

Multimodal Deep Generative Models for Trajectory Prediction: A Conditional Variational Autoencoder Approach

Boris Ivanovic^{1*}, Karen Leung^{1*}, Edward Schmerling², Marco Pavone¹

Abstract—Human behavior prediction models enable robots to anticipate how humans may react to their actions, and hence are instrumental to devising safe and proactive robot planning algorithms. However, modeling complex interaction dynamics and capturing the possibility of many possible outcomes in such interactive settings is very challenging, which has recently prompted the study of several different approaches. In this work, we provide a self-contained tutorial on a conditional variational autoencoder (CVAE) approach to human behavior prediction which, at its core, can produce a multimodal probability distribution over future human trajectories conditioned on past interactions and candidate robot future actions. Specifically, the goals of this tutorial paper are to review and build a taxonomy of state-of-the-art methods in human behavior prediction, from physics-based to purely data-driven methods, provide a rigorous yet easily accessible description of a data-driven, CVAE-based approach, highlight important design characteristics that make this an attractive model to use in the context of model-based planning for human-robot interactions, and provide important design considerations when using this class of models.

I. INTRODUCTION

Human behavior is inconsistent across populations, settings, and even different instants, with all other factors equal—addressing this inherent uncertainty is one of the fundamental challenges in human-robot interaction (HRI). Even when a humans broader intent is known, there are often multiple distinct courses of action that person may pursue to accomplish their goals. For example, in Figure 1, a pedestrian crossing a road may pass to the left or right of an oncoming pedestrian; reasoning about the situation cannot be simplified to the “average” case, i.e., the pedestrians colliding. To an observer, the choice of mode may seem to have a random component, yet also depend on the evolution of the humans surroundings. Imbuing a robot with the ability to take into consideration the full breadth of possibilities in how humans may respond to its actions is a key component of enabling anticipatory and proactive robot decision-making policies which can result in safer and more efficient interactions.

For the goal of creating robots that interact intelligently with human counterparts, observing data from human-human interactions has provided valuable insight into modeling interaction dynamics (see [1] for an extensive survey). A robot may reason about human actions, and corresponding likelihoods, based on how it has seen humans behave in similar settings. To implement a robot’s policy, model-free methods tackle this problem in an end-to-end fashion—human behavior predictions are implicitly encoded in the

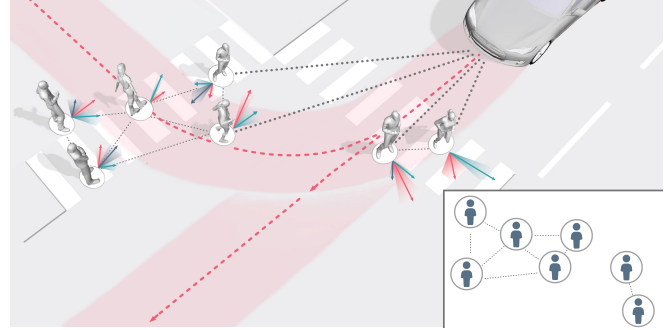


Fig. 1: There are many different ways an interaction (e.g., pedestrians crossing the road) may evolve. For safe human-robot interactions, robots (e.g., autonomous car) need to reason about the possibility of multiple outcomes (denoted by the colored shaded arrows), and understand how their actions influence the actions of others. Inset: Graphical representation of the interaction.

robot’s policy which is learned directly from data. On the other hand, model-based methods decouple the model learning and policy construction—a probabilistic understanding of the interaction dynamics is used as a basis for policy construction. By decoupling action/reaction prediction from policy construction, model-based approaches often afford a degree of transparency in a planners decision making that is typically unavailable in model-free approaches. In this paper, we take on a model-based approach to HRI and focus on learning a model of human behaviors, or more specifically, distributions over future human behaviors (e.g., trajectories).

Within model-based methods for HRI, there are many existing approaches to modeling human behaviors, and they can broadly be categorized as ontological or phenomenological as shown in Figure 2. To contextualize our work against other methods, we will build a taxonomy of different types of ontological and phenomenological state-of-the-art methods in this field. We note that these methods could be categorized differently across other dimensions (e.g., whether the model produces probabilistic or deterministic predictions). At a high-level, ontological approaches (sometimes referred to as “theory of mind”) postulate a core underlying structure about an agent’s behavior and build a mathematical model upon it. For instance, they may craft a set of rules that agents are required to follow, or an analytic model describing an agent’s internal decision-making scheme. In contrast, phenomenological approaches do not make such strong modeling assumptions, instead rely on a wealth of data to model agent behaviors without explicitly reasoning about underlying motivations.

We approach this problem phenomenologically, and in particular, focus on using a Conditional Variational Autoencoder (CVAE) [2] to learn a human behavior prediction model

Toyota Research Institute (TRI) provided funding to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

*Denotes equal contribution.

¹Department of Aeronautics and Astronautics, Stanford University. {borisi, karenl7, pavone}@stanford.edu

²Institute for Computational & Mathematical Engineering, Stanford University. schmerling@stanford.edu

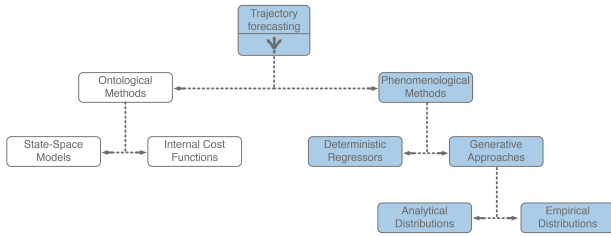


Fig. 2: A taxonomy of trajectory forecasting approaches, separated by high-level choices in methodology. In blue are the methods that will be mainly discussed in this paper.

well-suited for model-based planning and control [3]. We seek to explicitly characterize the multimodal uncertainty in human actions at each time step conditioned on interaction history as well as future robot action choices. Conditioning on interaction history allows a robot to reason about hidden factors like experience, mood, or engagement level that may influence the distribution, and conditioning on the robot’s next action choices takes into account response dynamics which is critical for planning in highly interactive scenarios.

