DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents

Namhoon Lee[1], Wongun Choi[2], Paul Vernaza[2], Christopher B. Choy[3],
Philip H. S. Torr[1], Manmohan Chandraker[2,4]

[1]University of Oxford, [2]NEC Labs America, [3]Stanford University, [4]University of California, San Diego

Abstract

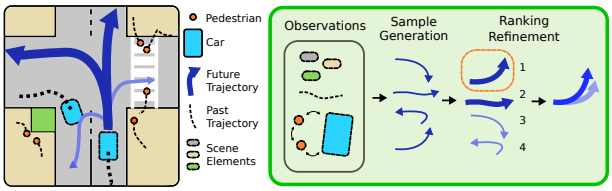
We introduce a Deep Stochastic IOC[1] RNN Encoder-decoder framework, DESIRE, for the task of future predictions of multiple interacting agents in dynamic scenes. DESIRE effectively predicts future locations of objects in multiple scenes by 1) accounting for the multi-modal nature of the future prediction (i.e., given the same context, future may vary), 2) foreseeing the potential future outcomes and make a strategic prediction based on that, and 3) reasoning not only from the past motion history, but also from the scene context as well as the interactions among the agents. DESIRE achieves these in a single end-to-end trainable neural network model, while being computationally efficient. The model first obtains a diverse set of hypothetical future prediction samples employing a conditional variational auto-encoder, which are ranked and refined by the following RNN scoring-regression module. Samples are scored by accounting for accumulated future rewards, which enables better long-term strategic decisions similar to IOC frameworks. An RNN scene context fusion module jointly captures past motion histories, the semantic scene context and interactions among multiple agents. A feedback mechanism iterates over the ranking and refinement to further boost the prediction accuracy. We evaluate our model on two publicly available datasets: KITTI and Stanford Drone Dataset. Our experiments show that the proposed model significantly improves the prediction accuracy compared to other baseline methods.

1. Introduction

> It is far better to foresee even without certainty than not to foresee at all.
>
> *Henri Poincaré (Foundations of Science)*

Considering the future as a consequence of a series of past events, a *prediction* entails reasoning about probable outcomes based on past observations. But predicting the future in many computer vision tasks is inherently riddled with uncertainty (see Fig. 1). Imagine a busy traffic intersection, where such ambiguity is exacerbated by diverse interactions of automobiles, pedestrians and cyclists with each other, as well as with semantic elements such as lanes, crosswalks and traffic lights. Despite tremendous recent interest in future prediction [3, 5, 17, 23, 26, 45, 46], existing state-of-the-art produces outcomes that are either deterministic, or do not fully account for interactions, semantic context or long-term future rewards.

(a) Future prediction example



(b) Workflow of *DESIRE*

Figure 1. (a) A driving scenario: The white van may steer into left or right while trying to avoid a collision to other dynamic agents. *DESIRE* produces accurate future predictions (shown as blue paths) by tackling multi-modaility of future prediction while accounting for a rich set of both static and dynamic scene contexts. (b) *DESIRE* generates a diverse set of hypothetical prediction samples, and then ranks and refines them through a deep IOC network.

In contrast, we present DESIRE, a Deep Stochastic IOC RNN Encoder-decoder framework, to overcome those limitations. The key traits of DESIRE are its ability to simultaneously: (a) generate *diverse hypotheses* to reflect a distribution over plausible futures, (b) reason about *interactions* between multiple dynamic objects and the scene context, (c) rank and refine hypotheses with consideration of *long-term future rewards* (see Fig. 1). These objectives are cast within a deep learning framework.

[1]IOC: Abbreviation for inverse optimal control, which will be more explained throughout the paper.

We model the scene as composed of semantic elements (such as roads and crosswalks) and dynamic participants or agents (such as cars and pedestrians). A static or moving observer is also considered as an instance of an agent. We formulate future prediction as determining the locations of agents at various instants in the future, relying solely on observations of the past states of the scene, in the form of agent trajectories and scene context derived from image-based features or other sensory data if available. The problem is posed in an optimization framework that maximizes the potential future reward of the prediction. Specifically, we propose the following novel mechanisms to realize the above advantages, also illustrated in Fig. 2:

- *Diverse Sample Generation*: Sec. 3.1 presents a conditional variational auto-encoder (CVAE) framework [41] to learn a sampling model that, given observations of past trajectories, produces a diverse set of prediction hypotheses to capture the multimodality of the space of plausible futures. The CVAE introduces a latent variable to account for the ambiguity of the future, which is combined with a recurrent neural network (RNN) encoding of past trajectories, to generate hypotheses using another RNN.
- *IOC-based Ranking and Refinement*: In Sec. 3.2, we propose a ranking module that determines the most likely hypotheses, while incorporating scene context and interactions. Since an optimal policy is hard to determine where multiple agents make strategic inter-dependent choices, the ranking objective is formulated to account for potential future rewards similar to inverse optimal control (IOC). This also ensures generalization to new situations further in the future, given limited training data. The module is trained in a multitask framework with a regression-based refinement of the predicted samples. In the testing phase, we iterate the above multiple times to obtain more accurate refinements of the future prediction.
- *Scene Context Fusion*: Sec. 3.3 presents the Scene Context Fusion (SCF) layer that aggregates interactions between agents and the scene context encoded by a convolutional neural network (CNN). The fused embedding is channeled to the aforementioned RNN scoring module and allows to produce the rewards based on the contextual information.

While DESIRE is a general framework that is applicable to any future prediction task, we demonstrate its utility in two applications – traffic scene understanding for autonomous driving and behavior prediction in aerial surveillance. Sec. 4 demonstrates outstanding accuracy for predicting the future locations of traffic participants in the KITTI raw dataset and pedestrians in the Stanford Drone dataset.

To summarize, this paper presents DESIRE, which is a deep learning based stochastic framework for time-profiled distant future prediction, with several attractive properties:

- *Scalability*: The use of deep learning rather than handcrafted features enables end-to-end training and easy incorporation of multiple cues arising from past motions, scene context and interactions between multiple agents.
- *Diversity*: The stochastic output of a deep generative model (CVAE) is combined with an RNN encoding of past observations to generate multiple prediction hypotheses that hallucinate ambiguities and multimodalities inherent in future prediction.
- *Accuracy*: The IOC-based framework accumulates long-term future rewards for sampled trajectories and the regression-based refinement module learns to estimate a deformation of the trajectory, enabling more accurate predictions further into the future.

