Socially-Aware Trajectory Prediction Guided by Motion Patterns

Ingrid Navarro-Anaya

CMU-RI-TR-22-38

August, 2022

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Dr. Jean Oh, *Chair*
Dr. Sebastian Scherer
Dr. Ji Zhang
Dr. Luis Ernesto Navarro-Serment
Jay Patrikar

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Abstract

As intelligent robots across domains start collaborating with humans in shared environments, *e.g.*, urban settings and airspace, algorithms that enable them to reason over human motion and intent are important to ensure seamless and safe interplay. Even beyond robotics, other domains, *e.g.*, surveillance and sports analysis, may also benefit from this type of algorithms.

In our work, we study human intent by focusing on the problem of predicting trajectories in dynamic environments. We are further interested in designing methods that are able to generalize across domains. Specifically, we target domains where navigation guidelines are relatively strictly defined yet not necessarily marked in their physical environments. We hypothesize that within these domains, in the short-term, agents tend to exhibit motion patterns that reveal important context information related to the agent's general direction, admissible motions, intermediate goals and social influences. From this intuition, we propose *Social-PatteRNN*, a new recurrent generative model that exploits motion patterns to encode the aforesaid context information and use it as conditioning signal for predicting trajectories. We assess our approach across three different problem domains: human motion in crowds, human motion in sports and aircraft motion in terminal airspace. Finally, we show that our approach achieves state-of-the-art results across these domains.

Acknowledgments

I would like to thank my advisor, Dr. Jean Oh, for giving me the opportunity to work with her and for the flexibility to explore various research subjects. Most importantly, I'm extremely grateful for her support, time and patience, especially during the moments where I felt the least confident in myself.

I would also like to thank Dr. Luis E. Navarro-Serment, Dr. John Dolan and Rachel Burcin for the opportunity to join the Robotics Institute Summer Scholars (RISS) program during my undergraduate studies, and for their constant encouragement and support over these past two years.

I'm very grateful also to Dr. Sebastian Scherer, Dr. Jon Francis, Jay Patrikar, Felix Labelle, Shaunak Halbe, Nariaki Kitamura and Xiaopeng Lu, for all of our project discussions and fruitful feedback.

Finally, I would like to thank my collaborators from the the Bot Intelligence Group (BIG) and the AirLab.

Funding

This work was supported, in part, by the Army Futures Command Artificial Intelligence Integration Center (AI2C) and the Ministry of Trade, Industry and Energy (MOTIE) and Korea Institute of Advancement of Technology (KIAT) through the International Cooperative R&D program: P0019782, Embeded AI Based fully autonomous driving software and Maas technology development.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Understanding and predicting the intended motion of dynamic agents in an environment is an important skill that autonomous robots across various domains, *e.g.*, social robotics [6, 22, 36], aerial robotics [27], and autonomous driving [8], must be equipped with in order to enable seamless and safe interactions. Even beyond robotics, other domains such as surveillance [25] and sports analysis [1, 23] may also benefit from algorithms capable of modeling how different agents behave in dynamic environments.

Motion and intent can be influenced by several factors which make the prediction task difficult. Three factors which are of particular interest in this work are; social behavior, motion constraints, and goals. Social behavior dictates how agents within the same setting behave and interact, and as such, it heavily depends on the context. For instance, in a sports-based setting, *e.g.*, a basketball game, close proximity to other agents may be an acceptable behavior, whereas this may not be true in an urban setting, *e.g.*, pedestrians on a side walk. In general, humans are naturally skilled to understand these, and even more nuanced contexts [34]. However, designing algorithms with the ability to reason about social context is yet a challenging task.

Respecting motion constraints is another relevant problem within this setting, as it is related to restrictions that may arise from the agent's own physical constraints or the constraints imposed by its environment, *e.g.*, the topology of a scene and the rules associated with it. Few existing works have accounted for agent's dynamic and

motion constraints and environmental information [26, 27, 33, 34].

Finally, another challenge involves inferring agent's goals. This is a difficult task because unless there's explicit communication of intent for a given agent, its actual goal is practically unknown to all other agents interacting with it. Recent works show promising results by considering the goal context in their predictions [20, 21, 23] but often follow assumptions that do not generalize across domains.

1.2 Social-PatteRNN

Motivated by the aforesaid, this work focuses on the problem of trajectory prediction in multi-agent settings.

Throughout this work, we refer to our approach as *Social-PatteRNN*. Briefly, *Social-PatteRNN*, is a generative model that follows an encoder-decoder design trained in an end-to-end manner to predict multi-future trajectories. It has a modular design that allows to break down the prediction problem into sub-tasks dedicated to learn from contextual information providing a more intuitive learning and prediction process. Furthermore, the model can be easily applied to various spatial navigation domains, *e.g.*, human motions in a 2D space or aircraft navigation in a 3D space, as well as, to easily add different contextual information.

As we will further explain below, our main interest is to explore pattern information as a mechanism for (1) generalizing across domains, and (2) extracting context that encodes social behavior, admissible and sub-goals.

Why generalizing across domains? The majority of existing works on trajectory prediction are mainly focused on modeling pedestrian behavior [2, 16, 40, 42]. Developing models for trajectory prediction that can be applied across multiple domains is challenging and less explored [27]. In this work, we conjecture that there are multiple domains where there exist navigation guidelines relatively strictly defined yet not necessarily marked in their physical environments. Figure 1.1, showcases this idea with a few examples. For instance, in aerial navigation, although there are strict rules analogous to ground navigation, the air space does not display visible lanes. Similarly for various scenarios within crowd navigation and sports, *e.g.*, moving around a museum, or a mall, the environment may be laid out but does not necessarily marking

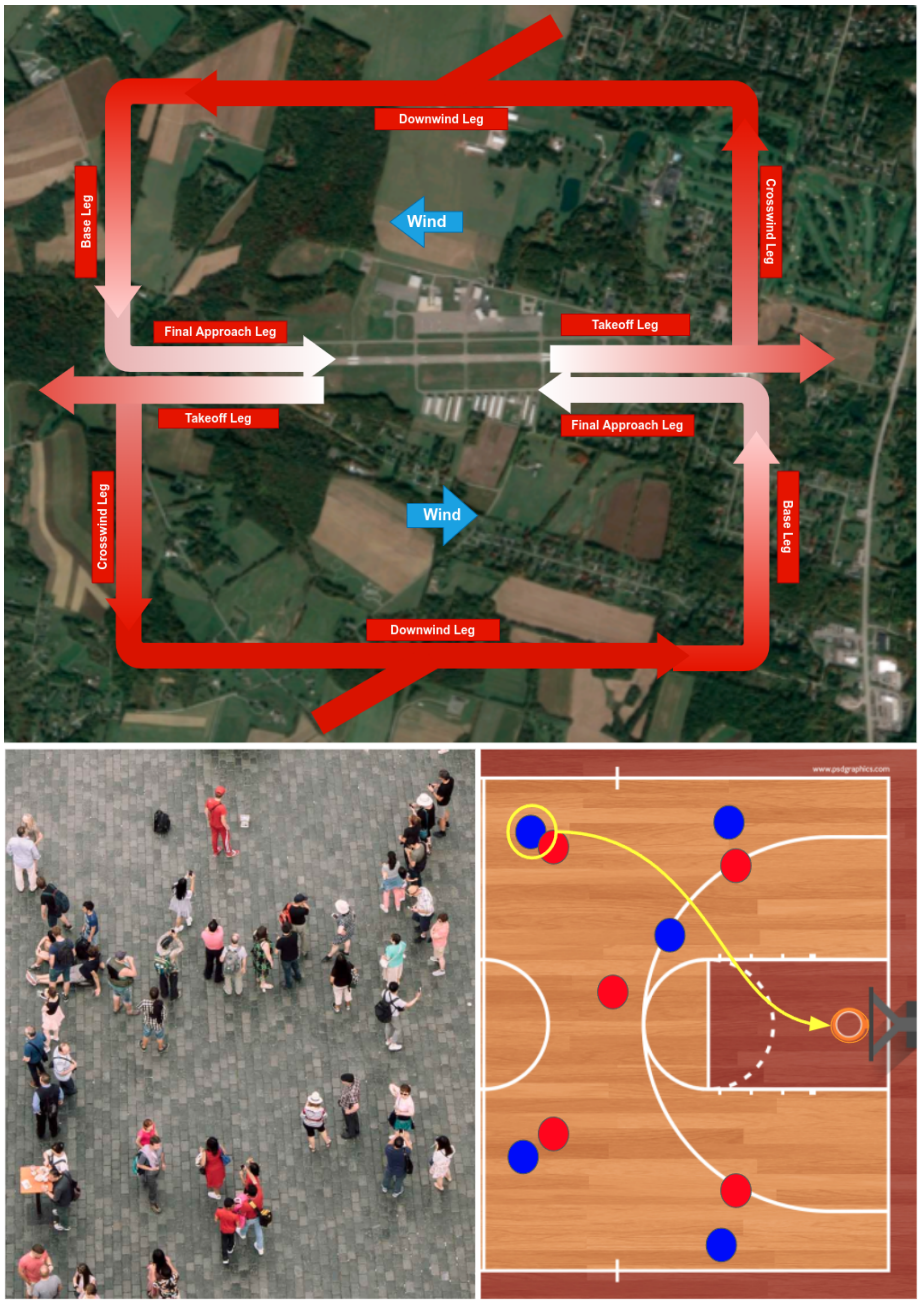


Figure 1.1: Examples where agents may exhibit patterns of motion in environments where topology and motion is loosely defined or not physically marked. **Top:** traffic patterns in airspace. **Bottom:** Humans in crowds (left) and sports settings (right).

how humans should move. The latter rises the interest of this work to design a general algorithm for learning motion in these types of contexts.

Why using motion patterns as context? Inspired by [42], we hypothesize that, irrespective of the domain, in the short term, agents tend to exhibit some motion patterns that reveal both their general directions of motion and their intermediate goals. We also assume that such patterns encompass admissible motions, as well as, general rules of motion, which may be explicit, *e.g.*, sports rules or flying guidelines, or implicit, *e.g.*, social etiquette. From these intuitions, we propose a data-driven approach to learn these patterns of motion and use them as a conditioning signal for predicting multimodal trajectories. Then, we use the learned patterns to extract sub-goal information which we aggregate to our model as the social context.

Lastly, to assess the effectiveness of the proposed approach in predicting the trajectories of multiple agents across different domains, we evaluate our approach on three different scenarios: motion in crowds [23, 30], sports [23], and in non-terminal space [27]. The experimental results show that the proposed approach has consistent performance across these three domains.

1.3 Thesis Organization

The following chapters are organized as follows:

- Chapter 2 reviews and summarizes existing work in this field;
- Chapter 3 provides the problem formulation and describes the components of *Social-PatteRNN*;
- Chapter 4 compiles the set of experiments that were carried out for assessing *Social-PatteRNN*'s performance;
- Chapter 5 analyses the results obtained on the experiments, and;
- Chapter 6 provides final discussions and ongoing and future work.

Chapter 2

Related Work

2.1 Trajectory Prediction across Domains

The majority of research in multi-agent trajectory prediction has focused on pedestrian behavior [2, 16, 40, 42]. For benchmarking trajectory prediction algorithms within the pedestrian setting, datasets such as UCY [19], ETH [28], Stanford Drone Dataset (SDD) [30] among multiple others [31] have been employed. These datasets generally capture social behavior from a top-down view perspective in highly dynamic urban environments. Beyond predicting pedestrian behavior in urban contexts, STATS SportsVU [1], leverages a camera system that uses computer vision and statistical algorithms to capture human behavior in sports-based settings like basketball and football.

