EPFL

Thèse n° 9842

Deep Learning Methods for Socially-Aware Human Trajectory Forecasting

Présentée le 2 décembre 2022

Faculté de l’environnement naturel, architectural et construit
Intelligence Visuelle pour les Transports
Programme doctoral en génie électrique

pour l’obtention du grade de Docteur ès Sciences

par

Parth Ashit KOTHARI

Acceptée sur proposition du jury

Prof. M. Kamgarpour, présidente du jury
Prof. A. M. Alahi, directeur de thèse
Prof. . M. Elhoseiny, rapporteur
Prof. T. Brox, rapporteur
Prof. D. Tuia, rapporteur

École
polytechnique
fédérale
de Lausanne

2022

For Dhruti

Acknowledgements

This work would not have been possible without the support of many people. First and foremost, I would like to thank my advisor, Prof. Alexandre Alahi, for his continuous support and guidance throughout my Ph.D. Our regular interactions in these past years have not only helped me become a good researcher but also contributed to what I am today.

I would like to express my special recognition to the members of my thesis committee, Prof. Maryam Kamgarpour, Prof. Devis Tuia, Prof. Mohamed Elhoseiny, and Prof. Thomas Brox for their effort in reviewing this dissertation and for their meaningful discussion.

Working at VITA has been a great pleasure. I owe a special thanks to Yuejiang Liu, who has been like a big brother to me. I have learnt so much about research from you and working with you has been a terrific pleasure. I could not have asked for a better collaborator throughout these years. I would like to thank George Adaimi for being an amazing friend and for all those times when he has been around when I needed to chat with someone. I would like to thank Lorenzo Bertoni for being an awesome roommate, helping create lots of memories in GC C1 392.

I am grateful to Sven Kreiss for the fruitful discussions and guidance in the early years. I would like to acknowledge Taylor Mordan for his meticulous attention to detail and his timely feedbacks. I want to thank Saeed Saadatnejad, Hossein Bahari, Brian Sifringer and Bastien Van Delft for the insightful discussions and fun conversations.

I would like to specially thank two of my closest friends Sudeep Salgia and Arka Sadhu for bearing me during the tough phases, motivating me and reviewing all my works at the very last minute. I will be always remain indebted for your support.

I would like to thank my wife, Dhruti Shah for being my rock. This dissertation would not have been possible without your constant encouragement and support. Finally, I would like to express my utmost gratitude to my parents. Mom and Dad, thanks a lot for all the sacrifices you have made for me over the years. Without the two of you, there would be no me.

Lausanne, 8 November 2022 P. K.

Abstract

The ability to forecast human motion, called “human trajectory forecasting", is a critical requirement for mobility applications such as autonomous driving and robot navigation. Humans plan their path taking into account what might happen in the future. Similarly, the decision-making algorithm of autonomous systems should predict how their environment will evolve in the future. This thesis focuses on developing deep learning methods for forecasting human motion.

In the first part of this thesis, we tackle the fundamental challenges of social interaction modelling and multimodality. Social interactions dictate how the motion of a human is affected by others. Current deep learning methods often struggle to model these interactions between trajectory sequences. To promote interaction-awareness in forecasting models, we develop a training paradigm that explicitly focuses on samples that undergo interactions and incorporates model uncertainty. Furthermore, we build a taxonomy of existing interaction encoders and propose an optimal design that is robust to the real-world noise. In addition to modelling interactions, a good trajectory forecasting model must account for the multimodal nature of the prediction, *i.e.*, the possibility of having multiple plausible futures given the past observations. To tackle multimodality, we present a socially-aware generative adversarial network that leverages recent advances in sequence modelling, and has the ability to model the temporal evolution of social interactions. Furthermore, we develop a collaborative sampling technique that refines the bad generated predictions at test time.

In the second part of this thesis, we focus on two challenges specific to the real-world deployment of forecasting models: interpretability and adaptability. While neural networks have the capacity to learn complex interactions, it is difficult to understand the reason behind their predictions. Thus, we develop a framework that combines the interpretability of the classical models with the predictive power of neural networks. With regards to adaptability, existing deep forecasting models suffer from inferior performance when they encounter novel scenarios. We develop a strategy to adapt a pre-trained forecasting model to a target domain using limited samples. In particular, we introduce motion style adapters that identify and adjust the target domain-specific features. Throughout this thesis, experiments on synthetic and real-world forecasting datasets validate the effectiveness of our proposed methods.

Keywords: trajectory forecasting, deep learning, social interactions, generative adversarial networks, interpretability, adaptation, benchmark

Résumé

La capacité de prévoir le mouvement humain, appelée "prévision de la trajectoire humaine", est essentielle pour les applications de mobilité telles que la conduite autonome et la navigation des robots. Les humains planifient leur trajectoire en tenant compte de ce qui pourrait se passer dans le futur. De même, l'algorithme de prise de décision des systèmes autonomes doit prévoir comment leur environnement évoluera dans le futur. Cette thèse se concentre sur le développement de méthodes d'apprentissage profond pour la prévision du mouvement humain.

Dans la première partie de cette thèse, nous abordons les défis de la modélisation des interactions sociales et de la multimodalité. Les interactions sociales dictent la façon dont le mouvement d'un humain est affecté par les autres. Les méthodes actuelles d'apprentissage profond ont souvent du mal à modéliser ces interactions entre les séquences de trajectoires. Pour promouvoir la prise en compte des interactions dans les modèles de prévision, nous développons un paradigme de formation qui se concentre explicitement sur les échantillons qui subissent des interactions. En outre, nous construisons une taxonomie des codeurs d'interaction existants et proposons une conception optimale qui est robuste au bruit du monde réel. Pour aborder la multimodalité, nous présentons un réseau adversarial génératif socialement conscient qui tire parti des avancées récentes en matière de modélisation des séquences, et qui a la capacité de modéliser l'évolution temporelle des interactions sociales. En outre, nous développons une technique d'échantillonnage collaborative qui affine les mauvaises prédictions générées au moment du test.

Dans la deuxième partie, nous nous concentrons sur deux défis spécifiques au déploiement de modèles de prévision dans le monde réel : l'interprétabilité et l'adaptabilité. Alors que les réseaux neuronaux ont la capacité d'apprendre des interactions complexes, il est difficile de comprendre la raison de leurs prédictions. Ainsi, nous développons un cadre qui combine l'interprétabilité des modèles classiques avec le pouvoir prédictif des réseaux neuronaux. En ce qui concerne l'adaptabilité, les modèles de prévision profonde existants souffrent de performances inférieures lorsqu'ils sont confrontés à de nouveaux scénarios. Nous développons une stratégie pour adapter un modèle de prévision pré-entraîné à un domaine cible en utilisant des échantillons limités. Nous introduisons des adaptateurs de style de mouvement qui identifient et ajustent les caractéristiques spécifiques au domaine cible. Tout au long de cette thèse, des expériences sur des ensembles de données de prévision synthétiques et réelles valident l'efficacité de nos méthodes proposées.

Mots clés : prévision de trajectoire, apprentissage profond, interactions sociales, réseaux adversariens génératifs, interprétabilité, adaptation, référence

Contents

Acknowledgements **i**

Abstract (English/French) **iii**

List of figures **xi**

List of tables **xiii**

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Our Approach . . . 2
- 1.3 Thesis Contributions . . . 5
- 1.4 Thesis Structure . . . 6
- 1.5 Related Publications . . . 7

2 Learning Social Interactions for Human Trajectory Forecasting **9**
- 2.1 Introduction . . . 9
- 2.2 Related Work . . . 12
- 2.3 Method . . . 14
  - 2.3.1 Problem statement . . . 14
  - 2.3.2 Interaction module . . . 15
  - 2.3.3 Forecasting model . . . 19
  - 2.3.4 Training paradigm . . . 19
  - 2.3.5 Equivalence to Graph Neural Networks . . . 21
  - 2.3.6 Explaining trajectory forecasting models . . . 22
- 2.4 TrajNet++: A Trajectory Forecasting Benchmark . . . 22
  - 2.4.1 Trajectory categorization . . . 23
  - 2.4.2 Evaluation metrics . . . 25
  - 2.4.3 Forecasting benchmark: a missing piece . . . 27
  - 2.4.4 TrajNet++ datasets . . . 28
- 2.5 Experiments . . . 31
  - 2.5.1 Implementation details . . . 31
  - 2.5.2 Interaction modules: synthetic experiments . . . 31
  - 2.5.3 Interaction modules: real world experiments . . . 33

2.5.4 Socially-aware training recipe 35
2.5.5 Explaining forecasting model decisions 36
2.6 Conclusion 38

3 Generative Adversarial Networks for Multimodal Trajectory Forecasting 39
3.1 Introduction 39
3.2 Related Work 41
3.3 Method 43
3.3.1 Problem statement 43
3.3.2 Generative Adversarial Networks 43
3.3.3 Interaction modelling designs 43
3.3.4 SGANv2 architecture 44
3.3.5 Training 47
3.4 Collaborative sampling in GANs 47
3.4.1 General formulation 48
3.4.2 Collaborative sampling in SGANv2 49
3.5 Experiments 50
3.5.1 Limitations of current multimodal evaluation scheme 51
3.5.2 Synthetic experiments 53
3.5.3 Real-world experiments 53
3.5.4 Multimodal analysis 55
3.5.5 Key attributes 56
3.5.6 Implementation details 58
3.6 Conclusion 58

4 Interpretable Social Anchors for Human Trajectory Forecasting 61
4.1 Introduction 61
4.2 Related Work 63
4.3 Method 63
4.3.1 Problem statement 63
4.3.2 Discrete Choice Models 64
4.3.3 Social Anchor framework 64
4.3.4 Anchor selection 66
4.3.5 Training 67
4.4 Experiments 68
4.4.1 Implementation details 68
4.4.2 Quantitative analysis 69
4.4.3 Interpretability of the intents 70
4.4.4 Direction normalization 71
4.5 Conclusion 74

5 Modular Low-Rank Adaptation for Trajectory Forecasting 75
5.1 Introduction . . . 75
5.2 Related Work . . . 76
5.3 Method . . . 77
  5.3.1 Motion style transfer . . . 78
  5.3.2 MoSA: motion style adapters . . . 79
  5.3.3 Modularized adaptation strategy . . . 80
5.4 Experiments . . . 81
  5.4.1 Motion style transfer across agents on Stanford Drone Dataset . . . 82
  5.4.2 Motion style transfer across scenes on Intersection Drone Dataset . . . 83
  5.4.3 Motion style transfer across scenes on Lyft Level 5 dataset . . . 83
  5.4.4 Modularized adaptation strategy . . . 84
  5.4.5 Implementation Details . . . 87
5.5 Conclusion . . . 88

6 Democratising Reinforcement Learning for Autonomous Driving 89
6.1 Introduction . . . 89
6.2 Related Work . . . 90
6.3 Method . . . 91
  6.3.1 State Representation . . . 91
  6.3.2 Action Spaces . . . 92
  6.3.3 Reactive Agents . . . 92
  6.3.4 Rewards . . . 92
  6.3.5 Closed-Loop Evaluation Protocol . . . 93
6.4 Experiments . . . 94
  6.4.1 Model Architecture . . . 94
  6.4.2 Training . . . 95
  6.4.3 Hyperparameters . . . 95
  6.4.4 Results . . . 96
  6.4.5 Visualizations . . . 96
6.5 Conclusion . . . 96

7 Concluding Remarks 97
7.1 Findings . . . 97
7.2 Future Directions of Research . . . 98

Bibliography 101

Curriculum Vitae 118

List of Figures

1.1 Illustration of data-driven pipeline for trajectory forecasting. We build a taxonomy of the interaction encoder. 2
1.2 Illustration of the decision-making process of discrete choice models. 4

2.1 Toy illustration of the human trajectory forecasting problem. 10
2.2 The data-driven pipeline of human trajectory forecasting. 14
2.3 Illustration of the grid-based interaction modules. 15
2.4 Illustration of the non-grid-based interaction modules. 17
2.5 Illustration of our proposed training paradigm to train socially-aware forecasting models. 20
2.6 Illustration of the equivalence between our forecasting model and graph neural networks. 21
2.7 Our proposed hierarchy for trajectory categorization in TrajNet++. 23
2.8 Visualization of the main trajectory categories. 23
2.9 Visualization of the interacting trajectory categories. 24
2.10 Visualization of sample scenes from our TrajNet++ benchmark. 24
2.11 Illustration of distance-based unimodal metrics. 25
2.12 Illustration of our proposed collision metrics. 26
2.14 Illustration of our filtering procedure to generate Trajnet++ synthetic dataset. . 29
2.15 Samples from the TrajNet++ synthetic dataset generated using ORCA. 30
2.16 Visualizing the decision-making of grid-based interaction modules using layer-wise relevance propagation. 37

3.1 Toy illustration of the multimodal trajectory forecasting problem. 40
3.2 System figure of the SGANv2 architecture. 44
3.3 Illustration of trajectory refinement using collaborative sampling. 49
3.4 Visualization of the outputs of a simple uniform predictor. 51
3.5 Illustration of collaborative sampling reducing model collisions on TrajNet++ synthetic and real-world datasets. 52
3.6 Qualitative illustration of effectiveness of SGANv2 to prevent mode collapse on Forking Paths dataset. 56
3.7 Illustration of the Distance-To-Goal metric. 57

4.1 Toy illustration of the social anchors framework. 62

4.2 System figure of the social anchors framework. 65
4.3 Qualitative illustration of the ability of social anchors framework to output high-level interpretable intents. 71
4.4 Qualitative illustration of the effect of direction normalization on model training. 73

5.1 System figure of the motion style adapters. 78
5.2 Illustration of the modularized adaptation strategy. 81
5.3 Distribution of trajectories of pedestrians (blue) and bikers (red) on *deathCircle*. 83
5.4 Qualitative illustration of effectiveness of motion style adapters on Stanford Drone dataset. 83
5.5 Quantitative evaluation of motion style adapters on Level 5 prediction dataset. . 84
5.6 Training-Adaptation split for Level 5 prediction dataset. 84
5.7 Illustration of the attention heatmaps before and after adaptation on Level 5 using motion style adapters. 85
5.8 Y-Net-Mod encoder architecture 86
5.9 Car and truck trajectory distribution in inD dataset *scene1*. Both share the same scene and differ largely in velocity distribution. 86
5.10 Velocity distribution of *car* and *truck* in inD scene1. Static trajectories have been filtered. 86
5.11 Quantitative evaluation of modularized adaptation strategy on agent motion transfer. 87
5.12 Quantitative evaluation of modularized adaptation strategy on scene transfer. . . 87

6.1 System figure of DriverGym. 90
6.2 Visualization of an episode rollout in DriverGym. 92
6.3 Example Rasterization Modes 93
6.4 Evaluation Plan 93

List of Tables

2.1 Acronyms for the various designs of interaction modules. 18
2.2 Quantitative performance of interaction modules on TrajNet++ synthetic dataset. 32
2.3 Quantitative performance of interaction modules on TrajNet++ real world dataset. 33
2.4 Speed comparison of grid-based modules at test time. 34
2.5 Quantitative analysis of the effectiveness of the various components of our proposed training paradigm on TrajNet++ real world dataset. 35

3.1 High-level comparison of our proposed SGANv2 against selected common generative models for trajectory forecasting. 41
3.2 Quantitative comparison on ETH-UCY using Top-20 metric. 51
3.3 Quantitative evaluation on TrajNet++ synthetic dataset. 52
3.4 Quantitative evaluation on ETH/UCY datasets. 53
3.5 Quantitative evaluation on TrajNet++ real world dataset. 54
3.6 Quantitative analysis of importance of interaction modelling in generative models. 55
3.7 Quantitative evaluation of various SGANv2 design choices on TrajNet++ synthetic dataset. 57

4.1 Quantitative comparison of social anchors on TrajNet++ synthetic dataset. . . . 70
4.2 Quantitative comparison of social anchors on TrajNet++ real world dataset. . . 70
4.3 Effect of direction normalization on various baselines on TrajNet++ synthetic dataset. 72
4.4 Effect of direction normalization on various baselines on TrajNet++ real world dataset. 72

5.1 Quantitative evaluation of motion style adapters on Stanford Drone and Intersection Drone datasets. 82

6.1 Comparison of various open-source RL simulation environments for autonomous driving. 91
6.2 Description of the various metrics provided in DriverGym closed-loop evaluation protocol. 94
6.3 Evaluation of different training strategies in DriverGym. 95

1 Introduction

1.1 Motivation

One of the great endeavors of research is equipping machines with intelligence to co-exist alongside humans. Recent developments in deep learning have made giant strides in this direction, with models achieving remarkable performance on a variety of visual perception tasks ranging from image classification [1, 2] and object detection [3, 4, 5] to image segmentation [6, 7, 8] and remote sensing [9, 10]. Such advances promise to reshape the future of mobility where autonomous systems will be moving alongside humans in everyday life. In fact, several large-scale efforts are already underway to realize this goal [11, 12, 13]. Despite the great progress in this field, several important questions require further exploration before we can realize our objective of ubiquitous autonomous systems.

Till date, there have been several instances of erroneous human-robot interactions by autonomous systems, leading to failures like getting stuck in crowds, and in rare, unfortunate cases even resulting in human fatalities [14, 15]. Such cases highlight one of the most critical challenges preventing the widespread deployment of autonomous systems in society: the ability to forecast the motion of surrounding humans. Humans can naturally navigate through social situations as they have the ability to infer other people's intents, thereby making decisions based on predictions of what might happen in the future. Similarly, the decision-making algorithm of autonomous agents is required to not only perceive the surroundings but also predict how their dynamic environment will evolve in the future. Thus, developing methods to forecast the motion of humans forms the core objective of this thesis. To improve the performance of motion forecasting, we address the shortcomings of current methods and subsequently equip these models with desirable attributes to facilitate real world deployment.

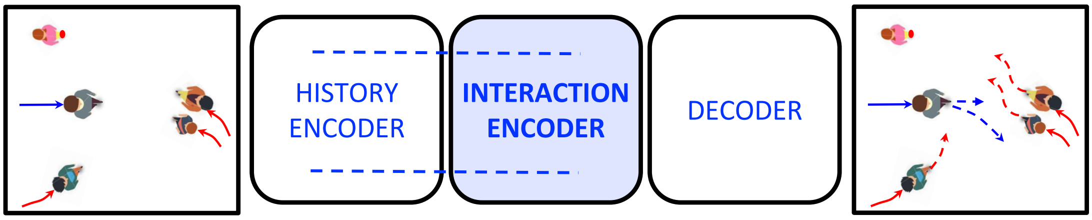


Figure 1.1: One of the fundamental challenges in human motion forecasting is modelling social interactions among pedestrians. A collection of interaction encoders have been proposed in literature, which are neural network blocks that learn social interactions in a data-driven fashion. In this work, we present a taxonomy of these interaction encoders.

1.2 Our Approach

This thesis represents a culmination of efforts towards the goal of improving human trajectory forecasting along four major fronts: (1) social interaction modelling, (2) multimodality and trajectory refinement, (3) injecting interpretability in the data-driven framework, and (4) efficient model adaptation to unseen scenarios.

To assist in the definition of human trajectory forecasting, we introduce the notion of a *trajectory* and a *scene*. We define a *trajectory* as the time-profile of human motion states. Generally, a motion state is composed of the position and velocity of a human. However, it can comprise more complex attributes like body pose that can help glean more information about a person's movement. A *scene* is defined as a collection of human trajectories in a social setting, which may include physical context that affects the human trajectories, *e.g.*, walls, doors, and elevators. We define human trajectory forecasting as follows:

Given the past trajectories of all humans in a scene, forecast the most likely future trajectories which conform to the social norms.

Human motion forecasting can be viewed as a sequence prediction problem: given the past motion states of all pedestrians, predict their future motion states. However, there is a crucial difference between this task and other sequence prediction tasks such as language modelling and weather forecasting: the trajectory of a pedestrian is affected by the motion of the others in its surroundings. Thus, one of the fundamental challenges in human motion forecasting is modelling these social interactions among pedestrians. Specifically, the forecasting model is required to capture the dependencies (social interactions) between multiple correlated sequences (pedestrian trajectories). We refer to this task as *social interaction modelling*.

In recent times, numerous neural network designs that learn social interactions in a data-driven fashion have been proposed in literature [16, 17, 18]. We refer to them as *interaction encoders*. Fig. 1.1 illustrates these encoders within the data-driven pipeline for trajectory forecasting. To shed light on the key design choices that contribute towards modelling social awareness, we build a taxonomy of these interaction encoders based on their underlying components. Given our findings, we propose a novel design that empirically outperforms existing encoders.