2. Related Works

Classical methods Path prediction problems have been studied extensively with different approaches such as Kalman filters [18], linear regressions [29] to non-linear Gaussian Process regression models [49, 33, 34, 48], autoregressive models [2] and time-series analysis [32]. Such predictions suffice for scenarios with few interactions between the agent and the scene or other agents (like a flight monitoring system). In contrast, we propose methods for more complex environments such as surveillance for a crowd of pedestrians or traffic intersections, where the locomotion of individual agents is severely influenced by the scene context (*e.g.*, drivable road or building) and the other agents (*e.g.*, people or cars try to avoid colliding with the other).

IOC for path prediction Kitani *et al.* recover human preferences (*i.e.*, reward function) to forecast plausible paths for a pedestrian in [23] using inverse optimal control (IOC), or inverse reinforcement learning (IRL) [1, 52], while [26] adapt IOC and propose a dynamic reward function to address changes in environments for sequential path predictions. Combined with a deep neural network, deep IOC/IRL has been proposed to learn non-linear reward functions and showed promising results in robot control [11] and driving [50] tasks. However, one critical assumption made in IOC frameworks, which makes them hard to be applied to general path prediction tasks, is that the goal state or the destination of agent should be given a priori, whereby feasible paths must be found to the given destination from the planning or control point of view. A few approaches relaxed this assumption with so-called goal set [28, 10], but these goals are still limited to a target task space. Furthermore, a recovered cost function using IOC is inherently static, thus it is not suitable for time-profiled prediction tasks. Finally, past approaches do not incorporate interaction between agents, which is often a key constraint to the motion of multiple agents. In contrast, our methods are designed for more natural scenarios where agent goals are open-ended, unknown or time-varying and where agents interact with each other while dynamically adapting in anticipation of future behaviors.

Future prediction Walker *et al.* [47] propose a visual pre-

diction framework with a data-driven unsupervised approach, but only on a static scene, while [5] learn scene-specific motion patterns and apply to novel scenes for motion prediction as a knowledge transfer. A method for future localization from egocentric perspective is also addressed successfully in [30]. But unlike our method, none of those can provide time-profiled predictions. Recently, a large dataset is collected in [36] to propose the concept of social sensitivity to improve forecasting models and the multi-target tracking task. However, their social force [14] based model has limited navigation styles represented merely using parameters of distance-based Gaussians.

Interactions When modeling the behavior of an agent, it should also be taken into account that the dynamics of an agent not only depend on its own, but also on the behavior of others. Predicting the dynamics of multiple objects is also studied in [24, 25, 3, 31], to name a few. Recently, a novel pooling layer is presented by [3], where the hidden state of neighboring pedestrians are shared together to joinly reason across multiple people. Nonetheless, these models lack predictive capacity as they do not take into account scene context. In [24], a dynamic Bayesian network to capture situational awareness is proposed as a context cue for pedestrian path prediction, but the model is limited to orientations and distances of pedestrians to vehicles and the curbside. A large body of work in reinforcement learning, especially game theoretical generalizations of Markov Decision Processes (MDPs), addresses multi-agent cases such as minmax-Q learning [27] and Nash-Q learning [16]. However, as noted in [38], typically learning in multi-agent setting is inherently more complex than single agent setting [40, 39, 6].

RNNs for sequence prediction Recurrent neural networks (RNNs) are natural generalizations of feedforward neural networks to sequences [42] and have achieved remarkable results in speech recognition [13], machine translation [4, 42, 7] and image captioning [19, 51, 9]. The power of RNNs for sequence-to-sequence modeling thus makes them a reasonable model of choice to learn to generate sequential future prediction outputs. Our approach is similar to [7] in making use of the encoder-decoder structure to embed a hidden representation for encoding and decoding variable length inputs and outputs. We choose to use gated recurrent units (GRUs) over long short-term memory units (LSTMs) [15] since the former is found to be simpler yet yields no degraded performance [8]. Despite the promise inherent in RNNs, however, only a few works have applied RNNs to behavior prediction tasks. Multiple LSTMs are used in [3] to jointly predict human trajectories, but their model is limited to producing fixed-length trajectories, whereas our model can produce variable-length ones. A Fusion-RNN that combines information from sensory streams to anticipate a driver's maneuver is proposed in [17], but again their model outputs deterministic and fixed-length predictions.

Deep generative models Our work is also related to deep generative models [37, 35, 44], as we have a sample generation process that is built on a variational auto-encoder (VAE) [22] within the framework. Since our prediction model essentially performs posterior-based probabilistic inference where candidate samples are generated based on conditioning variables (*i.e.*, past motions besides latent variables), we naturally extend our method to exploit a conditional variational auto-encoder (CVAE) [21, 41] during the sample generation process. Dense trajectories of pixels are predicted from a single image using CVAE in [46], while we focus on predicting long-term behaviors of multiple interacting agents in dynamic scenes.

Unlike our framework, all aforementioned approaches lack either consideration of scene context, modeling of interaction with other agents or capabilities in producing continuous, time-profiled and long-term accurate predictions.

3. Method

We formulate the future prediction problem as an optimization process, where the objective is to learn the posterior distribution $P(\mathbf{Y}|\mathbf{X}, \mathcal{I})$ of multiple agents' future trajectories $\mathbf{Y} = \{Y_1, Y_2, .., Y_n\}$ given their past trajectories $\mathbf{X} = \{X_1, X_2, .., X_n\}$ and sensory input $\mathcal{I}$ where n is the number of agents. The future trajectory of an agent i is defined as $Y_i = \{y_{i,t+1}, y_{i,t+2}, .., y_{i,t+\delta}\}$, and the past trajectory is defined similarly as $X_i = \{x_{i,t-\iota+1}, x_{i,t-\iota+2}, .., x_{i,t}\}$. Here, each element of a trajectory (*e.g.*, $y_{i,t}$) is a vector in $\mathbb{R}^2$ (or $\mathbb{R}^3$) representing the coordinates of agent i at time t, and δ and ι refer to the maximum length of time steps for future and past respectively. Since direct optimization of continuous and high dimensional $\mathbf{Y}$ is not feasible, we design our method to first sample a diverse set of future predictions and assign a probabilistic score to each of the samples to approximate $P(\mathbf{Y}|\mathbf{X}, \mathcal{I})$. In this section, we describe the details of DESIRE (Fig. 2) in the following structure: *Sample Generation Module* (Sec. 3.1), *Ranking and Refinement Module* (Sec. 3.2), and *Scene Context Fusion* (Sec. 3.3).

