Generative Multi-Agent Behavioral Cloning

Eric Zhan [1] **Stephan Zheng** [1] **Yisong Yue** [1] **Patrick Lucey** [2]

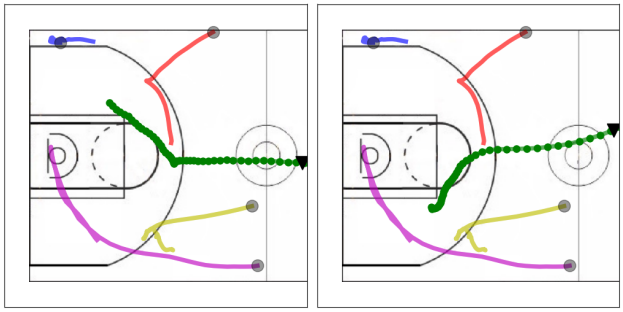
Abstract

We propose and study the problem of generative multi-agent behavioral cloning, where the goal is to learn a generative multi-agent policy from pre-collected demonstration data. Building upon advances in deep generative models, we present a hierarchical policy framework that can tractably learn complex mappings from input states to distributions over multi-agent action spaces. Our framework is flexible and can incorporate high-level domain knowledge into the structure of the underlying deep graphical model. For instance, we can effectively learn low-dimensional structures, such as long-term goals and team coordination, from data. Thus, an additional benefit of our hierarchical approach is the ability to plan over multiple time scales for effective long-term planning. We showcase our approach in an application of modeling team offensive play from basketball tracking data. We show how to instantiate our framework to effectively model complex interactions between basketball players, and generate realistic multi-agent trajectories of basketball gameplay over long time periods. We validate our approach using both quantitative and qualitative evaluations, including a user study comparison conducted with professional sports analysts.

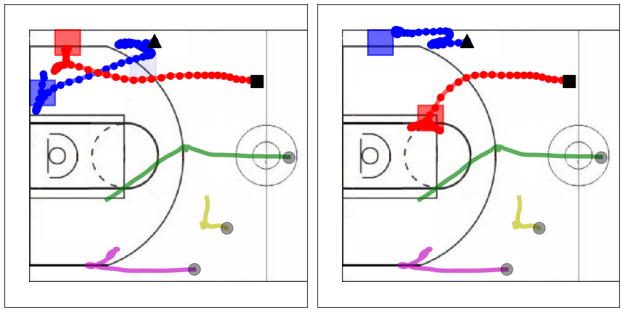
1. Introduction

The ongoing explosion of recorded tracking data is enabling the study of fine-grained behavior in many domains. Examples include sports (Miller et al., 2014; Yue et al., 2014; Zheng et al., 2016; Le et al., 2017), video games (Ross et al., 2011), video & motion capture (Suwajanakorn et al., 2017; Taylor et al., 2017; Xue et al., 2016), navigation & driving (Ziebart et al., 2009; Zhang & Cho, 2017; Li et al., 2017), laboratory animal behaviors (Johnson et al., 2016; Eyjolfsdottir et al., 2017), and tele-operated robotics (Abbeel & Ng, 2004; Lin et al., 2006). One popular research direction, which we also study, is to learn a policy that imitates demonstrated behavior, also known as imitation learning (Abbeel & Ng, 2004; Ziebart et al., 2008; Daumé et al., 2009; Ross et al., 2011; Ho & Ermon, 2016). The arguably simplest form of imitation learning is known as behavioral cloning, where the goal is to mimic a batch of pre-collected demonstration data, e.g. from human experts.

In this paper, we are interested in domains that involve multiple agents whose underlying policies are inherently non-deterministic. Consider Figure 1, which depicts several scenarios of offensive player behavior in basketball. On



(a) Players have multi-modal behavior. For instance, in the above examples the green player (▼) moves to either the top or bottom.



(b) Players are coordinated. **Left**: The red player (■) moves to the top-left corner of the court. **Right**: The blue player (▲) moves to the top-left corner so the red player goes elsewhere. Knowing each other's macro-goals (boxes) is crucial for team coordination.

Figure 1. Player behaviors for offense in basketball (ball not shown). Black symbols indicate starting positions. An interactive demo can be found at: `http://basketball-ai.com/`.

[1]California Institute of Technology, Pasadena, CA, USA [2]STATS, Chicago, IL, USA. Correspondence to: Eric Zhan <ezhan@caltech.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

offense, players typically behave non-deterministically and the distribution over possible trajectories is multi-modal (Figure 1a). Furthermore, all players display coherent team coordination over long time horizons; in Figure 1b, we observe that knowing each other's macro-goals (boxes that represent long-term intent) allows the red and blue players to avoid going to the same location. We aim to learn a policy that can capture all these aspects. One challenge is that the space of multi-agent trajectories is naively exponentially large. Beyond team sports, other domains include modeling social interactions in laboratory animals (Eyjolfsdottir et al., 2017), team-based video games, and music generation (each instrument is an "agent") (Thickstun et al., 2017).

We thus study the problem of *generative multi-agent behavioral cloning*, whereby the desired policy must map input states to distributions over multi-agent action spaces. We employ the simplest form of imitation learning, behavioral cloning, in order to keep the research contribution focused; one could also employ more advanced imitation learning techniques to train our policy class. Unlike conventional behavior cloning, generative behavior cloning implies that the desired policy does not necessarily perfectly mimic the demonstrations, but rather recovers the (latent) generating distribution from which the demonstrations were sampled. From the multi-agent perspective, we must also ensure that agents behave coherently over long time horizons.

We present a hierarchical policy class that builds upon recent advances in deep generative models and integrates them into the behavioral cloning setting. Our framework has three salient properties: 1) it is straightforward to incorporate domain knowledge such as macro-goals into the structure of the underlying deep graphical model; 2) the hierarchical decomposition allows for tractably modeling coordination between multiple agents; 3) the hierarchical decomposition also allows for generating plans at multiple time scales, which enables effective long-term planning as a by product. Our policy class is also compatible with existing variational methods for training deep generative models, which we can adapt to the behavioral cloning setting.