Trajectory prediction has also been widely explored within the autonomous vehicles (AV) domain where well-known datasets such as KITTI [14], CityScapes[12], Argoverse [9], nuScenes [7] and inD [5] have been used. In contrast to pedestrian-focused datasets, these datasets generally leverage sensor suites enabling them to capture much richer, multi-modal information including detailed maps, natural language [13], 2D and 3D semantic annotations, and trajectory information for different types of agents (*e.g.*, pedestrians, vehicles and bikers).

More recently, operation in non-towered airspace has been framed within the context of social navigation. In this particular setting, pilots are required to communicate with other pilots and self-organize following flying guidelines. To encourage the study

of social interaction within this domain, [27] introduced TrajAir, a large-scale dataset comprising weather data and 3D trajectory information capturing the interactions of pilots operating in a non-towered airport.

In this work, we explore whether our method is capable of generalizing across domains. As such, we train and evaluate on datasets for all of the aforementioned domains and provide insights for our results. We choose datasets where data was captured from birds-eye view perspective and focus on a single modality, *i.e.*, trajectory data. Thus, we select the following datasets: SDD [30] for pedestrian behavior, NBA [1] for sports, inD [5] for autonomous driving contexts, and TrajAir [27] for aircraft. We provide further details regarding these datasets and our insights in Chapter 4.

2.2 Modeling Techniques in Trajectory Prediction

Early work on trajectory prediction in pedestrian-based environments leveraged domain knowledge about social behavior to tackle this problem [17, 37, 38]. However, as has been pointed out by researchers within the field [2, 16, 23, 40, 42], models that leverage hand-crafted features have limited capability for generalizing to complex scenarios as well as different domains. The latter has motivated researchers to gear toward data-driven approaches where, instead, social behavior is learned from the data [2, 16, 23, 42].

Within this setting, three main components are generally taken into account which involve modeling (1) the temporal, (2) the multi-modal, and (3) the interactive or social aspect of trajectories in multi-agent contexts.

For modeling the temporal nature of trajectories, researchers have relied on mechanisms designed for exploiting the sequential dependencies of the data. Three widely used methods include Recurrent Neural Networks (RNNs) [2, 20, 21, 23], Temporal Convolutional Networks (TCN) [24, 27, 42] and Transformers [15, 41]. RNNs have been a prominent method for sequence modeling with a wide range of applications [3]. As such, they have generally been the preferred temporal mechanism in motion prediction tasks. Alternatively, TCN-based methods have recently been gaining more popularity within sequence-dependent tasks, achieving comparable performance to RNN-based techniques [3]. More recently, the ability of Transformers [39] to capture long-range sequences has motivated its use in trajectory prediction

tasks [15, 41].

Multiple future scenarios can be plausible given the same contextual information, *e.g.*, two pedestrians walking towards each other can make multiple decisions to avoid colliding with each other. A second component within trajectory prediction, thus, relates to modeling the multi-modality of trajectories. In order to account for the uncertainty of predicting future behavior, researchers have adopted generative frameworks such as the Conditional Variational Autoencoder (CVAE) [4, 18, 23, 27], as well as Generative Adversarial Networks (GANs) [16, 33, 43]. CVAEs are built upon the VAE [29], a type of model which learns to reconstruct input data by projecting it into a latent representation which is then used for reconstruction. The CVAE further allows to use conditioning variables as a method for controlling the learning process. CVAEs have been effectively employed across trajectory prediction domains [21, 23, 27] as they are well-suited for modeling multi-modality while conditioning on contextual information, *e.g.*, interactions, goals [21, 23], maps [34], and weather [27]. Alternatively, GANs do not explicitly learn a latent space. Rather, they follow a min-max objective where the aim is to directly learn to generate data points such that they are indistinguishable from the ground truth. As such, [16, 33] have trained GAN-based models to predict diverse multi-modal predictions.

The third component we listed above involves aggregating and modeling the interactions between agents. Some works have proposed learning interactions through pooling mechanisms [2, 16, 21]. For instance, [2] uses a neighboring-based pooling over the hidden states of a given set of agents. However, as explained in [42], one drawback to this type of mechanism lies in the difficulty of understanding the semantic representation of hidden states generated by RNNs. Recent works have geared toward attention-based mechanisms for encoding interactions [20, 23, 41], *e.g.*, Graph Attention Networks (GAT) [20, 23, 27], additive attention [34], and Muti-Head Attention (MHA) [41], all of which have shown promising results.

Inspired by state-of-the-art models for trajectory prediction [23, 34], our work is based on the a recurrent version of the CVAE, known as VRNN [11]. In particular, this model leverages the Gated Recurrent Units (GRU) [10], an efficient version of the RNN which exhibits similar performance compared to the TCN [3] as the mechanism for temporal modeling. In Section 3.2.1, we provide further details on the VRNN, and in Section 3.2.3 we describe the mechanism used for learning and

encoding social interactions. Finally, we account for social interactions by encoding the displacement matrices between the current agent's state and the current prediction of the next sub-goal extracted from the motion patterns, and then we attend over the displacements through a a Multi-Head Attention (MHA). We discuss our interaction method in Section 3.2.3.

2.3 Exploiting Motion Patterns in Trajectory Prediction

Finally, another approach that also leverages the use of motion patterns for trajectory prediction tasks is Social Pattern Extraction Convolution (*S-PEC*) [42]. In particular, the *S-PEC* algorithm begins by randomly initializing and distributing a set of patterns within an agent's ego-centric scene. Then, the algorithm is trained to detect patterns in the data and match the local scene to said patterns. Finally, for inference, it uses the learned scene and projects new trajectories based on the similarities between the patterns.

Through experimentation with the algorithm we noticed that within motions in human crowds this idea works well because pedestrians tend to move linearly and more slowly. However, we observed that this model exhibited difficulty generalizing to different spatial domains, *e.g.*, navigating in 3D space, or where motion is highly dynamic, *e.g.*, sports. Another drawback to this approach is that at every step it requires transforming the global scene to each agent's egocentric frame in order to predict the next location. The latter is becomes a computationally expensive operation as the number of agents in a scene grows. Finally, in order to create the local scene, the algorithm relies on various hyper-parameters, and performance is highly sensitive to this initialization.

Nonetheless, inspired by this idea, we propose a different approach to leverage pattern information within the data, and we believe it addresses the limitations listed above. Briefly, rather than learning an abstract representation of a scene, we equip a trajectory prediction algorithm with a sub-module that directly learns to predict a motion pattern relative to the agent's current location and current state information, *e.g.*, other agent's locations and a hidden state summarizing the past. In this manner,

the algorithm does not require transforming the scene to another frame of reference, nor is it sensitive to initialization. In our approach, motion patterns are used as context to inform the agent about its general direction as opposed to enforce that motion. The reason behind this, is to account for dynamic and unexpected changes in the scene that may require the agent to deviate from the prior intended direction. As we will show in Chapter 4, through this mechanism, we were able to obtain consistent results across different domains.

Chapter 3

Approach

We consider the problem of predicting the distribution of feasible future trajectories of multiple agents in a shared environment, given both, the observations of their past trajectories and contextual information. In this chapter, we first formulate the problem as estimating the conditional probability distribution over future trajectories, and then we provide the technical details of the proposed approach.

3.1 Problem Formulation

Let the state of an agent i at time-step t, denoted by $\mathbf{x}_t^i$, represent the coordinates localizing the agent, *e.g.*, $\mathbf{x}_t^i = (x, y, z)_t^i$ in a 3D environment. The trajectory of a given agent i is then defined as a sequence of states from time-step $t = 1$ to some ending time-step $t = T$, *i.e.*, $\mathbf{X}^i = [\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_T^i]$, where T denotes the total length of the trajectory. In this work, we transform the trajectories to relative motions, where a relative motion is a displacement vector between two consecutive steps, $\mathbf{x}_t^i$ and $\mathbf{x}_{t+1}^i$, *e.g.*, $\hat{\mathbf{x}}_\mathbf{t}^\mathbf{i} = \mathbf{x}_{t+1}^i - \mathbf{x}_t^i$. For simplicity, we keep the notation $\mathbf{x}_t^i$ and $\mathbf{X}^i$ to refer to relative motions and their corresponding trajectories.

In this problem setup, a trajectory is split into two segments; a history segment of length H, $\mathbf{X}_H^i = [\mathbf{x}_1^i, \ldots, \mathbf{x}_H^i]$, and a future segment of length F, $\mathbf{X}_F^i = [\mathbf{x}_{H+1}^i, \ldots, \mathbf{x}_F^i]$. Furthermore, we define the current context, which represents additional information that can be inferred from the observations, learned from previous experiences, or provided from external sources. Let $\mathbf{c}_t^i$ and $\mathbf{C}_H^i = [c_1^i, \ldots, c_H^i]$ denote the context

at time-step t and the context history, respectively. In the proposed approach, for instance, the context refers to short-term motion patterns as well as social awareness relative to other agents which are both learned from training data.

In this paper, the trajectory prediction problem is defined as finding the distribution of the future trajectories $\hat{\mathbf{X}}_F^i$ for each agent $i = \{1, \ldots, N\}$ in a scene, given their past trajectories and the context history. Formally,

$$\hat{\mathbf{X}}_F^i \sim p(\mathbf{X}_F^i | \mathbf{X}_H^i, \mathbf{C}_H^i) \quad \forall i \in \{1, \ldots, N\}.$$

3.2 Social-PatteRNN

The proposed approach, *Social-PatteRNN*, is an end-to-end model which we illustrate in Figure 5.8. Following prior work on trajectory prediction [20, 21, 23], we adopt a recurrent variational encoder-decoder as the backbone of the model which is shown as Ⓒ **VRNN** in the figure. This module takes features from the Ⓑ **Location Extractor**, the Ⓓ **Context Module**, and the Ⓒ2 **RNN**'s hidden state to learn to reconstruct the input displacements during training. During inference, it only relies on the context and the hidden state to generate new samples.

Our novelty here is focused on how the context information is learned and used for prediction, *i.e.*, in terms of motion patterns and social influences. Specifically, we propose a motion pattern learning module, Ⓓ1 **PatternNet**, that learns to predict short motion patterns given previous context summarized by the recurrent network. The motion patterns are intended to serve as guidance for predicting the next location as opposed to be enforced as the next set of motions. The predicted patterns are further used to reflect social influences by the interaction module, Ⓓ2 **InteractionNet**. This module extracts, attends, and aggregates social context in the form of sub-goals relative to each agent.

Our model can be seamlessly adapted to various spatial navigation domains, *e.g.*, human motions in a 2D space or aircraft navigation in a 3D space, as well as, to easily add different context information. The sections that follow describe each of these components in more detail.