Goals of this paper: The main goal of this paper is to provide a self-contained tutorial on the CVAE-based human trajectory prediction model proposed and developed in [3], [4], [5], and [6]. Before we delve into the details of our approach, we build a taxonomy of state-of-the-art methods for human behavior prediction in interactive settings in order to provide insight into the problem setups for which our work is best suited. Thus, the contributions of this paper are five-fold: (i) provide a concise taxonomy of ontological and phenomenological methods for human behavior prediction for interactive settings (Section II), (ii) introduce CVAEs in a self-contained manner and detail the proposed neural network architecture for human trajectory prediction (Section III), (iii) demonstrate the benefits of the model with a focus on its scalability to multi-agent settings, use of heterogeneous data, and ability to produce an analytic representation of the output trajectory distribution rooted with a dynamics model (Section IV and V), and (iv) compare the performance of such an approach against other state-of-the-art phenomenological approaches as well as discuss important implementation considerations for practitioners using this model (Section VI).

II. RELATED WORK

Illustrated in Figure 2, methods for predicting human behaviors can be classified as either ontological or phenomenological. On the left are ontological models—models that make assumptions about an agent’s dynamics or motivation. One direction is to make assumptions about the underlying physics that govern the system and then derive a state-space model via first principles. For instance, the Social Forces model [7] formulates the interaction dynamics by making assumptions on the attractive and repulsive forces between agents. Similarly, the Intelligent Driver Model (IDM) [8] derives a continuous-time car-following differential equation model. Due to the simplicity of these models, they are very useful in simulating large-scale interactions, such as crowd dynamics in panic situations [9] or traffic flow [10]. Despite these methods capturing the coupling between agents, they

are fundamentally unimodal representations of the interaction (i.e., do not account for the possibility of multiple distinct futures) and do not utilize knowledge of past interactions. These methods fall under *state-space models* in Figure 2.

Rather than formulating the interaction dynamics explicitly, we can instead make assumptions about a human’s internal decision-making process. This falls under the *internal cost functions* box in Figure 2. Game theoretic approaches model the interaction dynamics by making assumptions on whether the other agent is cooperative [11] or adversarial [12] and leverage this information for robot planning. Another popular approach is to model humans as optimal planners and represent their motivations at each time step as a state/action dependent reward (equivalently, negative cost) function. Maximizing this function, e.g., by following its gradients to select next actions, may be thought of as a computational proxy for human decision-making processes.

Inverse Reinforcement Learning (IRL) [13], [14] is a generalization of this idea whereby a parameterized family of reward functions is fit to a dataset of human state-action demonstrations. The reward function is typically represented as a linear combination of possibly nonlinear features $r(x, u) = \theta^T \varphi(x, u)$, where the weight parameters θ are fit to minimize a measure of error between the actions that optimize r and the true human actions. One of the typical strengths of IRL is its interpretability, both in terms of the ability to include handcrafted features, as well as what learned linear weights reveal regarding feature importance. Maximum entropy (MaxEnt) IRL [15] applies this principle in a probabilistic fashion; the probability distribution over human actions is proportional to the exponential of the reward $p(u) \propto \exp(r(x, u))$. This framework has been employed to model human behaviors in the context of driving [16] and social navigation [17] and then used to inform a robot’s planning strategy. In theory, with sufficiently complex and numerous features in the reward function, MaxEnt IRL could approximate any (including multimodal) distributions arbitrarily well, making this an attractive candidate for our application of HRI. However, there are three main drawbacks with MaxEnt IRL that prompt us to consider an alternative approach. First, to yield a unified and tractable framework for MaxEnt IRL-based prediction and policy construction, resulting policies typically employ gradient-based optimization using the IRL model which ultimately results in a unimodal assumption on interaction outcome regardless of the nature of the distribution [16], [17]. Second, IRL is typically applied to learn importance weights for a handful of human-interpretable features. Using more complex, possibly deep-learned, features to increase expressivity of the model removes one of the key benefits of IRL, and instead promotes the use of phenomenological methods. Even if the features are hand-crafted by the designer and are interpretable, the potential for “reward hacking” still exists; the robot may exploit the reward function, leading to unintended consequences [18]. Third, and most critically, is that IRL approaches rely on a Markovian assumption. Without some form of state augmentation (which would increase the computational complexity of the problem), this formulation is incapable of conditioning on interaction history when reasoning about the future. In general, reward-based approaches can be effective in settings with limited data as there are only a few param-

ters to learn, and can transfer to new and unseen tasks [19]. However, in the presence of large amounts of data and with the desire to condition on interaction history, it is natural to consider phenomenological approaches.

Phenomenological approaches (on the right in Figure 2) are methods that do not make inherent assumptions about the structure of the interaction dynamics and/or agents decision-making process. Instead, they rely on powerful modeling techniques and a wealth of observation data to infer and replicate the complex interactions. Recently, there have been a plethora of deep learning-based regression models for predicting future human trajectories (e.g., [20], [21]) following the success of Long Short-Term Memory (LSTM) networks [22], a purpose-built deep learning architecture for modeling temporal sequence data. However, such methods only produce a single deterministic trajectory output and therefore neglect to capture the uncertainty inherent in human behaviors. These methods fall under the *deterministic regressors* category in Figure 2. Safety-critical systems need to reason about many possible future outcomes to guard against worst-case scenarios, ideally with the likelihoods of each occurring, to enable safe decision-making. As a result, there has been recent interest in methods that simultaneously forecast multiple possible futures, or produce a distribution over possible future outcomes.