While there has been some work in multi-agent imitation learning (Chernova & Veloso, 2007; Le et al., 2017) and imitation learning with stochastic polices (Ziebart et al., 2008; Ho & Ermon, 2016; Li et al., 2017), no previous work has focused on learning generative polices as a core research direction, not to mention simultaneously addressing generative and multi-agent imitation learning. For instance, all the experiments in (Ho & Ermon, 2016) lead to policies with highly peaked distributions, and the stochastics are essentially used to help with learning (i.e., not get stuck in local optima). In contrast, we are interested in settings where the desired behavior is a complex multi-modal distribution that reflects the non-determinism inherent in the true policy that generated the demonstrations.

We showcase our approach in an application on modeling team offense in basketball. We show that our approach can learn to generate high-quality trajectories of multi-agent gameplay. Our approach also allows for conditional inference such as grounding the macro-goal variables to manipulate agent behavior. We validate our approach both quantitatively and qualitatively, including a user study comparison with professional sports analysts, and demonstrate significant improvements over standard baselines. To summarize, our contributions are as follows:

- We introduce a novel problem setting for behavioral cloning, where the optimal agent policy is probabilistic and multi-modal in nature, and must reason over the behavior of multiple interacting agents.
- We propose a hierarchical generative policy class that can use macro-goals to capture coordination between agents, as well as long-term intent.
- We derive a principled learning approach that utilizes amortized variational inference.
- We demonstrate empirically that our model generates higher quality behavior than conventional baselines.

2. Related Work

Imitation Learning & Behavioral Cloning. Broadly speaking, one can decompose imitation learning along two dimensions. The first dimension spans learning to mimic batched pre-collected demonstrations, also known as behavioral cloning (Abbeel & Ng, 2004; Ziebart et al., 2008; Ho & Ermon, 2016), versus actively querying an oracle for feedback during the learning process (Daumé et al., 2009; Ross et al., 2011). Along this dimension, behavioral cloning is often regarded as the simplest form of imitation learning.

The second dimension spans learning value functions, also known as inverse reinforcement learning (Abbeel & Ng, 2004; Ziebart et al., 2008), versus direct policy learning (Daumé et al., 2009; Ross et al., 2011; Ho & Ermon, 2016). In value function learning, one assumes that there exists an unknown value function, and demonstrations are rational with respect to that value function (i.e., the demonstrated actions maximize the value). In a sense, the value function approach imposes a certain model structure, whereas direct policy learning is essentially model-free.

Within this landscape of imitation learning research, our work can be viewed as learning via behavioral cloning while imposing new forms of model structure. By learning via behavioral cloning, we simplify the complexity along the first dimension to focus our research on the modeling challenges that arise from learning generative multi-agent policies.

As mentioned in the introduction, there have been some prior work in multi-agent imitation learning (Chernova & Veloso, 2007; Le et al., 2017) as well as learning stochastic policies (Ziebart et al., 2008; Ho & Ermon, 2016; Li et al., 2017). As also discussed before, no previous work has focused on learning generative polices as a core research direction, not to mention simultaneously addressing generative and multi-agent imitation learning. For instance all the experiments in (Ho & Ermon, 2016) led to highly peaked distributions, while (Li et al., 2017) captures multi-modal distributions by assigning each demonstration to one of a fixed number of experts and learning policies for each expert that turn out to be unimodal. In contrast, we are interested in learning mappings between input states and complex multi-modal distributions over multi-agent action spaces.

Long-term planning Another issue that our work addresses is long-term planning. In this regard, the closest prior work is (Zheng et al., 2016), which also reasoned over long sequences using macro-goals. However, their approach was only for a single agent and used relatively simple stochastics. Beyond imitation learning, designing hierarchical policies is a topic of both historical and contemporary interest in reinforcement learning (Dayan & Hinton, 1993; Sutton et al., 1999; Kulkarni et al., 2016). From that perspective, one can view our work as developing generative policies to capture complex non-deterministic behaviors.

Deep generative models. The study of deep generative models is an increasingly popular research area, due to their ability to inherit both the flexibility of deep learning and the probabilistic semantics of generative models. Broadly speaking, there are two ways that one can incorporate stochastics into deep models. The first approach is to model an explicit distribution over actions in the output layer, e.g., via logistic regression (Chen et al., 2015; Oord et al., 2016a;b; Zheng et al., 2016; Eyjolfsdottir et al., 2017). The second approach is to use deep neural nets to define a transformation from a simpler distribution (e.g., unit Gaussian) to the distribution of interest (Goodfellow et al., 2014; Kingma & Welling, 2014; Rezende et al., 2014). This second approach cannot explicitly specify the output distribution (i.e., one can typically only sample from the implicitly defined distribution), but can more readily be extended to incorporate additional structure, such as a hierarchy of random variables (Ranganath et al., 2016) or dynamics (Johnson et al., 2016; Chung et al., 2015; Krishnan et al., 2017; Fraccaro et al., 2016). Our framework can incorporate both variants as appropriate.

3. Background on Deep Generative Models

We give a brief overview of deep generative models used to instantiate our policy class in our experiments. In particular, we will briefly review recurrent neural networks (RNNs), variational autoencoders (VAEs) and variational RNNs.

Recurrent neural networks. A RNN models the conditional probabilities in Eq. (15) with a hidden state $\mathbf{h}_t$ that summarizes the information in the first $t-1$ timesteps:

$$p_\theta(\mathbf{x}_t|\mathbf{x}_{<t}) = \varphi(\mathbf{h}_{t-1}), \tag{1}$$

$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}), \tag{2}$$

where φ maps the hidden state to a probability distribution over states and f is a deterministic function such as LSTMs (Hochreiter & Schmidhuber, 1997) or GRUs (Cho et al., 2014). RNNs with simple output distributions often struggle to capture highly variable and structured sequential data. Recent work in sequential generative models aim to address this issue by injecting stochastic latent variables into the model and using amortized variational inference to infer the latent variables from the data (Fraccaro et al., 2016).