Another pivotal aspect that affects the interaction modelling capability of deep forecasting models is the training strategy. The prevalent paradigm is to predict and penalize all pedestrians in a scene simultaneously. However, a significant portion of trajectories in real world datasets are fairly easy to forecast while the safety-critical scenarios are rare [19]. We tackle this data-imbalance issue using a novel training strategy that forecasts and penalizes only one interacting pedestrian per scene, while utilizing the ground truth for the others at training time. Additionally, we stress the importance of incorporating the model variance within the training objective so that the model is penalized less for uncertain samples.

Correctly evaluating a network's ability to model social interactions is as important as designing the architecture itself. Currently, the improvement in distance-based metrics serves as the indicator of a model's ability to learn social interactions [16, 17, 20]. While these metrics are appropriate for evaluating forecasting results on an individual pedestrian level, they do not explicitly evaluate the predictions on the scene level. We posit that it is equally, if not more, important to evaluate the collision rate as the measure of social awareness. Furthermore, there does not exist a fixed test set to benchmark various forecasting models, unlike other fields like vision [21, 22] and language [23]. Thus, to objectively and meaningfully evaluate the performance of forecasting models, we develop a interaction-centric trajectory forecasting benchmark, a significant yet missing component in the field.

Given the past observations in a scene, multiple socially plausible futures exist. We refer to this property as *multimodality*. In addition to social interaction modelling, a forecasting network is required to exhibit multimodality. The problem of multi-modal trajectory forecasting is often approached from the perspective of learning a generative model over future trajectories. Generative adversarial networks (GANs) [24] are a popular choice of generative models owing to successful applications in vision. Indeed, several forecasting works utilize GAN-based designs to output multiple futures [17, 25, 26]. However, we discover that these models output predictions that undergo collisions as well as fail to cover all the future modes.

The failure to output collision-free trajectories can be attributed to the fact that the current discriminator designs do not fully model human-human interactions; hence the discriminator is incapable of differentiating real trajectory data from fake data. Only when the discriminator is capable of differentiating real data from fake data, can the supervised signal from it be meaningful to teach the generator. Additionally, current works do not leverage the recent advances in sequence modelling and utilize an LSTM-based discriminator. These observations motivate our proposed GAN design for safety-compliant multimodal forecasting.

Learning a perfect generative model for trajectory forecasting remains a challenging task. An imperfect generator can potentially output *bad* predictions at test time: predictions which undergo collision. It is easier for the discriminator to recognize that the real distribution is not being modeled correctly than for the generator to model the underlying distribution accurately. Thus,

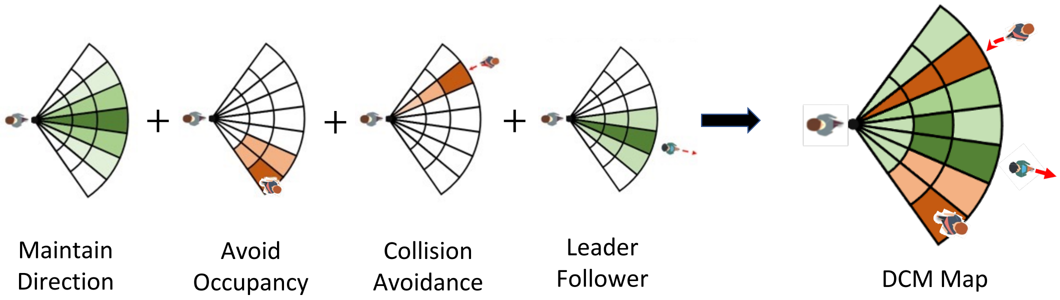


Figure 1.2: In discrete choice modelling, the next-step pedestrian choice can be explained by the underlying expert-designed functions like leader follower, collision avoidance.

to handle bad predictions, a potential solution is to leverage the information within the weights of the learned discriminator at test time. Given this intuition, we develop a novel collaborative sampling technique between the generator and discriminator to refine the predictions during test time.

Forecasting models are envisioned to be a key component of autonomous systems. The safety-critical nature of such applications brings forth the requirement of *interpretability*: the model should provide an easy-to-interpret rationale behind its behavior, so that one can understand what triggered a particular decision. While neural networks have the capacity to learn complex interactions, it is difficult to understand the "why" or simply the impact of pedestrians on others' trajectories. On the other hand, analyzing the independent components behind the decision-making of the hand-crafted classical models [27, 28] naturally provides a backbone for interpretability (as shown in Fig. 1.2). However, the manually chosen functions limit the expressibility of these models. Thus, we develop a framework that combines the interpretability of discrete choice models [27] with the predictive power of neural networks.

During real world deployment, forecasting models can come across unseen scenarios, which can result in a distribution shift. Such shifts can negatively impact the performance and lead to undesirable consequences [29, 30]. Thus, it is important to develop an *adaptation algorithm* that can update the pre-trained forecasting model to perform well in the unseen target domain. The standard approach is to fine-tune the whole pre-trained model on data collected from target domain. However, directly updating the model is often sample inefficient, as it fails to exploit the inherent structure of the distributional shifts in motion context.

In the forecasting setup, the physical laws behind motion dynamics are generally invariant across different domains such as geographical locations and agent types: all agents move towards their goal and avoid collisions. The distribution shift can be largely attributed to the changes in the *motion style*: the way an agent interacts with its surroundings. For instance, pedestrians maintain a larger inter-personal distance in a park as compared to a railway station. Given this decoupling of motion dynamics, we propose an efficient adaptation algorithm that only accounts for the updates in the target motion style.

One of the main applications of developing socially-aware trajectory forecasting methods is improving downstream tasks like planning in autonomous robots [31, 32]. Current reinforcement learning policies are trained on real-world data using log-replay or using simulators like CARLA [33] and SUMO [34]. However, these simulators utilize hand-coded rules for surrounding agents' motion that tends to be unrealistic and display a limited variety of behaviors. To address this issue, we develop an open-source environment that allows to simulate the surrounding agents using deep trajectory forecasting methods.

1.3 Thesis Contributions

The thesis focuses on identifying and addressing the shortcomings of current motion forecasting methods in modelling social interactions and outputting multiple futures, developing a data-driven forecasting framework that incorporates interpretability in its decision-making and devising a novel adaptation strategy for forecasting models. Further, we present a large-scale interaction-centric benchmark TrajNet++, a significant yet missing component in the field of human trajectory forecasting. Concretely, our key contributions are:

Social Interaction Modelling. We provide an analysis of existing designs of data-driven interaction encoders. We propose a new design that utilizes domain knowledge and is empirically robust to noise in real-world datasets. To promote interaction awareness in forecasting models, we develop a novel training paradigm that explicitly focuses on samples that undergo interactions and incorporates model uncertainty. [Details in : Chapter 2]

Multimodality. We present a novel generative adversarial network, coined SGANv2, that incorporates three components for safety-compliant multimodal forecasting: (1) spatio-temporal interaction modelling in both the generator and the discriminator, (2) transformer-based discriminator, (3) collaborative sampling between the generator and discriminator at test time. Our architecture reduces the collisions and distance-based metrics, while preventing mode collapse. [Details in : Chapter 3]

Incorporating Interpretability. We present a framework that combines the interpretability of discrete choice models with the expressibility of neural networks. We show that our framework outputs accurate future trajectories as well as provides a high-level rationale behind its predictions. [Details in : Chapter 4]

Adapting Forecasting Models. We propose a sample-efficient adaptation strategy that exploits the inherent structure of motion style shifts. We introduce a low-rank motion style adapters that project and adjust the target style features at a low-dimensional bottleneck. To further boost efficiency, we propose a modular adaptation strategy facilitating a fine-grained choice of adaptation layers. [Details in : Chapter 5]

Forecasting Benchmark. We present TrajNet++, an interaction-centric trajectory forecasting benchmark comprising explicit agent-agent scenarios. Our benchmark provides proper indexing of trajectories by defining a hierarchy of trajectory categorization. In addition, we provide an extensive evaluation system to benchmark various models in an objective manner. [Details in : Chapter 2]

Realistic Simulator. We present DriverGym, an open-source gym-compatible environment specifically tailored for developing planning algorithms for autonomous driving. In addition to access to more than 1000 hours of expert logged data, DriverGym also supports reactive and data-driven agent behavior. Moreover, the performance of a policy can be easily validated on real-world data using our extensive and flexible closed-loop evaluation protocol. This chapter results from an applied research project in collaboration with the autonomous driving company Lyft Level 5 (now, Woven Planet). [Details in : Chapter 6]

1.4 Thesis Structure

The body of this thesis is organized into seven chapters, the first of which is the introduction.

Chapter 2 focuses on modelling social interactions which dictate how the motion of a human affects others. We present a taxonomy of the interaction encoders based on their underlying components. Subsequently, we propose a novel design that incorporates domain knowledge, and is empirically robust to the noise in real-world datasets. To promote interaction-awareness in forecasting models, we propose a novel training paradigm that explicitly focuses on samples that undergo interactions and incorporates model uncertainty. Finally, we describe our interaction-centric benchmark TrajNet++ to objective compare trajectory forecasting models. Experiments on TrajNet++ quantitatively validate the effectiveness of our proposed interaction encoder and training strategy, as well as the need for incorporating collision-based metrics in model evaluation. This chapter expands on the paper published in T-ITS'21 [35].

Chapter 3 tackles the challenge of predicting socially acceptable multimodal output. We present our novel GAN design called SGANv2 for safety-compliant multimodal forecasting. In particular, we equip both the generator and the discriminator with the ability to model the temporal evolution of spatial interactions. In addition, we design a transformer-based discriminator to better guide the learning process of the generator. Furthermore, we develop a novel collaborative sampling technique between the generator and discriminator that re-uses the learned discriminator at test time to refine the generator predictions. Experiments on real-world datasets demonstrate the effectiveness of our proposed architecture to reduce collisions, and its potential to prevent mode collapse. The work in this chapter condenses two publications: a T-ITS (accepted) and a AAAI'20 [36] one.

Chapter 4 incorporates high-level interpretability within the deep motion forecasting models. We introduce our social anchor framework that combines interpretability of discrete choice models with the expressibility of neural networks. Our framework can be viewed as disentangling high-level interpretable intents and lower-level scene-specific nuances of human motion. Experiments on TrajNet++ show that our framework outputs a high-level rationale behind its predictions without compromising on the distance-based metrics. This chapter expands on the paper we presented at CVPR'21 [37].

Chapter 5 tackles the problem of adapting pre-trained forecasting models to unseen scenarios using limited samples. We introduce motion style adapters (MoSA), that are new modules inserted in parallel to the encoder layers, which project and adapt the target domain features at a low-dimensional bottleneck. To further improve efficiency, we describe a modularized adaptation strategy that disentangles the features of scene context and motion history to facilitate a fine-grained choice of adaptation layers. We demonstrate that our method outperforms existing fine-tuning methods on three real-world datasets in low-shot transfer. This chapter is based on the paper accepted at CoRL'22 [38].

Chapter 6 presents the internal working of DriverGym, an open-source gym environment for training reinforcement learning (RL) policies for autonomous driving. We address the issue of non-realism of surrounding agent behavior during simulations and provide the flexibility to incorporate data-driven forecasting models. Furthermore, our environment supports more than 1000 hours of real-world expert data and provides a configurable and extensible evaluation protocol. We provide behavior cloning baselines using supervised learning and RL, trained in DriverGym. This chapter is based on the paper accepted at NeurIPS'21 Workshop on Autonomous Driving [39].

Finally, Chapter 7 concludes with a summary of the contributions presented in this thesis and outlines directions for future work. Note that we do not have a common *Related Work* chapter, rather we build the related works of each chapter on top of one another.

1.5 Related Publications

This thesis is based on the material published in the following papers:

- **Parth Kothari**, Danya Li, Yuejiang Liu and Alexandre Alahi, *Motion Style Transfer: Modular Low-Rank Adaptation for Deep Motion Forecasting*, CoRL (2022) [URL]
- **Parth Kothari** and Alexandre Alahi, *Safety-compliant Generative Adversarial Networks for Human Trajectory Forecasting*, Accepted, IEEE Transactions on Intelligent Transportation Systems
- Yuejiang Liu, **Parth Kothari**, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan,

and Alexandre Alahi, *TTT++: When Does Self-Supervised Test-Time Training Fail or Thrive?*, NeurIPS (2021) [URL]

- **Parth Kothari**, Christian Perone, Luca Bergamini, Alexandre Alahi and Peter Ondruska, *Drivergym: Democratising reinforcement learning for autonomous driving*, Machine Learning for Autonomous Driving, NeurIPS Workshop (2021) [URL]
- **Parth Kothari**, Brian Sifringer and Alexandre Alahi, *Interpretable Social Anchors for Human Trajectory Forecasting in Crowds*, CVPR (2021) [URL]
- **Parth Kothari**, Sven Kreiss, and Alexandre Alahi, *Human Trajectory Forecasting in Crowds: A Deep Learning Perspective*, IEEE Transactions on Intelligent Transportation Systems (2021) [URL]
- Yuejiang Liu*, **Parth Kothari***[I] and Alexandre Alahi, *Collaborative Sampling in Generative Adversarial Networks*, AAAI (2020) [URL]
- **Parth Kothari** and Alexandre Alahi, *Adversarial Loss for Human Trajectory Prediction*, hEART 2019: 8th Symposium of the European Association for Research in Transportation [URL]

[I] denotes equal contribution

2 Learning Social Interactions for Human Trajectory Forecasting

This chapter is based on the article:

Parth Kothari, Sven Kreiss, and Alexandre Alahi, *Human Trajectory Forecasting in Crowds: A Deep Learning Perspective*, IEEE Transactions on Intelligent Transportation Systems (2021)

2.1 Introduction

This chapter tackles the most crucial aspect of human trajectory forecasting: modelling social interactions between pedestrians. Humans possess the natural ability to easily navigate through complex social scenarios. In other words, when people move in crowded spaces like airports and mall, they effortlessly follow the social etiquette of human motion like respecting personal space, yielding right-of-way, avoiding to walk through people belonging to the same group. These social interactions lead to various interesting pattern-formation phenomena in crowds, for instance, the emergence of lanes of pedestrians with uniform walking direction, oscillations of the pedestrian flow at bottlenecks.

The ability to model these interactions and consequently forecast motion dynamics in real-world environments is extremely valuable for a wide range of applications: infrastructure design [40, 41, 42], traffic operations [43], crowd abnormality detection systems [44], evacuation situation analysis [45, 46, 47, 48, 49], planning of intelligent transport systems [31, 32, 50, 51, 52] and recently helping in the broad quest of building a digital twin of our built environment.

We re-iterate the notion of a *trajectory* and a *scene*. We define a *trajectory* as the time-profile of human motion states. Generally, the motion state is composed of the position and velocity of a human. However, it can comprise more complex attributes like body pose that can help glean more information about a person's movement. A *scene* is defined as a collection of human trajectories interacting in a social setting. Additionally, a scene can include physical objects and non-navigable areas that affect the human trajectories, *e.g.,* walls, doors, and elevators. Wherever necessary, we refer to a particular human of interest in the scene as the *primary pedestrian.* We

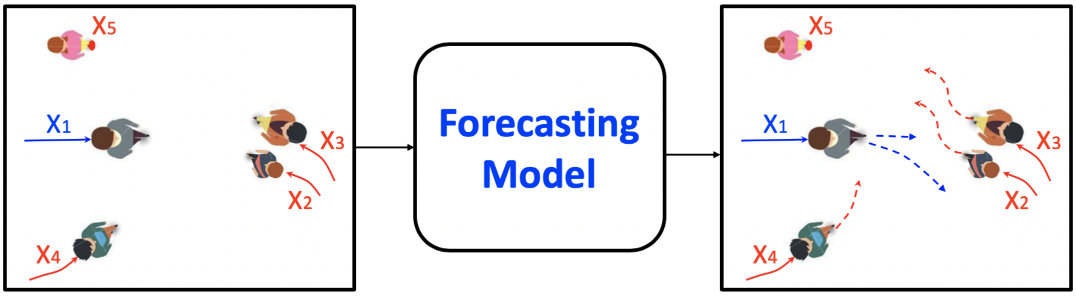


Figure 2.1: Human trajectory forecasting is the task of forecasting the future trajectories (dashed) of all humans which conform to the social norms, given the past observed scene (solid). The presence of social interactions distinguish human trajectory forecasting from other sequence modelling tasks: the primary pedestrian (X1) deviates from his direction of motion to avoid a collision, by forecasting the trajectory of the child (X2).

tackle the following objective:

Given the past trajectories of all humans in a scene, forecast the future trajectories which conform to the social norms.

Human trajectory forecasting is primarily a sequence modelling task. It comprises the typical challenges for a sequence modelling task like encoding observation sequence and modelling long-term dependencies. However, there exist two crucial challenges that differentiate it from other sequence prediction tasks such as language modelling, weather forecasting, and stock market forecasting (see Fig 2.1):

- **Presence of social interactions**: the trajectory of a person is affected by the motion of other people in his/her surroundings. Modelling how the observation of one sequence affects the forecast of another is an essential requirement for a good human trajectory forecasting model.
- **Physically acceptable outputs**: a good human trajectory forecasting model should provide physically acceptable outputs, for instance, the model prediction should not undergo collisions. Further, quantifying the physical feasibility of a model prediction is crucial for safety-critical applications.

We view the problem of human trajectory forecasting from the perspective of learning a representation of human social interactions. Modelling social interactions is an extremely challenging task as there exists no fixed set of rules that govern human motion. Early works handcrafted this representation based on domain knowledge [27, 53]. However, social interactions in crowded environments are not only diverse but often subtle. Following the success of Social LSTM [16], a variety of neural network-based modules have been proposed in literature that model social interactions in a data-driven manner.

In this chapter, we focus on the design and evaluation of interaction modules, which are neural networks that capture social interactions in crowds. The challenge in designing these interaction modules lies in handling a variable number of neighbours, identifying the neighbours of interest and modelling how they collectively influence the future trajectories of a pedestrian. Given the large number of existing designs, we present a taxonomy encompassing most of the designs of interaction modules. Based on our taxonomy, we propose an effective interaction module which incorporates domain knowledge into the neural network design. As a consequence, our module is better equipped to learn social etiquette like collision avoidance and leader-follower.

In addition to designing interaction modules, it is equally important to develop training techniques that help to better learn the underlying social interactions. We propose a training strategy to train socially-aware trajectory forecasting models. Specifically, we introduce a novel training paradigm that only forecasts the trajectory of pedestrians that undergo interactions during training. Furthermore, to improve the model training, we highlight the importance of incorporating the model variance within the loss objective such that the model is penalized less for uncertain samples. We empirically demonstrate that our training algorithm helps to output accurate as well as physically acceptable trajectories.

A long-standing question in data-driven trajectory forecasting models is to explore techniques that help to explain the model decisions. Explaining model decisions is particularly important in our setup due to the potential model deployment in safety-critical applications. In this chapter, we propose to utilize the popular Layer-wise Relevance Propagation (LRP) [54] to explain the decisions of our trajectory forecasting models. To the best of our knowledge, this is the first work that applies LRP, in a *forecasting* setting, to infer inter-sequence (neighbours) effects on the model output.

To demonstrate the efficacy of a trajectory forecasting model, one needs to have the means to objectively compare with other forecasting baselines on good quality datasets. However, current methods have been evaluated on different subsets of available data without a proper sampling of scenes in which social interactions occur. As our final contribution, we introduce **TrajNet++**, a large scale interaction-centric trajectory forecasting benchmark comprising explicit agent-agent scenarios. Our benchmark provides proper indexing of trajectories by defining a hierarchy of trajectory categorization. In addition, we provide an extensive evaluation system to test the gathered methods for a fair comparison. In our evaluation, we go beyond the standard distance-based metrics and introduce novel collision metrics that measure the capability of a model to emulate pedestrian behavior in crowds.

We demonstrate the efficacy of our proposed methods and training techniques on TrajNet++. Our proposed interaction module reduces the collisions in the real-world by 20%, without compromising the distance-based metrics. We highlight the effectiveness of our proposed training strategies in reducing both distance-based and collision metrics. Finally, we illustrate that the decisions of our proposed model architecture can be well-explained using LRP in real-world scenarios.