3.1. Diverse Sample Generation with CVAE

Future prediction can be inherently ambiguous and has uncertainties as multiple plausible scenarios can be explained under the same past situation (*e.g.*, a vehicle heading toward an intersection can make different turns as seen in Fig. 1). Thus, learning a deterministic function f that directly maps $\{\mathbf{X}, \mathcal{I}\}$ to $\mathbf{Y}$ will under-represent potential prediction space and easily over-fit to training data. Moreover, a naively trained network with a simple loss will produce predictions that average out all possible outcomes.

In order to tackle the uncertainty, we adopt a deep generative model, conditional variational auto-encoder (CVAE) [41], inside of DESIRE framework. CVAE is a generative model that can learn the distribution $P(Y_i|X_i)$ of

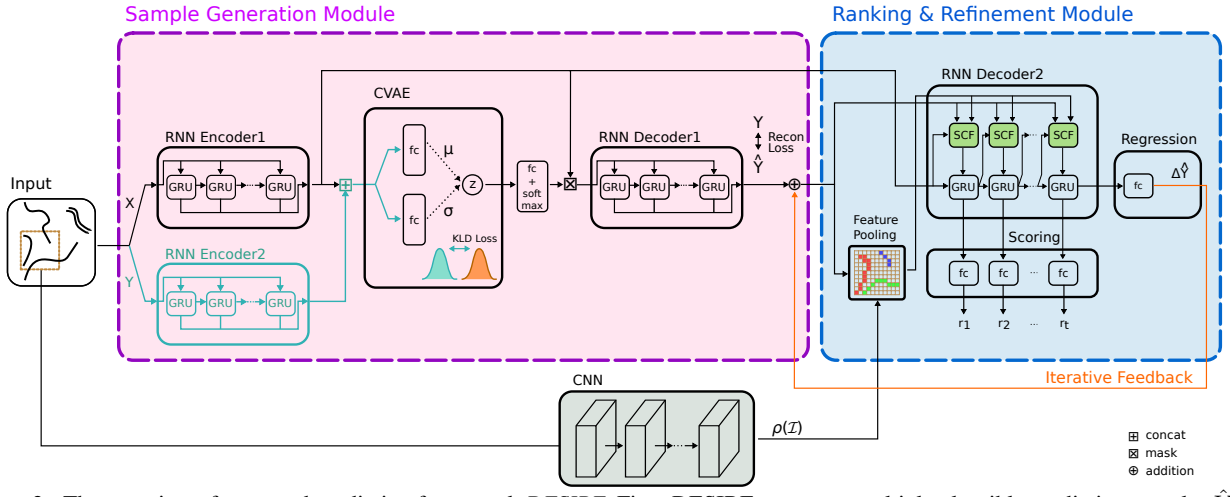


Figure 2. The overview of proposed prediction framework *DESIRE*. First, DESIRE generates multiple plausible prediction samples $\hat{Y}$ via a CVAE-based RNN encoder-decoder (Sample Generation Module). Then the following module assigns a reward to the prediction samples at each time-step sequentially as IOC frameworks and learns displacements vector $\Delta\hat{Y}$ to regress the prediction hypotheses (Ranking and Refinement Module). The regressed prediction samples are refined by iterative feedback. The final prediction is the sample with the maximum accumulated future reward. Note that the flow via aquamarine-colored paths is only available during the training phase.

the output Y_i conditioned on the input X_i by introducing a stochastic latent variable z_i[2]. It is composed of multiple neural networks, such as recognition network $Q_\phi(z_i|Y_i, X_i)$, (conditional) prior network $P_\nu(z_i|X_i)$, and generation network $P_\theta(Y_i|X_i, z_i)$. Here, θ, ϕ, ν denote the parameters of corresponding networks. The prior of the latent variables z_i is modulated by the input X_i, however, this can be relaxed to make the latent variables statistically independent of input variables, *i.e.*, $P_\nu(z_i|X_i) = P_\nu(z_i)$ [21, 41]. Essentially, a CVAE introduces stochastic latent variables z_i that are learned to encode a diverse set of predictions Y_i given input X_i, making it suitable for modeling one-to-many mapping. During training, $Q_\phi(z_i|Y_i, X_i)$ is learned such that it gives higher probability to z_i that is likely to produce a reconstruction $\hat{Y}_i$ close to actual prediction given the full context X_i and Y_i. At test time z_i is sampled randomly from the prior distribution and decoded through the decoder network to produce a prediction hypothesis. This enables probabilistic inference which serves to handle multi-modalities in the prediction space.

Train phase: Firstly, the past and future trajectories of an agent i, X_i and Y_i respectively, are encoded through two RNN encoders with separate set of parameters (*i.e.*, RNN Encoder1 and RNN Encoder2 in Fig. 2). The resulting two encodings, $\mathcal{H}_{X_i}$ and $\mathcal{H}_{Y_i}$, are concatenated and passed through one fully connected (fc) layer with a non-linear activation (*e.g.*, $relu$). Two side-by-side fc layers are followed to produce both the mean μ_{z_i} and the standard deviation σ_{z_i} over z_i. The distribution of z_i is modeled as a Gaussian distribution (*i.e.*, $z_i \sim Q_\phi(z_i|X_i, Y_i) = \mathcal{N}(\mu_{z_i}, \sigma_{z_i})$) and is regularized by the $\mathcal{KL}$ divergence against a prior distribution $P_\nu(z_i) := \mathcal{N}(0, I)$ during the training. Upon successful training, the target distribution is learned in the latent variable z_i, which allows one to draw a random sample z_i from a Gaussian distribution to reconstruct Y_i at test time. Since back-propagation is not possible through random sampling, we adopt the standard *reparameterization trick* [22] to make it differentiable.