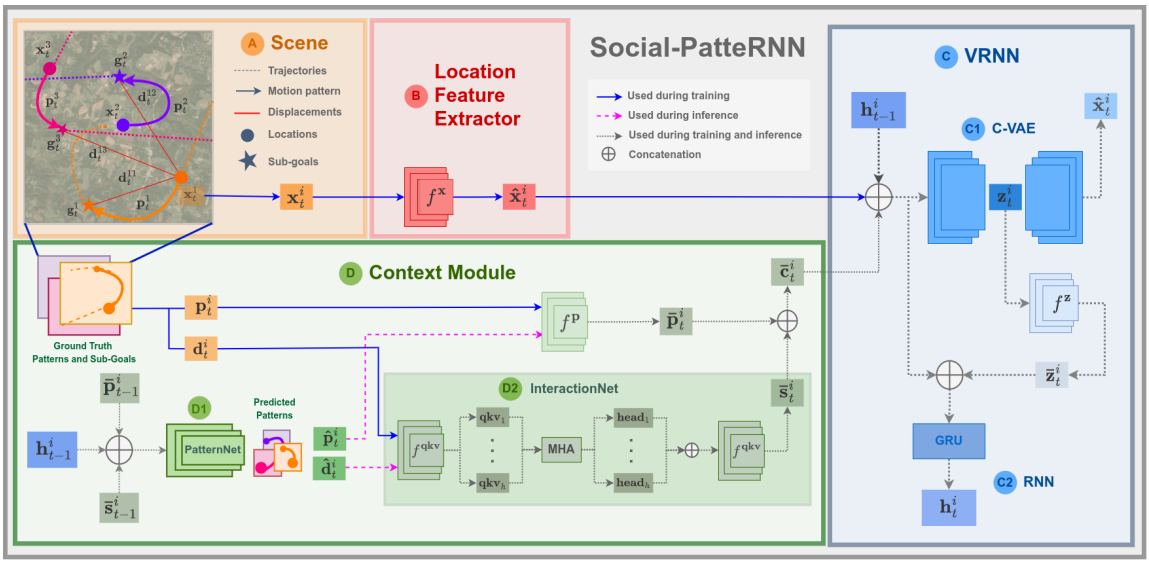


Figure 3.1: An overview of *Social-PatteRNN*. We show the prediction process for agent i at time-step t. (A) **Scene** represents a ground truth scene showing the locations, patterns and sub-goals of all agents. (B) **Location Extractor** is used for extracting current location features. (D) **Context Module** is the context module comprised by (D1) **PatternNet** which predicts the current motion pattern and (D2) **InteractionNet** which aggregates social context based on agents' sub-goals obtained from the predicted patterns. Finally, (C) **VRNN** comprises the backbone of the model; a VRNN consisting of a (C1) **C-VAE**, used for generating displacements given the features computed by (B) and (C), and a (C2) **RNN**, for encoding temporal state representations.

3.2.1 Backbone

To model the sequential nature of trajectory data and its multi-modality, we adopt the recurrent version of the C-VAE, *i.e.*, the Variational Recurrent Neural Network (VRNN) [11]. In Figure 5.8, we label the components of the VRNN as (C1) **C-VAE**, and (C2) **RNN**.

On one hand, C-VAEs [35] are an extension to the VAE [29], a type of generative model that is generally well-suited for representing multi-modal distributions which capture the variations of the input data into a latent representation. C-VAEs additionally allow to condition on context variables as a method for controlling the data generation process. RNNs, on the other hand, model sequential dependencies in the data, capturing hidden information from previous inputs and feeding it to

subsequent ones.

The VRNN module undergoes the following training process; first, it takes a relative location at time-step t for some agent i, $\mathbf{x}_t^i$, and encodes it into a latent representation, $\mathbf{z}_t^i$. Next, the latent representation is decoded into, $\hat{\mathbf{x}}_t^i$, representing the reconstructed input location. This is done for the entire history sequence $\mathbf{X}_H^i$ while also summarizing the hidden state representation at each step t, $\mathbf{h}_t^i$. After training, the prediction paradigm works as follows: the VRNN receives a history segment as before. Upon processing all history points, it obtains a final hidden state $\mathbf{h}_H$ which is used to kick-start the sampling process for generating a future sequence of length F, $\hat{\mathbf{X}}_F^i$, which is the predicted sequence of the ground truth future trajectory of the agent, $\mathbf{X}_F^i$.

The model consists of four main components, all of which are modeled as neural networks; the encoder (or posterior), $p(\cdot)$, the decoder (or likelihood), $q(\cdot)$, the prior, $p_{\mathbf{pr}}(\cdot)$, and the recurrent unit, $\text{RNN}(\cdot)$. We also incorporate additional feature extractor modules for the input data, the latent space and the context vectors, denoted $f^{\mathbf{x}}(\cdot)$, $f^{\mathbf{z}}(\cdot)$, and $f^{\mathbf{c}}(\cdot)$, respectively. We note that we do not explicitly implement $f^{\mathbf{c}}(\cdot)$, it rather refers to the modules introduced in Section 3.2.2 and Section 3.2.3 but here it is used to simplify notation.

Mathematically, we can express the model as,

$$\bar{\mathbf{x}}_t^i = f^{\mathbf{x}}(\mathbf{x}_t^i) \tag{3.1}$$

$$\bar{\mathbf{c}}_t^i = f^{\mathbf{c}}(\mathbf{c}_t^i) \tag{3.2}$$

$$\mathbf{z}_t^i \sim p(\mathbf{z}_t^i | \bar{\mathbf{x}}_t^i \oplus \bar{\mathbf{c}}_t^i \oplus \mathbf{h}_{t-1}) \quad \text{during training} \tag{3.3}$$

$$\mathbf{z}_t^i \sim p_{\mathbf{pr}}(\mathbf{z}_t^i | \bar{\mathbf{c}}_t^i \oplus \mathbf{h}_{t-1}) \quad \text{during inference} \tag{3.4}$$

$$\bar{\mathbf{z}}_t^i = f^{\mathbf{z}}(\mathbf{z}_t^i) \tag{3.5}$$

$$\hat{\mathbf{x}}_t^i \sim q(\mathbf{x}_t^i | \bar{\mathbf{z}}_t^i \oplus \hat{\mathbf{c}}_t^i \oplus \mathbf{h}_{t-1}) \tag{3.6}$$

$$\mathbf{h}_t^i = \text{RNN}(\bar{\mathbf{x}}_t^i \oplus \bar{\mathbf{c}}_t^i \oplus \bar{\mathbf{z}}_t^i \oplus \mathbf{h}_{t-1}) \tag{3.7}$$

where for all agents $i \in \{1, \ldots, N\}$ at a given time-step t, Equation (3.1) and Equation (3.2) represent the location and context feature extractor networks, respectively. We denote the extracted features with a bar symbol. Equation (3.3) and Equation (3.4) show the encoder and prior networks. These are used for learning a latent

space $\mathcal{Z}$ and for sampling a corresponding latent variable at some time-step t, $\mathbf{z}_t \in \mathcal{Z}$, conditioned on the extracted features and the hidden state from the RNN. The encoder is used only during training to guide the prior–which does not have access to the input data–toward the latent space, enabling it to generate future trajectories during the inference stage. The decoder network, expressed in Equation (3.6), is in charge of reconstructing the location $\hat{\mathbf{x}}_t$ conditioned on the latent variable and additional context. The RNN in Equation (3.7) produces the hidden representation of the current time-step using the extracted features and previous state information.

The loss function used for optimizing the model described above is expressed as follows:

$$\mathcal{L}^{cvae} = \sum_{t=1}^{H} \big[\log q(\mathbf{x}_t^i | \bar{\mathbf{z}}_t^i \oplus \bar{\mathbf{c}}_t^i \oplus \mathbf{h}_{t-1}) \quad - \\ D_{KL}(p(\bar{\mathbf{z}}_t^i | \bar{\mathbf{x}}_t^i \oplus \bar{\mathbf{c}}_t^i \oplus \mathbf{h}_{t-1}) \;||\; p_{\mathbf{pr}}(\bar{\mathbf{z}}_t^i | \bar{\mathbf{c}}_t^i \oplus \mathbf{h}_{t-1}))) \big]$$

The first term represents the log-likelihood for reconstructing the input location while the second one is the KL-divergence between the encoder and the prior distributions. For further details regarding this mathematical formulation and loss function, we refer the reader to [23, 29, 35].

In our approach, we condition the VRNN not only on the input data but also on the context features, $\bar{\mathbf{c}}_t^i = \bar{\mathbf{p}}_t^i \oplus \bar{\mathbf{s}}_t^i$. As will be described in Section 3.2.2 and Section 3.2.3, $\bar{\mathbf{c}}_t^i$ embeds features from short-term motion patterns, $\mathbf{p}_t^i$, and interactions between an agent i and all other agents in a scene, $\mathbf{s}_t^i$.

3.2.2 Learning Motion Patterns

Our approach is based on the intuition that social interactions in navigation are more local, short-term interactions rather than global, long-term ones. Whereas previous approaches [21, 23, 27] use the agents' long-term goals as the context to guide the process of learning the underlying distribution of the data, we use short-term motion patterns to represent social context instead.

A motion pattern can be represented as absolute coordinates, relative displacements, motion primitives, or control outputs such as accelerations that can be transformed to position information. In this section, we explore patterns that repre-

sent a sequence P relative displacements from a given location. An example of this is depicted in Ⓐ **Scene** in Figure 5.8. Here, we aim to predict a motion pattern from the current summary of the recurrent network and the previous pattern, as well as the previous' pattern features $\bar{\mathbf{p}}_t^i$. These features are obtained from $f^{\mathbf{p}}$ shown in Equation (3.8) which is part of the context feature extraction module defined in the problem definition:

$$\bar{\mathbf{p}}_t^i = f^{\mathbf{p}}(\mathbf{p}_t^i) \tag{3.8}$$

We refer to the pattern predictor as *PatternNet*, and it is defined by Equation (3.9).

$$\hat{\mathbf{p}}_t^i = PatternNet(\bar{\mathbf{p}}_{t-1}^i, \mathbf{h}_{t-1}^i) \tag{3.9}$$

This module is optimized such that, during inference it learns to predict short-term patterns that can be used to generate new samples. For doing so, we measure how close the predicted pattern resembles the ground truth pattern using the mean squared error (MSE) loss:

$$\mathcal{L}^{pat} = \sum_{t=1}^{H} MSE(\mathbf{p}_t^i, \hat{\mathbf{p}}_t^i)$$

3.2.3 Learning Social Interactions

In addition to using motion pattern features to guide the trajectory prediction process, we are interested in aggregating context relating to the interactions between agents. For this, we take advantage of our predicted patterns to extract short-term sub-goal information from all agents in a scene. To do so, for any given agent i in the scene, we deem the endpoint of its predicted pattern as their next sub-goal, $\mathbf{g}_t^i$. Then, we compute the relative displacement vectors between its current location $\mathbf{x}_t^i$ and its next sub-goal as well as the sub-goals of all other agents. We denote these displacement vectors as $\mathbf{d}_t^{ij}$, where i and j are the indices of the agents used to calculate the displacement, *i.e.*, $\mathbf{d}_t^{ij} = \mathbf{g}_t^j - \mathbf{x}_t^i$. As a result, for each agent we have a set of

displacements $\mathbf{d}_t^i = [\mathbf{d}_t^{i1}, \ldots, \mathbf{d}_t^{iN}]$ which we use to extract social context features:

$$\bar{\mathbf{s}}_t^i = f^{\mathbf{d}}(\mathbf{d}_t^i) \tag{3.10}$$

This module is also part of the feature context module in Equation (3.2) and is shown in Ⓓ **Context Module** in Figure 5.8.