To summarize, our main contributions are as follows:

1. We provide an in-depth analysis of existing designs of data-driven interaction encoders along with their source code. We propose an interaction model design driven by domain knowledge that reduces collisions by 20% in the real-world.
2. To promote interaction-awareness in forecasting models, we develop a novel training paradigm that explicitly focuses on samples that undergo interactions, and incorporates model uncertainty.
3. We explain the decision-making of our models by extending layer-wise relevance propagation to the trajectory forecasting setting.
4. We present TrajNet++, a large scale interaction-centric trajectory forecasting benchmark with novel evaluation metrics that quantify the *physical feasibility* of a model.

2.2 Related Work

Developing models of pedestrian motion in crowds has been an active area of research for the last few decades [16, 27, 28, 55, 56, 57, 58, 59]. In the 1980s, the motivation of developing such models lay in their application for designing and planning pedestrian areas such as railway stations and shopping malls. The earliest works focused on macroscopic approaches [55, 56] that directly modelled the flow of crowds as a whole. The crowd modelling paradigm later shifted to microscopic approaches, where the behavior of individual pedestrians became the basis for developing models that describe pedestrian groups or pedestrian crowds.

On a high level, the classical microscopic models [27, 28, 57, 58] cast the observed *rules* of individual pedestrian behavior into different mathematical frameworks such that the resulting formulation fitted the collected crowd data well. These rules refer to commonly observed individual pedestrian behavior such as (i) the attraction of an individual towards its destination, (ii) influence of neighbours on an individual's motion, and (iii) the attraction to specific individuals (e.g., friends) and physical objects (e.g., window display). For instance, the social force model [28] casts the rules into hand-crafted equations of motion, similar to Newtonian mechanics. These equations resulted in the physical production of an acceleration or deceleration force that governed the motion of individuals in crowds. Another prominent model for simulating human motion is Reciprocal Velocity Obstacles (RVO) [60], which guarantees safe and oscillation-free motion, assuming that each agent follows identical collision avoidance reasoning.

Classical microscopic models showcased a variety of commonly observed crowd phenomenon very realistically such as formation of dynamically varying lanes of pedestrians walking in the same direction [28], formation of freely forming groups and queue behavior [61]. They have also proved useful in applications like automatic video surveillance [27], tracking [62] and anomaly detection [44]. Crowd behavior modelling has also been approached from several other perspectives such as continuum dynamics [59], interaction gaussian processes [63], and additional functions have been defined to model different groups types [64, 65, 66]. Despite such intensive modelling process, the manually designed functions and potentials within the various mathematical frameworks limit the predictability power of these models. These functions impose not only strong priors but also have limited capacity when modelling complex interactions. In recent times, methods based on neural networks that infer interactions in a data-driven fashion have been shown to outperform the works mentioned above.

In the last decade, with the advent of computational resources and publicly available large-scale datasets [21, 22], deep learning began to demonstrate great potential to learn several vision and language tasks [1, 67, 68]. This served as an inspiration to explore neural network architectures that can learn human-human interactions and forecast human motion in a data-driven fashion. Inspired by the application of recurrent neural networks (RNNs) in diverse sequence prediction tasks [69, 70, 71, 72], Alahi *et al.* [16] proposed Social LSTM, the first neural network model for human trajectory forecasting. Social LSTM is a Long-Short Term Memory network (LSTM) [73] with a novel social pooling layer to capture social interactions of nearby pedestrians.

Following the success of Social LSTM, the social pooling module has been extended to incorporate physical space context [74, 75, 76, 77, 78, 79, 80] and various other grid designs interaction modelling have been proposed [17, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93]. Pfieffer *et al.* [81] proposed an angular pooling grid for efficient computation while Shi *et al.* [83] proposed an elliptical pooling grid placed along the direction of movement of the pedestrian with more focus on the pedestrians in the front. Bisagno *et al.* [84] proposed to consider only pedestrians not belonging to the same group during social pooling while, Hasan *et al.* [91, 92] only consider the pedestrians in the visual frustum of attention [94].

Other designs of interaction modules incorporated different aggregation strategies to pool neighbour information. Gupta *et al.* [17] proposed to use of a permutation-invariant (symmetric) max-pooling function, while Ivanovic *et al.* [87] and Salzmann *et al.* [93] proposed to sum-pool the neighbour states. Several works [25, 26, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107] propose attention mechanisms [108, 109] to identify the neighbours which affect the trajectory of the person of interest. The attention weights are either learned or handcrafted based on domain knowledge (*e.g.*, euclidean distance). Zhang *et al.* [85] proposed to refine the state of the LSTM cell using message passing algorithms.

Many interaction module designs differ in their input modalities. Liang *et al.* [88] proposed to utilize geometric relations obtained from the spatial distance between pedestrians, to derive the interaction representation. [89, 90] propose to input the relative position and relative velocity of k nearest neighbours. For an extensive survey of all human forecasting methods , one can refer to Rudenko *et al.* [110]. In this work, we focus on the underlying components of data-driven interaction modules and present a taxonomy encompassing most designs in literature.

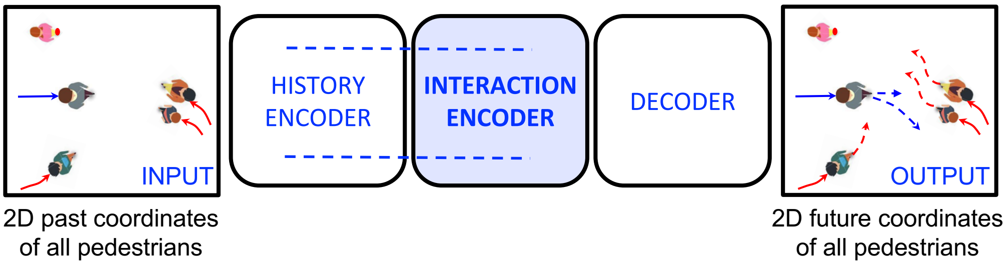

INPUT
2D past coordinates of all pedestrians
HISTORY ENCODER
INTERACTION ENCODER
DECODER
OUTPUT
2D future coordinates of all pedestrians

Figure 2.2: The data-driven pipeline for human trajectory forecasting. Different designs of interaction encoders have been proposed in literature, which are neural network blocks that learn social interactions in a data-driven fashion. In this chapter, we build a taxonomy of these interaction encoders.

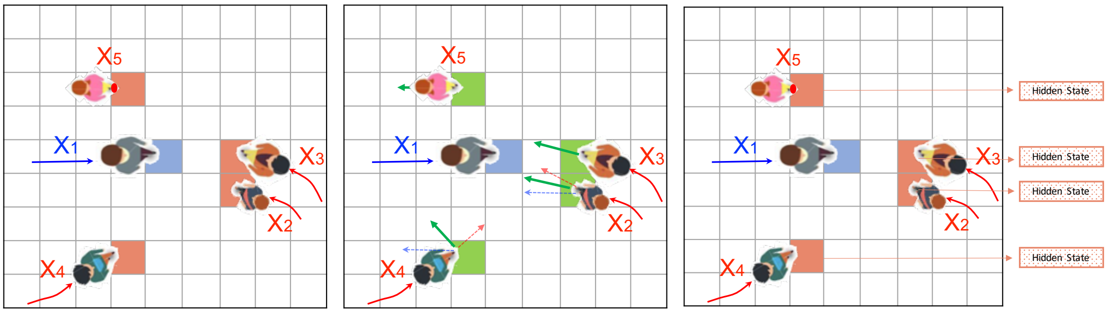
2.3 Method

A global data-driven pipeline for forecasting human motion is illustrated in Fig 2.2. It comprises of the motion encoding module, the interaction module and the decoder module. On a high level, the motion encoding module is responsible for encoding the past motion of pedestrians. The interaction module learns to capture the social interactions between pedestrians. The motion encoding module and the interaction module are not necessarily mutually exclusive. The output of the interaction module is the social representation of the scene. The social representation is passed to the decoder module to predict a single trajectory or a trajectory distribution depending on the decoder architecture. Since the objective of this chapter is to model human social interactions, we focus on investigating the design choices for the interaction module.

2.3.1 Problem statement

The forecasting model inputs the trajectories of all the people within a scene denoted by $\mathbf{X} = \{\mathbf{X_1}, \mathbf{X_2}, \ldots, \mathbf{X_N}\}$, where $\mathbf{X_i}$ is the trajectory of person i and N denotes the total number of people in the scene. The objective is to forecast the corresponding ground-truth future trajectories $\mathbf{Y} = \{\mathbf{Y_1}, \mathbf{Y_2}, \ldots, \mathbf{Y_N}\}$. Specifically, we are provided the motion states of all pedestrians at time-steps $t = 1, \ldots, T_{obs}$ (input trajectory) and we need to forecast the future motion states from time-steps $t = T_{obs+1}$ to T_{pred} (output trajectory). We denote the model predictions using $\hat{\mathbf{Y}} = \{\hat{\mathbf{Y}}_\mathbf{1}, \hat{\mathbf{Y}}_\mathbf{2}, \ldots, \hat{\mathbf{Y}}_\mathbf{N}\}$.

At time-step t, we denote the motion state of pedestrian i by $\mathbf{s}_i^t$. The state can comprise different attributes of a person, for instance, the position concatenated with velocity ($\mathbf{s}_i^t = [\mathbf{x}_i^t, \mathbf{v}_i^t]$, where $\mathbf{x}_i^t$ denotes the x-y position coordinates and $\mathbf{v}_i^t$ denotes the x-y velocity coordinates of pedestrian i at time t). The problem statement can be extended to input additional attributes like body pose, as well as predicting k most-likely future trajectories.



(a) Occupancy pooling [16] (b) Directional pooling [Ours] (c) Social pooling [16]

Figure 2.3: Illustration of the grid-based interaction encoding modules. (a) Occupancy pooling: each cell indicates the presence of a neighbour (b) Our proposed directional pooling: each cell contains the relative velocity of the neighbour with respect to the primary pedestrian. (c) Social pooling: each cell contains the LSTM hidden-state of the neighbour. The constructed grid tensors are passed through an MLP-based neural network to obtain the interaction vector.

2.3.2 Interaction module

Humans have the capability to navigate with ease in complex, crowded environments by following unspoken social rules, which result in social interactions. In recent years, these social interactions are captured effectively by designing novel interaction modules. In this section, we broadly categorize the different data-driven interaction encoders studied in literature, based on their underlying components. Following this, in the experimental section we empirically analyze the effectiveness of each of these components and provide recommendations for designing improved interaction modules. The existing designs can be broadly categorized into (1) **Grid based** and (2) **Non-Grid based**. We now discuss in detail the different components of these interaction encoders.

Grid Based Interaction Models

In grid-based models, the interaction module takes as input a local grid constructed around the primary pedestrian. Each cell within the grid represents a particular spatial position relative to the primary pedestrian. The design of grid-based modules largely differ based on neighbour state representation.

Neighbour state: Consider a grid constructed around the primary pedestrian, where each cell contains information about neighbours located in the corresponding position as shown in Fig. 2.3. Existing designs provide the information of neighbours in two main forms: (a) *Occupancy Pooling* [16, 77] where each cell in the grid indicates the presence of a neighbour (Fig 2.3a) (b) *Social Pooling* [16, 75, 76, 77, 79, 80, 84] where each cell contains the entire history of the neighbour, represented by, *e.g.*, the LSTM hidden state of the neighbours (Fig 2.3c). The obtained grid is flattened and subsequently embedded using an MLP to get the interaction vector p_i^t.

Directional pooling: In this chapter, based on our domain knowledge, we propose to additionally provide the relative velocity of each neighbour in the corresponding grid cell. When humans navigate in crowded environments, in addition to relative positions of the neighbours, they naturally tend to focus on their relative velocities [18]. For the same positional configuration, the relative velocities of neighbours lead to the concepts of leader-follower [111] and collision avoidance. In particular, leader-follower behavior arises when a neighbour is in front of the primary pedestrian and walks along the same direction, while this same positional configuration results in collision avoidance when the neighbour moves in the opposite direction. Having access to relative velocities can therefore significantly improve accuracy and reducing model prediction collisions.

Due to the complex nature of real-world movements combined with the possibility of noisy measurements, the current design of social pooling [16] may fail to become robust with regards to avoiding collisions. The forecasting models are expected to learn the notion of collision avoidance implicitly, as they are trained to minimize the displacement errors [16, 25] and not collisions. Explicit focus on relative velocity configurations can help to alleviate this issue as they provide more control over the design of the input grid. When the model explicitly focuses only on relative velocity configuration (rather than abstract hidden-state configurations), the resulting simple design has the potential to output safer predictions.

With regards to providing domain knowledge-driven inputs to the interaction module, it can be beneficial to only consider the neighbours in front of the primary pedestrian as proposed in [94]. We will demonstrate in the experimental section that directional pooling discovers this concept of focusing on the neighbours in the field-of-view of the primary pedestrian, directly from the data. In terms of computational efficiency, our proposed directional pooling is faster to deploy in real-time applications compared to social pooling due to the reduced size of input ($N \times N \times 2$ in comparison to $N \times N \times H_{dim}$ where H_{dim} is the hidden-state dimension).

Non-Grid Based Interaction Models

Non-grid based modules, as the name suggests, capture the social interactions in a grid-free manner. The challenge in designing non-grid based models lies in (1) handling a variable number of neighbours and (2) aggregating the state information of multiple neighbours to obtain the interaction vector p_t^i. As illustrated in Fig 2.4, the design choices of these modules can be categorized based on four factors: (a) neighbour state, (b) state embedding, (c) neighbour information aggregation strategy, and (d) aggregated vector embedding.

Neighbour state: Non-grid based methods do not contain an implicit notion of the spatial position of neighbours with respect to the primary pedestrian, unlike the grid-based counterparts. Hence, almost all the existing designs in literature take as input the relative spatial position of the neighbours. Another popular input choice is the hidden-state of the neighbouring pedestrian [17, 25] as the hidden-state has the ability to encode information regarding the motion history of the

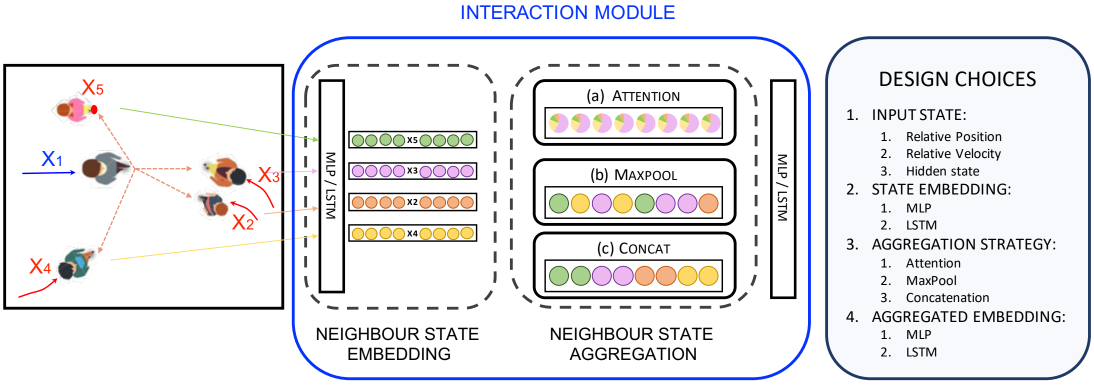


Figure 2.4: Illustration of the non-grid based encoding modules to obtain the interaction vector. The challenge lies in handling a variable number of neighbours and aggregating their state information to construct the interaction vector (a) Neighbour information is aggregated via attention mechanism (b) Neighbour information is aggregated utilizing a symmetric function (c) Neighbour information is aggregated via concatenation.

corresponding pedestrian. Amirian *et al.*[26] models the neighbour state using interaction-centric geometric features like bearing angle between agents and distance of closest approach [112]. Ivanovic *et al.* [87] takes as input the velocity of neighbours.

Input state embedding: The input states of the neighbours are then embedded, usually with an MLP. However, recent works [18, 100] based on graph neural network designs[113], embed the input states using an LSTM. Each connection of the primary pedestrian to his neighbour is modelled using a different LSTM. The LSTM helps to capture the temporal evolution of the neighbour states, unlike the first-order MLP.

Aggregation strategy: Given the embeddings of the neighbours, the most important challenge for non-grid based models is to design the ideal strategy to aggregate the information of all the neighbours. Gupta *et al.* [17] proposed to aggregate the interaction information by applying a symmetric max-pool function on the obtained neighbour state embeddings. Ivanovic *et al.* [87] and Hasan et al. [91] utilized the symmetric sum-pooling function.

A large body of works utilize the attention mechanism [108, 109] to determine the weights of different neighbours in predicting the future trajectory. These weights can be either hand-crafted [96] or learnt in a data-driven manner [25, 26, 98]. The attention mechanism can be applied multiple times to model higher-order spatial interactions [25].

The above aggregation strategies are benchmarked against a simple baseline that concatenates the neighbour embeddings. To handle a variable number of neighbours, the top-k neighbours are selected based on a defined criterion, (*e.g.*, euclidean distance). Despite the simplicity, we demonstrate that the concatenation strategy performs at par with its sophisticated counterparts.

Aggregated vector embedding: The aggregated neighbour vector is passed through an MLP,

Acronym (P-Q-R-S)	Input (P)	Embed-I (Q)	Aggreg. (R)	Embed-II (S)	References
O-Grid	Position	None	Grid	MLP	[16, 77, 81]
S-Grid	H-State	None	Grid	MLP	[16, 75, 76, 77, 79, 80, 84, 91]
D-Grid	Velocity	None	Grid	MLP	**Directional Pooling** [**Ours**]
D-MLP-Attn-MLP	Velocity	MLP	Attn	MLP	[83]
S-MLP-Attn-MLP	H-State	MLP	Attn	MLP	[25, 26, 95, 96, 98, 101, 105]
S-MLP-MaxP-MLP	H-State	MLP	MaxPool	MLP	[17]
D-MLP-ConC-MLP	Velocity	MLP	Concat	MLP	[89, 90]
D-MLP-SumP-LSTM	Velocity	MLP	SumPool	LSTM	[87]
O-LSTM-Att-MLP	Position	LSTM	Attn	MLP	[18, 100]
D-MLP-ConC-LSTM	Velocity	MLP	Concat	LSTM	**DirectConcat** [**Ours**]

Table 2.1: Model Acronyms: Acronyms for the various designs of interaction modules. We observe that most of the existing interaction encoder designs fall under our defined categorization.

with the exception of Ivanovic *et al.* [87] that pass the sum-pooled vector through an LSTM, to obtain the interaction vector p_t^i. We argue that encoding the aggregated vector using LSTMs offers the advantage of modelling higher-order interactions in the temporal domain. In other words, the interaction module learns how the interaction representations evolve over time.

For brevity, the interaction modules are denoted using acronyms determined by their designs. The acronyms are of the form **P-Q-R-S** where **P** denotes the input to the module, **Q** denotes the state embedding module, **R** denotes the information aggregation mechanism and **S** denotes aggregated vector embedding module. Table 2.1 illustrates how our categorization encompasses various designs of interaction modules in literature.

DirectConcat

Equivalent to our proposed `D-Grid`, we describe its non-grid counterpart *DirectConcat*. Grid-based models, based on their design, implicitly consider only those neighbours that are within the grid constructed around the primary pedestrian. We argue that modelling interactions of all pedestrians (even those far away) may lead to the model learning spurious correlations. Thus, we propose to consider only the top-*k* neighbours closest to the primary pedestrian.

Similar to aggregating the obtained directional grid by flattening the obtained grid, in *DirectConcat* we propose to concatenate the relative-velocity and relative-position embeddings of top-*k* neighbours. This preserves the unique identity of the neighbours as compared to mixing the different embeddings like in max-pooling [17] or sum-pooling [87]. Finally, we pass the aggregated vector through an LSTM as compared to an MLP. This design choice helps to model higher-order spatio-temporal interactions better and is more robust to noise in the real-world measurements. We demonstrate in the experimental section that indeed the LSTM embedding helps to improve the collision metric. *DirectConcat* is equivalent to `D-MLP-ConC-LSTM` given our taxonomy. We use the terms *DirectConcat* and `D-MLP-ConC-LSTM` interchangeably.

2.3.3 Forecasting model

We now describe the rest of the components of the forecasting model. To claim that a particular interaction module is superior, it is essential to keep the rest of the forecasting model components constant. Only then we can be sure that it was the interaction module that boosted performance, and not one of the extra added components. We choose the time-sequence encoder to be an LSTM due to its capability to handle varying input length and capture long-term dependencies. Moreover, most works utilize LSTMs as their base motion-encoding architecture.