Variational Autoencoders. A variational autoencoder (VAE) (Kingma & Welling, 2014) is a generative model for non-sequential data that injects latent variables $\mathbf{z}$ into the joint distribution $p_\theta(\mathbf{x}, \mathbf{z})$ and introduces an inference network parametrized by ϕ to approximate the posterior $q_\phi(\mathbf{z} \mid \mathbf{x})$. The learning objective is to maximize the evidence lower-bound (ELBO) of the log-likelihood with respect to the model parameters θ and ϕ:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] + D_{KL}(q_\phi(\mathbf{z} \mid \mathbf{x})||p_\theta(\mathbf{z})) \tag{3}$$

The first term is known as the reconstruction term and can be approximated with Monte Carlo sampling. The second term is the Kullback-Leibler divergence between the approximate posterior and the prior, and can be evaluated analytically (i.e. if both distributions are Gaussian with diagonal covariance). The inference model $q_\phi(\mathbf{z} \mid \mathbf{x})$, generative model $p_\theta(\mathbf{x} \mid \mathbf{z})$, and prior $p_\theta(\mathbf{z})$ are often implemented with neural networks.

Variational RNNs. VRNNs combine VAEs and RNNs by conditioning the VAE on a hidden state $\mathbf{h}_t$ (see Figure 3a):

$$p_\theta(\mathbf{z}_t|\mathbf{x}_{<t}, \mathbf{z}_{<t}) = \varphi_{\text{prior}}(\mathbf{h}_{t-1}) \quad \text{(prior)} \tag{4}$$

$$q_\phi(\mathbf{z}_t|\mathbf{x}_{\leq T}, \mathbf{z}_{<t}) = \varphi_{\text{enc}}(\mathbf{x}_t, \mathbf{h}_{t-1}) \quad \text{(inference)} \tag{5}$$

$$p_\theta(\mathbf{x}_t|\mathbf{z}_{\leq t}, \mathbf{x}_{<t}) = \varphi_{\text{dec}}(\mathbf{z}_t, \mathbf{h}_{t-1}) \quad \text{(generation)} \tag{6}$$

$$\mathbf{h}_t; = f(\mathbf{x}_t, \mathbf{z}_t, \mathbf{h}_{t-1}). \quad \text{(recurrence)} \tag{7}$$

VRNNs are also trained by maximizing the ELBO, which in this case can be interpreted as the sum of the ELBOs over each timestep of the sequence:

$$\mathbb{E}_{q_\phi(\mathbf{z}_{\leq T}|\mathbf{x}_{\leq T})}\Bigg[\sum_{t=1}^{T} \log p_\theta(\mathbf{x}_t \mid \mathbf{z}_{\leq T}, \mathbf{x}_{<t}) - D_{KL}\Big(q_\phi(\mathbf{z}_t \mid \mathbf{x}_{\leq T}, \mathbf{z}_{<t})||p_\theta(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t})\Big)\Bigg] \tag{8}$$

Note that the prior distribution of latent variable $\mathbf{z}_t$ depends on the history of states and latent variables (Eq. (4)). This temporal dependency of the prior allows VRNNs to model complex sequential data like speech and handwriting.

4. Generative Multi-Agent Behavioral Cloning

We are given training data $\mathcal{D}$ with demonstrations of K (ordered) cooperative agents over fixed time horizon T. We first establish some notation:

- Let $\mathcal{X}, \mathcal{A}$ denote the state, action space.
- Let $\mathbf{s}_{\leq T} = \{\mathbf{s}_t\}_{1 \leq t \leq T}$ denote a demonstration, where $\mathbf{s}_t = (\mathbf{x}_t, \mathbf{a}_t) = (\{\mathbf{x}_t^k\}_{\text{agents } k}, \{\mathbf{a}_t^k\}_{\text{agents } k})$. $\mathbf{x}_t^k \in \mathcal{X}$, $\mathbf{a}_t^k \in \mathcal{A}$ are the state, action of agent k at time t.
- Let $\tau_t = \{(\mathbf{x}_u, \mathbf{a}_u)\}_{1 \leq u \leq t}$ denote the history of state-action pairs.
- Let $\pi_\theta(\mathbf{x}_t, \tau_{t-1})$ denote a (multi-agent) stochastic policy parametrized by θ that samples actions from the probability distribution $p_\theta(\mathbf{a}_t | \mathbf{x}_t, \tau_{t-1})$.
- Let π_E denote the (multi-agent) expert stochastic policy that generated the data $\mathcal{D}$, and $\mathbf{s}_{\leq T} \sim \pi_E$ to denote that $\mathbf{s}_{\leq T}$ was generated from policy π_E.
- Let $\mathcal{M}(\mathbf{x}_t, \mathbf{a}_t)$ denote a (possibly probabilistic) transition function on states: $\mathbf{x}_{t+1} \sim p_{\mathcal{M}}(\mathbf{x}_{t+1} | \mathbf{x}_t, \mathbf{a}_t)$.

4.1. Learning Objective

The goal in behavioral cloning is to find a policy that behaves like the expert demonstrations $\mathcal{D}$, e.g., by solving an optimization problem with respect to a loss function ℓ:

$$\begin{aligned}\theta^* &= \text{argmin}_\theta \, \mathbb{E}_{\mathbf{s}_{\leq T} \sim \pi_E} \left[\sum_{t=1}^{T} \ell\big(\mathbf{a}_t, \pi_\theta(\mathbf{x}_t, \tau_{t-1})\big) \right] \\ &\approx \text{argmin}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \ell\big(\mathbf{a}_t, \pi_\theta(\mathbf{x}_t, \tau_{t-1})\big) \end{aligned} \tag{9}$$

Simplifying assumptions. In our problem setting of human motion tracking, the transition $\mathcal{M}$ is deterministic:

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \mathbf{a}_t. \tag{10}$$

As such, we can simplify the problem by absorbing $\mathcal{M}$ into the policy π_θ and predict $\mathbf{x}_t + \mathbf{a}_t$ directly:

- Since actions are now implicitly tied into the state, we can denote each demonstration as $\mathbf{x}_{\leq T} = \{\mathbf{x}_t\}_{1 \leq t \leq T}$, and the history of states as $\tau_t = \mathbf{x}_{\leq t}$.
- Similarly, the stochastic policy $\pi_\theta(\mathbf{x}_t, \tau_{t-1}) = \pi_\theta(\tau_t)$ now samples the next state directly from $p_\theta(\mathbf{x}_{t+1} | \tau_t)$. The policy is implicitly sampling an action.

For example, the initial state $\mathbf{x}_1$ of the green player in Figure 1a is marked by ▼. The player's action $\mathbf{a}_1$ is to move left, which results in the next state $\mathbf{x}_2 = \mathbf{x}_1 + \mathbf{a}_1$.