Due to recent advancements in generative modeling, [2], [23], there has been a paradigm shift from deterministic regressors to generative models, i.e., models that produce a distribution over possible future behaviors. In particular, *deep* generative methods (neural-network based models that learn an approximation of the true underlying probability distribution from which the dataset was sampled from) have emerged as state-of-the-art approaches. There are two main deep generative methods that dominate the field, (Conditional) Generative Adversarial Networks ((C)GANs) [23], [24], and (Conditional) Variational Autoencoders ((C)VAEs) [25], [2]. Both these methods have been widely used in the context of future human trajectory prediction in interactive settings (e.g., [26], [27], [28], [29]). GANs are composed of a generator and discriminator network—to produce realistic outputs, the generator outputs empirical samples which are then “judged” by the discriminator. Although GAN-based models show promising results, there are two main limitations. First, GANs often suffer from mode collapse, a phenomenon where the model converges to the mode of the distribution and is unable to capture and produce diverse outputs [30]. This is incompatible with safety-critical applications where it is important to capture rare yet potentially catastrophic outcomes. Second, GANs are notoriously difficult to train because the conflict between the generator and discriminator causes instability in the training process [31], [32]. Additionally, despite offering flexibility in the definition of the objective function, GANs fundamentally output an *empirical* distribution and could limit the types of model-based planners/controllers that can be used (e.g., planners that rely on a parameterized distribution).

Alternatively, (C)VAEs take on a variational Bayesian approach; they learn an approximation of the true underlying probability distribution by distilling latent attributes as probability distributions and then “decode” samples from the latent distribution to produce desired outputs. In contrast to

GANs, (C)VAEs optimize the negative log-likelihood over *all* examples in the training set, meaning all the modes of the distribution are considered and is less likely to suffer problems of mode collapse and lack of diversity seen in GANs. Additionally, (C)VAEs can produce either empirical samples from the distribution, or an analytical representation of the distribution, making it more attractive than GANs to use in the context of model-based planning and control.

Having considered each element in our taxonomy tree (Figure 2), there are many considerations when selecting a method to model interaction dynamics and perform human behavior prediction. In settings with high amounts of data available, and the need for high expressivity to capture interaction nuances and multimodal distribution coverage over the output space, we place our focus on using CVAEs for human trajectory prediction in the context of HRI.

III. THE CONDITIONAL VARIATIONAL AUTOENCODER FOR INTERACTION-AWARE BEHAVIOR PREDICTION

We describe a general CVAE model and apply it in the context of human behavior prediction. We highlight the core characteristics of our proposed CVAE trajectory prediction model and illustrate them with a traffic-weaving case-study.

A. Conditional Variational Autoencoder (CVAE)

Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, the goal of *conditional* generative modeling is to fit a model of the conditional probability distribution $p(y | x)$, which may be used for downstream applications such as inference (i.e., calculating the likelihood of observing a particular sample y given x), or to generate new samples y given x . In this work we consider parametric models, whereby we consider $p(y | x)$ within a family of distributions defined by a fixed set of parameters, which we fit to the dataset with the objective of maximizing the likelihood of the observed data. Due to their expressivity, neural networks are often used to represent complex and high-dimensional distributions.

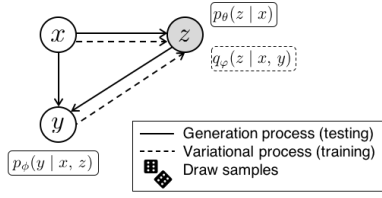
A CVAE [2] is a *latent* conditional generative model. The goal is still to approximate $p(y | x)$, but before outputting $p(y | x)$ the model first projects the inputs into a lower dimensional space, called the *latent space*, which acts as a bottleneck to encourage the model to uncover salient features with the intended purposes of improving performance, and potentially aiding in interpretability. Figure 3a illustrates the graphical model of a CVAE. An encoder, parameterized by θ , takes the input x and produces a distribution $p_\theta(z | x)$ where z is a latent variable that can be continuous or discrete [33], [34].¹ A decoder, parameterized by ϕ , uses x and samples from $p_\theta(z | x)$ to produce $p_\phi(y | x, z)$. In practice, the encoder and decoder are neural networks. The latent variable z is then marginalized out to obtain $p(y | x)$,

$$p(y | x) = \sum_z p_\phi(y | x, z) p_\theta(z | x). \quad (1)$$

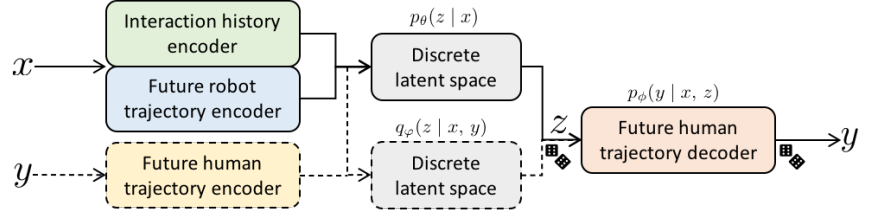
To efficiently perform the marginalization in (1), we desire values of z that are likely to have produced y , otherwise $p_\theta(z | x) \approx 0$ and will contribute almost nothing to $p(y | x)$.² As such, we perform *importance sampling* by instead

¹For this work, we focus on a discrete latent space, but note the following equations still apply by replacing the summation with an integral.