The state of person i at time-step t, $\mathbf{s}_i^t$, is embedded using a single layer MLP to get the state embedding e_i^t. We represent each person's state using his/her velocity, as switching the input representation from absolute coordinates to velocities increases the generalization power of sequence encoder [17]. We obtain the interaction vector p_i^t of person i from the interaction encoder. We concatenate the interaction vector with the velocity embedding and provide the resultant vector as input to the sequence-encoding module. Mathematically, we obtain the following recurrence:

$$e_i^t = \phi(\mathbf{v}_i^t; W_{emb}), \tag{2.1}$$

$$h_i^t = LSTM(h_i^{t-1}, [e_i^t; p_i^t]; W_{encoder}), \tag{2.2}$$

where ϕ is the embedding function, $W_{emb}, W_{encoder}$ are the weights to be learned. The weights are shared between all persons in the scene.

The hidden-state of the LSTM at time-step t of pedestrian i is then used to predict the distribution of the velocity at time-step $t+1$. Similar to Graves [67], we output a bivariate Gaussian distribution parametrized by the mean $\mu_i^{t+1} = (\mu_x, \mu_y)_i^{t+1}$, standard deviation $\sigma_i^{t+1} = (\sigma_x, \sigma_y)_i^{t+1}$ and correlation coefficient ρ_i^{t+1}:

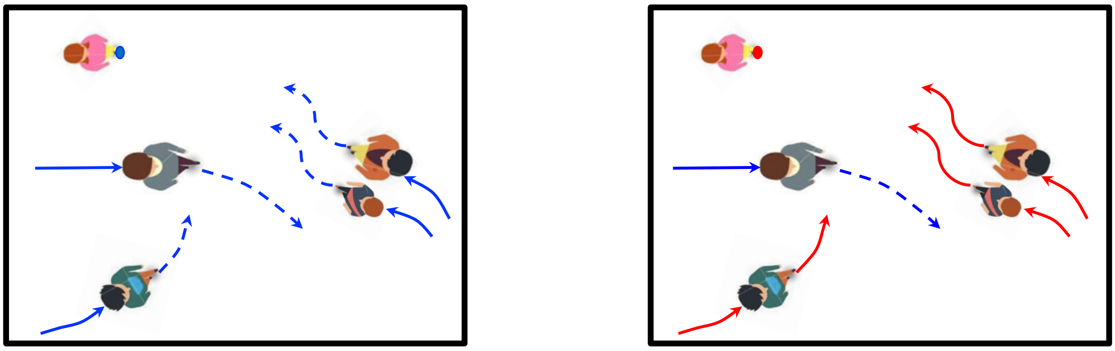
$$[\mu_i^t, \sigma_i^t, \rho_i^t] = \phi_{dec}(h_i^{t-1}, W_{dec}), \tag{2.3}$$

where ϕ_{dec} is modelled using an MLP and W_{dec} is learned.

2.3.4 Training paradigm

In addition to interaction module design, it is important to focus on the training procedure to ensure that forecasting models learn the underlying social interactions. As seen in recent works [17, 87], models without social pooling tend to perform better than their counterparts. It is not easy to train socially-aware forecasting models. We now highlight key ingredients to help forecasting models learn the underlying interactions.

Training Objective: We propose to learn the parameters of the forecasting model by minimizing the negative log-likelihood (NLL) loss:



(a) The prevalent training paradigm.

(b) Our proposed training paradigm.

Figure 2.5: The prevalent training paradigm of predicting all pedestrians (blue) struggles to learn the social interactions, as a significant portion of trajectories in the dataset are either static or linear. We propose a novel training strategy that forecasts and penalizes only one interacting pedestrian per scene (blue), while providing the ground-truth for the rest (red). At test time, all the pedestrians are forecasted simultaneously.

$$\mathscr{L}(w) = -\sum_{t=T_{obs}+1}^{T_{pred}} \log(\mathbb{P}(\mathbf{v}^t|\mu^t,\sigma^t,\rho^t)). \tag{2.4}$$

While utilizing the NLL loss is not new [16, 18], we posit that incorporating the predicted variance σ^t within the training objective helps to stabilize model training, as the model is not penalized heavily for samples for which it is highly uncertain (high σ^t). Such a loss formulation has also been shown to make the network robust to noisy data [114]. We demonstrate in the experimental section, that the gaussian-based loss formulation performs superior in comparison to its counterpart L_2 loss [17, 25, 76, 101, 115] that does not incorporate the model variance. In the experiments, we interchangeably use the term gaussian loss to refer to NLL loss.

Penalizing only the primary pedestrian: Many trajectories within a real-world dataset comprise static pedestrians standing in a group or pedestrians moving linearly without interacting. Such trajectories can make the model training inefficient as they are easy to forecast [116] and contribute no useful learning signal. Furthermore, when present in significant amounts, such static or linear data can bias the model training and lead to degenerate models.

Therefore, contrary to the standard practice of minimizing the loss objective for all the trajectories within a sample, we propose to minimize the loss for only those pedestrians that undergo interactions within each scene of the training dataset. As shown in Fig. 2.5, during training, the neighbours follow the ground-truth trajectory and only the trajectory of the primary pedestrian is forecasted in a closed-loop. In the next section, we describe how to identify these primary pedestrian within a dataset.

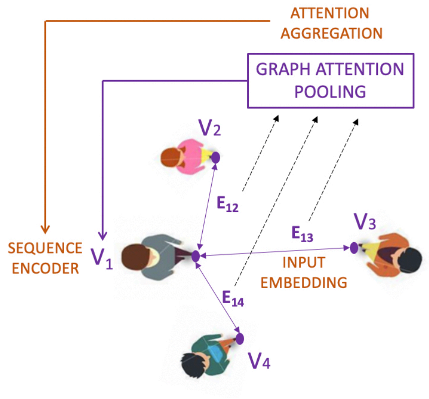


Figure 2.6: Illustration of Graph neural networks (purple) as a special case of our data-driven pipeline (brown). Each vertex V_i is modelled using a sequence encoder, the neighbour edges E_{ij} correspond to our input embeddings which are aggregated via attention mechanism. The resulting interaction vector is provided as input to the sequence encoder (vertex V_i).

During test time, we follow standard protocol and predict the trajectories of all the pedestrians simultaneously. Specifically, till time-step T_{obs}, we provide the ground truth position of all the pedestrians as input to the forecasting model. From time T_{obs+1} to T_{pred}, we use the predicted position (derived from the predicted velocity) of each pedestrian as input to the forecasting model and predict the future trajectories of all the pedestrians.

2.3.5 Equivalence to Graph Neural Networks

Recently, graph neural networks (GNNs) have become popular for forecasting human motion. In the GNN setup, each pedestrian is represented as a node/vertex V_i and two interacting pedestrians are connected via an edge E_{ij}. V_i models the sequence representation of the associated pedestrian and edge E_{ij} updates according to the interactions between the associated pedestrians. We show an equivalence between dynamic-interaction-based GNNs and our proposed LSTM-based pipeline with `S-X-Attn-MLP` (where `X` $\in$ {`MLP`, `LSTM`}) interaction encoding scheme, visually illustrated in Figure 2.6.

Without loss of generality, let pedestrian i be the primary pedestrian. Vertex V_i is modelled using an LSTM sequence encoder. Edge E_{ij} takes as input the state of the neighbours and updates over time using an MLP or an LSTM (*input state embedding*). At each time-step, the information of all connected edges is aggregated using attention mechanism (*aggregation strategy*), popularly referred to as graph attention (GAT) pooling [117] in GNN literature. Finally the aggregated vector is optionally passed through an MLP to obtain the interaction vector p_i which is the input to the LSTM sequence encoder for V_i. Social-BiGAT [25] utilizes the `S-MLP-Attn-MLP` design, Social Attention [18] utilizes the `O-LSTM-Attn-MLP` design while recently, STAR [105] utilizes the `S-MLP-Attn-MLP` design with the sequence encoder for vertex V_i being a Transformer [108].

2.3.6 Explaining trajectory forecasting models

Trajectory forecasting models are deployed in many safety-critical applications like autonomous systems. In such scenarios, it becomes really important to gain insight into the decision-making of the *blackbox* neural networks. Several works in literature attempt to explain the rationale behind the NN decisions [54, 118, 119, 120, 121]. Out of these techniques, Layer-wise Relevance Propagation (LRP) is one of the most prominent methods in explainable machine learning.

As a quick background, LRP re-distributes the model output decision back to each of the input variables indicating the extent to which each input contributes to the output. This is done by reverse-propagating the model prediction through the network by means of heuristic rules that apply to each layer of a neural network [54]. These propagation rules are based on a local conservation principle: the net quantity or relevance, received by any higher layer neuron is redistributed in the same amount to neurons of the layer below. Mathematically, if j and k are indices for neurons in two consecutive layers, and denoting by $R_{j \to k}$ the relevance flowing between two neurons, we have the equations:

$$\Sigma_j R_{j \to k} = R_k \tag{2.5}$$

$$R_j = \Sigma_k R_{j \to k} \tag{2.6}$$

On applying the local conservation principle across all the layers, we obtain global conservation of the output score when reverse propagated back to the inputs. Recently, Arras *et al.* [122] have demonstrated that the principle of LRP can also be applied to LSTMs.

LRP has largely been explored in the domain of model classification *i.e.* the outputs are classification scores. In this chapter, we utilize LRP to determine which neighbours (via the input interaction vector) and past velocities (via the input velocity embedding) of the primary pedestrian our model focuses on, when regressing to the next predicted velocity. We achieve this by reverse-propagating both the x-component v_x as well as y-component v_y of predicted velocity ($\mathbf{v_{pred}} = (v_x, v_y)$) and adding the obtained input relevance scores. To the best of our knowledge, we are the first work to empirically demonstrate that LRP provides reasonable explanations when extended to the regression task of trajectory forecasting. Moreover, the LRP technique is generic and can be applied on top of any trajectory forecasting network to analyze its predictions.

2.4 TrajNet++: A Trajectory Forecasting Benchmark

In this section, we present *TrajNet++*, our interaction-centric human trajectory forecasting benchmark. To demonstrate the efficacy of a trajectory forecasting model, the standard practice is to evaluate these models against baselines on a standard benchmark. However, current methods have been evaluated on different subsets of available data without proper sampling of scenes in which social interactions occur. In other words, a data-driven method cannot learn to model agent-agent interactions if the benchmark comprises primarily of scenes where the agents are

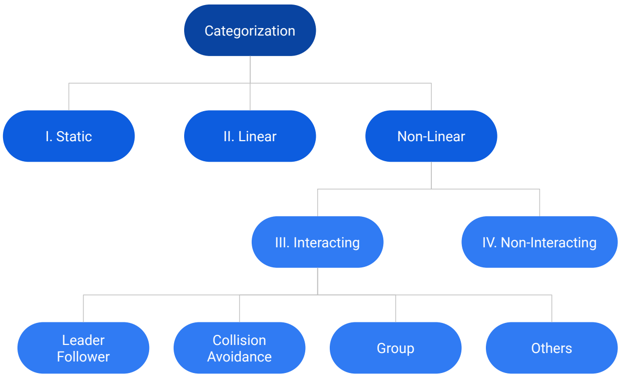
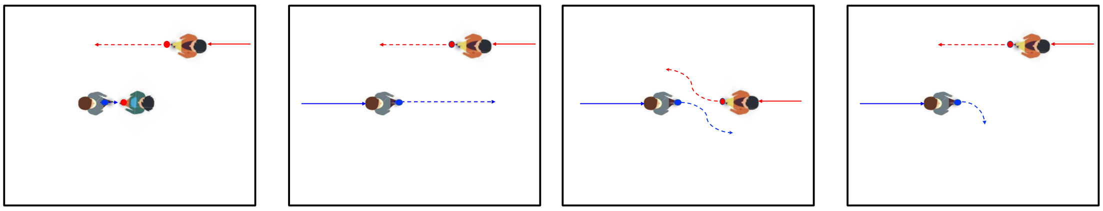


Figure 2.7: Our proposed hierarchy for trajectory categorization. Using our defined trajectory categorization, we construct the TrajNet++ benchmark by sampling trajectories corresponding largely to 'Type III: Interacting' category.



(a) Static (b) Linear (c) Interacting (d) Non-Interacting

Figure 2.8: Visualization of our high-level defined trajectory categories.

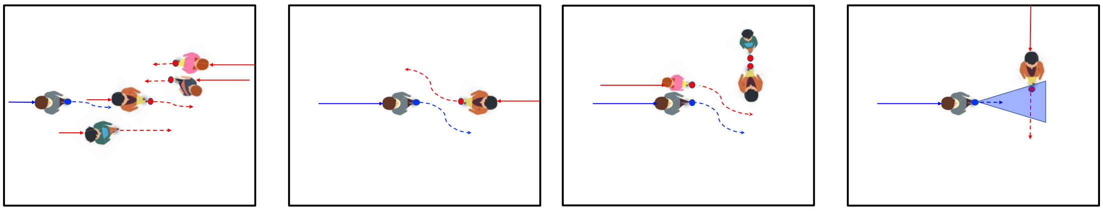
static or move linearly. Therefore, our benchmark comprises largely of scenes where social interactions occur. To this extent, we propose the following trajectory categorization hierarchy.

2.4.1 Trajectory categorization

We provide a detailed trajectory categorization (Fig 2.7). This detailed categorization helps us not only to better sample trajectories for TrajNet++ dataset but also glean insights into the model performance in diverse scenarios, *i.e.*, to verify whether the model captures all the different kinds of interactions.

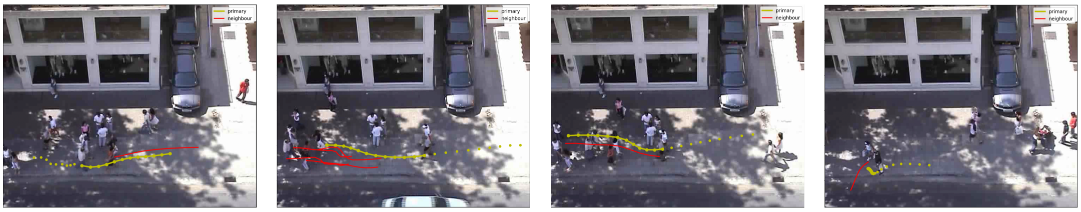
Every TrajNet++ scene sample is identified with respect to the pedestrian of interest, called the primary pedestrian. In particular, if there are two pedestrians of interest present within one real-world scene, it results in two TrajNet++ scene samples, one corresponding to each primary pedestrian. Subsequently, each scene is categorized with respect to its primary pedestrian. We now explain in detail our proposed hierarchy for trajectory categorization. We also provide example scenarios for the same in Fig 2.8:

1. **Static (Type I)**: If the euclidean displacement of the primary pedestrian in the scene is less than a specific threshold.



(a) Leader Follower (b) Collision avoidance (c) Group (d) Others

Figure 2.9: Visualization of our Type III interactions commonly occurring in real world crowds.



(a) Leader Follower (b) Collision Avoidance (c) Group (d) Others

Figure 2.10: Sample scenes from our benchmark. In each of the samples, we illustrate a different social interaction between the primary pedestrian (yellow) and the corresponding interacting neighbours (red) in real world datasets.

2. **Linear (Type II)**: If the trajectory of the primary pedestrian can be *correctly forecasted* with the help of an Extended Kalman Filter (EKF). A trajectory is said to be *correctly forecasted* by EKF if the final displacement error between the ground truth trajectory and forecasted trajectory is less than a specific threshold.

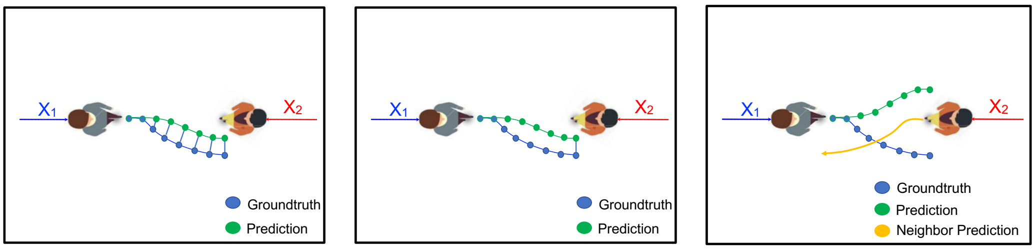
The rest of the scenes are classified as ‘Non-Linear’. We further divide non-linear scenes into Interacting (Type III) and Non-Interacting (Type IV).

3. **Interacting (Type III)**: These correspond to scenes where the primary pedestrian undergoes social interactions. For a detailed categorization coherent with commonly observed social interactions, we divide interacting trajectories into the following sub-categories (see Fig 2.9).

(a) **Leader Follower [LF] (Type IIIa)**: Leader follower phenomenon refers to the tendency to follow pedestrians going in relatively the same direction. The follower tends to regulate his/her speed and direction according to the leader. If the primary pedestrian is a follower, we categorize the scene as Leader Follower.

(b) **Collision Avoidance [CA] (Type IIIb)**: Collision avoidance phenomenon refers to the tendency to avoid pedestrians coming from the opposite direction. We categorize the scene as Collision avoidance if the primary pedestrian is involved in collision avoidance.

(c) **Group (Type IIIc)**: The primary pedestrian is said to be a part of a group if he/she maintains a close and roughly constant distance with at least one neighbour on his/her side during the entire scene.



(a) ADE (b) FDE (c) Physically feasible, high FDE

Figure 2.11: Distance-based unimodal evaluation metrics commonly used in literature: Average Displacement Error (ADE) and Final Displacement Error (FDE).

(d) **Other Interactions (Type IIId)**: These are scenes where the primary pedestrian undergoes social interactions other than LF, CA and Group. We define *social interaction* as follows: We look at the angular region in front of the primary pedestrian. If any neighbouring pedestrian is present in the defined region at any time-instant during prediction, the scene is classified as having the presence of social interactions.

4. **Non-Interacting (Type IV)**: If a trajectory of the primary pedestrian is non-linear and undergoes no social interactions during prediction, the scene is categorized as non-interacting.

Using our defined trajectory categorization, we construct the TrajNet++ benchmark by sampling trajectories corresponding mainly to the Type III category. Moreover, having many Type-I scenes in a dataset (scenes where the primary pedestrian is static) can hamper the training of the model as well as result in misleading evaluation. Therefore, we remove such samples in the construction of our benchmark. The details of the categorization thresholds as well as the datasets which comprise our TrajNet++ benchmark are provided in the supplementary material. A few examples of our categorization in the real world are displayed in Fig 2.10. In addition to a well-sampled dataset, TrajNet++ provides an extensive evaluation system to understand model performance better.

2.4.2 Evaluation metrics

Unimodal Evaluation: Unimodal evaluation refers to the evaluation of models that propose a single future mode for a given past observation. The most commonly used metrics of human trajectory forecasting in the unimodal setting are Average Displacement Error (ADE) and Final Displacement Error (FDE) defined as follows:

1. **Average Displacement Error (ADE)**: Average L_2 distance between ground truth and model prediction over all predicted time steps.
2. **Final Displacement Error (FDE)**: The L_2 distance between the predicted final destination and the ground truth final destination at the end of the prediction period T_{pred}.