To further capture the relationships between agents' sub-goals, we use multi-head attention (MHA) [39] to perform cross-agent attention over the displacement features and get a final social context vector $\bar{\mathbf{s}}_t^i$. In this process, sub-goal information is projected into h input queries $\mathbf{q}$, keys $\mathbf{k}$, and values $\mathbf{v}$ (Equation (3.11)). Then, the projections are transformed into h independent attention heads (Equation (3.12)). These heads are then combined together and linearly transformed to produce the attended social context vector (Equation (3.13)),

$$[(\mathbf{q}, \mathbf{k}, \mathbf{v})_1, \ldots, (\mathbf{q}, \mathbf{k}, \mathbf{v})_h] = f^{\mathbf{qkv}}(\mathbf{d}_t^i) \tag{3.11}$$

$$[\text{head}_1, \ldots, \text{head}_h] = \text{MHA}([\mathbf{q}, \mathbf{k}, \mathbf{v}]_{i=1}^h) \tag{3.12}$$

$$\bar{\mathbf{s}}_t^i = f^{\mathbf{d}}(\text{cat}([\text{head}_1, \ldots, \text{head}_h])) \tag{3.13}$$

Here, we refer to the above process as *InteractionNet*,

$$\bar{\mathbf{s}}_t^i = InteractionNet(\mathbf{d}_{t-1}^i) \tag{3.14}$$

In Chapter 4, we assess the effect of using context features with (Equation (3.10)) and without (Equation (3.13)) the attention mechanism, as well as, using the two pattern methods introduced in Section 3.2.2.

We can finally define the context feature vector introduced in Section 3.1 as $\bar{\mathbf{c}}_t^i = \bar{\mathbf{p}}_t^i \oplus \bar{\mathbf{s}}_t^i$. As shown in Figure 5.8, context features are both fed to the C-VAE backbone. We also feed the context vector to the previously introduced *PatternNet*. Thus, Equation (3.9) now becomes:

$$\hat{\mathbf{p}}_t^i = PatternNet(\bar{\mathbf{c}}_{t-1}^i, \mathbf{h}_{t-1}^i)$$

Finally, the full loss function for optimizing our model is,

$$\mathcal{L}^{total} = \mathcal{L}^{cvae} + \mathcal{L}^{pat}$$

Chapter 4

Experiments and Evaluation

4.1 Datasets

As we conveyed in Chapter 1, we are interested in assessing the performance of our algorithm across multiple domains. Thus, for our evaluation, we consider three datasets; (1) *TrajAir* [27], which consists of aircraft trajectories, (2) *Stanford Drone Dataset (SDD)* [30], featuring pedestrian behavior in urban scenarios; (3) *SportVU NBA Dataset (NBA)* [1], featuring human behavior in sports. Below, we provide further details on each dataset.

4.1.1 TrajAir

TrajAir [27] is a dataset consisting of 111 days of aircraft trajectory data collected in non-towered terminal space at the Pittsburgh-Butler Regional Airport. The trajectory data was captured using an Automatic Dependent Surveillance-Broadcast (ADS-B) receiver placed within the airport at 978MHx and 1090Mhz. The ADS-B system uses satellite navigation to produce accurate (x, y, z) locations.

In [27], the authors train their model on a subset of the dataset consisting of 28 days of data. We consider all of the 111 days of data; however, we partition the dataset differently. We observed that the dataset is heavily unbalanced toward single agent scenes (Table 4.1). In our work, we are interested in predicting the intent of aircraft when there are multiple other aircraft in the scene. Thus, upon analyzing

the dataset by number of agents per scene, we chose to train our models on a subset where scenes contain at least 4 agents.

Table 4.1: Number of train/validation samples (in thousands) by number of agents per scene using trajectories of $T = 140$s.

# agents	all	1	2	3	4	+4
# trajectories	425 / 177	298 / 177	90 / 42	26 / 14	9 / 3	2 / 0.2

4.1.2 Stanford Drone Dataset (SDD)

SDD [30] consists of top-down videos capturing social behavior in 8 different complex and crowded scenarios with different types of agents, *e.g.*, pedestrians, bikers, skateboarders, and vehicles, as well as environmental complexity including physical structures such as roundabouts. Similar to [23], we use the TrajNet benchmark [32] version of SDD. This version of the dataset features trajectories expressed in (x, y) world coordinates recorded at 2.5 FPS, and the number of agents in a scene can vary from 2 to 21.

4.1.3 SportVU NBA Dataset (NBA)

NBA consists of motion tracking data of basketball players collected with a SportVU system surrounding a basketball court [1]. The system provides top-down view of the players. Scenes within this dataset feature 10 players; 5 attackers and 5 defenders. Scenes are split into offensive plays, each starting once the ball crosses the middle of the court. Each scene consists of 50 time-step trajectories sampled at 5Hz, where trajectories are expressed in (x, y) world coordinates. We use the version of NBA as in [23], where trajectories are normalized and zero-centered and scenes develop to the right basket.

We note that in [23], models are trained on each team separately as they consider the nature of the teams to be intrinsically different. We train on both teams jointly as we argue that agent behaviors influence the interactions within their own team as well as the opponent team.

4.2 Evaluation Metrics

Following prior work on trajectory prediction [2, 16, 20, 21, 23, 27, 40, 42], we consider Average Displacement Error (ADE) and Final Displacement Error (FDE) as our metrics for assessing performance of our models. ADE measures the average Euclidean distance between the ground truth future sequence and the predicted one;

$$ADE = \frac{\sum_{i \in N} \sum_{t=H+1}^{t=T} \sqrt{((\hat{\mathbf{x}}_t^i - \mathbf{x}_t^i))^2}}{N \cdot F}$$

while FDE measures the average Euclidean distance between the endpoints of the sequence;

$$FDE = \frac{\sum_{i \in N} \sqrt{((\hat{\mathbf{x}}_T^i - \mathbf{x}_T^i))^2}}{N}$$

We report the best ADE and FDE values of $K = 20$ samples.

4.3 Baselines

We first assess our model by performing ablations on each of its components. As such, we define the following ablations or baselines;

1. **VRNN**: for this ablation, we remove the context module, *i.e.*, no social interactions, nor motion patterns are used as inputs to the model;
2. **VRNN + (S/L)-PAT**: for this ablation, we only remove the interaction-learning module to asses the performance of the pattern-learning module. Here, we also want to evaluate the effect of different pattern lengths. Here, we use S to refer to a short pattern, and L for a long one;
3. **VRNN + (S/L)-PAT + SOC-MLP**: for this ablation, we include both the pattern-learning module and social encoding, but we exclude the cross-agent attention mechanism, and;
4. **VRNN + (S/L)-PAT + SOC-MHA**: which comprises the full model described in Chapter 3.

We also compare our work to other models that are specifically related to this work;

1. **A-VRNN** [23]: an Attentive VRNN that learns interactions by encoding all agent's hidden states through a GAT;
2. **DAGNet** [23], an extension to the A-VRNN that adds a second GAT which is used for encoding one-hot encodings representing each agent's goals;
3. **TrajAirNet** [27], a TCN-based model designed for aircraft trajectory prediction, and which also leverages GATs for social encoding, and;
4. **SPEC** [42], a TCN-based baseline which uses the notion of motion patterns for encoding social interactions in pedestrian-based settings.

4.4 Implementation Details

As illustrated in Figure 5.8, our pipeline consists of three sub-modules; the VRNN, the location extractor, and the context module. Within these sub-modules, all of the feature extractors are implemented as fully-connected networks (FCN). Similarly, for he encoder and decoder networks in the C-VAE and the pattern learning network. The cross-agent attention mechanism uses 8 attention heads. And we use a Gated Recurrent Unit (GRU)[10] as the temporal encoding mechanism. We use Adam optimizer, we train for at most 1000 epochs with a learning rate of $1e-4$, and perform early stopping.

For the *SDD* and *NBA* datasets, we followed [23], where trajectories are split into $H = 8$ and $F = 12$, and $H = 10$ and $F = 40$, respectively. Since we carry an ablation on the pattern length, the pattern lengths for *SDD* are set to $P(L) = 6$ and $P(S) = 3$, for *NBA* are to $P(L) = 8$ and $P(S) = 4$. For the *TrajAir* dataset we take in trajectories of 140s, sampled every $5s$. This yields to trajectories of length $T = 28$ where we split the history and futures into $H = 8$ and $F = 20$, respectively. Pattern lengths are also set as $P(L) = 6$ and $P(S) = 3$.

Chapter 5

Results

5.1 Quantitative Analysis

5.1.1 Ablations

As explained in section 4.3, we validated the benefit of each component of the model by performing ablations. The corresponding results, summarized in Table 5.1, show that using the pattern learning module improves the results in term of displacement errors in the three datasets. Moreover, exploiting pattern information to aggregate *social* features, achieves further reductions in error. We note that the more pronounced improvements correspond to *TrajAir*'s results; where, adding pattern learning reduces ADE / FDE by $\sim 12/9\%$, while aggregating social features yields a further improvement of $\sim 5\%$. We find this specially relevant since in aircraft navigation, pilots are expected to follow flying patterns in addition to procuring safe interactions with other pilots [27]. We also observe that in the sports domain, where interactions are more frequent, aggregating social encoding achieves more significant improvements in ADE / FDE, of $\sim 11/14\%$, than only considering motion patterns, $\sim 3/2\%$.

Regarding the pattern length experiments, we observe that for both, *TrajAir* and *NBA*, the model performs better when the long patterns (L) are employed, while the performance for *SDD* is similar in both of the pattern length tests. We hypothesize that the latter may be related to the nature of these domains; whereas pedestrian

Table 5.1: The ablation results in ADE / FDE (↓).

		TrajAir (km)	**SDD (m)**	**NBA (ft)**
1	VRNN	0.660 / 1.433	0.605 / 1.181	9.176 / 14.375
2	+ PAT (S)	0.634 / 1.424	0.573 / 1.140	9.058 / 14.142
3	+ PAT (S) + SOC-MLP	0.613 / 1.394	0.566 / 1.126	8.821 / 12.728
4	+ PAT (S) + SOC-MHA	0.569 / 1.290	0.578 / 1.153	8.537 / 12.716
5	+ PAT (L)	0.590 / 1.331	0.587 / 1.177	8.877 / 14.110
6	+ PAT (L) + SOC-MLP	**0.548** / **1.179**	**0.552** / **1.099**	8.312 / 12.604
7	+ PAT (L) + SOC-MHA	0.562 / 1.295	0.565 / 1.118	**8.125** / **12.342**

motion and interactions in urban settings may be more *linear*, using short or long patterns may not make a significant difference in this setting. In contrast, more dynamic settings may benefit more from learning to model longer-term motion and interactions, *e.g.*, plays in sports, traffic patterns in aircraft navigation.

5.1.2 Main Results

Table 5.2 summarizes the results for the *socially*-aware baselines introduced in Section 4.3 and ours. We now show our models as *Social-PatteRNN-MLP* and *Social-PatteRNN-MHA* for the models without and with attention introduced in the previous section. These results show that our models outperform the corresponding baselines in both, *TrajAir* in terms of ADE / FDE by $\sim 28/21\%$ w.r.t to *TrajAirNet*, and *NBA* by $\sim 5/1\%$ w.r.t. *DAG-Net*. Finally, although for *SDD* dataset our model does not outperform the state-of-the-art model, *i.e.*, *DAG-Net*, it performs comparably with a difference of $\sim 4/5\%$ in the reported error metrics.

Another finding from our experimentation with different baselines is that their performance across different domains appears to degrade on domains in which they were not evaluated, suggesting sensitivity to the problem domain. For instance, *SPEC*, which was assessed on human crowds from the ETH/UCY [19, 28] benchmarks, significantly under-performs in the aerial navigation domain, as well as the sports domains. The latter may indicate that the algorithm is sensitive to different spatial domains, *i.e.*, 2D vs 3D, in addition to other aspects such as highly dynamic interactions, *e.g.*, sports, and longer-term prediction tasks. Similarly, *DAG-Net* exhibits lower

Table 5.2: Comparison against related models. Numbers shown as: ADE / FDE (↓).