Our learning objective becomes:

$$\theta^* = \text{argmin}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \ell\big(\mathbf{x}_t, \pi_\theta(\tau_{t-1})\big) \tag{11}$$

If the policy is deterministic, i.e. probability peaked around a single action, then we can define ℓ to be an L_2 reconstruction loss:

$$\ell\big(\mathbf{x}_t, \pi_\theta(\tau_{t-1})\big) = \|\mathbf{x}_t - \pi_\theta(\tau_{t-1})\|_2^2. \tag{12}$$

For a stochastic policy that returns parameters of a distribution, ℓ can be the negative log-likelihood and we can re-write the objective as a maximization problem:

$$\ell\big(\mathbf{x}_t, \pi_\theta(\tau_{t-1})\big) = -\log p_\theta\big(\mathbf{x}_t | \tau_{t-1}\big), \tag{13}$$

$$\theta^* = \text{argmax}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \log p_\theta\big(\mathbf{x}_t | \tau_{t-1}\big). \tag{14}$$

Eq. (14) is exactly the objective for sequential generative models that maximize the log-likelihood of data $\mathcal{D} = \{\mathbf{x}_{\leq T}\}$ by factorizing the joint distribution of the sequence:

$$\begin{aligned}\theta^* &= \text{argmax}_\theta \sum_{\mathcal{D}} \log p_\theta(\mathbf{x}_{\leq T}) \\ &= \text{argmax}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \log p_\theta(\mathbf{x}_t | \mathbf{x}_{<t}). \end{aligned} \tag{15}$$

As we empirically verify in Section 6, models trained with Eq. (15) have difficulty learning representations of the data that generalize well over long time horizons. Our solution is to introduce macro-goals (as seen in Figure 1) as an effective means in learning low-dimensional representations of the data that extend in time and space for multiple agents.

5. Generative Multi-Agent Policy Class

We now present our generative multi-agent policy class. Our policy class is hierarchical and incorporates macro-goals within the higher layer of the hierarchy.

We first assume conditional independence between the agent states $\mathbf{x}_t^k$ given history $\tau_{t-1} = \mathbf{x}_{<t}$. This lets us decompose the loss ℓ and policy π_θ in Eq. (11):

$$\begin{aligned}\theta^* &= \text{argmin}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \ell\big(\mathbf{x}_t, \pi_\theta(\tau_{t-1})\big) \\ &= \text{argmin}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \sum_{k=1}^{K} \ell\big(\mathbf{x}_t^k, \pi_\theta^k(\tau_{t-1})\big) \\ &= \text{argmax}_\theta \sum_{\mathcal{D}} \sum_{t=1}^{T} \sum_{k=1}^{K} \log p_\theta^k(\mathbf{x}_t^k | \mathbf{x}_{<t}) \end{aligned} \tag{16}$$

For our experiments, We model our policies π_θ^k with VRNNs using stochastic latent variables $\mathbf{z}_t^k$ for each agent:

$$\pi_\theta^k(\tau_{t-1}) \sim p_\theta^k(\mathbf{x}_t^k|\mathbf{x}_{<t}) = \varphi^k(\mathbf{z}_t^k, \mathbf{h}_{t-1}^k), \quad (17)$$

$$\mathbf{h}_t^k = f^k(\mathbf{x}_t^k, \mathbf{z}_t^k, \mathbf{h}_{t-1}^k). \quad (18)$$

5.1. Hierarchical Policy with Macro-goals

Next, we introduce macro-goal variables $\mathbf{g}_t$ for policies π_θ^k:

$$\pi_\theta^k(\tau_{t-1}) \sim p_\theta(\mathbf{x}_t^k|\mathbf{x}_{<t}) = \varphi^k(\mathbf{z}_t^k, \mathbf{h}_{t-1}^k, \mathbf{g}_t) \quad (19)$$

The motivations behind introducing a hierarchical structure with macro-goals are: 1) to provide a tractable way to capture coordination between agents; and 2) to encode long-term intents of agents and enable long-term planning at a higher-level timescale. The space of all possible combinations of trajectories for multiple agents is exponentially large. A hierarchical decomposition using macro-goals can compactly represent some low-dimension structure in the trajectory space that is easier to learn and represent.

Figure 2 illustrates the macro-goals for two players. The blue player moves from the top-right to bottom-left of the court, while the green player moves from the bottom-right to the middle-left. At each timestep t, the player moves towards the current macro-goal $\mathbf{g}_t^k$. The macro-goal changes only once in 50 timesteps.

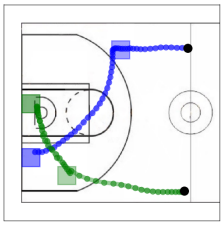
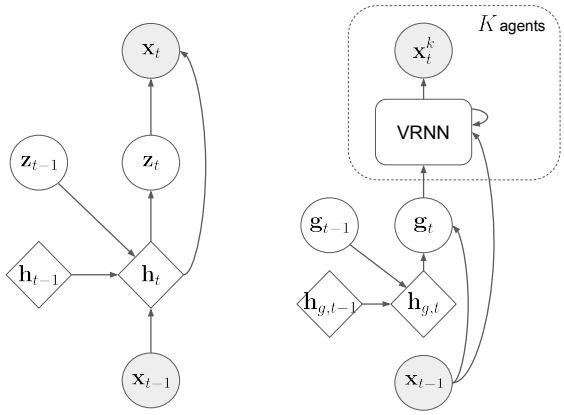


Figure 2. Showing macro-goals (boxes) for two players.

Our modeling assumptions for macro-goals are:

- subsequences of agent states $\{\mathbf{x}_t^k\}$ in an episode $[t_1, t_2]$ lead to some macro-goal $\mathbf{g}_t^k$,
- the start and end times of an episode can vary,
- macro-goals change slowly over time relative to the agent states: $d\mathbf{g}_t^k/dt \ll 1$.
- due to their reduced dimensionality, we can model (near-)arbitrary dependencies between macro-goals (e.g., coordination) via black box learning.