In order to model $P_\theta(Y_i|X_i, z_i)$, z_i is combined with X_i as follows. The sampled latent variable z_i is passed to one fc layer to match the dimension of $\mathcal{H}_{X_i}$ that is followed by a $softmax$ layer, producing $\beta(z_i)$. Then that is combined with the encodings of past trajectories $\mathcal{H}_{X_i}$ through a masking operation $\boxtimes$ (*i.e.*, element-wise multiplication). One can interpret this as a *guided drop out* where the guidance β is derived from the full context of individual trajectory during the training phase, while it is randomly drawn from X_i, Y_i agnostic prior distribution $z_i^{(k)} \sim P_\nu(z_i)$ in the testing phase. Finally, the following RNN decoder (*i.e.*, RNN Decoder1 in Fig. 2) takes the output of the previous step, $\mathcal{H}_{X_i} \boxtimes \beta(z_i^{(k)})$, and generates K number of future prediction samples, *i.e.*, $\hat{Y}_i^{(1)}, \hat{Y}_i^{(2)}, .., \hat{Y}_i^{(K)}$.

There are two loss terms in training the CVAE-based RNN encoder-decoder.

- Reconstruction Loss: $\ell_{Recon} = \frac{1}{K}\sum_k \|Y_i - \hat{Y}_i^{(k)}\|$. This loss measures how far the generated samples are from the actual ground truth.
- $\mathcal{KLD}$ Loss: $\ell_{KLD} = D_{\mathcal{KL}}(Q_\phi(z_i|Y_i, X_i)\|P_\nu(z_i))$. This regularization loss measures how close the sampling distribution at test time is to the distribution of latent variable that we learn during training.

Test phase: At test time, the encodings of future trajectories $\mathcal{H}_{Y_i}$ are not available, thus the encodings of past trajectories $\mathcal{H}_{X_i}$ are combined with multiple random samples of latent variable $z_i^{(k)}$ drawn from the prior $z_i^{(k)} \sim P_\nu(z_i)$. Similar to the training phase, $\mathcal{H}_{X_i} \boxtimes \beta(z_i^{(k)})$ is passed to the following

[2]Notice that we learn the distribution independently over different agents in this step. Interaction between agents is considered in Sec. 3.2.

RNN decoder (*i.e.*, RNN Decoder1 in Fig. 2) to generate a diverse set of prediction hypotheses.

Further details: For both train and test phases, we pass trajectories through a temporal convolution layer before encoding to encourage the network to learn the concept of velocity from adjacent frames before getting passed into RNN encoders. Also, RNNs are implemented using gated recurrent units (GRU) [7] to learn long-term dependencies, yet they can be easily replaced with other popular RNNs like long short-term memory units (LSTM) [15]. In summary, this sample generation module produces a set of diverse hypotheses critical to capturing the multimodality of the prediction task, through a effective combination of CVAE and RNN encoder-decoder. Unlike [46], where CVAE is used to predict for short-term visual motion from a single image, our CVAE module generates diverse set of future trajectories based on a past trajectory.

3.2. IOC-based Ranking and Refinement

Predicting a *distant* future can be far more challenging than predicting one close by. In order to tackle this, we adopt the concept of decision-making process in reinforcement learning (RL) where an agent is trained to choose its actions that maximizes *long-term rewards* to achieve its goal [43]. Instead of designing a reward function manually, however, IOC [50, 11] learns an unknown reward function. Inspired by this, we design an RNN model that assigns rewards to each prediction hypothesis $\hat{Y}_i^{(k)}$ and measures their *goodness* $s_i^{(k)}$ based on the accumulated long-term rewards. Thereafter, we also directly refine prediction hypotheses by learning displacements $\triangle\hat{Y}_i^{(k)}$ to the actual prediction through another fc layer. Lastly, the module receives iterative feedbacks from regressed predictions and keeps adjusting so that it produces precise predictions at the end. The model is illustrated in the right side of Fig. 2. During the process, we combine 1) past motion history through the embedding vector $\mathcal{H}_{\mathbf{X}}$, 2) semantic scene context through a CNN with parameters ρ, and 3) interaction among multiple agents by using interaction features (Sec. 3.3). Notice that unlike typical robotics applications [50, 11], we do not assume that the goal (final destination) is known or the dynamics of the agents are given. Our model learns the agents dynamics as well as the scene context in a coherent framework.

Learning to score: For an agent i, there are K number of samples (*i.e.*, $\hat{Y}_i^{(1)}, \hat{Y}_i^{(2)}, .., \hat{Y}_i^{(K)}$) that are generated by our CVAE sampler. Let the score s of individual prediction hypothesis $\hat{Y}_i^{(k)}$ for the agent i be defined as follows,

$$s(\hat{Y}_i^{(k)}; \mathcal{I}, \mathbf{X}, \hat{\mathbf{Y}}_{j\setminus i}^{(\forall)}) = \sum_{t=1}^{T} \psi(\hat{y}_{i,t}^{(k)}; \mathcal{I}, \mathbf{X}, \hat{\mathbf{Y}}_{\tau<t}^{(\forall)}), \quad (1)$$

where $\hat{\mathbf{Y}}_{j\setminus i}^{(\forall)}$ is the prediction samples of other agents (*i.e.*, $\forall j$, where $j \neq i$), $\hat{y}_{i,t}^{(k)}$ is the k^{th} prediction sample of an agent i at time t, $\hat{\mathbf{Y}}_{\tau<t}^{(\forall)}$ is all the prediction samples until a time-step t, T is the maximum prediction length, and ψ is the reward function that assigns a reward value at each time-step. ψ is implemented as an fc layer that is connected to the hidden vector of RNN cell at each time step. We share the parameters of the fc layer over all the time steps (each RNN cell outputs the hidden state of the same dimension). Therefore, the score s is accumulated rewards over time, accounting for the entire future rewards being assigned to each hypothesis. This enables our model to make a strategic decision by allowing us to rank samples as in other sampling-based IOC frameworks [11]. In addition, the reward function ψ incorporates both scene context $\mathcal{I}$ as well as the interaction between agents (see Sec. 3.3).