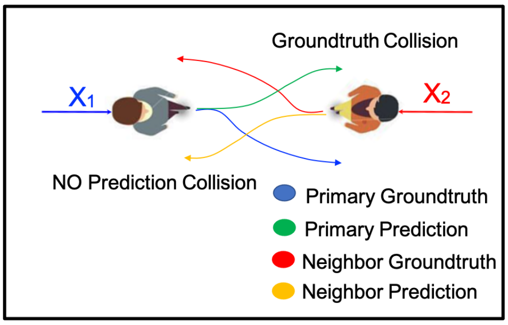


Figure 2.12: Visual illustration of our proposed collision metrics. In the example, the model prediction exhibits ground-truth collision (Col-II = 1) but no prediction collision (Col-I = 0).

$$\text{ADE}\ (\widehat{\mathbf{Y}}_{\mathbf{i}}, \mathbf{Y}_{\mathbf{i}}) = \frac{1}{(T_{pred} - T_{obs})} \sum_{t=T_{obs}+1}^{T_{pred}} ||\widehat{\mathbf{x}}_{\mathbf{i}}^{\mathbf{t}} - \mathbf{x}_{\mathbf{i}}^{\mathbf{t}}||_2^2, \tag{2.7}$$

$$\text{FDE}\ (\widehat{\mathbf{Y}}_{\mathbf{i}}, \mathbf{Y}_{\mathbf{i}}) = ||\widehat{\mathbf{x}}_{\mathbf{i}}^{\mathbf{T_{pred}}} - \mathbf{x}_{\mathbf{i}}^{\mathbf{T_{pred}}}||_2^2. \tag{2.8}$$

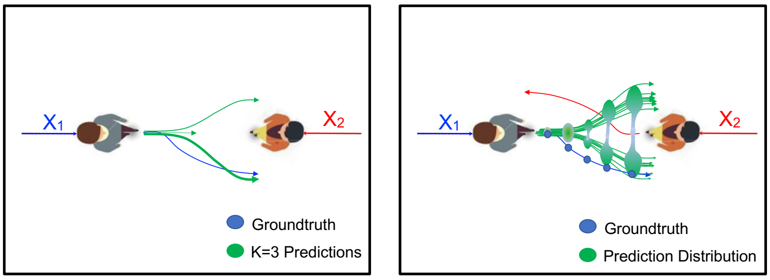
These metrics essentially define different distance measures between the forecasted trajectory and the ground truth trajectory. With respect to our task, one of the most important aspects of human behavior in crowded spaces is collision avoidance. To ensure that models forecast collision-free trajectories, we propose two new collision-based metrics in our framework (see Fig 2.12):

3. **Collision I - Prediction collision (Col-I)**: This metric calculates the percentage of collision between the primary pedestrian and the neighbors in the *predicted future* scene. This metric indicates whether the predicted model trajectories collide, *i.e.*, whether the model learns the notion of collision avoidance.
4. **Collision II - Groundtruth collision (Col-II)**: This metric calculates the percentage of collision between the primary pedestrian's prediction and the neighbors in the *groundtruth future* scene.

$$\text{Col-I}\ (\widehat{\mathbf{Y}}) = \min\left(1, \sum_{t=1}^{T_{pred}} \sum_{j=2}^{N} \mathbb{1}[||\widehat{\mathbf{x}}_{\mathbf{1}}^{\mathbf{t}} - \widehat{\mathbf{x}}_{\mathbf{j}}^{\mathbf{t}}||_2^2 \leq \delta]\right), \tag{2.9}$$

$$\text{Col-II}(\widehat{\mathbf{Y}}, \mathbf{Y}) = \min\left(1, \sum_{t=1}^{T_{pred}} \sum_{j=2}^{N} \mathbb{1}[||\widehat{\mathbf{x}}_{\mathbf{1}}^{\mathbf{t}} - \mathbf{x}_{\mathbf{j}}^{\mathbf{t}}||_2^2 \leq \delta]\right). \tag{2.10}$$

We want to stress further the importance of the collision metrics in the unimodal setup. As mentioned earlier, human motion is multimodal. A model may forecast a physically-feasible future, which is different from the actual ground truth. Such a physically-feasible prediction can



(a) Top-K (b) Average NLL [87]

result in a large ADE/FDE, which can be misleading. Our Col-I metric can help overcome this limitation of ADE/FDE metrics and provides a solution to measure the physical feasibility of a prediction (aversion to a collision in this case). Col-II metric indicates whether the model understood the intention of the neighbours and predicted the desired trajectory mode indicated by fewer collisions with neighbours in ground truth. We believe our proposed collision metrics are an important step towards capturing the understanding of the model of human social etiquette in crowds.

Multimodal Evaluation: For models performing multimodal forecasting, *i.e.*, outputting a future trajectory distribution, we provide the following metrics to measure their performance:

5. **Top-k ADE**: Given k output predictions for an observed scene, this metric calculate the ADE of the primary prediction *closest* to the groundtruth trajectory, similar in spirit to Variety Loss proposed in [17].
6. **Top-k FDE**: Given k output predictions for an observed scene, this metric calculate the FDE of the primary prediction *closest* to the groundtruth trajectory, similar in spirit to Variety Loss proposed in [17].

For the Top-k metrics, we propose k be small (3 as opposed to 20) as a model outputting uniformly-spaced predictions, irrespective of the input observation, can result in a much lower Top-20 ADE/FDE (explained in detail in the next chapter).

7. **Average NLL** [87]: At each time-step, we obtain a Kernel Density Estimate (KDE) [123] of the predicted distribution. From these estimates, the log-likelihood of ground truth trajectory is computed at each time step and is subsequently averaged over the prediction horizon. This metric provides a good indication of the probability of the ground truth trajectory in the model prediction distribution.

2.4.3 Forecasting benchmark: a missing piece

We would like to further stress on the relevance of TrajNet++ from a benchmarking perspective. The train-test split for trajectory forecasting is *not* explicitly fixed in literature, and the raw datasets need to be processed to generate the samples for training and testing. During this preprocessing, one has to be careful about information leakage: the information of the future (ground-truth) should not creep into the past observation. This can lead to significant differences in results [87, 124].

Furthermore, the test set across all baselines needs to be identical. Some methods do not model incomplete trajectories while others do not model scenes with single pedestrians. Therefore, such trajectories and scenes are often filtered out leading to different test sets and theoretically, an unobjective evaluation. Similarly, the definition of metrics also needs to be consistent across baselines. For instance, for a given test scene, some works [106] utilize the same prediction (out of K) to calculate the Top-K ADE and Top-K FDE, while others [17, 102, 124] utilize different predictions (out of K) to calculate the two metrics, optimizing both independently.

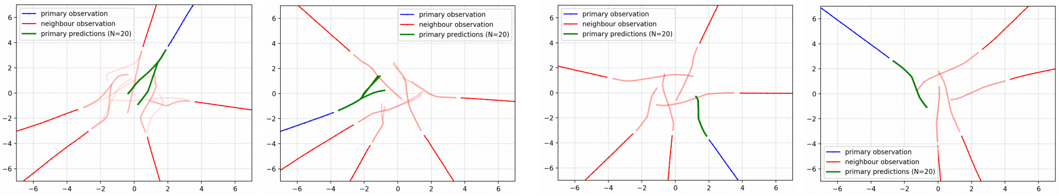
Thus, it is important to measure model performances on a forecasting benchmark. We encourage researchers to submit their models to TrajNet++, so that the quality of trajectory forecasting models can keep increasing in tackling more challenging scenarios.

2.4.4 TrajNet++ datasets

We now describe the datasets used in the TrajNet++ benchmark. Since the focus of this chapter is to tackle agent-agent interactions in crowded settings, we explicitly select datasets where scene constraints do not play a significant role in determining the future trajectory. For each real world dataset, we utilize only the information regarding the pedestrian locations from the respective annotations files, *i.e.*, spatial coordinates of each pedestrian in each time frame. Furthermore, we provide no information regarding the destination of each pedestrian or structure of the scene. Our goal is to forecast only the 2D spatial coordinates for each pedestrian.

TrajNet++ Real Dataset

- **ETH:** ETH dataset provides for two locations: Univ and Hotel, where pedestrian trajectories are observed. This dataset contains a total of approximately 750 pedestrians exhibiting complex interactions (Pellegrini *et. al.* [125]). The dataset is one of the widely used benchmarks for pedestrian trajectory forecasting. It captures diverse real-world social interactions like leader follower, collision avoidance, and group forming and dispersing.
- **UCY:** UCY dataset consists of three scenes: Zara01, Zara02 and Uni, with a total of approximately 780 pedestrians (Lerner *et. al.* [41]). This dataset, in addition to the ETH dataset, is widely used as benchmarks for pedestrian trajectory forecasting, offering a wide range of non-linear trajectories arising out of social interactions.
- **WildTrack:** This is a recently proposed benchmark [126] for pedestrian detection and tracking captured in front of ETH Zurich. Since the dataset comprises of diverse crowd interactions in the wild, we utilize it for our task of trajectory forecasting.
- **L-CAS:** This is a recently proposed benchmark for pedestrian trajectory forecasting (Sun *et. al.* [127]). The dataset, comprising over 900 pedestrian tracks, comprises diverse



(a) Sensitive Scenes (b) Insensitive Scenes

Figure 2.14: Illustration of our filtering procedure to generate Trajnet++ synthetic dataset. Given ground-truth observations (primary pedestrian in blue, neighbours in red), we perturb the positions of all agents at the end of the observation period with small noise and forecast the future with ORCA multiple times, to obtain a distribution (in green). This procedure helps us identify the *sensitive scenes* and consequently remove them.

social interactions that are captured within indoor environments. Some of the challenges scenarios in this dataset include people pushing trolleys and running children.

- **CFF:** This is a large-scale dataset of 42 million trajectories extracted from real-world train stations [65]. It is one of the biggest datasets that capture agent-agent interactions in crowded settings during peak travel times. Due to the high density of people, we observe higher instances of social interactions like leader-follower in this dataset.

TrajNet++ Synthetic Dataset Interaction-centric synthetic datasets can provide the necessary controlled environment to compare the performances of different model components. We provide synthetic data in TrajNet++ to evaluate the performance of a model under controlled interaction scenarios. It is important to understand what works and what does not in clean simpler scenarios.

Simulator Selection: It is a necessary condition that the interactions in the synthetic dataset are similar to those in the real world. Empirically, we find that in comparison to Social Force [28], ORCA [60] provides a better similarity to real world human motion with respect to avoiding collisions. We select the ORCA parameters which minimize the average displacement error (ADE) on the Type-III trajectories of TrajNet++ real-world training set.

Dataset Generation: Given the ORCA parameters, we generated the synthetic dataset using the following procedure: n pedestrians were initialized at random on a circle of radius r keeping a certain minimum distance d_min between their initial positions. The goal of each pedestrian was defined to be the point diametrically opposite to the initial position on the circle. For the TrajNet++ synthetic dataset: We ran different simulations with n chosen randomly from the range [4, 7) on a circle of radius $r = 10$ meters and $d\_min = 2$ meters.

Given the generated trajectories, we selected only those scenes which belonged to the Type III: ‘Interacting’ category. The ORCA simulator demonstrates sensitive dependence on initial conditions. This can be attributed to the fact that all the agents are expected to collide near the same point (at the origin), so slight perturbations can greatly affect the future trajectory of all

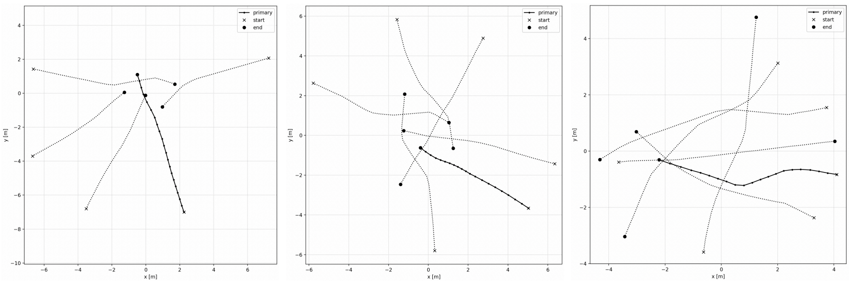


Figure 2.15: Examples of samples in the TrajNet++ synthetic dataset generated using ORCA.

agents. Sensitivity to initial conditions, also known as the *Butterfly Effect*, is a well-studied phenomenon of *Chaos theory*[128]. To identify such sensitive initial conditions, the practice which is often followed is to perturb the initial conditions with arbitrary small noise and observe the effect. Along similar lines, we propose an additional step to filter out such *sensitive* scenes: in each scene, we perturb all trajectories at the point of observation with a small uniform noise ($\text{noise} \in U[-\text{noise\_thresh}, \text{noise\_thresh}]$), and forecast the future trajectories using ORCA. We perform this procedure N times. If any of the N ORCA predictions has a significant ADE compared to the ground truth, we filter out such scenes. Fig 2.14 visualizes the sample outputs of our filtering process (with $\text{noise\_thresh} = 0.01$, $N = 20$). We pass the selected scenes through a final additional filter that identifies sharp unrealistic turns in trajectories. Fig 2.15 illustrates a few sample scenes of our TrajNet++ synthetic dataset.

TrajNet++ categorization rules. We provide the mathematical conditions to be satisfied to classify a scene into a particular category in TrajNet++.

- **Static (Type I)**: The total distance traveled by the primary pedestrian is *less than 1 meter*.
- **Linear (Type II)**: A trajectory is categorized to be *Linear* if the final displacement error (FDE) of the *extended kalman filter prediction is less than 0.5m.*
- **Leader Follower (Type IIIa)**: The primary pedestrian follows a neighbour, moving in the same direction within an angular range of $\pm 15\deg$ and having relative velocity in the range of $\pm 15\deg$, for more than 2 seconds.
- **Collision Avoidance (Type IIIb)**: The primary pedestrian is faced head-on by a neighbour, coming from the opposite direction within an angular range of $\pm 15\deg$ and relative velocity of movement in the range $[180 \pm 15\deg]$ at any given point of time.
- **Group (Type IIIc)**: There exists a neighbour on the side of the primary pedestrian (angular range $[90 \pm 15\deg]$ or $[-90 \pm 15\deg]$) for the entire duration of the sample with mean distance $\leq 1m$ and standard deviation $\leq 0.2m$.
- **Other Interactions (Type IIId)**: If at any point of time, a neighbour exists in the front (angular range of $\pm 15\deg$) of the primary pedestrian within a distance of 5 m, we categorize this trajectory of undergoing social interactions.
- **Non-Interacting (Type IV)**: All the other trajectories that fail to satisfy any of the above conditions.

2.5 Experiments

In this section, we perform extensive experimentation on both TrajNet++ synthetic and real-world datasets to understand the efficacy of various interaction module designs for human trajectory forecasting. Moreover, we demonstrate how our proposed metrics help to provide a more meaningful picture of model performance.

2.5.1 Implementation details

The velocity of each pedestrian is embedded into a 64-dimensional vector. The dimension of the interaction vector is 256. The dimension of the goal direction vector is 64. For grid-based interaction encoding, we construct a grid of size 16×16 with a resolution of 0.6 meters. The dimension of the hidden state of both the encoder LSTM and decoder LSTM is 128. As mentioned earlier, each pedestrian has its own encoder and decoder. The batch size is fixed to 8. We train using ADAM optimizer [129] with a learning rate of 1e-3. We perform interaction encoding at every time-step. For the concatenation based models, we consider top-4 nearest neighbours based on euclidean distance unless stated otherwise. For the attention aggregation strategy, we utilize the self-attention mechanism proposed in [108].

Data augmentation is another technique that can help increase accuracy, which can get wrongly attributed to the interaction encoder. We use rotation augmentation as the data augmentation technique to regularize all the models.

2.5.2 Interaction modules: synthetic experiments

We utilize synthetic datasets to validate the efficacy of various interaction modules in a controlled setup. Since ORCA [60], our underlying simulator for synthetic data generation, has access to the goals of each pedestrian, we embed the direction to the goal using an MLP, and concatenate it to the velocity embedding (in Eq 2.1).

Table 2.2 quantifies the performance of the different designs of interaction modules published in the literature on TrajNet++ synthetic dataset. Observing only the distance-based metrics, one might wrongly conclude that all interaction modules perform similarly. However, ADE/FDE metrics do not indicate the ability of the model to learn social etiquette of collision avoidance. In safety-critical scenarios, it is more important for a model to prevent collisions in comparison to minimizing ADE/FDE.

Grid-Based Modules

Our proposed `D-Grid` outperforms `O-Grid`, especially in terms of Col-I, *i.e.*, `D-Grid` learns better to avoid collisions. Even though the motion encoder (LSTM) has the potential to infer

Model (Acronym)	ADE/FDE	Col-I	Col-II
Grid based methods			
Vanilla	0.32/0.62	19.0	7.1
O-Grid [16]	0.27/0.52	11.7	4.9
S-Grid [16]	0.24/0.50	**2.2**	**4.6**
D-Grid [**Ours**]	0.24/0.49	**2.2**	4.8
Non-Grid based methods			
S-MLP-MaxP-MLP [17]	0.27/0.52	6.4	5.2
S-MLP-Attn-MLP [25]	0.26/0.52	3.7	5.4
D-MLP-SumP-LSTM [87]	0.29/0.57	13.8	6.6
O-LSTM-Attn-MLP [18]	0.24/0.48	0.8	5.2
D-MLP-MaxP-MLP	0.28/0.55	14.3	6.1
D-MLP-Attn-MLP	0.27/0.52	8.1	**5.0**
D-MLP-ConC-MLP	0.25/0.50	1.3	5.6
D-MLP-ConC-LSTM [**Ours**]	0.24/0.48	**0.6**	5.3

Table 2.2: Unimodal Comparison of interaction encoder designs when forecasting 12 future time-steps, given the previous 9 time-steps, on TrajNet++ synthetic dataset. Errors reported are ADE / FDE in meters, Col-I / Col-II reported in %.

the relative velocity of neighbours over time, there is a significant difference in performance when we explicitly provide relative velocity of the neighbours as input. Further, since ORCA is a first-order trajectory simulator dependent only on relative configuration of neighbours, the performance of `D-Grid` is at par with `S-Grid` in the controlled setup.

Non-Grid-Based Modules

First, we focus on the information aggregation strategies for non-grid based encoders. It is evident that max-pooling strategy performs the worst as it fails to preserve the identity of the surrounding neighbours. Indeed, it has been shown that max-pooling operation is leaky in information [130]. Furthermore, the simple concatenation baseline performs at par with attention-based aggregation, implying that further research needs to be conducted to design more effective attention-based strategies.

Among the non-grid LSTM-based designs, encoding the interaction information using LSTM [`O-LSTM-Att-MLP`, `D-MLP-Conc-LSTM`], improves performance over its MLP-based counterparts. The drop in performance of `D-MLP-SumPool-LSTM` module [87] can be attributed largely to (1) sum pooling which provides equal weights to all neighbours and (2) the state encoding scheme that considers absolute neighbour coordinates instead of relative coordinates: relational coordinates of agents to the target agent are easier to train than exact coordinates of agents [17]. MLP encoders, due to their non-recurrent nature, have no information regarding the interaction representation at the previous step. We argue that LSTMs can capture the evolution of interaction and therefore provide a better social representation as the scene evolves.

Model (Acronym)	ADE/FDE	Col-I	Col-II
Hand-crafted methods			
Kalman Filter	0.87/1.69	16.20	22.1
Constant Velocity	0.68/1.42	14.30	15.2
Social Force	0.89/1.53	**0.0**	**13.1**
ORCA	0.68/1.40	**0.0**	15.0
Top submitted methods*[1]			
AMENet [131]	0.62/1.30	14.1	16.90
AIN [132]	0.62/1.24	10.7	17.10
PecNet [106]	0.57/1.18	15.0	14.3
Grid based methods			
Vanilla	0.6/1.3	13.6 (0.2)	14.8 (0.1)
O-Grid [16]	0.58/1.24	9.1 (0.4)	15.1 (0.3)
S-Grid [16]	0.53/1.14	6.7 (0.2)	13.5 (0.5)
D-Grid [**Ours**]	0.56/1.22	**5.4 (0.3)**	**13.0 (0.5)**
Non-Grid based methods			
S-MLP-MaxP-MLP [17]	0.57/1.24	12.6 (0.9)	14.6 (0.7)
S-MLP-Att-MLP [25]	0.56/1.22	7.2 (0.8)	14.8 (0.4)
D-MLP-SumP-LSTM [87]	0.60/1.28	13.9 (0.7)	15.4 (0.5)
O-MLP-Att-LSTM [18]	0.56/1.21	9.0 (0.3)	15.2 (0.4)
D-MLP-ConC-MLP	0.58/1.23	7.6 (0.6)	14.3 (0.2)
D-MLP-MaxP-MLP	0.60/1.25	12.9 (0.6)	14.8 (0.5)
D-MLP-Attn-MLP	0.56/1.22	6.9 (0.3)	14.3 (0.6)
D-MLP-ConC-LSTM (k=8)	0.56/1.22	8.5 (0.5)	**14.0 (0.1)**
D-MLP-ConC-LSTM [**Ours**]	0.55/1.19	**6.8 (0.4)**	14.5 (0.5)

Table 2.3: Unimodal Comparison of interaction encoder designs when forecasting 12 future time-steps, given the previous 9 time-steps, on Type III *interacting* trajectories of TrajNet++ real world dataset. Errors reported are ADE / FDE in meters, Col-I / Col-II in mean % (std. dev. %) across 5 independent runs.

2.5.3 Interaction modules: real world experiments

Now, we discuss the performances of forecasting models on TrajNet++ real-world data. With the help of our defined trajectory categorization, we construct the TrajNet++ real-world benchmark by sampling trajectories corresponding mainly to Type III *interacting* category. Having gained insights on the performance of different modules on controlled synthetic data, we explore the question, ‘Do these findings generalize to the noisy real world datasets comprising much more diverse and subtle interactions?’