²We note that if the size of the discrete latent space is small, we can tractably compute the summation in (1) exactly.



(a) Graphical model of a CVAE.



(b) Sequence-to-sequence CVAE architecture for human behavior prediction.

Fig. 3: A graphical model of a CVAE, and a neural network architecture of a CVAE for human behavior prediction. Solid lines represent components for the generation process (during testing), and dashed lines represent components used for variational inference (during training).

sampling from $q(z | x, y)$, a proposal distribution, which will help us select values of z that are likely to have produced y . Since we are free to choose $q(z | x, y)$, we parameterize it with φ (often a neural network), denoted by $q_\varphi(z | x, y)$. We can rewrite (1) by multiplying and dividing by the proposal distribution, and using the definition of expectation,

$$\begin{aligned} p(y | x) &= \sum_z \frac{p_\phi(y | x, z) p_\theta(z | x)}{q_\varphi(z | x, y)} q_\varphi(z | x, y) \\ &= \mathbb{E}_{q_\varphi(z | x, y)} \left[\frac{p_\phi(y | x, z) p_\theta(z | x)}{q_\varphi(z | x, y)} \right]. \end{aligned}$$

The goal is to fit parameters ϕ , θ , and φ that maximize the log-likelihood of $p(y | x)$ over the dataset \mathcal{D} . By taking the log of both sides, using Jensen's inequality, and rearranging the terms, the *evidence lower-bound* (ELBO) is derived,

$$\log p(y | x) \geq \mathbb{E}_{q_\varphi(z | x, y)} [\log p_\phi(y | x, z)] - \mathcal{D}_{\text{KL}} [q_\varphi(z | x, y) \| p_\theta(z | x)] \quad (2)$$

where $\mathcal{D}_{\text{KL}}(p \| q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$ is the Kullback-Liebler divergence (for discrete distributions). The ELBO is a lower bound on $\log p(y | x)$, the term that we are trying to maximize, but is intractable to compute. Instead, we maximize the ELBO as a proxy. By using the reparameterization trick [25], [33], [34], the ELBO is tractable to compute and can be optimized via stochastic gradient descent. The loss for a single training example (x, y) is,

$$\mathcal{L}(x, y) = -\mathbb{E}_{q_\varphi(z | x, y)} [\log p_\phi(y | x, z)] + \mathcal{D}_{\text{KL}} [q_\varphi(z | x, y) \| p_\theta(z | x)]. \quad (3)$$

During training, we minimize the Monte Carlo estimate of the expected loss over the training set.

B. Interaction-aware Human Behavior Prediction

We are interested in learning a model that is able to predict future trajectories of intelligent agents (i.e., we assume these agents are humans, or human-controlled) interacting with other intelligent agents in the environment. Specifically, we desire a model that (i) is history dependent in order to capture behavioral tendencies or intent, (ii) accounts for the coupled interaction dynamics between all agents, (iii) produces a *multimodal* distribution over future human trajectories because there are many different ways a human may behave in an interactive setting, and (iv) is well-suited for model-based planning since our ultimate goal is to design robots that can leverage these predictions to interact smoothly with humans. Our proposed sequence-to-sequence CVAE trajectory prediction architecture, shown in Figure 3b, is able to address these desiderata in the following ways.

To address (i) and (ii) above, the input conditioning variable x consists of features representing *interaction history*, a sequence of features from all agents (e.g., positions, velocities, actions) since the start of the interaction, and a *future robot trajectory*, a sequence of states and/or actions that the robot plans to follow over the planning horizon. We can additionally include other features that may be relevant to the application, such as a map of the environment, or camera images from the robot (see Section V). The output y is a sequence of future states/actions of all human agents that we are interested in. Since the output is conditioned on what the robot will do in the future (and interaction history), this model learns the coupled interaction dynamics. We discuss later in Section V how we can integrate predicted actions to produce dynamically-feasible trajectory predictions.

To address (iii), a multimodal distribution is constructed by using a *discrete* latent space. Each latent vector instantiation of z corresponds to a discrete mode (i.e., a mixture component), and its probability $p_\theta(z | x)$, is produced by the encoder (corresponding to the mixture weight). For example, one of the discrete modes might correspond to a human driver braking, while another might correspond to turning right. Note that enforcing semantic meaning to each latent value is by no means guaranteed, and is an area of active research [35]. A continuous latent space could be used, though in our work, we found a discrete latent space to be more effective. For a given mode, there are variations to how this behavior might occur (e.g., slightly different ways to turn right). To cater for these variations and account for dependencies in successive states or actions, the decoder outputs an autoregressive Gaussian mixture model (GMM). We want to highlight that the use of GMMs here is not the mechanism behind creating a multimodal distribution, albeit that is what GMMs are typically known for. Rather, the GMMs are used to create arbitrarily complex distributions which correspond to one of the mixture components. At each time step of the prediction horizon, the decoder outputs GMM components describing the distribution of the output features, then a sample is drawn from the GMM and is used to generate a GMM at the next time step. Repeating this process will create a *sample* drawn from $p(y | x)$. For the case with a single GMM component (i.e., a Gaussian), the mean and variance at each time step can be propagated instead of a sample, enabling an *analytic* representation of the output distribution (see Section V).

The flexibility in how the output distribution is represented addresses (iv); we can tailor the outputs to the needs of a model-based planner. Specifically, we can choose to

describe the learned distribution empirically (i.e., output samples directly), or analytically (i.e., output parameters of the distribution). Additionally, there are many options for how to construct the encoder and decoder. We primarily leverage recurrent neural networks (RNN) to process time-series data with potentially variable length without increasing the problem size (in contrast to IRL which relies on a Markovian assumption and would require some form of state-augmentation). As we will describe in Sections IV and V, we can augment the model to consider spatio-temporal relationships between multiple agents and heterogeneous data inputs (e.g., state trajectories, images, and maps).