Capturing coordination. The last assumption motivates us to capture coordination by learning a single black-box model to predict the macro-goals of all the players. In contrast, the models in Eq. (17) have separate hidden states $\mathbf{h}_t^k$, so the agents are uncoordinated because their policies are independent. We induce coordination between agents by sharing the macro-goal variable $\mathbf{g}_t$ among all agents in Eq. (19). Intuitively, this means that all agents share a common macro-goal. In our case where the macro-goal decomposes for individual agents, each agent knows the macro-goals of all other agents. This can help the players move more cohesively (see Figure 1b).



(a) VRNN (b) Our model

Figure 3. Computation graph of VRNN (Chung et al., 2015) and our model. Circles are stochastic variables whereas diamonds are deterministic states. Macro-goal $\mathbf{g}_t$ is shared among all agents.

Capturing long-term intent. The above assumptions also imply that macro-goals can encode long-term intent. We aim to leverage these macro-goals to ensure more consistent behavior over long horizons.

5.2. Modeling Macro-goals

Ultimately, we want our model to generate macro-goals rather than depend on conditional input, so we train a policy modeled by a RNN to sample macro-goals:

$$p(\mathbf{g}_t|\mathbf{g}_{<t}) = \varphi_g(\mathbf{h}_{g,t-1}, \mathbf{x}_{t-1}), \quad (20)$$

$$\mathbf{h}_{g,t} = f_g(\mathbf{g}_t, \mathbf{h}_{g,t-1}). \quad (21)$$

We also choose to condition the macro-goal policy on the previous states $\mathbf{x}_{t-1}$ in Eq. (20). Then we generate trajectories by first sampling a macro-goal $\mathbf{g}_t$, and then sampling $\mathbf{x}_t^k$ conditioned on $\mathbf{g}_t$ (see Figure 3b for full graphical model). In addition, macro-goals that are interpretable and can be manipulated to control an agent's behavior (see Section 6.3).

In this paper, we let $\mathbf{g}_t$ be the concatenation of macro-goals $\mathbf{g}_t^k$ for each agent k. We further assume that macro-goals are available in the demonstration data, or can be easily extracted. Interesting future directions include richer macro-goals spaces and learning from less supervision.

5.3. Multi-stage Training.

Our agent and macro-goal policies can be trained independently. For our macro-goal policy, we directly maximize the log-likelihood of macro-goals $\mathbf{g}_{\leq T}$. For each of our agent

policies, we maximize the ELBO for VRNNs from Eq (8):

$$\mathbb{E}_{q^k(\mathbf{z}^k_{\leq T}|\mathbf{x}^k_{\leq T},\mathbf{g}_{\leq T})}\Bigg[\sum_{t=1}^{T}\log p^k_\theta(\mathbf{x}^k_t|\mathbf{z}^k_{\leq T},\mathbf{x}^k_{<t},\mathbf{g}_{\leq T}) - D_{KL}\Big(q^k_\phi(\mathbf{z}^k_t|\mathbf{x}^k_{\leq T},\mathbf{z}^k_{<t},\mathbf{g}_{\leq T})\Big\|p^k_\theta(\mathbf{z}^k_t|\mathbf{x}^k_{<t},\mathbf{z}^k_{<t},\mathbf{g}_{<t})\Big)\Bigg]. \tag{22}$$

6. Experiments

We apply our approach to modeling team basketball gameplay on offense (team with possession of the ball). We present both quantitative and qualitative experimental results. Our quantitative results include a user study comparison with professional sports analysts, who significantly preferred rollouts by our approach to standard baselines. Examples of the user study is in the supplementary material. Our qualitative results demonstrate the ability of our approach to generate high-quality rollouts under various conditions. An interactive demo is available at `http://basketball-ai.com/`.

6.1. Experimental Setup

Training data. Each demonstration in our data contains trajectories of $K = 5$ players on the half-court, recorded for $T = 50$ timesteps at 6 Hz. The offensive team has possession of the ball for the entire sequence. $\mathbf{x}^k_t$ contains 2 dimensions: the coordinates of player k at time t on the court (50×94 feet). We normalize and mean-shift the data. Players are ordered based on their relative positions, similar to the role assignment in (Lucey et al., 2013). Overall, there are 107,146 training and 13,845 test examples.

For simplicity, we ignore the defensive players to focus on capturing the coordination of the offensive team. In addition, the defense is usually reactionary whereas the offense takes the initiative and will tend to have more multi-model behavior. In principle, we can provide the defensive positions as conditional input for our model and update the defensive positions using methods such as (Le et al., 2017). We also ignore the ball since the dynamics of the ball are difficult to learn (e.g. oscillations indicate dribbling while straight lines indicate passing). We leave the task of modeling the ball for future work.

Weak macro-goal labels. We extract weak macro-goals labels $\hat{\mathbf{g}}^k_t$ for each player k as done in (Zheng et al., 2016). We segment the left half-court into a 10×9 grid of 5ft $\times$5ft cells. The weak macro-goal $\hat{\mathbf{g}}^k_t$ at time t is a 1-hot encoding of dimension 90 of the next cell in which player k is stationary (i.e. speed $\|\hat{\mathbf{v}}^k_t\|_2 = \|\mathbf{x}^k_{t+1} - \mathbf{x}^k_t\|_2$ below a threshold). Macro-goals change slowly over time relative to player positions (see Figure 2). Figure 6 shows the distribution of extracted weak macro-goal labels for each player.

Model	Log-Likelihood	# Parameters
RNN-gauss	1931	7,620,820
VRNN-single	$\geq$ 2302	8,523,140
VRNN-indep	$\geq$ 2360	4,367,340
Ours	$\geq$ **2362**	4,372,190

Table 1. We report the average log-likelihood per sequence in the test set as well as the number of trainable parameters for each model. "$\geq$" indicates a lower bound on the log-likelihood.