		TrajAir (km)	SDD (m)	NBA (ft)
1	A-VRNN [23]	0.64 / 1.31	0.56 / 1.14	8.88 / 14.06
2	DAG-Net [23]	0.78 / 1.53	**0.53** / **1.04**	8.55 / 12.37
3	TrajAirNet [27]	0.77 / 1.50	0.92 / 1.68	9.91 / 15.27
4	S-PEC [42]	0.96 / 2.05	0.70 / 1.11	11.03 / 12.54
5	Social-PatteRNN-MLP (Ours)	**0.55** / **1.18**	0.55 / 1.10	8.47 / 12.40
6	Social-PatteRNN-MHA (Ours)	0.56 / 1.30	0.57 / 1.12	**8.10** / **12.05**

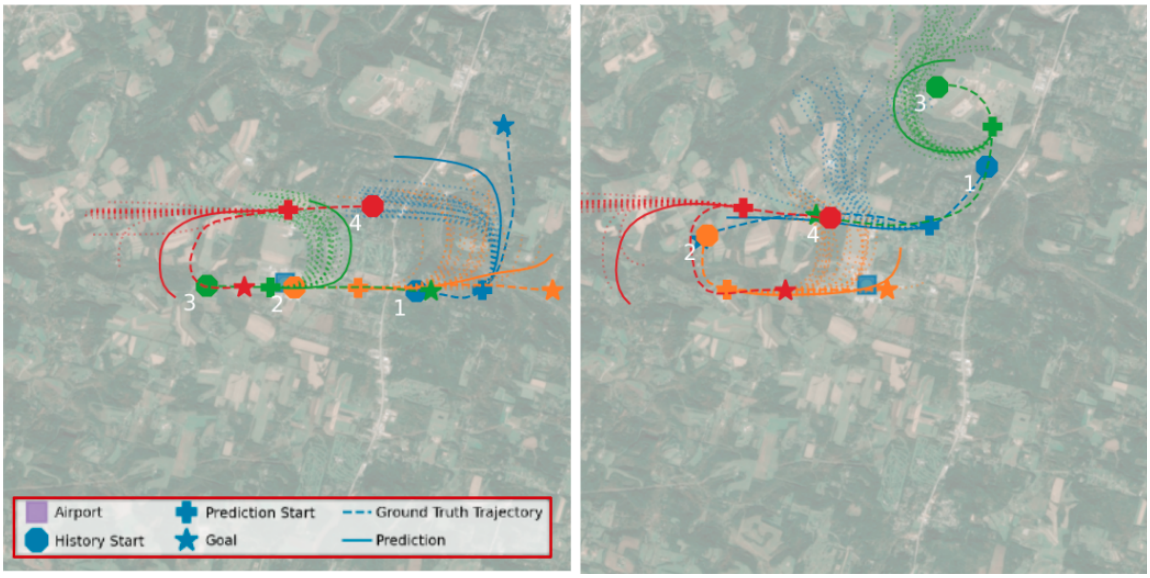
performance in the aerial domain in contrast to the other two domains, where their performance is akin to ours. One difference that may explain the latter, is that our approach not only focuses on *where* the agents intend to go, but also on *how* they intend to do it, *i.e.*, we explicitly condition and learn pattern information in addition to sub-goals, whereas *DAG-Net* conditions only on sub-goals. Finally, the performance of *TrajAirNet*, a model designed for long-term prediction in airspace, degrades on both, *SDD* and *NBA*. We believe that the latter may be related to the lack of a better mechanism for encoding social interactions.

As a final clarification, we note that the authors of both, *DAG-Net* and *SPEC*, suggested that their methods are general enough to be applied to other scenarios, whereas the authors of *TrajAirNet* did not claim generality. Nonetheless, the type of backbone employed by *TrajAirNet* for predicting trajectories and encoding social interactions has been widely used by other prediction algorithms [20, 23]. Thus, we deemed reasonable to assess the algorithm's capabilities to generalize.

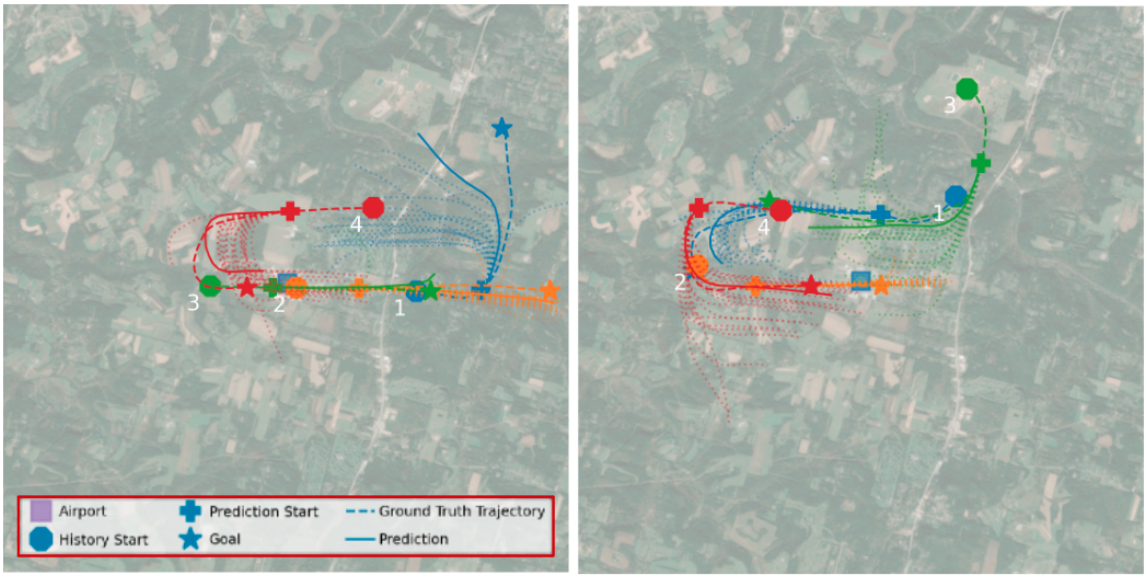
5.2 Qualitative Analysis

5.2.1 Visualizing Results Across Domains

We provide visual comparisons between the VRNN's and *Social-PatteRNN*'s predictions across our three domains of interest together with their corresponding analysis. Figure 5.1 shows predictions from two different scenes drawn from the *TrajAir* dataset. Each scene consists of four agents whose ground-truth (GT) trajectories are shown with a dashed line. The models' predictions start at the location indicated with a



(a) VRNN

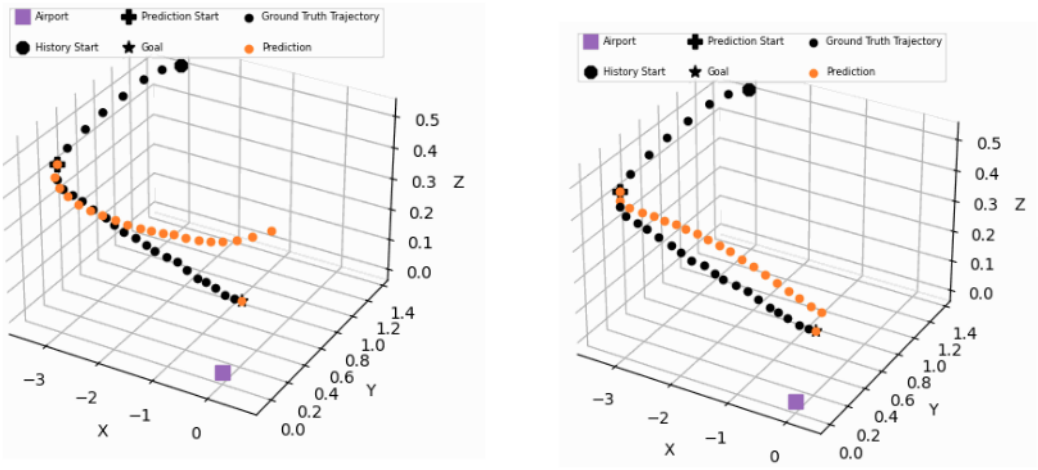


(b) SocialPatteRNN

Figure 5.1: Visual multi-future results for two scenes on the *TrajAir* dataset using (a) VRNN and (b) *Social-PatteRNN*. Dashed lines are the ground-truth trajectories, solid lines are the best prediction and dotted lines are the remaining $K - 1$ predictions. Agents are numbered and color-coded; 1 - blue, 2 - orange, 3 - green and 4 red. Trajectories start at the octagonal landmark, the predictions start at the cross landmark, and the goal is at the star landmark.

cross landmark, and are denoted with a solid line for the best prediction, and dotted lines for the other $K - 1$ samples (See Section 4.4). From the visualizations, we notice that the VRNN baseline tends to predict trajectories that do not follow the traffic patterns of the aircraft, whereas *Social-PatteRNN* does better at following them while also producing more diverse trajectories. Another significant difference between the two models is in the prediction of altitude. As an example, consider the agent shown in orange on the right-most scene in Figure 5.1. The GT trajectory for this agent appears mostly as a horizontal line in the figure. Therefore, it seems as if both of the models made a *reasonable* prediction of the trajectory. However, the agent in this particular scene is landing but the changes in altitude are not visible from the top-down view perspective. Figure 5.2 shows the trajectory as a 3D plot, where the altitude corresponds to the z coordinates. In this figure the GT trajectory is shown in black, the predictions are shown in orange and the goal is the orange star at the endpoint of the GT trajectory. From these plots, we can see that the VRNN's prediction is most notably separated from the goal location, roughly maintaining the same altitude throughout the entire prediction. In contrast, *Social-PatteRNN* performed better at predicting the *landing* trajectory; the altitude gets reduced with each step and the final prediction is closer to the GT endpoint.

Figure 5.3 shows results for the NBA dataset. These follow a similar format as described above except that now we color-code each of the teams; red for the attacker (ATK) team, blue for the defense (DEF) team. Since each scene consists of 10 agents with highly dynamic motion, for ease of visualization, we only depict the predictions for two randomly selected agents; agent 3 from ATK, and agent 10 from DEF. The GT motions of the remaining agents are shown in transparent color for visual context. In the figure, we observe that the VRNN does poorly at predicting the future trajectories with most of the predictions being significantly shorter in length compared to the GT ones. For instance, the VRNN's predictions on the right scene are barely visible, which may suggest that model predicted that the two agents were *static* during that play. Conversely, *Social-PatteRNN* is better at capturing the general motion of the basketball players. For instance, in the left image, *Social-PatteRNN* was able to predict the DEF agent's turn, whereas the VRNN was not. Then, on the scene where the VRNN predicted *static* agents, *Social-PatteRNN* was able to predict longer trajectories and in the correct direction of motion. It is also