C. Traffic-Weaving Case Study

We describe the traffic-weaving scenario studied in [3] to illustrate the key characteristics of our approach. In the traffic-weaving scenario, two cars initially side-by-side must swap lanes in a short amount of time and distance, emulating cars merging onto/off of a highway. This is a challenging negotiation due to the inherent multimodal uncertainty of who will pass whom. Before we begin, we make two remarks. First, we use an LSTM for the encoder and decoder networks as we found this RNN architecture provided the best performance. Second, we chose to predict future human *action* sequences and use future robot *action* sequences as inputs as this aligned with our case study. However, for other applications, states can be used instead of actions.

The interaction history is defined as the sequence of states and actions from both agents since the start of the interaction. RNNs are a desirable choice since they can account for variable input lengths. We consider future robot action sequences as inputs to the future robot trajectory encoder. The LSTM decoder produces GMM components describing a distribution over human actions at each time step; an action is sampled from the GMM and is fed into back into the LSTM cell to produce the next action, and so forth. During the training process (i.e., variational process) where we want to learn the proposal distribution, $q_\phi(y | x, z)$, the future human action sequences are inputs to the future human trajectory encoder. This helps ensure that at test time (i.e., generation process), $p_\theta(z | x)$ will be effective in estimating $p(y | x)$ since we will not have access to y , the ground truth (see (1)).

In Figure 4, when the robot is deciding its next action to take, it can predict how the human may respond to each of its candidate future action sequences (dashed blue line). The different colors in the predictions (thin lines) showcase the different modes in the output distribution. For instance, light blue trajectories correspond to the human speeding up, while dark yellow trajectories correspond to the human slowing down. Given this interaction model, the robot can select the optimal action to take by searching over the (finite) set of possible future action sequences and selecting the one that results in the highest expected reward. This model-based planner was tested and validated in simulation [3] and on a full-scale test vehicle [36].

IV. SCALING UP TO MULTI-AGENT INTERACTIONS

In the real world, agents simultaneously interact with *many* other agents, e.g., pedestrians walking through crowds, vehicles passing through intersections or merging on highways. Thus, the model discussed in the previous section needs to

be extended to consider a general number of agents as well as the spatio-temporal relationships between them.

A. Modeling a General Number of Agents

A natural approach to modeling such interactions is to abstract the scene as a spatio-temporal graph (STG) $G = (V, E)$, named so because it represents agents as nodes and their interactions as edges, which evolve in time. An edge $(u, v) \in E$ is present if agent u “interacts” with agent v . Since the concept of an interaction is difficult to define mathematically, spatial proximity is a commonly used proxy [20], [21], [27], [28]. Specifically, agents u and v are said to be interacting if $\|\mathbf{p}_u - \mathbf{p}_v\|_2 \leq d$ where $\mathbf{p}_u, \mathbf{p}_v$ are the spatial coordinates of agents u, v in the world, and d is a distance threshold that sets the interaction range of agents. A benefit of abstracting scenes in this way is that it enables any similarly-structured approach to be applied to various environments, and even different problem domains (e.g., modeling human-object interactions in computer vision [21]), as STGs are general abstractions. Figure 1 shows an example of a STG abstraction of an autonomous driving scene.

This changes the trajectory forecasting problem from one of modeling agents and their interactions to one of modeling nodes and their edges. The key challenge here is that an agent can have a general number of neighbors which change from one scenario to another. Thus the resulting model needs to be able to handle a general number of inputs for a fixed architecture (since neural network weights have fixed sizes). To do this, one can extend the architecture discussed in Section III-C so that it mimics the structure of the scene’s STG. In particular, an LSTM is added for each edge that connects to a node (directly modeling edges), with an intermediate aggregation step in order to combine the influence from neighboring nodes of the same type. This is the approach taken in [4], which demonstrated that this structure can model the influence that neighboring agents have.

While this enables one to model a general number of agents, an additional consideration needs to be made for the fact that V , the set of agents, and E , the set of agent-agent interactions, are *time-varying*. This is especially noticeable in autonomous driving as a vehicle’s sensors have limited range. As a result, agents can appear and disappear at every timestep, e.g., due to merging on or off the highway near the ego-vehicle. Even if the number of agents was constant, their interactions are necessarily time-varying as agents’ spatial proximity to others changes as they move. Thus, the edge encoding scheme discussed in the previous subsection needs to be further extended to capture time-varying structure.

B. Modeling Time-varying Interactions

Introducing time-variance modifies our STG representation from $G = (V, E)$ to $G_t = (V_t, E_t)$. Unfortunately, naively recreating a new STG per timestep and applying the method discussed in the previous section will be expensive and inefficient as it does not recycle information that may persist over multiple timesteps (e.g., keeping track of which edges are new, established, or recently removed).