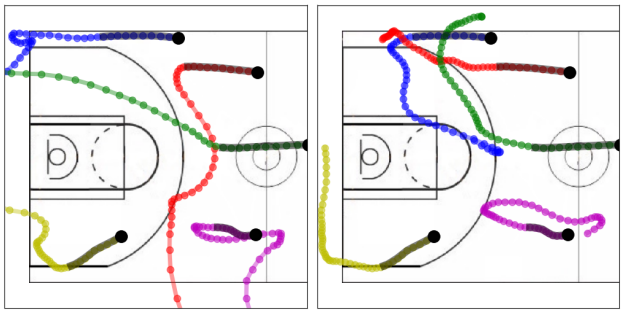
Model details. We model each latent variable $\mathbf{z}^k_t$ as a multivariate Gaussian with diagonal covariance of dimension 16 (so the KL-term in Eq. (22) can be computed analytically). All policies (and inference/prior functions for VRNNs, where applicable) are implemented with memoryless 2-layer fully-connected neural networks with a hidden layer of size 200. Our agent policies sample from a multivariate Gaussian with diagonal covariance while our macro-goal policies sample from a multinomial distribution over the macro-goals. All hidden states ($\mathbf{h}_{g,t}, \mathbf{h}^1_t, \ldots \mathbf{h}^K_t$) are modeled with 200 2-layer GRU memory cells each. We maximize the log-likelihood/ELBO with stochastic gradient descent using the Adam optimizer (Kingma & Ba, 2014) and a learning rate of 0.0001.

Baselines. We compare our approach with 3 baselines that do not use a hierarchy of macro-goals:

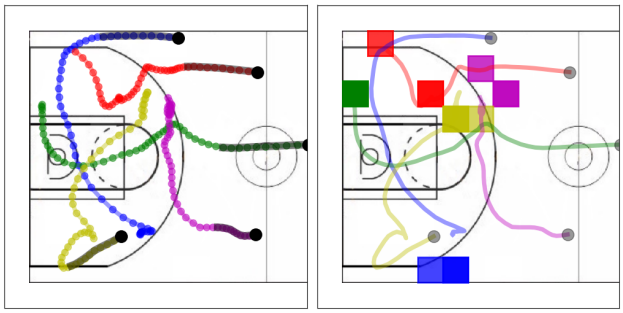
1. **RNN-gauss:** RNN model Eq. (1-2) with 900 2-layer GRU cells for the hidden state.
2. **VRNN-single:** We concatenate all player positions together and use model Eq. (17-18) for $K = 1$ with 900 2-layer GRU cells for the hidden state and a 80-dimensional latent variable.
3. **VRNN-indep:** Model Eq. (17-18) with 250 2-layer GRU cells for the hidden states and 16-dimensional latent variables. We also provide the previous positions of all players as conditional input for each policy, so Eq. (17) becomes $p^k_\theta(\mathbf{x}^k_t|\mathbf{x}_{<t}) = \varphi^k(\mathbf{z}^k_t, \mathbf{h}^k_{t-1}, \mathbf{x}_{t-1})$.

6.2. Quantitative Evaluation

Log Likelihood. Table 1 reports the ELBO (log-likehood for RNN-gauss) on the test data. We see that our approach outperofrms RNN-gauss and VRNN-single using fewer parameters and is comparable with VRNN-indep. However, higher log-likelihoods do not necessarily indicate higher quality of generated samples (Theis et al., 2015). As such, we also conduct a human preference study to assess the relative quality of generated rollouts.



(a) Baseline rollouts of representative quality. **Left**: VRNN-single. **Right**: VRNN-indep. Common problems in baseline rollouts include players moving out of bounds or in the wrong direction. Players do not appear to behave cohesively as a team.



(b) **Left**: Rollout from our model. All players remain in bounds. **Right**: Corresponding macro-goals for left rollout. Macro-goal generation is stable and suggests that the team is creating more space for the blue player (perhaps setting up an isolation play).

Figure 4. Rollouts from baselines and our model starting from black dots, generated for 40 timesteps after an initial burn-in period of 10 timesteps (marked by dark shading). An interactive demo of our model is availanle at: http://basketball-ai.com/.

Model Comparison	Win/Tie/Loss	Avg Gain
vs. VRNN-single	25/0/0	0.57
vs. VRNN-indep	15/4/6	0.23

Table 2. Preference study results. We asked 14 professional sports analysts to judge the relative quality of the generated rollouts. Judges are shown 50 comparisons that animate one rollout from our model and another from a baseline. Win/Tie/Loss indicates how often our model is preferred over baselines. Gain scores are computed by scoring +1 when our model is preferred and -1 otherwise. The average gain is computed over the total number of comparisons (25 per baseline) and judges. Our results are 98% significant using a one-sample t-test.

Human preference study. We recruited 14 professional sports analysts as judges to compare the quality of rollouts. Each comparison animates two rollouts, one from our model and another from a baseline. Both rollouts are burned-in for 10 timesteps with the same ground-truth states from the test set, and then generated for the next 40 timesteps. Judges decide which of the two rollouts looks more realistic. Example comparisons are in the supplementary material.

Table 2 shows the results from the preference study. We tested our model against two baselines, VRNN-single and VRNN-indep, with 25 comparisons for each. All judges preferred our model over the baselines with 98% statistical significance. These results suggest that our model generates rollouts of significantly higher quality than the baselines.

6.3. Qualitative Evaluation of Generated Rollouts

We next conduct a qualitative visual inspection of rollouts. Figure 4 shows rollouts generated by VRNN-single, VRNN-indep, and our model. Rollouts are generated by sampling states for 40 timesteps after an initial burn-in period of 10 timesteps with ground truth states from the test set. An interactive demo to generate more rollouts from our model can be found at: http://basketball-ai.com/.

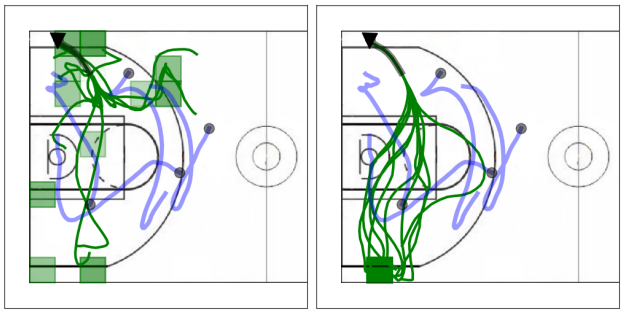


Figure 5. 10 rollouts of the green player (▼) overlayed on top of each other. A burn-in period of 20 timesteps is applied. Blue trajectories (●) are ground truth and black symbols indicate starting positions. **Left**: The model generates macro-goals. **Right**: We ground the macro-goals at the bottom-left. In both cases, we observe a multi-modal generating distribution of trajectories.