Learning to refine: Alongside the scores, our model also estimates a regression vector $\triangle\hat{Y}_i^{(k)}$ that refines each prediction sample $\hat{Y}_i^{(k)}$. The regression vector for each agent i is obtained with the regression function η defined as follows,

$$\triangle\hat{Y}_i^{(k)} = \eta(\hat{Y}_i^{(k)}; \mathcal{I}, \mathbf{X}, \hat{\mathbf{Y}}_{j\setminus i}^{(\forall)}). \quad (2)$$

Represented as parameters of a neural network, the regression function η accumulates both scene contexts and all other agents dynamics from the past to entire future frames, and estimates the best displacement vector $\triangle\hat{Y}_i^{(k)}$ over entire time-horizon T. Similarly to the score s, it accounts for what happens in the future both in terms of scene context and interactions among dynamic agents to produce the output. We implement η as another fc layer that is connected to the last hidden vector of the RNN which outputs $M \times T$ dimensional vector. $M = 2$ (or 3) is the dimension of the location state.

Iterative feedback: Using the displacement vector $\triangle\hat{Y}_i^{(k)}$, we iteratively refine the prediction hypothesis $\hat{Y}_i^{(k)}$. After each cycle, $\hat{Y}_i^{(k)}$ is updated by $\hat{Y}_i^{(k)} + \triangle\hat{Y}_i^{(k)}$, and fed into the IOC module. This process is similar to the gradient descent optimization of $\hat{Y}_i$ over the score function s, but it does not require to compute the gradient over RNN which can be very unstable due to the recurrent structure (i.e., vanishing or exploding gradient). We observe that iterative refinement indeed improves the quality of prediction samples in the experiments (see Fig. 4 and Fig. 5).

Losses: There are two loss terms in training the IOC ranking and refinement module.

- Cross-entropy Loss: $\ell_{CE} = H(p, q)$ of which the target distribution q is obtained by $softmax(-d(Y_i, \hat{Y}_i^{(k)}))$, where $d(Y_i, \hat{Y}_i^{(k)}) = \max \|\hat{Y}_i^{(k)} - Y_i\|$.
- Regression Loss: $\ell_{Reg} = \frac{1}{K}\sum_k \|Y_i - \hat{Y}_i^{(k)} - \triangle\hat{Y}_i^{(k)}\|$

Finally, the total loss of the entire network is defined as a multi-task loss as follows, where N is the number of agents in one batch.

$$\ell_{Total} = \frac{1}{N}\sum_{i \in N} \ell_{Recon} + \ell_{KLD} + \ell_{CE} + \ell_{Reg} \quad (3)$$

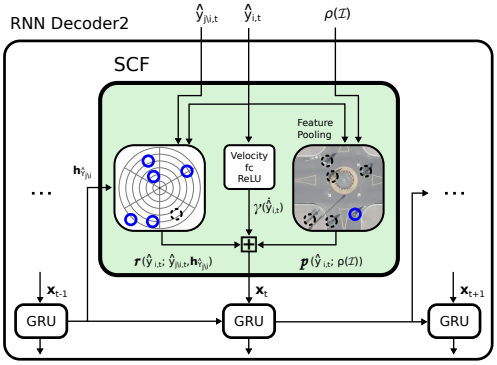


Figure 3. Details of Scene Context Fusion unit (SCF) in RNN Decoder2 in Fig. 2. Note that the input to the GRU cell at each time-step, $\mathbf{x}_t$, integrates multiple cues (*i.e.*, the dynamics of agents, scene context and interaction between agents).

3.3. Scene Context Fusion

As discussed in the previous section, our ranking and refinement module relies on the hidden representation of the shared RNN module. Thus, it is important that the RNN must contain the information about 1) individual past motion context, 2) semantic scene context and 3) the interaction between multiple agents, in order to provide proper hidden representations that can score and refine a prediction $\hat{Y}_i^{(k)}$.

We achieve the goal by having an RNN that takes following input $\mathbf{x}_t$ at each time step:

$$\mathbf{x}_t = \left[\gamma(\hat{v}_{i,t}), p(\hat{y}_{i,t}; \rho(\mathcal{I})), r(\hat{y}_{i,t}; \hat{y}_{j\setminus i,t}, \mathbf{h}_{\hat{Y}_{j\setminus i}})\right] \quad (4)$$

where $\hat{v}_{i,t}$ is a velocity of $\hat{Y}_i^{(k)}$ at t, γ is a fc layer with a *ReLU* activation that maps the velocity to a high dimensional representation space, $p(\hat{y}_{i,t}; \rho(\mathcal{I}))$ is a pooling operation that pools the CNN feature $\rho(\mathcal{I})$ at the location $\hat{y}_{i,t}$, $r(\hat{y}_{i,t}; \hat{y}_{j\setminus i,t}, \mathbf{h}_{\hat{Y}_{j\setminus i}})$ is the interaction feature computed by a fusion layer that spatially aggregates other agents hidden vectors, similar to SocialPooling (SP) layer [3]. The embedding vector $\mathcal{H}_{X_i}$ (the output of the RNN Encoder1 in Fig. 2) is shared as the initial hidden state of the RNN, in order to provide the individual past motion context. We share this embedding with the CVAE module since both require the same information to be embedded in the vector.

Interaction Feature: We implement a spatial grid based pooling layer similar to SP layer [3]. For each sample k of an agent i at t, we define spatial grid cells centered at $\hat{y}_{i,t}^{(k)}$. Over each grid cell g, we pool the hidden representation of all the other agents' samples that are within the spatial cell, $\forall j \neq i, \forall k, \hat{y}_{j,t}^{(k)} \in g$. Instead of using the max pooling operation with rectangular grids, we adopt log-polar grids with an average pooling. Combined with CNN features, the SCF module provides the RNN decoder with both static and dynamic scene information. It learns consistency between semantics of agents and scenes for reliable prediction.

3.4. Characteristics of DESIRE

This section highlights particularly distinctive features of DESIRE that naturally enable higher accuracy and reliability.

- The framework is based on deep neural network and is trainable end-to-end, rather than relying on hand-crafted parametric representation and interactions terms. Trajectories of each agent are represented using RNN encoders and are combined together through a fusion layer within the architecture. Scene context is represented through CNN and is not solely restricted to images (*i.e.*, can handle non-visual sensors too). Overall, the algorithm is scalable and flexible.
- CVAE is combined with RNN encodings to generate stochastic prediction hypothesis, which handles ambiguities and multimodalities inherent in future prediction.
- A novel RNN module coherently integrates multiple cues that have critical influence on behavior prediction such as dynamics of all neighboring agents and scene semantics.
- An IOC framework is used to train the trajectory ranking objective by measuring potential long-term future rewards. This makes the model less reactive, and enables more accurate predictions further into the future.
- A regression vector is learned to refine trajectories and an iterative feedback mechanism sequentially adjusts the predicted behavior, resulting in more accurate predictions.