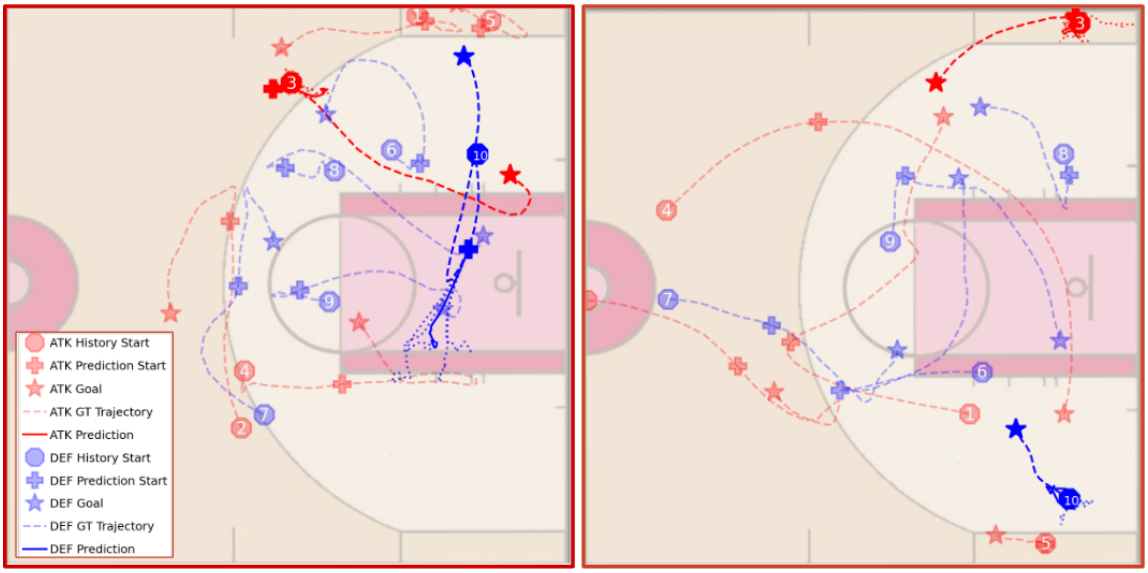
(a) VRNN

(b) SocialPatteRNN

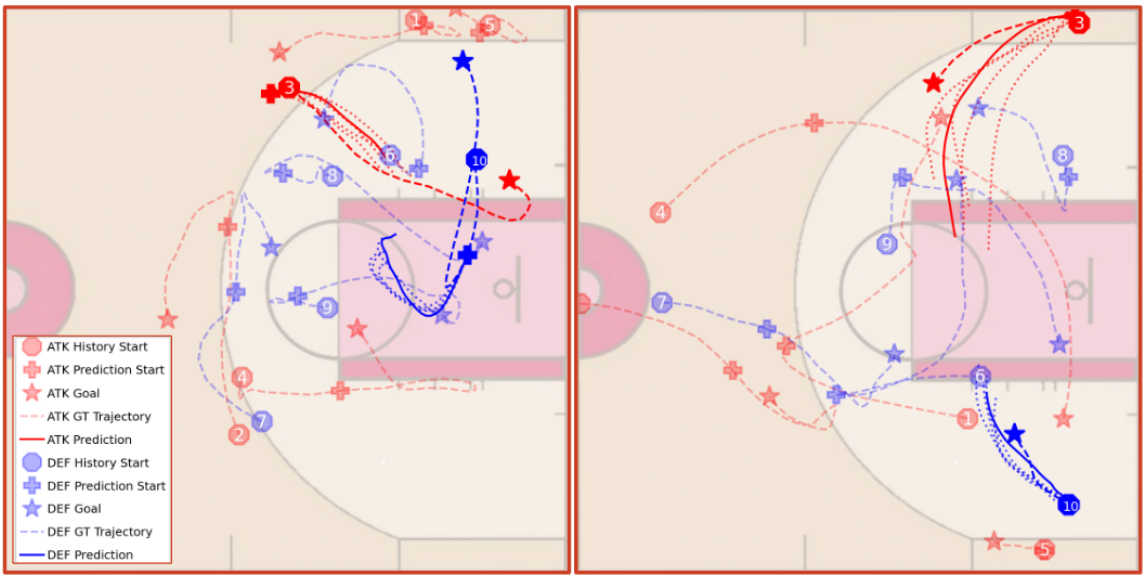
Figure 5.2: Visual results for a TrajAir trajectory in 3D using (a) VRNN and (b) Social-PatteRNN. Ground truth is shown in black, and best prediction is shown in orange. The remaining $K-1$ samples are not shown for simplicity.

clear that *Social-PatteRNN*'s results are not perfect either, *e.g.*, on the left scene, the predicted turn was not as sharp as the GT one, and on the right scene, the predicted trajectories were longer than the GT. However, we emphasize that the *NBA* dataset comprises a set of highly dynamic scenes with complex interactions. As such, significant improvements from our models, as well as the models presented in previous sections are still necessary.

Finally, Figure 5.4 shows the results from two scenes in the *SDD* dataset. In both of these scenes there are only two agents interacting. Here, the differences between the VRNN and *Social-PatteRNN*'s results are less evident in contrast to the latter datasets. One reason for this may be that motion in the pedestrian-based version of *SDD* is mostly *linear*. Nonetheless, there are two main differences between the two baselines. First, *Social-PatteRNN* does a better at predicting longer trajectories, suggesting a better understanding of the pedestrian velocity of motion, whereas the VRNN often predicts shorter trajectories. Second, we can observe that the span of multi-future trajectories is larger in the *Social-PatteRNN*'s predictions, which suggests that the model does better at predicting more diverse trajectories. As said before, this is important because for a given context history, multiple futures may be

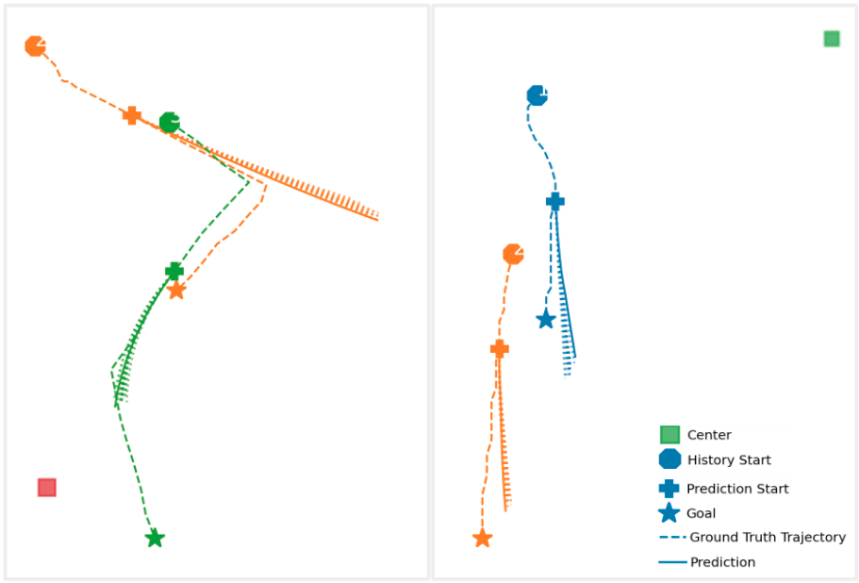


(a) VRNN

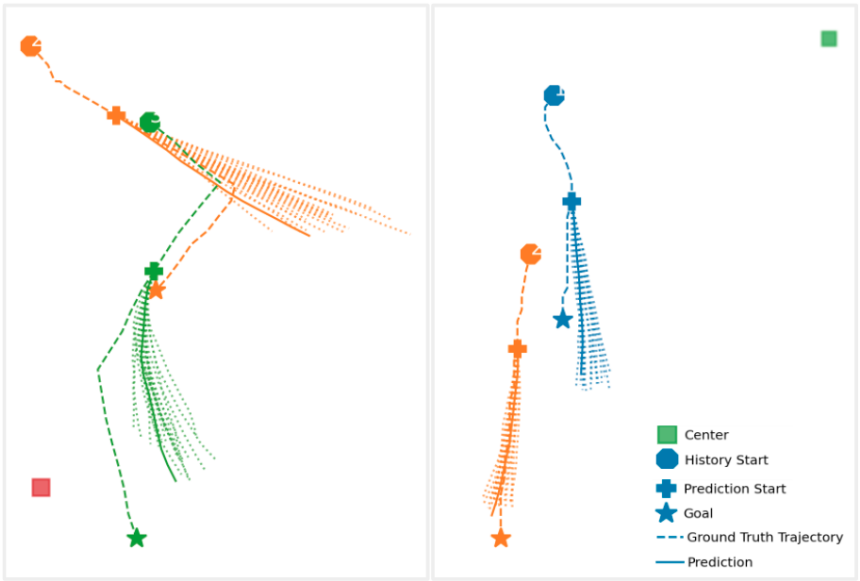


(b) SocialPatteRNN

Figure 5.3: Visual multi-future results for two scenes on the NBA dataset using (a) VRNN and (b) Social-PatteRNN. Agents are numbered and color-coded by team; agents 1 to 5 in red for attackers (ATK) and 6 to 10 in blue for defenders (DEF). For simplicity, each figure shows the prediction for one player in the ATK team, and one player in the DEF team. The ground truth trajectories for all other players are plotted for reference in transparent color.



(a) VRNN



(b) Social-PatteRNN

Figure 5.4: Visual multi-future results for two scenes on the Stanford Drone Dataset using (a) VRNN and (b) *Social-PatteRNN*. Dashed lines are the ground-truth trajectories, solid lines are the best prediction and dotted lines are the remaining $K-1$ predictions. Trajectories start at the octagonal landmark, the predictions start at the cross landmark, and the goal is at the star landmark.

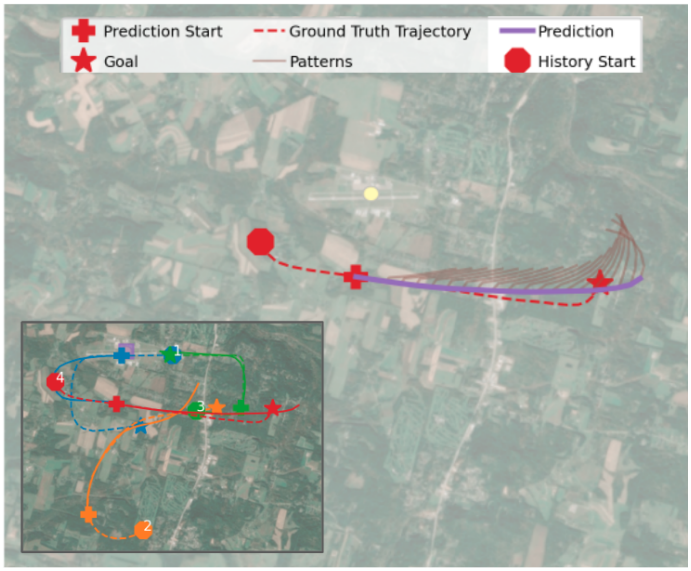
appropriate. As a concluding comment, we believe further analysis pedestrian-based settings may be useful, *e.g.*, evaluating our approach on the *ETH/UCY* benchmark or on the multi-agent version of *SDD*.

5.2.2 Visualizing Motion Patterns

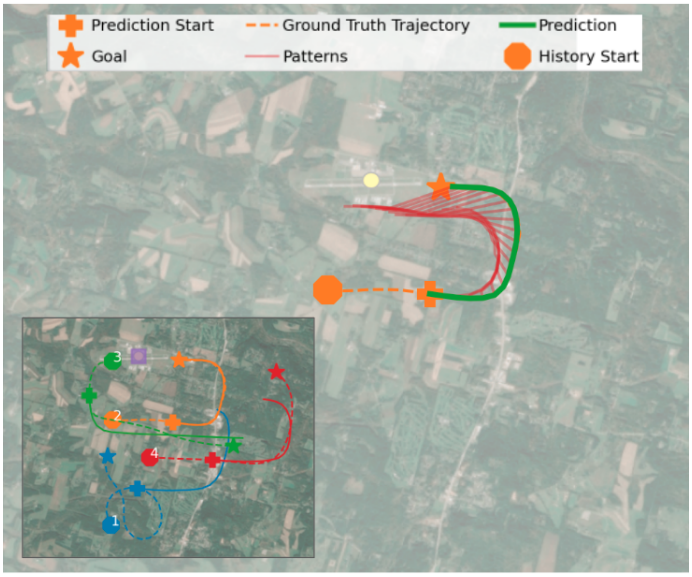
We provide an analysis of the motion pattern predictions across domains. To reiterate, one of our core ideas is to use these patterns as *guidance* to the goal location, as opposed to enforcing the motion of the predicted patterns. One reason behind this is to avoid unstable motions that may arise from unprecedented scenarios. Another associated reason is related to how we predict the motion patterns for each agent, *i.e.*, at each step we account for the previous belief of the sub-goals of other agents in order to make the predictions. However, due to the dynamic nature in this problem setup, sub-goals may need to change, specially when unexpected situations occur. In this subsection, we are interested in analyzing whether the predicted patterns properly *guide* the agents. We do so by considering whether the learned patterns (1) generally point in the correct direction of motion, (2) capture rules-of-motion, *e.g.*, traffic patterns in airspace, and (3) appear admissible and stable, *e.g.*, there are no abrupt changes in motion, both within the same pattern and subsequent ones when abrupt motion would be inappropriate.

