two primary advantages: 1) the task of modeling the latent distribution is simplified, given that a normalizing flow's purpose is to convert a standard distribution into a complex data distribution, and 2) the decoder maintains the ability to generate realistic basketball gameplays even when the sampled latent representation slightly deviates from the true latent distribution.

4.2 Decoder

The decoder G is composed of four transformer blocks and two multilayer perceptrons (MLPs). The first MLP escalates the dimensionality of the latent representation z from $\mathbb{R}^L$ to $\mathbb{R}^{L\times m}$, where m represents the number of patches. The m representations are subsequently concatenated with positional encodings and then input into the transformer blocks. The output is further processed through an MLP layer to reconstruct the comprehensive basketball gameplay. Specifically, we train the VAEGAN by optimizing the reconstruction loss for each generated gameplay:

$$\mathcal{L}_r(\hat{x}) = \left|x - \hat{x}\right|, \tag{2}$$

where $\hat{x} = G(z)$, and x is the actual gameplay. Given that the latent representation z does not incorporate defensive information, the decoder has the responsibility to synthesize defensive plays based on the encoded offensive strategy. Hence, we employ a deeper decoder G than the encoder E to simulate basketball gameplays. We also utilize a discriminator D, also comprised of transformer blocks, to assess the realism of the generated gameplays and guide the decoder's training. In our implementation, we use the WGAN-GP [15] method to train the VAEGAN as well as the discriminator. The loss can be articulated as:

$$\mathcal{L}_{adv} = E_{\hat{x}}[D(\hat{x})] - E_x[D(x)] + \lambda E_{\tilde{x}}\left[(||\nabla_{\tilde{x}} D(\tilde{x})||_2 - 1)^2\right], \tag{3}$$

where $\tilde{x}$ is an interpolation of a generative and a real game plays.

The discriminator's inputs are also segmented patch-wise, in line with the discriminator's backbone architecture. As the discriminator evaluates the realism of each trajectory segment independently, the synthesized gameplays could suffer from discontinuity artifacts. Therefore, akin to the strategy used in VITGAN [19], we feed overlapping patches into the discriminator to ensure the continuity of time series game data. In addition, training a GAN is notoriously challenging due to its inherent instability. To mitigate this, we employ feature loss optimization to assist in the training of BasketballFlow. Specifically, when the discriminator encodes the gameplays into features, referred to as $D_f(x)$ and $D_f(\hat{x})$, for the purpose of assessing their realism, we aim to have these encoded features be as similar as possible. The feature loss is calculated as:

$$\mathcal{L}_f(\hat{x}) = \left|D_f(x) - D_f(\hat{x})\right|. \tag{4}$$

Despite the potential for unexpected pathways to minimize this feature loss, we maximize the entropy of both $D_f(x)$ and $D_f(\hat{x})$ to inhibit the D_f function from degenerating. Specifically, our goal is for $D_f(\hat{x})$ to possess sufficient information to reconstruct the offensive strategy s by utilizing an MLP P, as depicted in Figure 1. The loss for this entropy maximization is formulated as:

$$\mathcal{L}_e = \left|s - P(D_f(\hat{x}))\right|. \tag{5}$$

Note that an alternative approach to prevent the degeneration of D_f might be the reconstruction of the entire gameplay x. However, our experimental findings recommend the reconstruction of strategy s as a more favorable option. We hypothesize that the intricacy within x might be overwhelming. Tasking an MLP with the complete gameplay reconstruction could compromise the ability of $D_f(\hat{x})$ to evaluate the realism of the synthesized gameplays effectively.

In addition to the aforementioned loss functions, we incorporate the dribbler loss, defender loss, ball passing loss, and acceleration loss when training the BasketballFlow. These functions, grounded in basketball expertise, align closely with the ones outlined by [16]. For a comprehensive breakdown of these functions, please refer to the supplemental material.