Common problems in baseline rollouts (Figure 4a) include players moving out of bounds or in the wrong direction. These issues tend to occur at later timesteps, suggesting that the baselines do not perform well over long horizons. One possible explanation is due to compounding errors (Ross et al., 2011): if the policy makes a mistake and deviates from the states seen during training, it will make more mistakes in the future thus leading to poor generalization.

On the other hand, generated rollouts from our model (Figure 4b) are more robust to the types of errors made by the baselines. Generated macro-goals also allow us to intepret the intent of each individual player as well as a global team strategy (e.g. setting up a specific formation on the court). We highlight that our model learns a multi-modal generating distribution, as repeated rollouts with the same burn-in result in a dynamic range of generated trajectories, as seen in Figure 5 Left. Furthermore, Figure 5 Right demonstrates that we can ground macro-goals manually instead of sampling them to control agent behavior.

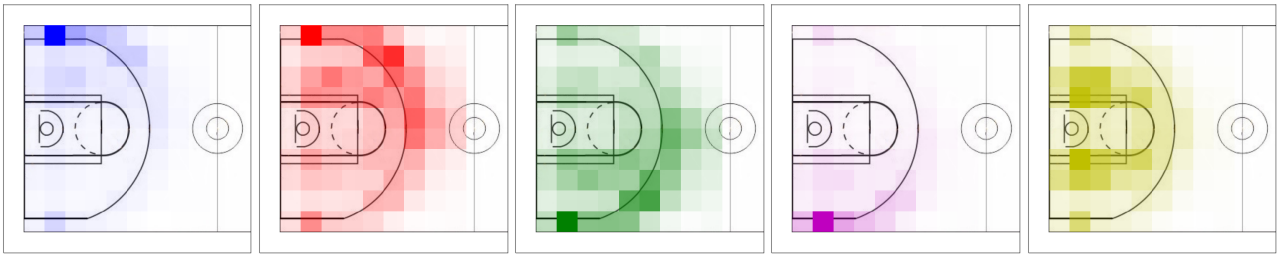


Figure 6. Distribution of weak macro-goal labels extracted for each player from the training data. Color intensity corresponds to frequency of macro-goal label. Players are ordered by their relative positions on the court, which can be seen from the macro-goals.

6.4. Analysis of Models

Output distribution for states. The agent policies in all our models (including baselines) sample from a multivariate Gaussian with diagonal covariance. We also experimented with sampling from a mixture of 2, 3, 4, and 8 Gaussian components, but discovered that the models would always learn to assign all the weight on a single component and ignore the others. The variance of the active component is also very small. This is intuitive because sampling with a large variance at every timestep would result in noisy trajectories and not the smooth ones that we see in Figure 4.

Consequently, each agent's inidividual policy is effectively deterministic when conditioned on its two inputs, one of which is deterministic (hidden state $\mathbf{h}_t^k$), while the other is stochastic (latent variable $\mathbf{z}_t^k$). So the variability of generated trajectories that we observe in Figure 5 must come from the only remaining source of randomness: the latent variable $\mathbf{z}_t^k$. We conclude that our model does not suffer from the Dying Units problem (Zhang et al., 2017; Chen et al., 2016), where the output policy model is sometimes sufficiently expressive on its own that the latent variables fail to encode anything meaningful.

Model for macro-goal policy. We chose to model our macro-goal policy in Eq. (20-21) with an RNN. In principle, we can also use more expressive models, like VRNNs, to model macro-goal policies over richer macro-goal spaces. In our case, we found that an RNN was sufficient in capturing the distribution of macro-goals shown in Figure 6. The RNN learns multinomial distributions over macro-goals that are peaked at a single macro-goal and relatively static through time, which is consistent with the behavior of macro-goals that we extracted from the data. Latent variables in a VRNN had little effect on the multinomial distribution.

Hidden state for macro-goal policy model. In our work, we defined a macro-goal $\mathbf{g}_t$ that consists of individual macro-goals $\mathbf{g}_t^k$ for each player. For our macro-goal policy, we compared a RNN model with a shared hidden state in Eq. (21), with a RNN model with independent hidden states, similar to Eq. (18). Intuitively, we expect the shared hidden state model to be better at capturing coordination.

To provide a more quantitative comparison, we computed the frequency that two or more players had the same individual macro-goal $\mathbf{g}_t^k$ at a given time, with the assumption that coordinated players do not have coinciding macro-goals very often. In the training data, 5.7% of all timesteps had coinciding macro-goals. In 10,000 rollouts from our macro-goal policy, 8.5% and 15.2% of all timesteps had coinciding macro-goals for the shared and independent hidden state models respectively. As a result, we used a RNN with a shared hidden state to model the macro-goal policy.

7. Conclusion and Discussion

We studied the problem of generative multi-agent behavioral cloning, where the goal is to learn a multi-agent policy that can tractably map input states to distributions over multi-agent action spaces. We proposed a hierarchical policy class that can effectively capture Our model incorporates macro-goals to reason over agent-coordination and ensure consistent behavior over long time horizons. We showcased our approach on modeling team offense in basketball, and demonstrated that our approach generates significantly higher-quality rollouts than non-hierarchical baselines.

The macro-goals used in our experiments are relatively simple. For instance, rather than simply using location-based macro-goals, we could also incorporate interactions such as "pick and roll". Another direction for future work is to explore how to adapt our approach to different domains. For example, instead of defining macro-goals for each agent, one can imagine learning a macro-goal representing "argument" for a dialogue between two agents, or a macro-goal representing "refrain" for music generation.

Another limitation is the need for macro-goal annotations or heuristics to extract weak macro-goal labels. Although high-level annotations in many datasets is feasible, such as in music (Thickstun et al., 2017), an interesting future direction is to develop algorithms to learn macro-goals in a semi-supervised or unsupervised setting, which has emerged in prior work. For example in STRAW, the agent learns to commit to a plan of future actions and is penalized for changing its plan (Vezhnevets et al., 2016). Incorporating such concepts is an interesting direction for future work.

References

Abbeel, Pieter and Ng, Andrew Y. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

Chen, Liang-Chieh, Schwing, Alexander, Yuille, Alan, and Urtasun, Raquel. Learning deep structured models. In *ICML*, 2015.

Chen, Xi, Kingma, Diederik P., Salimans, Tim, Duan, Yan, Dhariwal, Prafulla, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Variational lossy autoencoder. *CoRR*, abs/1611.02731, 2016.