4. Experiments

4.1. Datasets

KITTI Raw Data [12]: The dataset provides images of driving scenes and Velodyne 3D laser scan along with calibration information between cameras and sensors. To prepare data examples (*i.e.*, $X, Y, \mathcal{I}$), we performed the following: As the dataset does not provide semantic labels for 3D points (which we need for *scene context*), we first perform semantic segmentations of images and project Velodyne laser scans onto the image plane using the provided camera matrix to label 3D points. The semantically labeled 3D points are then registered into the world coordinates using GPS-IMU tags. Finally we create top-down view feature maps $\mathcal{I}$ of size $H \times W \times C$ (H, W: size of crop and C: number of classes for scene elements, *e.g.*, road, sidewalk, and vegitation shown as red, blue and green color in Fig. 6.). $\mathcal{I}$ is cropped with respect to the view point of the camera to simulate actual driving scenario ($H, W = 80m$ and the size of pixel is $0.5m$. The camera is located at the left-center.). Since laser scans on dynamic objects generate traces during registration, we remove moving objects and only use static scene elements. The trajectories X, Y are generated by extracting the center locations of the 3D tracklets and registering them in the world coordinates. We use all annotated videos from Road and City scenes for our experiments and generate approximately 2,500 training examples.

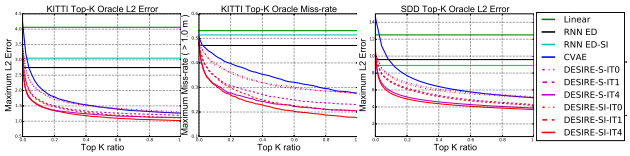


Figure 4. Oracle prediction errors over the number of samples on the KITTI dataset. X axis represents the ratio of top samples used in the oracle error evaluation (Y axis). Best viewed in color.

Stanford Drone Dataset [36]: The dataset contains a large volume of aerial videos captured in a university campus using a drone. There are various classes of dynamic objects interacting with each other, often in the form of high density crowds. Except for less stabilized cameras and lost labels, we used all videos to create examples to train/test our model, yielding approximately $16,000$ examples. Note that we directly use raw images to extract visual features, rather than semantically labeled feature maps. We resize the images by $1/5$ in following experiments to avoid memory overhead.

4.2. Evaluation Metrics and Baselines

The following metrics are used to measure the performance of future prediction task in various aspects: (i) L2 distance between the prediction and ground truth at multiple time steps, (ii) miss-rate with a threshold in terms of L2 distance at multiple time steps, (iii) maximum L2 distance over entire time frames, (iv) maximum miss-rate over entire time frames, and (v) *oracle* error over top K number of samples (*i.e.*, $\mathcal{E}_{oracle} = \min_{\forall k \in K} \mathcal{E}(\hat{Y}_i^{(k)} - Y_i)$) to account for the uncertainty in the future prediction (similar to MEE in [46]). We set K to be 50 throughout the main experiments.

We compare our method with the following baselines:

- *Linear*: A linear regressor that estimates linear parameters by minimizing the least square error.
- *RNN ED*: An RNN encoder-decoder model that directly regresses the prediction only using the past trajectories.
- *RNN ED-SI*: An *RNN ED* augmented with our SCF unit into the decoder similar to [17]. The model combines the scene and interaction features while making prediction and uses the same information as ours, but makes a prediction at $t+1$ solely based on the past information up to t.
- *DESIRE*: The proposed method. We denote our model with only semantic scene context in SCF module as *DESIRE-S* and our model with both scene context and interaction as *DESIRE-SI*. We also evaluate *DESIRE-X-IT*$\{N\}$, where N is the number of iterative feedbacks.

4.3. Learning Details

We train the model with Adam optimizer [20] with the initial learning rate of 0.004. The learning rate is decreased by half at every quarter of total epochs, albeit we do not observe clear improvement with this. All the models including Encoder-decoder baselines are trained for 600 epochs for KITTI and 8 epochs for SDD (about 50K iterations with a batch size 32). The full details on the architecture are discussed in the supplementary materials. In order to avoid exploding gradient in RNNs, we apply gradient clipping with L2 norm of 1.0. During the training procedure, we randomly rotate the scene and trajectories to augment data and reduce over-fitting. For all experiments, we run randomized 5 fold cross validation without overlapping videos in different splits. All models observe maximum of 2 seconds for past trajectories and make a prediction up to 4 seconds into the future. All models are implemented using TensorFlow and trained end-to-end with a NVIDIA Tesla K80 GPU. Training takes approximately one to two days per model.

Iteration 0 Iteration 1 Iteration 3

Figure 5. Improved *DESIRE-SI* prediction samples (red) over iterations. Iterative regression refines the predictions closer to the ground truth future trajectory (blue) matching with scene context.

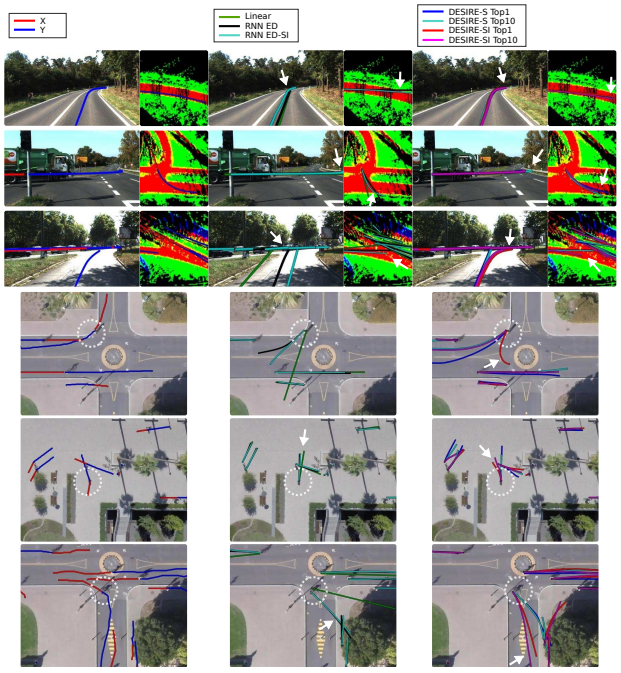
4.4. Analysis

Table 1 and Fig. 4 compare the oracle prediction errors[3] of various methods. We present L2 distance error for both datasets and miss-rate with $1m$ threshold for KITTI only, as trajectories in SDD are defined in image pixel space. Note that *Linear*, *RNN ED*, and *RNN ED-SI* output a single prediction, thus their results are shown as horizontal lines. *CVAE* samples are sorted randomly without confidence values.