Figure 5.5 shows two examples from the *TrajAir* dataset. Each one depicts the predicted future trajectory and the predicted motion patterns at each step for a single agent. For reference, the visualization is accompanied by the scene from which the agent was drawn. We believe that in these visualizations the predicted patterns satisfy our aforesaid criteria. First, the patterns generally point to the correct direction of the motion. Second, they capture rules-of-motion, *e.g.*, the beginning of the turn of the red agent in the top image, and the full turn of the orange agent in the bottom image. Finally, we can see that the predicted patterns are smooth, *i.e.*, they do not exhibit abrupt behavior such as *jerkiness*.

Following the format explained above, Figure 5.6 illustrates two examples from the *NBA* dataset. In particular, we show an example of an attacker's (ATK) trajectory and a defender's (DEF) one (sub-figures (a) and (b), respectively). In both of the results we observe that the patterns exhibit a similar behavior. First, the initial



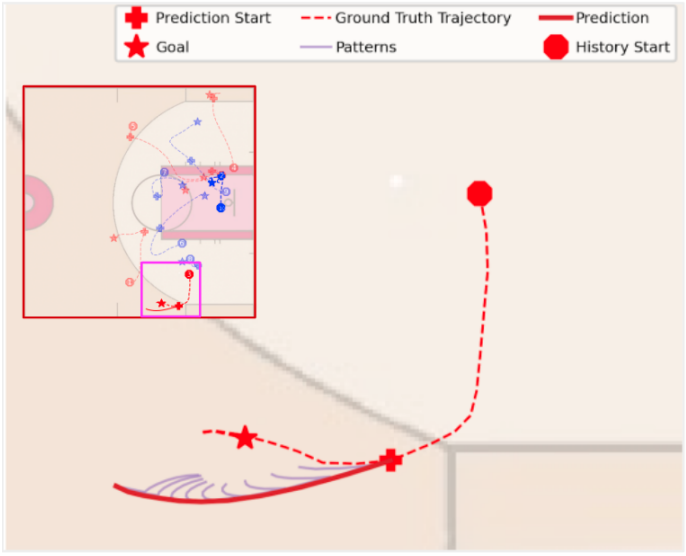
(a) Predicted motion patterns) in brown) for agent #4 (in red).



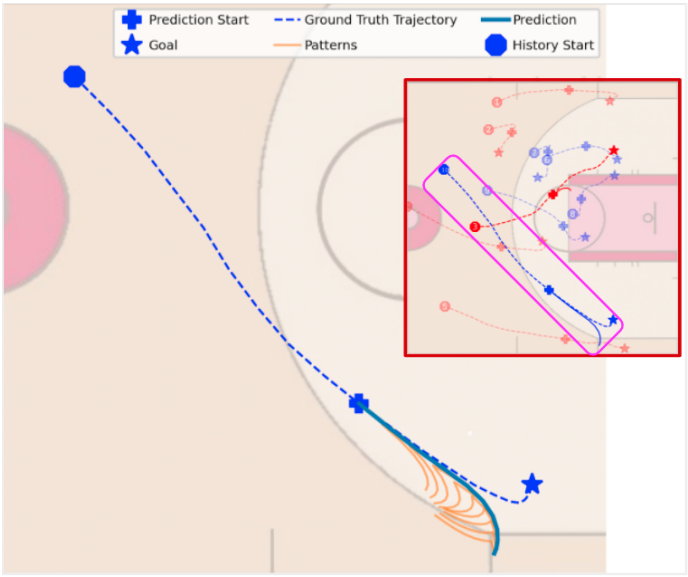
(b) Predicted motion patterns (in red) for agent #2 (in orange).

Figure 5.5: Inspecting *TrajAir* agents' predicted motion patterns at each step. Within each sub-image we show the scene from which the agent was drawn.

patterns predict a *turn* motion; but, with each step, the predictions reveal a more pronounced change in direction that appears like a *step-back* motion. In the ATK's



(a) Predicted patterns (in purple) for ATK agent #3 (in red).



(b) Predicted patterns (in orange) for DEF agent #10 (in blue).

Figure 5.6: Inspecting *NBA* agents' predicted motion patterns at each step. Within each sub-image we show the scene from which the agent was drawn.

trajectory, these predictions were appropriate, since the agent *did* stepped back. However, for the DEF trajectory, these predictions were incorrect, and further *guided* the agent into an incorrect direction. In contrast to *TrajAir*, the learned patterns

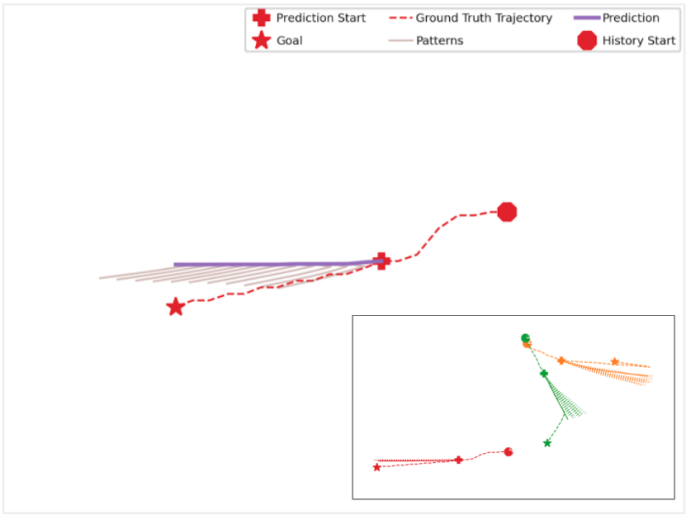
in this dataset are more dynamic, *i.e.*, the change in direction of the patterns with each step is more pronounced. Nonetheless, we believe that in the *NBA* dataset, this learned behavior is appropriate due to the dynamic and complex nature of the dataset. As was conveyed in the previous section, we acknowledge that significant improvements in this dataset are still compulsory. The latter is further observed from inspecting the motion patterns; where, in various scenarios we observed that the patterns guided the agent into an erroneous direction.

Finally, Figure 5.7 provides two examples from the *SDD* dataset. In these examples we observe that the motions predicted by *PatterNet* are generally straight and smooth, while also point to the correct direction. We deem our criteria is satisfied for this dataset as well. One limitation we note, however, is the lack of expressiveness due to the predictions being heavily linear.

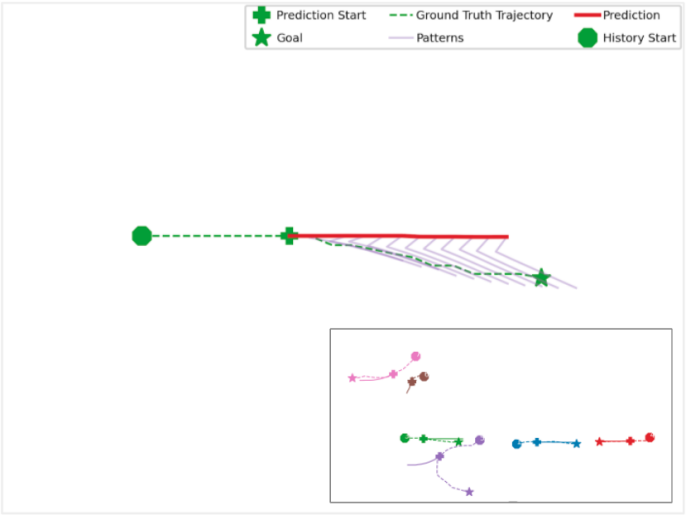
5.2.3 Visualizing Different Models

To finalize the qualitative analysis, in Figure 5.8 we provide examples comparing different models on *TrajAir*. On the left sub-figure we show the predictions for only one agent, denoted with a black dashed line. For reference, we show the trajectories of the other agents in the scene in transparent color.

In this example, we can observe that the predicted trajectories for *TrajAirNet* and the *VRNN* baseline do not align with the correct direction of the ground truth trajectory, while the models that were conditioned on additional context do, *i.e.*, *DAG-Net* and ours'. Note for instance, that the end-point of the trajectory predicted by *DAG-Net* gets close to the goal location. This is expected as the model was conditioned on goal-map information [23]. However, we observe that it does not follow the expected aircraft pattern, whereas our pattern-conditioned approaches do. Furthermore, our pattern-based models achieve smoother predictions than *TrajAirNet* and *DAG-Net*. Here, it is specially relevant to highlight that the ability to learn and follow motion patterns is a strength of our model. Also, to reiterate the importance of following flying guidelines in the airspace domain, as it enables and furthers safe cooperation. Moreover, the visualization also allows us to see the benefit of aggregating social context information by comparing the predicted trajectory by our approaches *with* social context (*Social-PatteRNN* and *Social-PatteRNN*-ATT),



(a) Predicted patterns (in brown) for red agent.



(b) Predicted patterns (in purple) for green agent.

Figure 5.7: Inspecting *SDD* agents' predicted motion patterns at each step. Within each sub-image we show the scene from which the agent was drawn.

shown in pink and red, respectively, against that *without* social context (VRNN-PAT), shown in orange. Quantitatively, Table 5.1 concurs that adding social context further improves the prediction performance.

On the right sub-figure we show predictions of two models, *Social-PatteRNN* (top) and *TrajAirNet* (bottom), for the same scene. For simplicity, we only show the best prediction for each agent. Similar to the previous analysis, we can see that *Social-*

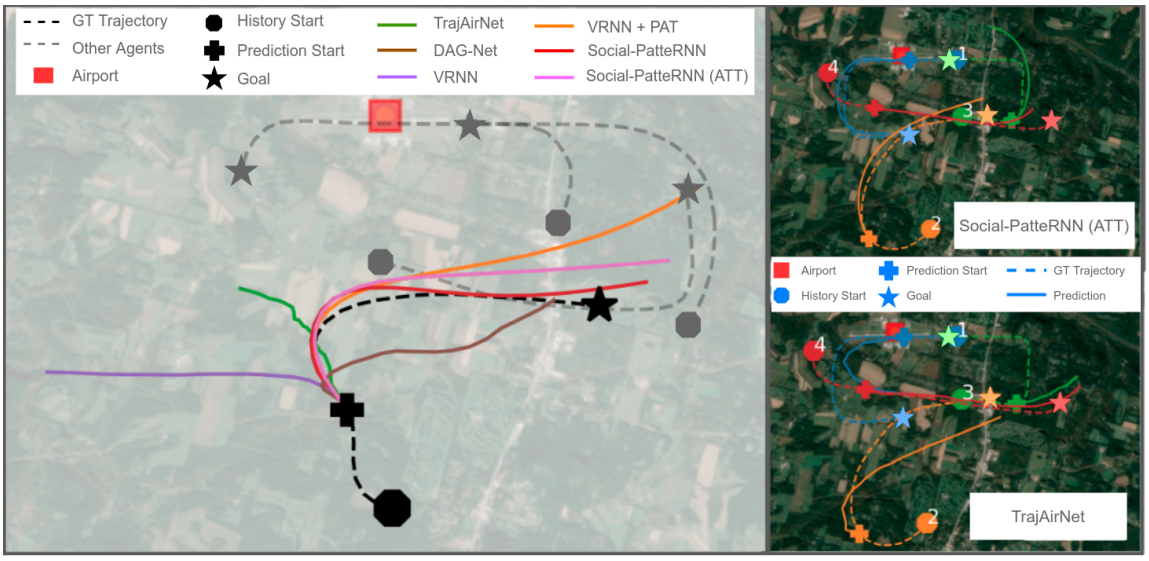


Figure 5.8: Comparison of predicted trajectories using different models (ground truth in dotted lines; prediction, solid lines). **Left:** Various models' predictions vs. ground-truth for a single agent. **Right:** SocialPatteRNN (Top) vs. TrajAirNet (Bottom) for multiple agents.