Another approach is to introduce a scalar that modulates the outputs of each edge-encoding LSTM depending on how recently the edge was added or removed. This is the approach taken in [5], where the scalar varies from 0 to 1 and acts as

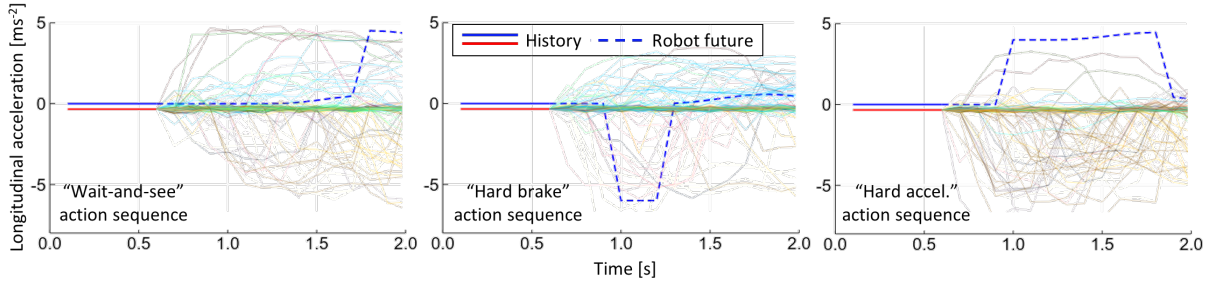


Fig. 4: Predictions of future human action sequences depend on the future action sequence of the robot (blue dashed line). The different colors of future human action sequences correspond to different discrete latent variable instantiations (i.e., different modes in the multimodal output distribution.)

an additional weighting factor before the edge’s influence is included in the rest of the model. This output re-weighting additionally serves as a low-pass filter so that newly added or removed edges do not wildly swing model outputs from one timestep to another, rejecting high-frequency noise produced by upstream perception systems (e.g., when vehicles dither near the limits of sensor range). A key benefit of this approach is that it is fast to update online, due to the model’s stateful representation only needing a few matrix multiplies to capture new observations [5]. This is especially important in robotic use cases, which frequently require the ability to run online from streaming data in real-time. We will further discuss runtime considerations in Section VI-C.

V. INCORPORATING AGENT DYNAMICS AND HETEROGENEOUS INPUT DATA

So far, we have seen how one can probabilistically model a general, time-varying number of interacting agents in a scene. In this section, we will dive deeper into considerations for output structures, specifically those that ensure the feasibility of the output trajectories, as well as methods for including additional sources of information that are commonly available on modern robotic platforms, such as high-definition (HD) maps of the surrounding environment.

A. Producing Dynamically-feasible Outputs

Common to most approaches in behavior prediction is the eventual need to produce outputs in spatial coordinates, mostly because a majority of evaluation metrics in academic behavior prediction settings are over spatial coordinates [1]. As a result, most methods either directly produce trajectory samples (e.g., GANs) or utilize intermediate models to convert internal representations to positions with uncertainty (e.g., CVAE-based approaches with decoders that output bivariate GMMs), such as the architecture discussed in the previous sections. However, both of these output structures make it difficult to enforce dynamics constraints, e.g., non-holonomic constraints such as those arising from no slip conditions. The absence of such considerations might lead to predictions that are unrealizable by the underlying actions (e.g., predicting that a car will move sideways).

To remedy this, we can leverage established ideas in dynamics modeling. When selecting a dynamics model to enforce, one usually finds a trade-off between modeling complexity and computational efficiency. In the case of autonomous driving, however, there is an additional complicating factor in the form of perception requirements. Ideally, agent models would be chosen to best match their semantic

type. For example, one would usually model cars on the road using a bicycle model [37]. However, estimating the bicycle model parameters or actions of another vehicle from perception online is very difficult as it requires estimation of the vehicle’s center of mass, wheelbase, and front wheel steer angle. A related model which does not have such high estimation requirements is the dynamically-extended unicycle model [38]. It strikes a good balance between accuracy (accounting for key vehicular non-holonomic constraints, e.g., no slip constraints) and efficiency (having only four states and two actions), without requiring complex online parameter estimation procedures (one only needs to estimate the vehicle’s position and velocity). This choice of dynamics model follows the one made in [6], which shows through experiments that such a simplified model is already quite impactful on improving prediction accuracy.

To incorporate such dynamics considerations, one should instead view their learning architecture as producing distributions over an agent’s actions rather than positions (as in the previous sections), and focus on the process of integration from actions to position through the agent’s dynamics. Notably, this scheme can also propagate the model’s uncertainty in its generated actions to uncertainty over the resulting positions. The reason for this is that the only uncertainty at prediction time stems from the model’s normally-distributed output. Thus, in the case of linear dynamics (e.g., single integrators, frequently used to model pedestrians), the system dynamics are linear Gaussian. Formally, for a single integrator with actions $\mathbf{u}^{(t)} = \dot{\mathbf{p}}^{(t)}$, the position mean at $t + 1$ is $\mu_{\mathbf{p}}^{(t+1)} = \mu_{\mathbf{p}}^{(t)} + \mu_{\mathbf{u}}^{(t)} \Delta t$, where $\mu_{\mathbf{u}}^{(t)}$ is produced by the learning architecture. In the case of nonlinear dynamics (e.g., unicycle models, used to model vehicles), one can still (approximately) use this uncertainty propagation scheme by linearizing the dynamics about the agent’s current state and action.³ This dynamics integration scheme is used in [6] and enables the model to produce *analytic* output distributions.

Importantly, even with this additional inclusion of dynamics, no additional data is required for training (e.g., the loss was not amended to be over actions). The model still directly learns to match a dataset’s ground truth position, with gradients backpropagated through the agent’s dynamics to the rest of the model. Thus, without any extra data, this inclusion of dynamics enables the model to generate explicit action sequences that lead to dynamically-feasible trajectory predictions. Overall, this output scheme is able to

³Full mean and covariance equations for the single integrator and dynamically-extended unicycle models can be found in the appendix of [6].

TABLE I: Comparison of our CVAE-based method against GAN-based methods for pedestrian modeling. Lower is better, bold is best.