4.3 Normalizing Flows

The main objective of the normalizing flow within our framework is to obtain a latent representation from which the decoder can synthesize complete basketball gameplays. As previously mentioned, normalizing flows operate as bi-directional neural networks, wherein their operations during the training and testing phases are in opposite directions. During the training phase, the normalizing flow, denoted by f, is designed to map a latent distribution $\mathbf{Z}$ to a standard normal distribution $\mathbf{W}$ while integrating the encoded strategy sketch $\mathcal{E}(S)$, as illustrated in Figure 1. Specifically, the training procedure minimizes the negative log-likelihood loss, which is articulated as:

$$\begin{aligned}\mathcal{L}_{n\ell\ell} &= -\log p_{\mathbf{Z}|\mathcal{E}(S)}(\mathbf{z}|\mathcal{E}(s)) \\ &= -\log p_w(\mathbf{w}_{t_0}) + \int_{t_0}^{t_1} tr\left(\frac{\partial f}{\partial \mathbf{w}_t}\right) dt.\end{aligned} \tag{6}$$

Here, $\mathbf{w}_{t_0}$ is a noise vector drawn from the standard normal distribution, and $z \in \mathbf{Z}$ is a representation consistent with $\mathcal{E}(S)$. On the other hand, in the testing phase, the normalizing flow ingests a sample randomly drawn from a standard normal distribution and yields a latent representation z in line with the specified strategy condition. This representation z is computed as:

$$\mathbf{z} = \mathbf{w}_{t_0} + \int_{t_0}^{t_1} f_\theta(t, \mathcal{E}(s), \mathbf{w}_t) dt. \tag{7}$$

Note that we train the normalizing flow f to model the distribution $\mathbf{Z}_\mu$ (Figure 1), rather than $\mathbf{Z}$. The underlying rationale is that $\mathbf{Z}_\mu$ offers a more constrained representation compared to $\mathbf{Z}$. By subjecting the normalizing flow to this more challenging task, we anticipate enhancing its capacity to faithfully capture the latent distribution, which is essential for simulating basketball gameplay.

The normalizing flow's network structure comprises a moving batch normalizing layer [25], four concatsquash layers [13, 25], and another batch normalization layer. The concatsquash layer is mathematically expressed as:

$$CCS(t, \mathbf{c}, \mathbf{u}) \quad = \sigma_1((Y_u\mathbf{u} + b_u) \times gate + bias).$$

Here, $gate = \sigma_2(Y_{tt}t + Y_{tc}\mathbf{c} + b_t)$ and $bias = (Y_{bt}t + Y_{bc}\mathbf{c} + b_b t)$. The terms $Y_u, Y_{tt}, Y_{tc}, Y_{bt}, Y_{bc}, b_u, b_t, b_b$ are learnable parameters. The activation functions, σ_1 and σ_2 correspond to *tanh* and *sigmoid*.

4.4 Implementation Details

We trained BasketballFlow, encompassing both the VAEGAN and normalizing flows, using the Adam optimizer. The batch size, learning rate, parameters β_1 and β_2 used in the optimizer were set to 64, 10^{-4}, 0.9, and 0.999, respectively. The ODE solver's error tolerance