PatteRNN's predictions are smoother and better capture the appropriate aircraft motion, whereas *TrajAirNet*'s results exhibit more abrupt motion and more difficulty following the GT motion.

To conclude, we believe that our observations from the comparisons above support our initial hypothesis that through motion patterns additional context information can be learned, *e.g.*, general direction and rules of motion, and used as conditioning information for improving intent prediction models.

Chapter 6

Conclusions

In our work, we hypothesized that short-term motion patterns reveal context information related to the agents' general direction, rules of motion, and how they interact with each other. From this intuition, we introduced *Social-PatteRNN*, a trajectory prediction algorithm that leverages motion patterns to capture these contexts.

Our experiments show that the performance of existing approaches are sensitive to problem domain and how the dataset is composed. The results support our intuition that short-term patterns are robust indicators for trajectory prediction and are also less sensitive to problem domains. As future work, we aim to explore various avenues. We are interested in further exploiting and learning the dependencies between the agents by exploring other attention mechanism, as well as other backbone structures such as Transformers [41]. We are also interested in incorporating more context information from input modalities other than trajectory data. Finally, we plan to extend our work to develop a safe and socially-aware robot navigation algorithm for urban driving and aerial navigation in crowded scenarios.

Bibliography

[1] STATS Perform - SportVU NBA Dataset. https://www.statsperform.com/team-performance/basketball/optical-tracking/. 1.1, 2.1, 4.1, 4.1.3

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: human trajectory prediction in crowded spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 961–971. IEEE Computer Society, 2016. 1.2, 2.1, 2.2, 4.2

[3] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR*, abs/1803.01271, 2018. 2.2

[4] Apratim Bhattacharyya, Michael Hanselmann, Mario Fritz, Bernt Schiele, and Christoph-Nikolas Straehle. Conditional flow variational autoencoders for structured sequence prediction. *CoRR*, abs/1908.09008, 2019. 2.2

[5] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934, 2020. doi: 10.1109/IV47402.2020.9304839. 2.1

[6] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. The interactive museum tour-guide robot. In Jack Mostow and Chuck Rich, editors, *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA*, pages 11–18. AAAI Press / The MIT Press, 1998. 1.1

[7] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*

2020, Seattle, WA, USA, June 13-19, 2020, pages 11618–11628. Computer Vision Foundation / IEEE, 2020. doi: 10.1109/CVPR42600.2020.01164. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Caesar_nuScenes_A_Multimodal_Dataset_for_Autonomous_Driving_CVPR_2020_paper.html. 2.1

[8] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8748–8757. Computer Vision Foundation / IEEE, 2019. 1.1

[9] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8748–8757. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00895. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Chang_Argoverse_3D_Tracking_and_Forecasting_With_Rich_Maps_CVPR_2019_paper.html. 2.1

[10] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL, 2014. doi: 10.3115/v1/d14-1179. URL https://doi.org/10.3115/v1/d14-1179. 2.2, 4.4

[11] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2980–2988, 2015. 2.2, 3.2.1

[12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3213–3223. IEEE Computer Society, 2016. doi:

10.1109/CVPR.2016.350. URL https://doi.org/10.1109/CVPR.2016.350. 2.1

[13] Thierry Deruyttere, Simon Vandenhende, Dusan Grujicic, Luc Van Gool, and Marie-Francine Moens. Talk2car: Taking control of your self-driving car. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2088–2098. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1215. URL https://doi.org/10.18653/v1/D19-1215. 2.1

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 3354–3361. IEEE Computer Society, 2012. doi: 10.1109/CVPR.2012.6248074. URL https://doi.org/10.1109/CVPR.2012.6248074. 2.1

[15] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, pages 10335–10342. IEEE, 2020. doi: 10.1109/ICPR48806.2021.9412190. URL https://doi.org/10.1109/ICPR48806.2021.9412190. 2.2

[16] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: socially acceptable trajectories with generative adversarial networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2255–2264. Computer Vision Foundation / IEEE Computer Society, 2018. 1.2, 2.1, 2.2, 4.2

[17] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical Review E*, 51, 05 1998. doi: 10.1103/PhysRevE.51.4282. 2.2

[18] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B. Choy, Philip H. S. Torr, and Manmohan Chandraker. DESIRE: distant future prediction in dynamic scenes with interacting agents. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2165–2174. IEEE Computer Society, 2017. 2.2

[19] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. *Comput. Graph. Forum*, 26(3):655–664, 2007. doi: 10.1111/j.1467-8659.2007.01089.x. URL https://doi.org/10.1111/j.1467-8659.2007.01089.x. 2.1, 5.1.2

[20] Congcong Liu, Yuying Chen, Ming Liu, and Bertram E. Shi. AVGCN: trajectory prediction using graph convolutional networks guided by human attention. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an,*

China, May 30 - June 5, 2021, pages 14234–14240. IEEE, 2021. 1.1, 2.2, 3.2, 4.2, 5.1.2

[21] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part II*, volume 12347 of *Lecture Notes in Computer Science*, pages 759–776. Springer, 2020. 1.1, 2.2, 3.2, 3.2.2, 4.2

[22] Christoforos I. Mavrogiannis, Francesca Baldini, Allan Wang, Dapeng Zhao, Pete Trautman, Aaron Steinfeld, and Jean Oh. Core challenges of social robot navigation: A survey. *CoRR*, abs/2103.05668, 2021. 1.1

[23] Alessio Monti, Alessia Bertugli, Simone Calderara, and Rita Cucchiara. Dagnet: Double attentive graph neural network for trajectory forecasting. In *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, pages 2551–2558. IEEE, 2020. 1.1, 1.2, 2.2, 3.2, 3.2.1, 3.2.2, 4.1.2, 4.1.3, 4.2, 1, 2, 4.4, 5.2, 5.1.2, 5.2.3

[24] Nishant Nikhil and Brendan Tran Morris. Convolutional neural network for trajectory prediction. In Laura Leal-Taixé and Stefan Roth, editors, *Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11131 of *Lecture Notes in Computer Science*, pages 186–196. Springer, 2018. 2.2

[25] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, J. K. Aggarwal, Hyungtae Lee, Larry Davis, Eran Swears, Xioyang Wang, Qiang Ji, Kishore Reddy, Mubarak Shah, Carl Vondrick, Hamed Pirsiavash, Deva Ramanan, Jenny Yuen, Antonio Torralba, Bi Song, Anesco Fong, Amit Roy-Chowdhury, and Mita Desai. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR 2011*, pages 3153–3160, 2011. doi: 10.1109/CVPR.2011.5995586. 1.1

[26] Seong Hyeon Park, Gyubok Lee, Jimin Seo, Manoj Bhat, Minseok Kang, Jonathan Francis, Ashwin R. Jadhav, Paul Pu Liang, and Louis-Philippe Morency. Diverse and admissible trajectory forecasting through multimodal context understanding. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XI*, volume 12356 of *Lecture Notes in Computer Science*, pages 282–298. Springer, 2020. doi: 10.1007/978-3-030-58621-8_17. URL https://doi.org/10.1007/978-3-030-58621-8_17. 1.1

[27] Jay Patrikar, Brady G. Moon, Jean Oh, and Sebastian A. Scherer. Predicting

like A pilot: Dataset and method to predict socially-aware aircraft trajectories in non-towered terminal airspace. *CoRR*, abs/2109.15158, 2021. 1.1, 1.2, 1.2, 2.1, 2.2, 3.2.2, 4.1, 4.1.1, 4.2, 3, 5.1.1, 5.2

[28] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, pages 261–268. IEEE Computer Society, 2009. doi: 10.1109/ICCV.2009.5459260. URL https://doi.org/10.1109/ICCV.2009.5459260. 2.1, 5.1.2

[29] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1278–1286. JMLR.org, 2014. 2.2, 3.2.1, 3.2.1

[30] Alexandre Robicquet, Alexandre Alahi, Amir Sadeghian, Bryan Anenberg, John Doherty, Eli Wu, and Silvio Savarese. Forecasting social navigation in crowded complex scenes. *CoRR*, abs/1601.00998, 2016. 1.2, 2.1, 4.1, 4.1.2

[31] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M. Kitani, Dariu M. Gavrila, and Kai Oliver Arras. Human motion trajectory prediction: a survey. *Int. J. Robotics Res.*, 39(8), 2020. doi: 10.1177/0278364920917446. URL https://doi.org/10.1177/0278364920917446. 2.1

[32] Amir Sadeghian, Vineet Kosaraju, Agrim Gupta, Silvio Savarese, and A Alahi. Trajnet: Towards a benchmark for human trajectory prediction. *arXiv preprint*, 2018. 4.1.2

[33] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive GAN for predicting paths compliant to social and physical constraints. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1349–1358. Computer Vision Foundation / IEEE, 2019. 1.1, 2.2

[34] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVIII*, volume 12363 of *Lecture Notes in Computer Science*, pages 683–700. Springer, 2020. 1.1, 2.2

[35] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In Corinna Cortes,

Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3483–3491, 2015. 3.2.1, 3.2.1

[36] Sebastian Thrun, Maren Bennewitz, Wolfram Burgard, Armin B. Cremers, Frank Dellaert, Dieter Fox, Dirk Hähnel, Charles R. Rosenberg, Nicholas Roy, Jamieson Schulte, and Dirk Schulz. MINERVA: A second-generation museum tour-guide robot. In *1999 IEEE International Conference on Robotics and Automation, Marriott Hotel, Renaissance Center, Detroit, Michigan, USA, May 10-15, 1999, Proceedings*, pages 1999–2005. IEEE Robotics and Automation Society, 1999. 1.1

[37] Peter Trautman and Andreas Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 18-22, 2010, Taipei, Taiwan*, pages 797–803. IEEE, 2010. 2.2

[38] Adrien Treuille, Seth Cooper, and Zoran Popovic. Continuum crowds. *ACM Trans. Graph.*, 25(3):1160–1168, 2006. 2.2

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. 2.2, 3.2.3

[40] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1–7. IEEE, 2018. 1.2, 2.1, 2.2, 4.2

[41] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. *CoRR*, abs/2103.14023, 2021. 2.2, 6

[42] Dapeng Zhao and Jean Oh. Noticing motion patterns: A temporal CNN with a novel convolution operator for human trajectory prediction. *IEEE Robotics Autom. Lett.*, 6(2):628–634, 2021. 1.2, 1.2, 2.1, 2.2, 2.3, 4.2, 4, 5.2

[43] Haosheng Zou, Hang Su, Shihong Song, and Jun Zhu. Understanding human behaviors in crowds by imitating the decision-making process. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium*

on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, pages 7648–7656. AAAI Press, 2018. 2.2