Baselines: *RNN ED* performs significantly better than *Linear* since it can learn non-linear motion. We observe that *RNN ED-SI* performs worse than *RNN ED* on the KITTI since the model learns to behave *reactive* (see Fig. 6). This might be due to the small size of the dataset, which makes it hard to learn predictive CNN/interaction features (*i.e.*, features need to have high capacity to encode long-term information). On the contrary, *RNN ED-SI* significantly outperforms *RNN ED* on SDD dataset since SDD is much bigger and has a large number of interactions among agents.

Proposed models: With a single random sample (*CVAE* 1 in Table 1), *CVAE* performs worse than *RNN ED* since *RNN ED* directly optimizes for L2 distance during training. Given more than few samples (e.g., *CVAE* 10% in Table 1), *CVAE* outperforms *RNN ED* quickly on both datasets, which confirms the multi-modal nature of the prediction problem. *DESIRE-X-IT0* without iterative regression properly ranks the random *CVAE* samples achieving lower error with few samples. Note that *DESIRE-X-IT0* only ranks the samples without regression, thus achieves the same error as used all samples, *i.e.*, at Top K ratio of 1.0 in Fig. 4. As we iterate over, the outputs get refined and achieve smaller oracle error (*i.e.*, *DESIRE-X10%-IT0* vs. *DESIRE-X10%-IT4*). Fig. 5 shows an example of the iterative feedback. Finally, we observe that considering the interaction between agents further helps to achieve lower error. The difference between

[3]The maximum error in Table 1 might be different from Fig. 4 due to the test examples without ground truth labels at 4 seconds in the future.



(a) GT (b) Baselines (c) DESIRE

Figure 6. **KITTI results** (top 3 rows): The row 1&2 in (b) show highly *reactive* nature of *RNN ED-SI* (*i.e.*, prediction turns after it hits near non-drivable area). On the contrary, *DESIRE* shows its long-term prediction capability by considering potential future rewards. *DESIRE-SI* also produces more convincing predictions in the presence of other vehicles. **SDD results** (bottom 3 rows): The row 4 shows the multi-modal nature of the prediction problem. While the cyclist is making a right turn, it is also possible that he turns around the round-about (denoted with arrow). *DESIRE-SI* predicts such equally possible future as the top prediction, while covering the ground truth future within top 10 predictions. The row 5&6 also show that *DESIRE-SI* provides superior predictions by reasoning about both static and dynamic scene contexts.

DESIRE-S and *DESIRE-SI* is smaller in KITTI experiment, since KITTI has only few interactions between cars. However, we observe clear improvement on the SDD dataset since there are rich set of scenes with interactions between agents. Although our model with top 1 sample (*DESIRE Best*) achieves higher error compared to the direct regression baselines, using a few more samples yields much better prediction accuracy (*i.e.*, *DESIRE* 10%). Note that direct regression models with lower error are not necessarily better if averaging various futures (*e.g.*, going straight). We believe that in some applications, probabilistic prediction over a variety of outcomes is more desirable than a single MAP prediction. For both datasets, *DESIRE* achieves error on par with best baselines using as little as top 2 samples of *DESIRE-SI-IT4* predictions (see Fig. 4). Qualitative results are presented in Fig. 6 and in the supplementary material.

Ablative study: We conduct further experiments for varying K and past length to supplement the main experiments and report the results in Table 2 and Table 3.

Method	1.0 (sec)	2.0 (sec)	3.0 (sec)	4.0 (sec)
KITTI (error in meters / miss-rate with 1 m threshold)				
Linear	0.89 / 0.31	2.07 / 0.49	3.67 / 0.59	5.62 / 0.64
RNN ED	0.45 / 0.13	1.21 / 0.39	2.35 / 0.54	3.86 / 0.62
RNN ED-SI	0.56 / 0.16	1.40 / 0.44	2.65 / 0.58	4.29 / 0.65
CVAE 1	0.61 / 0.22	1.81 / 0.50	3.68 / 0.60	6.16 / 0.65
CVAE 10%	0.35 / 0.06	0.93 / 0.30	1.81 / 0.49	3.07 / 0.59
DESIRE-S-IT0 Best	0.53 / 0.17	1.52 / 0.45	3.02 / 0.58	4.98 / 0.64
DESIRE-S-IT0 10%	0.32 / 0.05	0.84 / 0.26	1.67 / 0.43	2.82 / 0.54
DESIRE-S-IT4 Best	0.51 / 0.15	1.46 / 0.42	2.89 / 0.56	4.71 / 0.63
DESIRE-S-IT4 10%	**0.27** / **0.04**	**0.64** / 0.18	**1.21** / 0.30	2.07 / 0.42
DESIRE-SI-IT0 Best	0.52 / 0.16	1.50 / 0.44	2.95 / 0.57	4.80 / 0.63
DESIRE-SI-IT0 10%	0.33 / 0.06	0.86 / 0.25	1.66 / 0.42	2.72 / 0.53
DESIRE-SI-IT4 Best	0.51 / 0.15	1.44 / 0.42	2.76 / 0.54	4.45 / 0.62
DESIRE-SI-IT4 10%	0.28 / 0.04	0.67 / **0.17**	1.22 / **0.29**	**2.06** / **0.41**
SDD (pixel error at 1/5 resolution)				
Linear	2.58	5.37	8.74	12.54
RNN ED	1.53	3.74	6.47	9.54
RNN ED-SI	1.51	3.56	6.04	8.80
CVAE 1	2.51	6.01	10.28	14.82
CVAE 10%	1.84	3.93	6.47	9.65
DESIRE-S-IT0 Best	2.02	4.47	7.25	10.29
DESIRE-S-IT0 10%	1.59	3.31	5.27	7.75
DESIRE-S-IT4 Best	2.11	4.69	7.58	10.66
DESIRE-S-IT4 10%	1.30	2.41	3.67	5.62
DESIRE-SI-IT0 Best	2.00	4.41	7.18	10.23
DESIRE-SI-IT0 10%	1.55	3.24	5.18	7.61
DESIRE-SI-IT4 Best	2.12	4.69	7.55	10.65
DESIRE-SI-IT4 10%	**1.29**	**2.35**	**3.47**	**5.33**

Table 1. Prediction errors over future time steps on KITTI and SDD datasets. Our method, *DESIRE-IT4*, achieves by far the lowest top 10% error, addressing the multimodal nature of the task effectively.