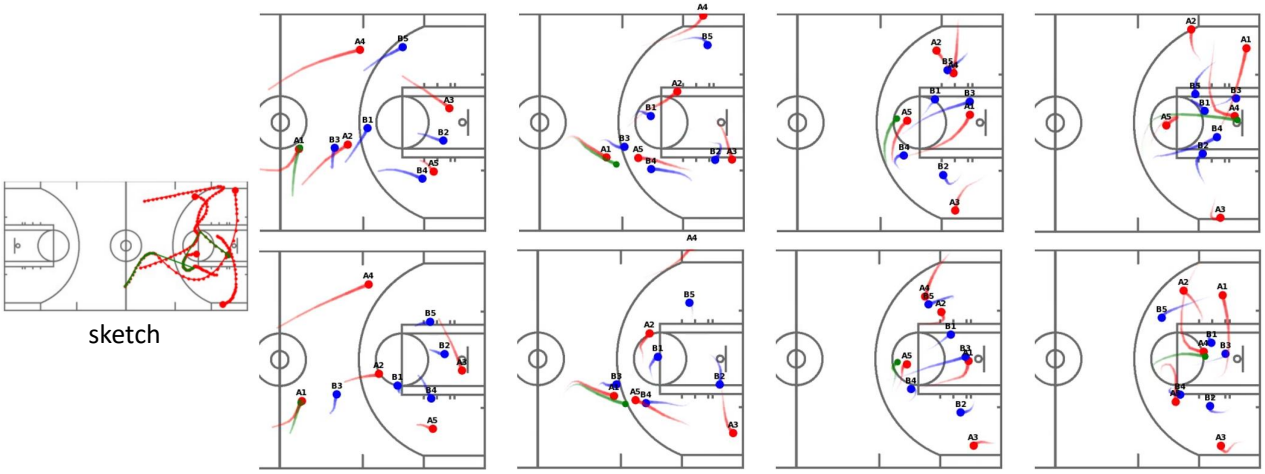

Figure 2: We show the input sketch on the left, and the generated game plays on the right. In the results, red and blue trajectories indicate offensive and defensive plays, and the game plays are depicted from left to right. As indicated, basketball flow generates two different realistic game plays, although the input is the same.

	BasketballGAN	BasketballFlow	CVAEGAN
Frèchet Distance	25.584	6.604	7.394

Table 1: The numbers indicate the similarity of the generated and actual basketball gameplays. The lower values are better.

was set to 10^{-5}. The training process spanned 1000 epochs, and the model achieving the best validation performance was selected for basketball gameplay generation. The training was carried out on an AMD R9 CPU with an NVIDIA GeForce RTX 4090 graphics card, taking approximately 8 to 10 hours to complete.

5 RESULTS AND EVALUATIONS

5.1 DataSets

Our data is sourced from SportsVU, which records the movements of ten players and the ball during a match. The recorded data pertains to the NBA's 2015-2016 season, encompassing approximately 600 basketball games. By combining this with the official play-by-play data, we partition these trajectory data into different gameplays. Each gameplay begins when the offense brings the ball past half-court and ends after a shot, regardless of whether points were scored. We also reduced the original data to 5 frames per second for training. Additionally, we simplify the offensive trajectories using the Ramer-Douglas-Peucker algorithm and then apply bezier curves to generate smooth strategy sketches [16]. This process achieves more effective and cost-efficient data generation for training.

5.2 Quality Comparison with Baseline Methods

We evaluated the realism of gameplays produced by BasketballFlow by benchmarking it against BasketballGAN [16] and CVAEGAN. BasketballGAN, a leading competitor, also synthesizes gameplay based on strategy sketches. CVAEGAN, on the other hand, shares similarities with our BasketballFlow but doesn't incorporate the normalizing flow element. For a balanced comparison, we employed the same backbones – specifically, the transformer – across the training of these three generative networks. After training, we computed the Frèchet distances between genuine and generated gameplays. Results, presented in Table 1, indicate superior quality from both BasketballFlow and CVAEGAN over BasketballGAN, as evidenced by their reduced Frèchet distances. This superiority might stem from the inherent challenge in conditioning GANs, as compared to non-conditioned models [1]. Moreover, while both models effectively captured the gameplay representation distribution, BasketballFlow demonstrated its ability to produce diverse gameplays from a single strategy sketch, showcased in Figure 2.

6 CONCLUSIONS

We present BasketballFlow, an innovative system designed to generate realistic and varied basketball game plays based on a provided sketch. BasketballFlow combines a variational autoencoder and a conditional normalizing flow. The variational autoencoder translates strategy sketches into latent forms and then reconstructs these forms into related basketball plays. Concurrently, the conditional normalizing flow captures the latent distribution of basketball plays, enabling the generation of varied plays from a single input sketch. The successful synthesis of realistic and diverse gameplays demonstrates the efficacy of blending variational autoencoders with normalizing flows. Moving forward, we aim to have coaches and players test the system to assess if BasketballFlow can enhance the efficacy of offensive strategies in their future competitions.