Table 2.3 provides an extensive evaluation of existing baselines on the Type III *interacting* trajectories of the TrajNet++ real dataset. Similar to the synthetic setup, we observe that Col-I metric is a critical differentiating factor for various module designs when compared on *identical grounds*. We hope that in future, researchers will incorporate the collision metrics while reporting their model performances on trajectory forecasting datasets. Another surprising observation is that the performance of ADE/FDE is similar (including submitted methods) indicating that there exists a lot of scope to improve the performance of current trajectory forecasting models on a well-sampled interaction-centric scenarios.

	Vanilla	O-Grid	S-Grid	D-Grid
Time	0.01	0.022	0.081	0.022
Speed-Up	8.1x	3.7x	1x	**3.7x**

Table 2.4: Speed (in seconds) comparison of grid-based modules at test time. D-Grid provides 3.7x speedup as compared to S-Grid rendering it more suitable for real-world deployment tasks.

Classical Methods

We first compare with the classical trajectory forecasting models, namely, Extended Kalman Filter (EKF), Constant Velocity (CV) [133], Social Force [28], and ORCA [60]. The high error of EKF and CV can be attributed to the fact that these methods do not model social interactions. Both Social Force and ORCA models forecast the future trajectory based on the assumption that each pedestrian has an intended direction of motion (driven by the goal) and a preferred velocity. We interpolate the observed trajectory to identify the *virtual goals* for each agent. `Social Force` and `ORCA` are calibrated to fit the TrajNet++ training data by minimizing ADE/FDE metrics. The interaction-based data-driven models outperform the handcrafted models in terms of the distance-based metrics, as neural networks have the ability to learn the subtle and diverse social interactions.

Grid-based modules

Our proposed `D-Grid` performs superior to `O-Grid` in the real world as well. It is interesting to compare the performances of `D-Grid` and `S-Grid`. The current design of `S-Grid` fails to learn the notion of avoiding collisions. This corroborates the fact that while training to minimize ADE/FDE, the abstract hidden-state of LSTM is unable to learn a representation necessary to avoid collisions. In the `D-Grid` design, the model explicitly focuses on relative velocities. The simplicity of our design slightly hampers the distance-based accuracy as we limit the expressibility of the model. However, it results in safer predictions as the task of the model to learn social concepts is made easier thanks to our domain-knowledge based design. Further, as shown in Table 2.4, the `D-Grid` provides significant computational speed-up in comparison to `S-Grid` rendering it useful for real-time deployment.

Aggregation strategy for non-grid modules

We evaluate the performance of various aggregation strategies [`D-MLP-Attn-MLP`, `D-MLP-\MaxP-MLP`, `D-MLP-ConC-MLP`] on real-world data keeping all the other factors constant. We observe that the max-pooling strategy continues to perform bad due to its leaky design to hard-merge the embeddings of various neighbours [130]. The concatenation strategy performs slightly worse in comparison to its sophisticated attention-based counterpart. We believe that the concatenation baseline is a simple yet powerful baseline to compare with, when designing future information aggregating modules.

Interaction Modelling	Objective	Penalize	ADE	FDE	Col-I
None	Gaussian	Primary	0.61	1.31	13.6
Once	Gaussian	Primary	0.59	1.27	13.4
Every Step	L2	Primary	0.64	1.39	6.4
Every Step	Gaussian	All	0.59	1.29	8.2
Every Step	Gaussian	Primary	**0.56**	**1.22**	**5.4**

Table 2.5: Our recipe to train socially-aware forecasting models that reduces the collision as well as distance-based metrics. We propose to (1) perform interaction modelling at all times, (2) incorporate model uncertainty in loss objective and (3) penalize only the primary pedestrian during training. Errors reported are ADE / FDE in meters, Col-I in %.

LSTM-based non-grid models

Among the LSTM-based non-grid designs, `D-MLP-SumPool-LSTM` module [87] demonstrates high Col-I metric due to (1) sum pooling strategy and (2) encoding of absolute neighbour coordinates. The Col-I metric for `O-LSTM-Att-MLP` [18] is relatively higher compared to `D-MLP-Concat-LSTM` in the real-world due to the absence of relative velocity as input to the interaction model. One can notice the importance of having an LSTM-based embedding in our proposed `DirectConcat` model by comparing the performance between `D-MLP-Concat-LSTM` and `D-MLP-Concat-MLP`. The LSTM-based interaction encoder helps to model higher-order spatio-temporal interactions better and is more robust to noise in the real-world measurements as LSTM controls the evolution of the interaction vector. The top-k neighbours are chosen based on euclidean distance. We believe that including domain knowledge by considering nearest neighbours is one of the reasons for improvement in Col-I metric as compared to its attention-based and max-pooling-based counterparts. We corroborate this by observing that considering a large number of neighbours ($k = 8$), in comparison to ($k = 4$), results in an increase in the Col-I metric.

2.5.4 Socially-aware training recipe

We empirically demonstrate the different aspects of our proposed training strategies on the real world data, namely (1) penalizing only the primary pedestrian during training, (2) utilizing learned loss attenuation in the form of gaussian loss, (3) performing spatio-temporal interaction modelling. Table 2.5 illustrates the performance of `D-Grid` module under various training schemes.

Penalty. In TrajNet++, the primary trajectories are largely interacting thanks to our defined categorization; however, there exist significant portion of trajectories among the neighbours which are static and linear. Penalizing such neighbouring trajectories during training might bias the network into learning linear behavior. Therefore, we employ a modified training objective where we penalize only the primary pedestrian and utilize ground-truth of neighbours. During test time, we do *not* provide the ground truth neighbour trajectories. Our scheme reduces the collisions by 30% (relative) while also improving the distance-based metrics.

Loss Objective. In forecasting, the loss objective also plays a significant role in helping the model learn better. Predicting the variance in addition to the next step velocity, allows the model to reduce the penalty it incurs on samples where it is highly uncertain. This results in a more stable training. As seen in Table 2.5, the gaussian loss objective improves the distance-based metrics by $\sim 15\%$ in comparison to the L2 loss counterpart (which does not consider the variance).

Interaction Modelling. In recent times, works in trajectory forecasting propose to perform interaction modelling only once at the end of the observation period [17, 25]. It greatly impacts the ability of the model to learn collision avoidance (2.5x more collision) and distance-based metrics as well [17]. Therefore, it is important to monitor the performance on the collision metric while designing interaction modelling strategies.

2.5.5 Explaining forecasting model decisions

Using the popular technique of Layer-wise Relevance Propagation (LRP), we investigate how various input factors affect the decision-making of the forecasting model at each time-step. This helps to verify whether the neural network decision-making process follows human intuition. Fig 2.16 illustrates the score of each neighbour obtained on applying the LRP procedure on our proposed `D-Grid` module and baseline `S-Grid` in real-world scenarios.

In Scene 1, we demonstrate the application of LRP on a simple real-world example. In case of `D-Grid`, the primary pedestrian starts focusing on the potential collider $N2$ despite it being distant compared to $N1$ thereby preventing collision by staying closer to $N1$. On the other hand, `S-Grid` keeps focusing on the $N1$ which is not desirable. It is interesting to note that once $N2$ passes the primary pedestrian, both `D-Grid` and `S-Grid` shift the attention of the primary pedestrian back to $N1$.

In Scene 2, we demonstrate the effectiveness of our proposed `D-Grid` module in a complex real-world scenario. For `D-Grid`, initially the primary pedestrian focuses on $N3$ to prevent collision. On successfully avoiding collision with $N3$, `D-Grid` immediately shifts the focus to the pair $N1$ and $N2$ as they would potentially lead to a collision. On coming in close proximity to $N1$ and $N2$, the focus significantly shifts towards $N1$ as it is closer to the primary pedestrian. Finally, on passing $N1$ and $N2$, the primary pedestrian attends to the pedestrian $N4$ in front. On the other hand, `S-Grid` passes in between $N1$ and $N2$, such behavior is not expected in human crowds.

Thus, we can see that LRP is an effective investigative tool to understand the rationale behind the NN decisions. We can observe that, along with having a lower Col-I metric as compared to

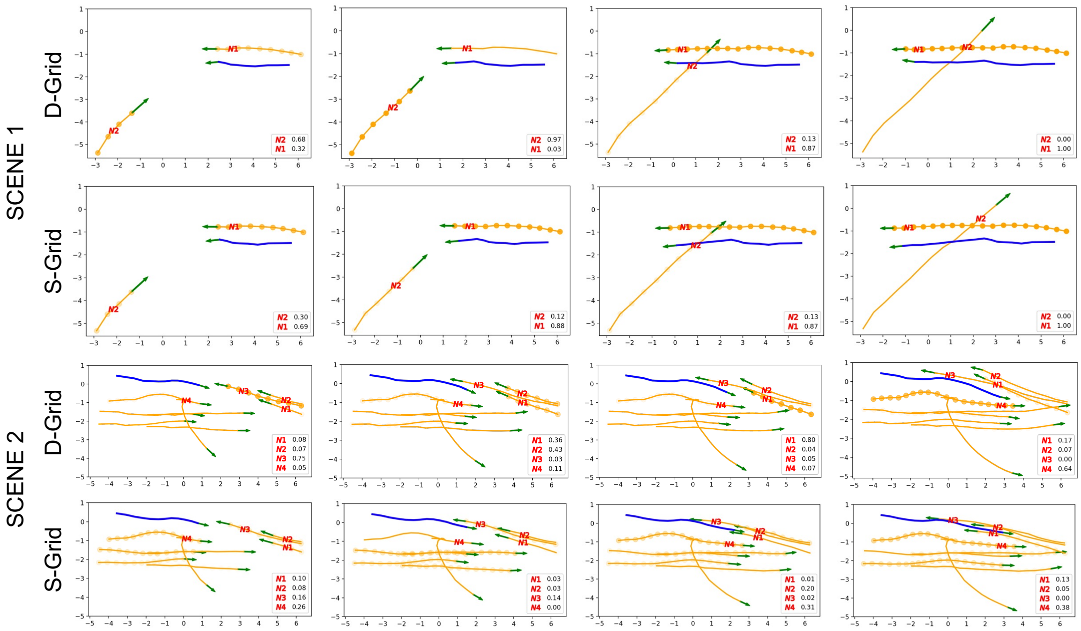


Figure 2.16: Visualizing the decision-making of grid-based interaction modules using layer-wise relevance propagation. The darker the yellow circles, the more is the weight (also shown in the legend) provided by the primary pedestrian (blue) to the corresponding neighbour (yellow). Our proposed `D-Grid`, driven by domain knowledge, outputs socially acceptable trajectories with more intuitive focus on surrounding neighbours as compared to `S-Grid`.

`S-Grid` in Table 2.3, the decision making of our domain-knowledge based `D-Grid` satisfies human intuition while navigating crowds. The LRP technique is generic and can be applied on top of any existing trained interaction module architecture.

Summary. Despite claims in literature that specific interaction modules better model interactions, we observe that under *identical* conditions, all modules perform similar in terms of the distance-based ADE and FDE metrics. The incorporation of Col-I metrics paints a more meaningful picture of model performance. Secondly, relative velocity plays a crucial role in learning collision avoidance in the real-world. Thirdly, a simple concatenation strategy performs at par with the sophisticated attention-based counterparts. We believe that the concatenation baseline should be a standard baseline to compare to when designing future information aggregating modules.

Finally, the LRP technique is a useful investigative tool to gain insights regarding the decision-making process of neural networks. We hope that such practices will help to accelerate the development of interaction modules in future research. There certainly exists room for improvement, and we hope that our benchmark provides the necessary resources to advance the field of trajectory forecasting. We open-source our code for reproducibility.

2.6 Conclusion

In this chapter, we tackled the challenge of modelling social interactions between pedestrians in crowds. While modelling social interactions is a central issue in human trajectory forecasting, the literature lacks a definitive comparison between the many existing interaction model designs on identical grounds. We presented an in-depth analysis of the design of interaction modules proposed in the literature and proposed an optimal domain-knowledge inspired interaction module.

A significant yet missing component in this field is an objective and informative evaluation of these interaction-based methods. To solve this issue, we propose *TrajNet++*: (1) TrajNet++ is interaction-centric as it largely comprises scenes where interactions take place thanks to our defined trajectory categorization, both in the real-world and synthetic settings, (2) TrajNet++ provides an extensive evaluation system that includes novel collision-based metrics that can help measure the *physical feasibility* of model predictions. The superior quality of TrajNet++ is highlighted by the improved performance of interaction-based models on real world datasets on all metrics (4 of the top 5 methods on TrajNet [134], an earlier benchmark, do not model social interactions). Further, we demonstrated how our collision-based metrics provide a more concrete picture regarding the model performance.

Our proposed models outperform competitive baselines on TrajNet++ synthetic dataset by benchmarking against several popular interaction module designs in the field. On the real dataset, there is no clear winner amongst all the designs in terms of distance-based metrics, when compared on equal grounds. Our proposed design shows significant gains in reducing model prediction collisions. There is room for improvement, and we hope that our benchmark facilitates researchers to objectively and easily compare their methods against existing works so that the quality of trajectory forecasting models can keep increasing, allowing us to tackle more challenging scenarios.

In this chapter, we proposed interaction module designs and training strategies that improve distance-based metrics and significantly reduce collisions. However, the current model architecture outputs a unimodal prediction. Given the high uncertainty of the future, a good forecasting model should be able to provide multiple possible futures or modes. In the next chapter, we tackle the challenge of designing generative models that can output multiple socially-acceptable futures.

3 Generative Adversarial Networks for Multimodal Trajectory Forecasting

This chapter is based on the following two articles:

Parth Kothari, and Alexandre Alahi, *Safety-compliant Generative Adversarial Networks for Human Trajectory Forecasting*, Accepted, IEEE Transactions on Intelligent Transportation Systems

Yuejiang Liu*, Parth Kothari*[I] and Alexandre Alahi, *Collaborative Sampling in Generative Adversarial Networks*, AAAI (2020)

3.1 Introduction

In the previous chapter, we studied interaction encoder designs and training strategies to learn socially-aware unimodal forecasting models. In this chapter, we focus on extending these models to the multimodal setting (see Fig. 3.1). Given the history, the forecasting model needs to be able to output all futures without missing any mode. To achieve our objective, we propose a novel generative adversarial network design for safety-compliant multimodal forecasting and a collaborative sampling technique to refine the bad predictions. Similar to the previous chapter, we discuss current evaluation methodologies albeit in the multimodal scenario.

The objective of multi-modal trajectory forecasting is to learn a generative model over future trajectories. Generative adversarial networks (GANs) [24] are a popular choice of generative models for trajectory forecasting as they can effectively capture all possible future modes by mapping samples from a given noise distribution to samples in real data distribution. Gupta *et al.* [17] proposed Social GAN (SGAN), GANs with social mechanisms, to learn human interactions and output multimodal trajectories. Following the success of SGAN, recent works [25, 26, 79, 98] have proposed improved GAN architectures to better model human interactions in crowds. Indeed, these designs have been successful in reducing the distance-based metrics on real-world datasets [25]. However, we discover that they fail to model social interactions *i.e.*, the

---

[I] denotes equal contribution

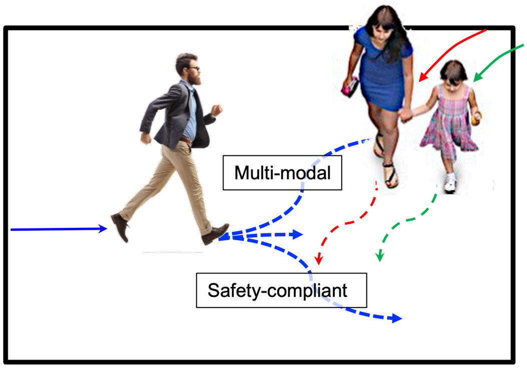


Figure 3.1: Given the history (solid), a forecasting model needs to account for social rules of human motion when predicting collision-free multimodal futures (dash). SGANv2 learns social interactions via spatio-temporal interaction modelling and a transformer-based discriminator. Additionally, it refines unsafe outputs using a collaborative sampling strategy.

models output colliding trajectories.

The failure to output collision-free trajectories can be attributed to the fact that the current discriminator designs do not fully model human-human interactions; hence they are incapable of differentiating real trajectory data from fake data. *Only when the discriminator is capable of differentiating real data from fake data, can the supervised signal from it be meaningful to teach the generator*. To tackle this issue, we propose two architectural changes to the SGAN design: (1) Spatio-temporal interaction modelling to better discriminate between real and generated trajectories. (2) A transformer-based discriminator design to strengthen the sequence modelling capability and better guide the generator training. Equipped with these structural changes, our proposed architecture *SGANv2*, learns to better model the underlying etiquette of human motion as evidenced by reduced collisions.

To further reduce the prediction collisions, SGANv2 leverages the trained discriminator even at test time. In particular, we perform collaborative sampling [135] between the generator and discriminator at test-time to guide the unsafe trajectories sampled from the generator. Additionally, we empirically demonstrate that collaborative sampling not only helps to refine trajectories but also has the potential to prevent mode collapse, a phenomenon where the generator fails to capture all modes in the output distribution.

We empirically validate the efficacy of SGANv2 in outputting socially compliant predictions on both synthetic and real-world trajectory datasets. First, we shed light on the shortcomings of the current metric commonly used to measure the multimodal performance, namely Top-20 ADE/FDE [17]. Specifically, we demonstrate that a simple predictor that outputs uniformly spaced predictions performs at par with the state-of-the-art methods when evaluated using only Top-20 ADE/FDE. To counter this limitation, we propose an alternate evaluation scheme to better measure the socially-compliant multimodal performance of a model. We demonstrate that SGANv2 outperforms competitive baselines on both synthetic and real-world trajectory datasets under the new evaluation scheme. Finally, we demonstrate the ability of collaborative

Method	Generative Model	Spatio-temporal Interaction Modelling in Generator	Multimodal	Spatio-temporal Interaction Modelling in Discriminator	Discriminator Design	Test-time Refinement
S-LSTM [16]	–	✓	✗	–	–	✗
DESIRE [76]	VAE	✓	✓	–	–	✓
Trajectron [87]	VAE	✓	✓	–	–	✗
SGAN [17]	GAN	✗	✓	✗	RNN	✗
S-BiGAT [25]	GAN	✓	✓	✗	RNN	✗
SGANv2 [Ours]	GAN	✓	✓	✓	Transformer	✓

Table 3.1: High-level comparison of our proposed architecture against selected common generative models for trajectory forecasting.

sampling to prevent mode collapse on the recently released Forking Paths [136] dataset. Our main contributions are:

1. We propose SGANv2, an improved SGAN architecture that incorporates spatio-temporal interaction modelling in both the generator and the discriminator. Moreover, our transformer-based discriminator better guides the learning process of the generator.
2. We demonstrate the efficacy of collaborative sampling between the generator and discriminator at test-time to reduce prediction collisions and prevent mode collapse in trajectory forecasting.

3.2 Related Work

In this section, we review forecasting model designs that predict multimodal outputs. Table 3.1 provides a high-level overview of how SGANv2 architecture differs from selected generative model-based designs.

Multimodal forecasting. Neural networks trained using L_2 loss are condemned to output the average of all possible outcomes. To tackle this, one line of work proposes L_2 loss variants [19, 101, 137, 138] capable of handling multiple hypotheses. However, these variants fail to penalize low quality predictions, for e.g. samples that are far away from the ground truth and undergo collisions. Thus, training using these variants can result in high diversity but low quality predictions.

Another line of work utilizes generative models [17, 25, 26, 76, 87, 139], with Variational Autoencoders (VAEs) and Generative Adversarial networks (GANs) being the most popular, to model future trajectory distribution. VAE models in trajectory forecasting [76, 87] employ a loss objective based on different variants of the euclidean distance. Such a formulation leads to low quality samples especially when the predictions are uncertain [140]. On the other hand, the discriminator of the GAN framework acts as a learned loss function that naturally penalizes the low quality samples under the adversarial training objective *i.e.* penalty is incurred on the generator if a sample does not look real [24]. Thus, we choose GANs as our generative model as they can effectively produce diverse and high-quality modes by transforming samples from a noise distribution to samples in the real data.

GANs in trajectory forecasting. SGAN [17] used an LSTM encoder-decoder with social mechanisms within the GAN framework [68] to perform multimodal forecasting. Following the success of SGAN, various GAN-based architectures have been proposed to better model multimodality in crowds [25, 26, 141] as well as on roads [142, 143]. Yuke Li [141] proposed to infer the latent decisions of the agents to model multimodality. Kosaraju *et. al.* [25] proposed to introduce two discriminators: a local discriminator for the local pedestrian trajectories, similar to [17, 26], and a global discriminator that accounted for the spatial interactions. Recent works advocated performing spatial interaction modelling only at the end of observation, as this strategy did not impact the distance-based metrics and saved computational time [17, 25]. In this chapter, we argue that it is crucial to equip the discriminator with the ability to model spatio-temporal interactions.