Metric	S-GAN [26]	S-BiGAT* [29]	Trajectron++ [6]
BoN ADE [26]	0.58	0.48	0.21
BoN FDE [26]	1.18	1.00	0.41
KDE NLL [5]	5.80	-	-1.14

*Model not public, so we could not evaluate it on the KDE NLL metric.

guarantee that its trajectory samples are dynamically feasible, in contrast to methods which directly output positions.

B. Incorporating Heterogeneous Data

Modern robotic systems host a plethora of advanced sensors which produce a wide variety of outputs and data modalities for downstream consumption. However, many current behavior prediction methodologies only make use of the tracked trajectories of other agents as input, neglecting these other sources of information from modern perception systems. Notably, HD maps are used by many real-world systems to aid localization as well as inform navigation. Depending on sensor availability and sophistication, maps can range in fidelity from simple binary obstacle maps, i.e., $M \in \{0, 1\}^{H \times W \times 1}$, to multilayered semantic maps, e.g., $M \in \{0, 1\}^{H \times W \times L}$ where each layer $1 \leq \ell \leq L$ corresponds to an area with semantic type (e.g., “driveable area,” “road block,” “walkway,” “pedestrian crossing”).

A major reason for this choice of map format is that it closely resembles images, which also have height, width, and channel dimensions. As a result, Convolutional Neural Networks (CNNs), which are efficient to evaluate online, are frequently used to encode them. [6] also makes this choice, using a relatively small CNN to encode local scene context around the agent being modeled. In particular, a CNN with 4 convolutional layers can be used to encode an 11-layer HD semantic map that is 120×120 pixels in size with a 0.33 m per pixel resolution. The CNN’s filters are respectively of size $\{5, 5, 5, 3\}$ with strides of $\{2, 2, 1, 1\}$. These are followed by a fully-connected layer, the output of which is a vector that represents the local context visible in the map. This vector is then concatenated with the vectors encoding node history and edge influence from previous sections. More generally, one can similarly include further additional information (e.g., raw LIDAR data, camera images, pedestrian skeleton or gaze direction estimates) in the encoder of an architecture by representing it as a vector via an appropriate model and concatenating the resulting output to this backbone of representation vectors.

VI. EXPERIMENTS AND PRACTICAL CONSIDERATIONS

In this section, we quantitatively compare the method described in Section V against state-of-the-art GAN-based approaches for the challenging problem of pedestrian motion prediction. Additionally, we discuss important implementation considerations for practitioners seeking to employ the approaches presented in this work.

A. Quantitative Performance

We compare Trajectron++ [6] against Social GAN [26] and Social BiGAT [29], all of which use similar RNN-based architectures to model temporal sequences. The approaches are evaluated on the real-world ETH [39] and UCY [40] pedestrian datasets, a standard benchmark in the field com-

prised of challenging multi-human interaction scenarios. We evaluate their performance with the Best-of- N (BoN) Average and Final Displacement Error (ADE and FDE) metrics proposed in [26] as well as the Kernel Density Estimate-based Negative Log-Likelihood (KDE NLL) proposed in [5]. As can be seen in Table I, the CVAE-based Trajectron++ significantly outperforms the others on the three specified metrics. Further experiments as well as ablation studies can be found in [6]. More broadly, the success of phenomenological approaches for large-data regimes has been reflected in modern trajectory forecasting competitions. For instance, all prize winners of the recent ICRA 2020 nuScenes [41] prediction challenge (one of which is Trajectron++ [6]) are phenomenological, using deep encoder-decoder architectures and leveraging heterogeneous input data in addition to past trajectory history.

B. Latent Space Size

The size of the latent space (i.e., the number of latent variables) is something not yet discussed in this work. While finding the “optimal” size ends up being a hyperparameter search, one should generally allocate a latent variable for each high-level behavior or effect they wish to model. In the (common) case where it is difficult to know exactly how many that is (e.g., in driver modeling), one should start high and let the CVAE prune out redundant modes by assigning them very low probabilities. For instance, in each of [3], [4], [5], [6] we use 25 latent variables (i.e., z can take 25 values). Of these, the CVAE only ends up assigning significant probability to a few modes at a time, e.g., moving straight, turning left, turning right, stopping.

In order to determine how many modes are being used, the CVAE’s learned weights can be analyzed through the lens of evidential theory, as proposed in [42]. Specifically, one can identify which latent variables have direct evidence that supports their existence, and prune the others without any loss in performance. For instance, [42] found that only 2 – 12 latent variables have direct evidence in [6] and that the rest can be pruned without any loss in performance.

C. Online Model Runtime

A key consideration in the development of models for robotic applications is their runtime complexity. To achieve real-time performance, one can leverage the stateful representation that spatiotemporal graphs provide. Specifically, the model can be updated online with new information without fully executing a forward pass. For instance, due to our method’s use of LSTMs, only the last LSTM cells in the encoder need to be fed the newly-observed data. The rest of the model can then be executed using the updated encoder representation. This update-and-predict scheme is applied in [5], [6], both of which achieve real-time online performance.

VII. CONCLUSIONS AND FUTURE WORK

We have provided a self-contained tutorial on a CVAE approach to multimodal trajectory prediction for multi-agent interactions. Additionally, we provide a taxonomy of existing state-of-the-art approaches, thereby identifying major methodological considerations and placing our proposed approach in perspective. In the presence of large amounts of data with potentially heterogeneous data types (e.g., spatial features, images, maps), and non-Markovian settings

where future behaviors depend on the *history* of interaction, our proposed CVAE approach is an attractive model for predicting future human trajectories in multi-agent interactive settings. In particular, our CVAE approach is very flexible, making it easy to include heterogeneous data, account for agent dynamics, and tailor it to different types of model-based planning algorithms.