Method	K (the number of prediction samples)			
	25	50	100	200
DESIRE-S-IT4 Best	4.87	4.71	4.81	4.70
*DESIRE-S-IT4 top*20	2.03	2.04	1.99	1.96

Table 2. Prediction errors of *DESIRE-S-IT4* on KITTI at $4s$ for varying K. The best sample errors remain similar, while top 20 oracle errors decrease slightly as K increases.

Method	Time length for past (sec)		
	1.0	2.0	4.0
DESIRE-S-IT4 Best	4.94	4.71	4.78
DESIRE-S-IT4 10%	2.11	2.07	2.05

Table 3. Prediction errors of *DESIRE-S-IT4* on KITTI at $4s$ for varying time length for past trajectory. The model trained with $1s$ past slightly worse than ours ($2s$), showing that 2 second past contains enough cues to encode motion context. Note also that prior works adopt similar past lengths (2.8s in [3, 36])

5. Conclusion

We introduce a novel framework *DESIRE* for distant future prediction of multiple agents in complex scene. The model incorporates both static and dynamic scene contexts with a deep IOC framework and produces stochastic, continuous, and time-profiled long-term predictions that can effectively account for the uncertainty in the future prediction task. Our empirical evaluations on driving and surveillance scenarios demonstrate clear improvement over other baselines. For future work, we believe that our model can be further improved on larger datasets and be applied to various robotics applications with a direct use of perspective images.

Acknowledgement

This work was part of N. Lee's summer internship at NEC Labs America and also supported by the EPSRC, ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1.

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004. 2

[2] H. Akaike. Fitting autoregressive models for prediction. *Annals of the institute of Statistical Mathematics*, 21(1):243–247, 1969. 2

[3] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971, 2016. 1, 3, 6, 8

[4] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 3

[5] L. Ballan, F. Castaldo, A. Alahi, F. Palmieri, and S. Savarese. Knowledge transfer for scene-specific motion prediction. *arXiv preprint arXiv:1603.06987*, 2016. 1, 3

[6] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, And Cybernetics-Part C: Applications and Reviews, 38 (2), 2008*, 2008. 3

[7] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 3, 5

[8] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 3

[9] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015. 3

[10] A. D. Dragan, N. D. Ratliff, and S. S. Srinivasa. Manipulation planning with goal sets using constrained trajectory optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4582–4588. IEEE, 2011. 2

[11] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. *arXiv preprint arXiv:1603.00448*, 2016. 2, 5

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 6

[13] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013. 3

[14] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995. 3

[15] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3, 5

[16] J. Hu and M. P. Wellman. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov):1039–1069, 2003. 3

[17] A. Jain, A. Singh, H. S. Koppula, S. Soh, and A. Saxena. Recurrent neural networks for driver activity anticipation via sensory-fusion architecture. In *International Conference on Robotics and Automation (ICRA)*, 2016. 1, 3, 7

[18] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960. 2

[19] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015. 3

[20] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[21] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014. 3, 4

[22] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3, 4

[23] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *European Conference on Computer Vision*, pages 201–214. Springer, 2012. 1, 2

[24] J. F. P. Kooij, N. Schneider, F. Flohr, and D. M. Gavrila. Context-based pedestrian path prediction. In *European Conference on Computer Vision*, pages 618–633. Springer, 2014. 3

[25] H. Kretzschmar, M. Kuderer, and W. Burgard. Learning to predict trajectories of cooperatively navigating agents. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4015–4020. IEEE, 2014. 3

[26] N. Lee and K. M. Kitani. Predicting wide receiver trajectories in american football. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016. 1, 2

[27] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. 3

[28] J. Mainprice, R. Hayne, and D. Berenson. Goal set inverse optimal control and iterative re-planning for predicting human reaching motions in shared workspaces. *arXiv preprint arXiv:1606.02111*, 2016. 2

[29] P. McCullagh and J. A. Nelder. *Generalized linear models*, volume 37. CRC press, 1989. 2

[30] H. S. Park, J.-J. Hwang, Y. Niu, and J. Shi. Egocentric future localization. 2016. 3

[31] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268. IEEE, 2009. 3

[32] M. B. Priestley. Spectral analysis and time series. 1981. 2

[33] J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005. 2

[34] C. E. Rasmussen. Gaussian processes for machine learning. 2006. 2

[35] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014. 3

[36] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European Conference on Computer Vision*, pages 549–565. Springer, 2016. 3, 7, 8

[37] R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *AISTATS*, volume 1, page 3, 2009. 3

[38] S. Shalev-Shwartz, N. Ben-Zrihem, A. Cohen, and A. Shashua. Long-term planning by short-term prediction. *arXiv preprint arXiv:1602.01580*, 2016. 3

[39] Y. Shoham and K. Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008. 3

[40] Y. Shoham, R. Powers, and T. Grenager. If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171(7):365–377, 2007. 3

[41] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015. 2, 3, 4

[42] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014. 3

[43] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. 5

[44] E. Thibodeau-Laufer, G. Alain, and J. Yosinski. Deep generative stochastic networks trainable by backprop. 2014. 3

[45] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating visual representations from unlabeled video. 2016. 1

[46] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision*, pages 835–851. Springer, 2016. 1, 3, 5, 7

[47] J. Walker, A. Gupta, and M. Hebert. Patch to the future: Unsupervised visual prediction. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3302–3309. IEEE, 2014. 2

[48] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):283–298, 2008. 2

[49] C. K. Williams. Prediction with gaussian processes: From linear regression to linear prediction and beyond. In *Learning in graphical models*, pages 599–621. Springer, 1998. 2

[50] M. Wulfmeier, D. Z. Wang, and I. Posner. Watch this: Scalable cost-function learning for path planning in urban environments. *arXiv preprint arXiv:1607.02329*, 2016. 2, 5

[51] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2(3):5, 2015. 3

[52] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008. 2