Current works do not leverage recent advances in sequence modelling and utilize an LSTM-based discriminator. Recently, Transformers [108] have been shown to outperform RNNs in almost all sequence modelling tasks, including trajectory forecasting [144, 145]. Therefore, we design our discriminator using the transformer and demonstrate that it better guides the generator training. Similar to our work, Giuliari *et al.* [144] utilizes transformers but does not take into account social interactions leading to high collisions in the outputs. The spatio-temporal transformer design of STAR [105] is most closely related to the design of our discriminator. However, as discussed above, their L_2 loss training objective can fail to effectively model multimodality. Further, in contrast to previous transformer and GAN-based works, SGANv2 performs test-time refinement that leads to further collision reduction, discussed next.

Test-time Refinement. Test-time Refinement refers to the task of refining model predictions at test-time. Lee *et al.* [76] propose an inverse optimal control based module to refine the predicted trajectories. Sun *et al.* [146] refine trajectories using a reciprocal network that reconstructs input trajectories given the predictions. However, they rely on the strong assumption that both forward and backward trajectories follow identical rules of human motion. We propose to refine trajectories by performing collaborative sampling between the trained generator and discriminator. This technique provides theoretical guarantees with respect to moving the generator distribution closer to real distribution.

Mode Collapse. Mode collapse is the phenomenon where the generator distribution fails to capture all modes of target distribution. SGAN [17] collapses to a single mode of behavior. Social Ways [147] utilizes InfoGAN that overcomes this issue albeit on a toy dataset. We empirically show that the SGANv2 overcomes mode collapse on the more-diverse Forking Path dataset [136].

3.3 Method

Modelling human trajectories using generative adversarial networks (GANs) has the potential to learn the underlying etiquette of human motion and output realistic multimodal predictions. Indeed, recent GAN-based trajectory forecasting models have been successful in reducing distance-based metrics, however they suffer from high prediction collisions. In this section, we present SGANv2, an improvement over the SGAN architecture to output safety-compliant predictions. On a high level, we propose three structural changes: (1) *Spatio-temporal interaction modelling* within the discriminator and generator to better understand social interactions, (2) *Transformer-based discriminator* to better guide the generator, (3) *Collaborative sampling mechanism* between the generator and discriminator to refine the colliding trajectories at test-time. Our proposed changes are generic and can be employed on top of any existing GAN-based architecture.

3.3.1 Problem statement

Given a scene, we receive as input the trajectories of all people within the scene denoted by $\mathbf{X} = \{X_1, X_2, ...X_n\}$, where n is the number of people in the scene. The trajectory of a person i, is defined as $X_i = (x_i^t, y_i^t)$, for time $t = 1, 2...T_{obs}$ and the future ground-truth trajectory is defined as $Y_i = (x_i^t, y_i^t)$ for time $t = T_{obs} + 1, ...T_{pred}$. The objective is to accurately and simultaneously forecast the future trajectories of all people $\hat{\mathbf{Y}} = \hat{Y}_1, \hat{Y}_2...\hat{Y}_n$, where $\hat{Y}_i$ is used to denote the predicted trajectory of person i and the corresponding ground-truth future trajectory is Y_i. The velocity of a pedestrian i at time-step t is denoted by v_i^t.

3.3.2 Generative Adversarial Networks

GANs consist of two neural networks, namely the generator G and the discriminator D, which are trained together in tandem. The objective of D is to correctly identify whether a sample belongs to the real data distribution or is generated by the generator. The objective of G is to produce realistic samples which can fool the discriminator. G takes as input a noise vector z sampled from a given noise distribution p_z and transforms it into a real looking sample $G(z)$. D outputs a probability score indicating whether a sample comes from the generator distribution p_g or the real data distribution p_r. Training GANs is essentially a minimax game between the generator and the discriminator:

$$\min_G \max_D \mathbb{E}_{x \sim p_r}[\log(D(x))] + \mathbb{E}_{z \sim p_z}[1 - \log(D(G(z)))]. \tag{3.1}$$

3.3.3 Interaction modelling designs

Modelling social interactions is the key to outputting safe and accurate future trajectories. In this chapter, we argue that current works do not model interactions between agents sufficiently within

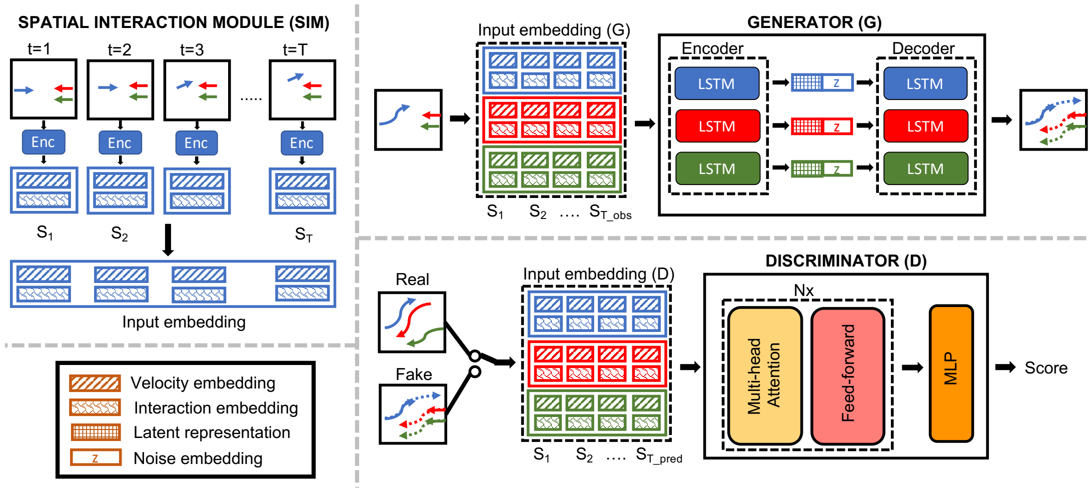


Figure 3.2: Our proposed SGANv2 model: Our model consists of three main parts: the spatial interaction embedding module (SIM), the generator (G) and the discriminator (D). At each time-step, for each pedestrian, the SIM outputs the motion embedding and the spatial interaction embedding. G encodes the input embedding sequence using the encoder LSTM to obtain the latent representation. The latent representation along with the sampled noise vector z is used to generate multimodal predictions using the decoder LSTM. D inputs the stacked embedding sequence of real (or fake) trajectories, encodes it using the transformer architecture to obtain the real (or fake) score.

both the generator and discriminator leading to large number of prediction collisions. Here, we differentiate between the notion of performing *spatial interaction modelling* and performing *spatio-temporal interactions modelling*. An architectural design is said to perform *spatial interaction modelling* if it models the interaction between pedestrians at a **single time-step only**. For instance, SGAN performs spatial interaction modelling within the generator as it encodes the neighbourhood information only once, at the end of the observation. On the other hand, an architectural design is said to perform *spatio-temporal interaction modelling* if it performs the spatial interaction modelling at **every** time-step (from $t = 1$ to $t = T_{pred}$) and the temporal evolution of the interactions are captured using any sequence encoding mechanism, *e.g.* an LSTM or a Transformer. We empirically demonstrate that spatio-temporal interactions modelling within both the generator and the discriminator are essential to output safer trajectories.

3.3.4 SGANv2 architecture

We now describe our proposed model design in detail (see Fig. 3.2). Our architecture consists of three key components: the Spatial Interaction embedding Module (SIM), the Generator (G), and the Discriminator (D). SIM is responsible for spatial interaction modelling while the G and D perform temporal modelling. Thus, *G and D in congregation with SIM perform spatio-temporal interaction modelling (STIM)*. In particular, SIM performs motion embedding and spatial interaction embedding for each pedestrian at each time-step. G encodes the embedded sequence through time and outputs multimodal predictions using an LSTM encoder-decoder framework. D, modelled using a transformer [108], inputs the entire sequence comprising the observed trajectory $\mathbf{X}$ and the future prediction $\hat{\mathbf{Y}}$ (or ground-truth $\mathbf{Y}$), and classifies it as real/fake.

Spatial Interaction Embedding Module. One important characteristic that differentiates human motion forecasting from other sequence prediction tasks is the presence of social interactions: the trajectory of a person is affected by other people in their vicinity. SIM performs the task of encoding human motion and human-human interactions in the spatial domain at a particular time-step. We embed the velocity v^t of pedestrian i at time t using a single layer MLP to get the motion embedding vector e_i^t given as:

$$e_i^t = \phi(v_i^t; W_{emb}), \tag{3.2}$$

where ϕ is the embedding function with weights W_{emb}.

The design of SIM is flexible and it can utilize any spatial interaction module proposed in literature (e.g. [25, 148]). It embeds the spatial configuration of the scene and outputs the interaction embedding p_i^t for pedestrian i at time-step t. We then concatenate the motion embedding with the spatial interaction embedding, *i.e.* $s_i^t = [e_i^t; p_i^t]$, and provide the concatenated embedding s_i^t to the G (or the D). The input embedding is constructed using the ground-truth observations from $[1, T_{obs}]$, and generator predictions from $[T_{obs}+1, T_{pred}]$.

Generator. Within the generator, the encoder LSTM encodes the input embedding sequence provided by the SIM. The encoder LSTM helps to model the temporal evolution of spatial interactions in the form of the following recurrence:

$$h_i^t = LSTM_{enc}(h_i^{t-1}, s_i^t; W_{\text{encoder}}), \tag{3.3}$$

where h_i^t denotes the hidden state of pedestrian i at time t, W_{encoder} are the weights of encoder LSTM that are learned.

The output of the LSTM encoder for each pedestrian at the end of the observation period represents his/her *observed scene representation*. Similar to SGAN, we utilize this representation to condition our GAN for prediction. In other words, SGANv2 take as input noise z and the observed scene representation to produce future trajectories that are conditioned on the past observations. The decoder hidden-state of each pedestrian is initialized with the final hidden-state of the encoder LSTM. The input noise z is concatenated with the inputs of the decoder LSTM, resulting in the following recurrence for the decoder LSTM:

$$h_i^t = LSTM_{dec}(h_i^{t-1}, [s_i^t; z_i]; W_{\text{decoder}}), \tag{3.4}$$

where W_{decoder} are the weights of decoder LSTM.

The decoder hidden-state at time-step t of pedestrian i is then used to predict the next velocity at time-step $t+1$. Similar to Alahi *et al.*[16], we model the next velocity as a bivariate Gaussian distribution parametrized by the mean $\mu^{t+1} = (\mu_x, \mu_y)^{t+1}$, standard deviation $\sigma^{t+1} = (\sigma_x, \sigma_y)^{t+1}$ and correlation coefficient ρ^{t+1}:

$$[\mu^t, \sigma^t, \rho^t] = \phi_{dec}(h_i^{t-1}, W_{\text{norm}}), \tag{3.5}$$

where ϕ_{dec} is an MLP and W_{norm} is learned.

Discriminator. The social interactions between humans evolve with time. Therefore, we design our discriminator to perform spatio-temporal interaction modelling. Also, in recent times, transformers [108] have become the de-facto model for modelling temporal sequences, replacing recurrent architectures [105, 144]. Therefore, we design the discriminator as a transformer to perform the temporal sequence modelling of the output provided by SIM.

The discriminator takes as input $Traj_{real} = [\mathbf{X}, \mathbf{Y}]$ or $Traj_{fake} = [\mathbf{X}, \hat{\mathbf{Y}}]$ and classifies them as real/fake. The discriminator has its own SIM, which provides the spatial interaction embedding s_i^t for each pedestrian i at each time-step t in the input sequence. Instead of passing s_i^t through an LSTM (similar to the generator), we stack these embedded vectors together to form an embedded sequence S_i for each pedestrian i (similar to an embedded sequence obtained after embedding word tokens in the field of natural language [108]):

$$S_i = [s_i^1; s_i^2; \dots s_i^{T_{pred}}]. \tag{3.6}$$

This sequence S_i is given as input to the encoder of the transformer proposed in [108]. The ability of transformers to capture the temporal correlations within the spatial interaction embedding lies mainly in its self-attention module. Within the attention module, each element of the sequence S_i is decomposed into query (Q), key (K) and value (V). The matrix of outputs is computed using the following equation [108]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{3.7}$$

where d_k is the dimension of the SIM embedding s_i^t. The output of the attention layer is normalized and passed through a feedforward layer to obtain the latent representation of the input sequence, denoted by R_i:

$$R_i = \max(0, A_i * W1 + b1) * W2 + b2, \tag{3.8}$$

where the weights $W1, W2, b1, b2$ are learned, $*$ denotes matrix multiplication and A_i denotes the normalized representation of the output of the attention module. We utilize the last element of R_i, as the representation of the input sequence. This embedding gets scored using an MLP ϕ_d to determine if the sequence is real or fake.

3.3.5 Training

As mentioned earlier, SGANv2 is a conditional GAN model. It takes as input noise vector z, sampled from $\mathcal{N}(0,1)$, and outputs future trajectories $\hat{\mathbf{Y}}$ conditioned on the past observations $\mathbf{X}$. We found the least-square training objective [149] to be effective in training SGANv2:

$$\min_G \mathscr{L}(G) = \frac{1}{2}\mathbb{E}_{z\sim p_z}[(D(X,G(X,z))-1)^2], \tag{3.9}$$

$$\min_D \mathscr{L}(D) = \frac{1}{2}\mathbb{E}_{x\sim p_r}[(D(X,Y)-1)^2] + \frac{1}{2}\mathbb{E}_{z\sim p_z}[(D(X,G(X,z)))^2]. \tag{3.10}$$

Additionally, we utilize the variety loss [17] to further encourage the network to produce diverse samples. For each scene, we generate k output predictions by randomly sampling z and penalize the prediction closest to the ground-truth based on L2 distance.

$$\mathscr{L}_{variety} = \min_k ||Y - G(X,z)^{(k)}||_2^2. \tag{3.11}$$

Following the strategy in [148], the generator predicts only the trajectory of the pedestrian of interest in each scene and uses the ground-truth future of neighbours during training. During test time, we predict the trajectories of *all* the pedestrians simultaneously in the scene. All the learnable weights are shared between all pedestrians in the scene.

3.4 Collaborative sampling in GANs

Despite successful applications in a wide variety of tasks, training GANs is notoriously unstable, often impacting the model distribution. An imperfect generator can potentially output bad predictions that can result in undesirable consequences. In this section, we propose to go beyond GAN training and explore methods for effective sampling. Our goal is to improve the model distribution by fully exploiting the value contained in both the generator and discriminator during sampling.

The standard practice in GANs discards the discriminator during sampling. However, this sampling method loses valuable information learned by the discriminator regarding the data distribution. Here, we propose a collaborative sampling scheme between the generator and the discriminator for improved data generation. Guided by the discriminator, our approach refines the generated samples through gradient-based updates at a particular layer of the generator, shifting the generator distribution closer to the real data distribution. Specifically, once training completes, we freeze the parameters of the generator and refine the generated samples using the gradients provided by the discriminator. This gradient-based sample refinement can be performed repeatedly at any layer of the generator, ranging from low-level feature maps to the final output space.

Algorithm 1 Collaborative Sampling

```
Input: a frozen generator $G$, a frozen discriminator $D$, the layer index for sample refinement
    $l$, the maximum number of steps $M$, the stopping criterion $\eta$
Output: a synthetic sample $x$
Randomly draw a latent code $z$
$x^0 \leftarrow$ ProposeSample($G, z$)
for $m = 0, 1, \dots, M-1$ do
    if $D(x^m) < \eta$ then
        $g_l^m \leftarrow$ GetGradient($D$,$x_l^m$),
        $x_l^{m+1} \leftarrow$ UpdateActivation($g_l^m$,$x_l^m$),    (Eq. 3.13)
        $x^{m+1} \leftarrow$ UpdateSample($G$,$x_l^{m+1}$),    (Eq. 3.14)
    else
        break
    end if
end for
```

3.4.1 General formulation

Consider a generator network that inputs a latent code z and produces an output x. It typically consists of multiple layers:

$$\begin{aligned} G(z) &= G_L \circ G_{L-1} \circ \cdots \circ G_1(z), \\ G_l(x_l) &= \sigma(\theta_l \cdot x_l) + b_l, \quad l = 1, 2, \dots, L, \end{aligned} \tag{3.12}$$

where G_l is the lth layer of the generator, x_l is the corresponding activation input, σ is a nonlinear activation function, θ_l and b_l are the model parameters. The input to the first layer is $x_1 = z$ and the output of the last layer is $G_L(x_L) = x$. For a randomly drawn sample from the generator distribution, *i.e.*, $x \sim p_g$, the discriminator outputs a real-valued scalar $D(x)$ which indicates the probability of x to be real. When the generator and the discriminator reach an equilibrium, the generated samples are no longer distinguishable from the real samples, *i.e.*, $D^*(x) = \frac{p_r(x)}{p_r(x)+p_g(x)} = 1/2$. However, such a saddle point of the minimax problem is hardly obtained in practice [150], indicating room for improvement over the model distribution p_g.

Our goal is to shift p_g towards p_r through sampling without changing the parameters of the generator. Inspired by the gradient-based MC methods using Langevin [151] or Hamiltonian [152] dynamics, we leverage the gradient information provided by the discriminator to continuously refine the generated samples through the following iterative updates:

$$x_l^{m+1} = x_l^m - \lambda \nabla_l \mathscr{L}_G(x_l^m), \tag{3.13}$$

$$x^{m+1} = G_L \circ G_{L-1} \circ \dots G_l(x_l^{m+1}), \tag{3.14}$$

where m is the iteration number, λ is the stepsize, l is the index of the generator layer for sample

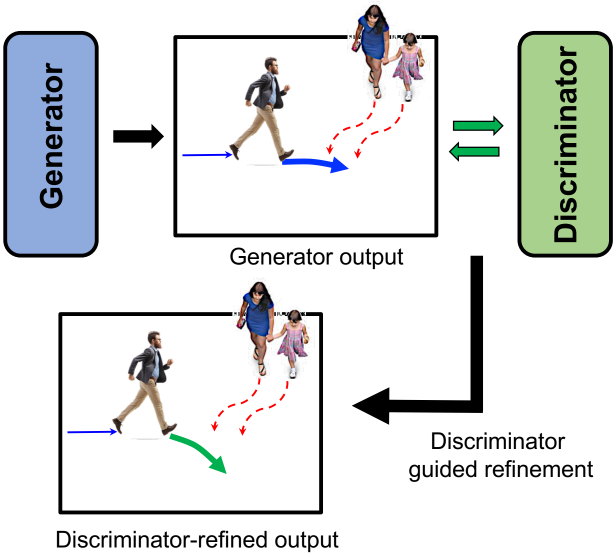


Figure 3.3: Illustration of trajectory refinement using collaborative sampling. The trained discriminator provides feedback to improve the generated samples during test time.

refinement, $\mathcal{L}_G$ is the loss of the generator, *e.g.*, the non-saturating loss advocated in [68]:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z}[\log D(G(z))] \tag{3.15}$$

The iterative sample update consists of two parts: in the backward pass, the discriminator provides the generator with gradient feedback to adjust the activation map of the selected layer l (Eq. 3.13); in the forward pass, the generator reuses part of its parameters to propose an improved sample (Eq. 3.14). A pseudo code is summarized in Algorithm 1.

Recall that an optimal discriminator outputs the density ratio between $p_r(x)$ and $p_g(x)$. The iterative updates shift samples to the regions in which $p_r(x)/p_g(x)$ is higher. In other words, samples are encouraged to move to regions where less samples are produced by the generator but more samples are expected in the real data distribution. Our method forms a closed-loop sampling process, allowing both the generator and the discriminator to contribute to sample generation.

Our proposed collaborative sampling method is generic and we have shown its successful application on various vision tasks in our AAAI paper [36]. In the following subsection, we re-formulate it for motion forecasting.

3.4.2 Collaborative sampling in SGANv2

In the context of motion forecasting, our trained discriminator has knowledge regarding the social etiquette of human motion. Therefore, we propose to utilize this knowledge to refine the *bad* predictions proposed by the generator. We define a prediction as *bad* if the pedestrian of interest undergoes collision in the model prediction. We refine such trajectories by performing collaborative sampling between the generator and discriminator at test-time, as demonstrated in Fig. 3.3.