ACKNOWLEDGMENTS

We thank the reviewers for their insightful comments. This work is partially supported by the National Science and Technology Council (NSTC), under Contract: 112-2425-H-A49 -001 - and 111-2221-E-A49 -129 -MY3, Industrial Technology Research Institute (ITRI), and by the Higher Education Sprout Project of the National Yang Ming Chiao Tung University and Ministry of Education (MOE), Taiwan.

REFERENCES

[1] Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. 2021. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics* 40, 3 (2021), 1–21.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875* (2017).

[3] Peter Beshai. 2014. Buckets: Basketball Shot Visualization. *University of British Columbia, published Dec* (2014), 547–14.

[4] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. 2021. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence* (2021).

[5] Chieh-Yu Chen, Wenze Lai, Hsin-Ying Hsieh, Yu-Shuen Wang, Wen-Hsiao Peng, and Jung-Hong Chuang. 2018. Adversarial generation of defensive trajectories in basketball games. In *2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE, 1–1.

[6] Chieh-Yu Chen, Wenze Lai, Hsin-Ying Hsieh, Wen-Hao Zheng, Yu-Shuen Wang, and Jung-Hong Chuang. 2018. Generating defensive plays in basketball games. In *Proceedings of the 26th ACM international conference on Multimedia*. 1580–1588.

[7] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366* (2018).

[8] Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516* (2014).

[9] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2016. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803* (2016).

[10] Carl Doersch. 2016. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908* (2016).

[11] Alexander Franks, Andrew Miller, Luke Bornn, and Kirk Goldsberry. 2015. Counterpoints: Advanced defensive metrics for nba basketball. In *9th Annual MIT Sloan Sports Analytics Conference, Boston, MA*.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

[13] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. 2018. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367* (2018).

[14] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. 2014. Deep autoregressive networks. In *International Conference on Machine Learning*. PMLR, 1242–1250.

[15] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. 2017. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*. 5767–5777.

[16] Hsin-Ying Hsieh, Chieh-Yu Chen, Yu-Shuen Wang, and Jung-Hong Chuang. 2019. BasketballGAN: Generating Basketball Play Simulation Through Sketching. *arXiv preprint arXiv:1909.07088* (2019).

[17] Diederik P Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039* (2018).

[18] Justin Kubatko, Dean Oliver, Kevin Pelton, and Dan T Rosenbaum. 2007. A starting point for analyzing basketball statistics. *Journal of Quantitative Analysis in Sports* 3, 3 (2007).

[19] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. 2021. Vitgan: Training gans with vision transformers. *arXiv preprint arXiv:2107.04589* (2021).

[20] Dean Oliver. 2004. *Basketball on paper: rules and tools for performance analysis*. Potomac Books, Inc.

[21] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2019. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762* (2019).

[22] Jaime Sampaio, Eric J Drinkwater, and Nuno M Leite. 2010. Effects of season period, team quality, and playing time on basketball players' game-related statistics. *European Journal of Sport Science* 10, 2 (2010), 141–149.

[23] Jaime Sampaio, Manuel Janeira, Sergio Ibáñez, and Alberto Lorenzo. 2006. Discriminant analysis of game-related statistics between basketball guards, forwards and centres in three professional leagues. *European journal of sport science* 6, 3 (2006), 173–178.

[24] Thomas Seidl, Aditya Cherukumudi, Andrew Hartnett, Peter Carr, and Patrick Lucey. 2018. Bhostgusters: Realtime Interactive Play Sketching with Synthesized NBA Defenses.(2018). (2018).

[25] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. 2019. Pointflow: 3d point cloud generation with continuous normalizing flows. In *IEEE/CVF International Conference on Computer Vision*. 4541–4550.

[26] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2022. Diffusion models: A comprehensive survey of methods and applications. *Comput. Surveys* (2022).

[27] Xianwen Yu, Xiaoning Zhang, Yang Cao, and Min Xia. 2019. VAEGAN: A Collaborative Filtering Framework based on Adversarial Variational Autoencoders.. In *IJCAI*. 4206–4212.