Chernova, Sonia and Veloso, Manuela. Multiagent collaborative task learning through imitation. In *International Symposium on Imitation in Animals and Artifacts*, 2007.

Cho, KyungHyun, van Merrienboer, Bart, Bahdanau, Dzmitry, and Bengio, Yoshua. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259, 2014.

Chung, Junyoung, Kastner, Kyle, Dinh, Laurent, Goel, Kratarth, Courville, Aaron C., and Bengio, Yoshua. A recurrent latent variable model for sequential data. In *NIPS*, 2015.

Daumé, Hal, Langford, John, and Marcu, Daniel. Search-based structured prediction. *Machine learning*, 75(3): 297–325, 2009.

Dayan, Peter and Hinton, Geoffrey E. Feudal reinforcement learning. In *NIPS*, 1993.

Eyjolfsdottir, Eyrun, Branson, Kristin, Yue, Yisong, and Perona, Pietro. Learning recurrent representations for hierarchical behavior modeling. In *ICLR*, 2017.

Fraccaro, Marco, Sø nderby, Søren Kaae, Paquet, Ulrich, and Winther, Ole. Sequential neural models with stochastic layers. In *NIPS*, 2016.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *NIPS*, 2014.

Ho, Jonathan and Ermon, Stefano. Generative adversarial imitation learning. In *NIPS*, 2016.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Johnson, Matthew, Duvenaud, David K, Wiltschko, Alex, Adams, Ryan P, and Datta, Sandeep R. Composing graphical models with neural networks for structured representations and fast inference. In *NIPS*, 2016.

Kingma, Diederik P. and Ba, Jimmy. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Kingma, Diederik P. and Welling, Max. Auto-encoding variational bayes. In *ICLR*, 2014.

Krishnan, Rahul G., Shalit, Uri, and Sontag, David. Structured inference networks for nonlinear state space models. In *AAAI*, 2017.

Kulkarni, Tejas D, Narasimhan, Karthik, Saeedi, Ardavan, and Tenenbaum, Josh. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *NIPS*, 2016.

Le, Hoang Minh, Yue, Yisong, Carr, Peter, and Lucey, Patrick. Coordinated multi-agent imitation learning. In *ICML*, 2017.

Li, Yunzhu, Song, Jiaming, and Ermon, Stefano. Infogail: Interpretable imitation learning from visual demonstrations. In *NIPS*, 2017.

Lin, Henry C, Shafran, Izhak, Yuh, David, and Hager, Gregory D. Towards automatic skill evaluation: Detection and segmentation of robot-assisted surgical motions. *Computer Aided Surgery*, 11(5):220–230, 2006.

Lucey, Patrick, Bialkowski, Alina, Carr, Peter, Morgan, Stuart, Matthews, Iain, and Sheikh, Yaser. Representing and discovering adversarial team behaviors using player roles. In *CVPR*, 2013.

Miller, Andrew, Bornn, Luke, Adams, Ryan, and Goldsberry, Kirk. Factorized point process intensities: A spatial analysis of professional basketball. In *ICML*, 2014.

Oord, Aaron van den, Dieleman, Sander, Zen, Heiga, Simonyan, Karen, Vinyals, Oriol, Graves, Alex, Kalchbrenner, Nal, Senior, Andrew, and Kavukcuoglu, Koray. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016a.

Oord, Aaron van den, Kalchbrenner, Nal, and Kavukcuoglu, Koray. Pixel recurrent neural networks. In *ICML*, 2016b.

Ranganath, Rajesh, Tran, Dustin, and Blei, David. Hierarchical variational models. In *ICML*, 2016.

Rezende, Danilo Jimenez, Mohamed, Shakir, and Wierstra, Daan. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

Ross, Stéphane, Gordon, Geoffrey J., and Bagnell, J. Andrew. No-regret reductions for imitation learning and structured prediction. In *AISTATS*, 2011.

Sutton, Richard S, Precup, Doina, and Singh, Satinder. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Suwajanakorn, Supasorn, Seitz, Steven M, and Kemelmacher-Shlizerman, Ira. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 36(4):95, 2017.

Taylor, Sarah, Kim, Taehwan, Yue, Yisong, Mahler, Moshe, Krahe, James, Rodriguez, Anastasio Garcia, Hodgins, Jessica, and Matthews, Iain. A deep learning approach for generalized speech animation. In *SIGGRAPH*, 2017.

Theis, L., van den Oord, A., and Bethge, M. A note on the evaluation of generative models. *ArXiv e-prints*, November 2015.

Thickstun, John, Harchaoui, Zaid, and Kakade, Sham. Learning features of music from scratch. In *ICLR*, 2017.

Vezhnevets, Alexander, Mnih, Volodymyr, Agapiou, John, Osindero, Simon, Graves, Alex, Vinyals, Oriol, and Kavukcuoglu, Koray. Strategic attentive writer for learning macro-actions. *CoRR*, abs/1606.04695, 2016.

Xue, Tianfan, Wu, Jiajun, Bouman, Katherine, and Freeman, Bill. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *NIPS*, 2016.

Yue, Yisong, Lucey, Patrick, Carr, Peter, Bialkowski, Alina, and Matthews, Iain. Learning fine-grained spatial models for dynamic sports play prediction. In *ICDM*. IEEE, 2014.

Zhang, Cheng, Bütepage, Judith, Kjellström, Hedvig, and Mandt, Stephan. Advances in variational inference. *CoRR*, abs/1711.05597, 2017.

Zhang, Jiakai and Cho, Kyunghyun. Query-efficient imitation learning for end-to-end simulated driving. In *AAAI*, 2017.

Zheng, Stephan, Yue, Yisong, and Lucey, Patrick. Generating long-term trajectories using deep hierarchical networks. In *NIPS*, 2016.

Ziebart, Brian D, Maas, Andrew L, Bagnell, J Andrew, and Dey, Anind K. Maximum entropy inverse reinforcement learning. In *AAAI*. Chicago, IL, USA, 2008.

Ziebart, Brian D, Maas, Andrew L, Bagnell, J Andrew, and Dey, Anind K. Human behavior modeling with maximum entropy inverse optimal control. In *AAAI Spring Symposium: Human Behavior Modeling*, pp. 92, 2009.