Future work includes further improvements on the model such as developing ways to make the latent space more interpretable, e.g., through the lens of temporal logic, robustifying against upstream sensor noise, and applying the learned model to generate more realistic simulation agents for testing and validation. More broadly, there are still many open questions regarding evaluation metrics and architectural considerations stemming from future integration with downstream planning and control algorithms. These questions are increasingly important now that phenomenological trajectory prediction methods have outweighed others in raw performance, and are targeting deployment on real-world safety-critical robotic systems.

Acknowledgment. We thank Matteo Zallio for his help in visually communicating our ideas.

REFERENCES

- [1] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras. (2019) Human motion trajectory prediction: A survey. Available at <https://arxiv.org/abs/1905.06113>.
- [2] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Conf. on Neural Information Processing Systems*, 2015.
- [3] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone, "Multimodal probabilistic model-based planning for human-robot interaction," in *Proc. IEEE Conf. on Robotics and Automation*, 2018.
- [4] B. Ivanovic, E. Schmerling, K. Leung, and M. Pavone, "Generative modeling of multimodal multi-human behavior," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2018.
- [5] B. Ivanovic and M. Pavone, "The Trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *IEEE Int. Conf. on Computer Vision*, 2019.
- [6] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conf. on Computer Vision*, 2020, in Press.
- [7] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, pp. 4282–4286, 1995.
- [8] M. Treiber, A. Hennecke, and D. Helbing, "Microscopic simulation of congested traffic," in *Traffic and Granular Flow '99*. Springer Berlin Heidelberg, 2000.
- [9] D. Helbing, I. Farkas, and T. Vicsek, "Simulating dynamical features of escape panic," *Nature*, vol. 407, pp. 487–490, 2000.
- [10] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, pp. 1805–1824, 2000.
- [11] S. Nikolaidis, S. Nath, A. D. Procaccia, and S. Srinivasa, "Game-theoretic modeling of human adaptation in human-robot collaboration," in *IEEE Int. Conf. on Human-Robot Interaction*, 2017.
- [12] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game theoretic planning for self-driving cars in competitive scenarios," in *Robotics: Science and Systems*, 2019.
- [13] A. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Int. Conf. on Machine Learning*, 2000.
- [14] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Int. Conf. on Machine Learning*, 2004.
- [15] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI Conf. on Artificial Intelligence*, 2008.
- [16] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions," in *Robotics: Science and Systems*, 2016.
- [17] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *Int. Journal of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, 2016.
- [18] D. Hadfield-Menell, S. Milli, P. Abbeel, S. Russell, and A. D. Dragan, "Inverse reward design," in *Conf. on Neural Information Processing Systems*, 2017.
- [19] R. Choudhury, G. Swamy, D. Hadfield-Menell, and A. D. Dragan, "On the utility of model learning in HRI," in *IEEE Int. Conf. on Human-Robot Interaction*, 2019.
- [20] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2016.
- [21] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on spatio-temporal graphs," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2016.
- [22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997.
- [23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Conf. on Neural Information Processing Systems*, 2014.
- [24] M. Mirza and S. Osindero. (2014) Conditional generative adversarial nets. Available at <https://arxiv.org/abs/1411.1784>.
- [25] D. P. Kingma and M. Welling. (2013) Auto-encoding variational bayes. Available at <https://arxiv.org/abs/1312.6114>.
- [26] A. Gupta, J. Johnson, F. Li, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2018.
- [27] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "DESIRE: distant future prediction in dynamic scenes with interacting agents," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.
- [28] M.-F. Deo and J. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms," in *IEEE Intelligent Vehicles Symposium*, 2018.
- [29] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, S. H. Rezatofighi, and S. Savarese, "Social-BiGAT: Multimodal trajectory forecasting using bicycle-GAN and graph attention networks," in *Conf. on Neural Information Processing Systems*, 2019.
- [30] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Conf. on Neural Information Processing Systems*, 2016.
- [31] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *Int. Conf. on Learning Representations*, 2017.
- [32] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Int. Conf. on Machine Learning*, 2017.
- [33] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *Int. Conf. on Learning Representations*, 2017.
- [34] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *Int. Conf. on Learning Representations*, 2017.
- [35] T. Adel, Z. Ghahramani, and A. Weller, "Discovering interpretable representations for both deep generative and discriminative models," in *Int. Conf. on Machine Learning*, 2018.
- [36] K. Leung, E. Schmerling, M. Zhang, M. Chen, J. Talbot, J. C. Gerdes, and M. Pavone, "On infusing reachability-based safety assurance within planning frameworks for human-robot vehicle interactions," *Int. Journal of Robotics Research*, 2019, in Press.
- [37] J. Kong, M. Pfeifer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *IEEE Intelligent Vehicles Symposium*, 2015.
- [38] S. M. LaValle, "Better unicycle models," in *Planning Algorithms*. Cambridge Univ. Press, 2006, pp. 743–743.
- [39] S. Pellegrini, A. Ess, K. Schindler, and L. v. Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *IEEE Int. Conf. on Computer Vision*, 2009.
- [40] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," *Computer Graphics Forum*, vol. 26, no. 3, pp. 655–664, 2007.
- [41] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," 2019.
- [42] M. Itkina, B. Ivanovic, R. Senanayake, M. J. Kochenderfer, and M. Pavone, "Evidential disambiguation of latent multimodality in conditional variational autoencoders," in *Conf. on Neural Information Processing Systems - Workshop on Bayesian Deep Learning*, 2019.