Our goal is to refine the generator prediction using gradients from the discriminator *without* updating the parameters of the generator. We leverage the gradient information provided by the discriminator to continuously refine the generator predictions of the pedestrian of interest i at test-time through the following iterative update:

$$\hat{Y}_i^{m+1} = \hat{Y}_i^m - \lambda \nabla \mathscr{L}_G(\hat{Y}_i^m), \tag{3.16}$$

where m is the iteration number, λ is the stepsize, $\mathscr{L}_G$ is the loss of the generator in Eq. 3.9. The trajectories are updated till either the discriminator score goes above a defined threshold or the maximum number of iterations is reached.

3.5 Experiments

In this section, we highlight the ability of SGANv2 to output socially-compliant multimodal futures. We evaluate the performance of our architecture against several state-of-the-art methods on the ETH/UCY datasets [41, 62] and on the interaction-centric TrajNet++ benchmark [148]. Additionally, we highlight the potential of collaborative sampling to prevent mode collapse on the Forking Paths [136] dataset. Additional ablation studies in synthetic environments, implementation details of experiments and the source code is provided in the appendix. We evaluate two variants of our model against various baselines:

- **SGANv2 w/o CS**: Our GAN architecture comprising of a transformer-based discriminator that performs spatio-temporal interaction modelling.
- **SGANv2**: Our complete GAN architecture in combination with collaborative sampling at test-time.

Evaluation metrics

1. **Top-k Average Displacement Error (ADE)**: Average l_2 distance between ground truth and closest prediction (out of k samples) over all predicted time steps.
2. **Top-k Final Displacement Error (FDE)**: The distance between the final destination of closest prediction (out of k samples) and the ground truth final destination at the end of the prediction period T_{pred}.
3. **Prediction collision (Col)** [148]: The percentage of collision between the primary pedestrian and the neighbors in the *forecasted future* scene.

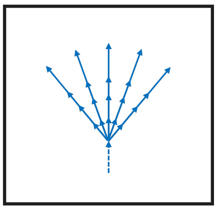


Figure 3.4: Outputs (solid) of a uniform predictor (UP) conditioned on the last observed velocity.

Model	**ETH**		**HOTEL**		**UNIV**		**ZARA1**		**ZARA2**	
	Top-20	Col	Top-20	Col	Top-20	Col	Top-20	Col	Top-20	Col
Transformer† [144]	0.6/0.9	5.8	0.3/0.5	8.2	0.8/1.3	10.9	0.3/0.4	7.1	**0.2/0.3**	11.3
STGAT† [101]	0.7/1.2	1.7	0.5/1.0	4.2	0.3/0.7	13.9	**0.2/0.4**	3.9	0.2/0.4	6.9
Social-STGCNN† [102]	0.7/1.2	6.7	0.3/0.6	10.4	0.5/0.8	25.0	0.3/0.5	12.1	0.3/0.5	19.4
Uniform Predictor (UP)	**0.6/0.9**	3.3	**0.2/0.4**	5.1	**0.3/0.6**	15.7	0.3/0.6	4.7	0.2/0.4	7.5
SGANv2 [Ours]	0.7/1.2	**1.0**	0.3/0.5	**1.2**	0.5/0.8	**8.3**	0.3/0.6	**1.3**	0.3/0.5	**2.2**

Table 3.2: Quantitative evaluation of various methods on ETH-UCY. Only observing the Top-20 metric (as done by previous work) can lead to incorrect conclusions. We show that a high-entropy uniform predictor is highly competitive with respect to state-of-the-art methods in multimodal forecasting using Top-20 metric. See Table 3.4 for a more informative evaluation.

3.5.1 Limitations of current multimodal evaluation scheme

Current multimodal forecasting works utilize metrics that measure model performance at the *individual level* such as the top-k ADE/FDE [17, 101]. This metric evaluates the quality of the predicted distribution per pedestrian; and does not measure the interaction between different pedestrians. Further, the value of k is very high (k=20 being most common). Almost all the recent works [17, 25, 101, 144, 153] in human trajectory forecasting utilize the Top-20 ADE/FDE metric [17] to quantify multimodal performance. We argue that measuring multimodal performance based solely on this metric can be misleading.

The Top-20 ADE/FDE metric can be easily cheated by predicting a high entropy distribution that covers all the space but is not precise [154]. We empirically validate this claim by comparing state-of-the-art baselines against a simple hand-crafted uniform predictor (`UP`). `UP` takes as input the last observed velocity of each pedestrian and outputs 20 uniformly spread trajectories (see Fig 3.4). `UP` outputs 20 predictions using the combination of 5 different relative direction profiles $[0, 25, 50, -25, -50]$ (in degrees relative to current direction of motion) and 4 different relative speed profiles $[1, 0.75, 1.25, 0.25]$ (factors of the current speed).

Table 3.2 compares the performance of recent state-of-the-art methods [101, 102, 144] and `UP` on ETH-UCY datasets. It is apparent that by observing the *Top-20 metric only*, `UP` seems to perform better (or at par) against the state-of-the-art baselines. If we note the prediction collisions, it is apparent that `UP` is not a good multimodal predictor. This corroborates our conjecture that a high entropy distribution can easily cheat the Top-20 metric leading to incorrect conclusions.

Method	Top-3	Col
CV* [133]	0.4/1.0	21.1
LSTM* [73]	0.3/0.6	19.0
S-LSTM* [148]	0.2/0.5	2.2
D-LSTM* [148]	0.2/0.5	2.2
CVAE [76]	0.2/0.5	4.6
WTA [138]	0.2/0.4	2.4
SGAN [17]	0.2/0.4	2.8
SGANv2 w/o CS [Ours]	0.2/0.4	1.9
SGANv2 [Ours]	**0.2/0.4**	**0.6**

Table 3.3: Quantitative evaluation on TrajNet++ synthetic dataset. SGANv2 with collaborative sampling greatly reduces the model prediction collisions (Col) without compromising on the distance-based metrics. *Unimodal methods

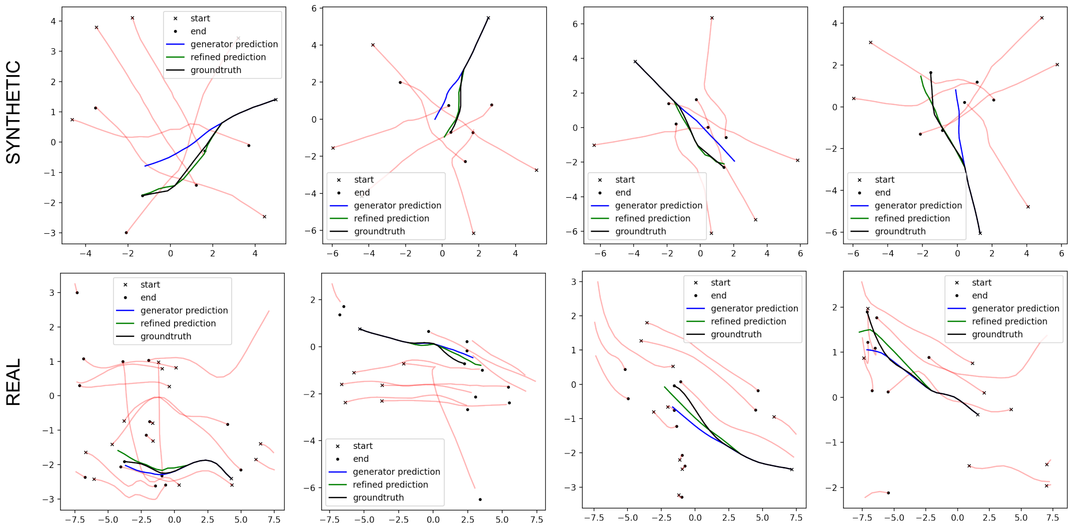


Figure 3.5: Illustration of collaborative sampling reducing model collisions on both TrajNet++ synthetic and real-world datasets. Given a generator prediction of the pedestrian of interest (blue) that undergoes collision with the neighbours (red), our discriminator, equipped with spatio-temporal interaction modelling, provides feedback based on its learned understanding of human-human interactions. The resulting refined prediction (green) does not undergo collision and in some cases, is closer to the ground-truth (black)

Multimodal evaluation scheme To counter the above issues with current multimodal evaluation strategy, we propose to set k to a lower value ($k = 3$) in our experiments; as a lower k is a better proxy for likelihood estimation for implicit generative models [154]. Further, to measure the interaction-modelling capability, we focus on the percentage of collisions between the primary pedestrian and the neighbors in the *forecasted future* scene.

Model	**ETH**		**HOTEL**		**UNIV**		**ZARA1**		**ZARA2**	
	Top-3	Col	Top-3	Col	Top-3	Col	Top-3	Col	Top-3	Col
CV* [133]	1.1/2.3	5.3	0.4/0.8	7.2	0.6/1.4	20.3	0.4/1.0	6.0	0.3/0.7	9.6
LSTM* [73]	1.0/2.1	5.8	0.5/0.9	6.7	0.6/1.3	20.2	0.5/1.0	5.2	0.4/0.8	9.5
Uniform Predictor (UP)	1.1/2.2	3.3	0.5/0.9	5.1	0.6/1.3	15.7	0.5/1.0	4.7	0.4/0.8	7.5
Transformer† [144]	1.0/1.9	5.8	0.5/0.9	8.2	2.3/4.2	10.9	0.5/1.0	7.1	0.4/0.8	11.3
S-LSTM* [16]	1.1/2.1	2.2	0.5/0.9	2.5	0.7/1.5	11.8	0.4/0.9	2.7	0.4/0.8	3.7
CVAE [76]	1.1/2.2	2.8	0.4/0.8	1.5	0.7/1.5	12.6	0.4/0.9	2.6	0.4/0.8	3.5
WTA [138]	1.0/1.9	2.5	0.4/0.7	2.3	0.6/1.3	12.7	0.4/0.8	2.2	0.3/0.7	4.1
SGAN [17]	1.0/2.0	2.2	0.4/0.7	1.7	0.6/1.3	11.8	0.4/0.8	2.3	0.3/0.7	3.2
STGAT† [101]	**0.9/1.8**	1.7	0.7/1.4	4.2	0.6/1.2	13.9	0.4/0.9	3.9	0.4/0.7	6.9
Social-STGCNN† [102]	1.0/1.8	6.7	0.4/0.8	10.4	0.7/1.3	25.0	0.5/0.9	12.1	0.4/0.8	19.4
S-BiGAT [25]	1.0/1.9	3.3	0.4/0.7	1.7	0.6/1.3	11.5	0.4/0.8	2.2	0.3/0.7	3.3
SGANv2 w/o CS [Ours]	1.0/1.9	1.7	0.4/0.7	1.4	0.6/1.3	11.5	0.4/0.8	2.1	0.3/0.7	3.6
SGANv2 [Ours]	1.0/1.9	**1.0**	**0.4/0.7**	**1.2**	**0.6/1.3**	**8.3**	**0.4/0.8**	**1.3**	**0.3/0.7**	**2.2**

Table 3.4: Quantitative evaluation of our proposed method on ETH/UCY datasets. We observe the trajectories for 8 times steps (3.2 seconds) and show prediction results for the next 12 time steps (4.8 seconds). Errors reported are Top-3 ADE / FDE in meters, Col in %. SGANv2 improves in collision metric without compromising on the distance-based metrics. *Unimodal

3.5.2 Synthetic experiments

We first demonstrate the efficacy of our proposed architectural changes in SGANv2 compared to other generative model designs in the TrajNet++ synthetic setup. We observe that SGANv2 greatly improves upon the Top-3 ADE/FDE metric with a lower collision metric compared to training a model using only variety loss (see Table 3.3).

Next, we utilize collaborative sampling technique to refine trajectories that undergo collision at test-time. The trained discriminator provides feedback to the colliding samples which helps to reduce the collisions. For each colliding prediction, we perform 5 refinement iterations with stepsize 0.01. We observe that this scheme greatly reduces the collision rate by $\sim \mathbf{70\%}$. The first row of Fig 3.5 illustrates the ability of collaborative sampling to refine predictions in the synthetic scenario.

The appendix presents an ablation which demonstrates that when training with only adversarial loss, SGAN and S-BiGAT result in high amount of collisions in its predictions. On the other hand, SGANv2 equipped with spatio-temporal interaction modelling and transformer-based discriminator leads to almost zero collisions.

3.5.3 Real-world experiments

Next, we evaluate the performance of our SGANv2 architecture in real-world datasets of ETH/UCY and the TrajNet++ benchmark. For ETH/UCY, we observe the trajectories for 8 times steps (3.2 seconds) and show prediction results for 12 (4.8 seconds) time steps. For TrajNet++, we observe the trajectories for 9 times steps (3.6 seconds) and show prediction results for 12 (4.8 seconds) time steps.

Method	**Top-3**	**Col**
CV* [133]	0.6/1.3	10.9
LSTM* [73]	0.5/1.2	9.3
S-LSTM* [16]	0.5/1.0	4.9
D-LSTM* [148]	0.5/1.1	3.9
CVAE [76]	0.5/1.1	3.9
WTA [138]	0.5/1.0	3.5
SGAN [17]	0.5/1.0	3.5
S-NCE [155]	0.5/1.1	4.0
PECNet [153]	**0.4/0.9**	10.7
Uniform Predictor (UP)	0.6/1.2	8.4
Transformer† [144].	0.7/1.3	9.4
STGCNN† [102]	0.6/1.1	12.6
STGAT† [101]	0.5/1.1	5.6
S-BiGAT [25]	0.5/1.0	3.3
SGANv2 w/o CS [Ours]	0.5/1.0	3.1
SGANv2 [Ours]	0.5/1.0	**2.3**

Table 3.5: Quantitative evaluation of our proposed method on TrajNet++ real-world dataset. SGANv2 in combination with collaborative sampling (CS) improves in collision metric without compromising on the distance-based metrics. *Unimodal

Table 3.4 provides the quantitative evaluation of various baselines and state-of-the-art forecasting methods on the ETH/UCY dataset. We observe that SGANv2 outputs safer predictions in comparison to competitive baselines without compromising on the prediction accuracy. Our Top-3 ADE/FDE are on par with (if not better than) state-of-the-art methods while our collision rate is significantly reduced thanks to spatio-temporal interaction modelling. It is further interesting to note that Trajectory Transformer [144] and the simple uniform predictor (`UP`) that performed the best on Top-20 ADE/FDE in Table 3.2 are not among the top performing methods when evaluated on the more-strict Top-3 ADE/FDE. Next, we benchmark on the TrajNet++ with interaction-centric scenes with a standardized evaluator that provides a more objective comparison [148].

Table 3.5 compares SGANv2 against other competitive baselines on TrajNet++ real-world benchmark. The first part of Table 3.5 reports simple baselines and the top-3 official submissions on AICrowd made by different works literature [148, 153, 155]. SGANv2 performs at par with the top-ranked PECNet [153] on the Top-3 evaluation while having **3x** lower collisions demonstrating that spatio-temporal interaction modelling is key to outputting safer trajectories [II]. Additionally, we utilize the open-source implementation of three additional state-of-the-art methods (denoted by †) and evaluate them on the TrajNet++ benchmark. Compared to these competing baselines, SGANv2 improves upon the Top-3 ADE/FDE metric by $\sim$ 10% and the collision metric by $\sim$ 40%.

We perform collaborative sampling to refine trajectories that undergo collision in real world datasets. For each colliding prediction, we perform 5 refinement iterations with stepsize 0.01. We observe that this procedure reduces the collision rate by $\sim$ **30%** on both ETH/UCY and TrajNet++. The trained discriminator understands human social interactions, and provides

---

[II] PECNet performs spatial interaction modelling once at end of observation

G_{Pool}	D_{Pool}	**TrajNet++ Synth**		**TrajNet++ Real**	
		Top-3	Col	Top-3	Col
✗	✗	0.3 / 0.5	18.3	0.5 / 1.1	9.6
✓	✗	0.2 / 0.4	4.1	0.5 / 1.0	3.9
✓	✓	0.2 / 0.4	**1.9**	0.5 / 1.0	**3.1**

Table 3.6: Quantitative analysis of importance of interaction modelling in GANs. Modelling interactions in both generator and discriminator is crucial to reduce collisions in outputs.

feedback to the bad samples, and consequently helps to reduce collisions. The second row of Fig 3.5 illustrates a few real-world scenarios where collaborative sampling demonstrates the ability to refine generator predictions that undergo collisions. In conclusion, we observe that SGANv2 beats competitive baselines in generating socially-compliant trajectories without compromising on the distance-based metrics.

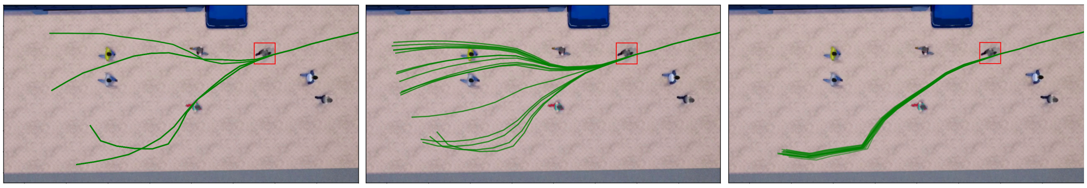
Interaction modelling. In Table 3.6, we empirically demonstrate that modelling interactions is the key to reducing prediction collisions. We consider the performance of different variants of our proposed SGANv2 architecture based on the interaction modelling schemes within the generator and discriminator. It is apparent that modelling interaction within both the generator and discriminator is necessary to output safe multimodal trajectories.

3.5.4 Multimodal analysis

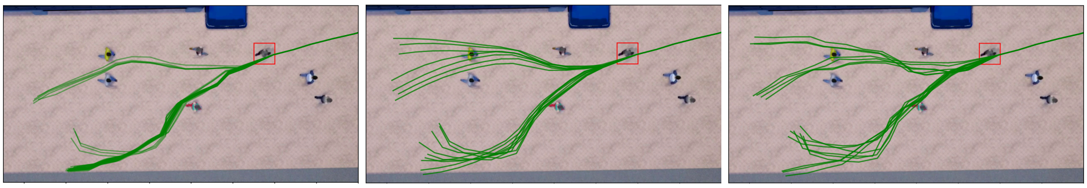
In this experiment, we demonstrate the potential of SGANv2 to prevent mode collapse in trajectory generation. We utilize the sample scene ‘Zara01’ from the Forking Paths dataset. We choose this scene as the multimodal futures of the ‘Zara01’ scene is only affected by social interactions, and not physical obstacles. It forms the ideal test ground to check the multimodal performance of forecasting models. In this experiment, we observe the trajectories for 8 times steps (3.2 seconds) and show prediction results for 13 (5.2 seconds) time steps.

Fig. 3.6 qualitatively illustrates the performance of a GAN model trained using variety loss [17, 138] and other GAN objectives on the chosen scene. As there are 4 dominant modes in the scene, we chose $k = 4$ for the variety loss. The model trained using variety loss (Fig. 3.6b) ends up learning a uniform distribution, *i.e.*, high diversity and low quality, as there is no penalty on the bad samples during training. Variety loss only penalizes the sample closest to the ground-truth. SGAN training [17] (Fig. 3.6c) results in mode collapse, *i.e.*, low diversity and high quality as standard GAN training is highly unstable. Social Ways [26] proposed infoGAN objective [156] to mitigate the mode collapse issue. The InfoGAN improves upon SGAN, however, it still fails to cover all the modes (Fig. 3.6d).

Empirically, we found that training SGANv2 provides a better mode coverage compared to InfoGAN, but the resulting distribution is still not accurate. As shown in Fig. 3.6f, our proposed collaborative sampling at test time helps to improve the accuracy of the SGANv2 predictions,



(a) Ground Truth (b) Variety Loss (c) SGAN



(d) Social Ways (e) SGANv2 w/o CS (f) SGANv2

Figure 3.6: Illustration of effectiveness of SGANv2 to prevent mode collapse on Forking Paths [136]. (b) Training models using variety loss [138] leads to uniform output distribution. (c) Training using SGAN objective [17] leads to mode collapse while (d) InfoGAN [26] helps to alleviate the issue to some extent. (e) SGANv2 helps to cover all the modes, and in combination with (f) collaborative sampling, we can successfully recover all modes with high accuracy.

recovering modes with low coverage. The trained discriminator guides the generated samples to these modes. Thus, we see that collaborative sampling is not only effective in refining trajectories at test time, but also can help to prevent mode collapse.

3.5.5 Key attributes

We now analyze the performance of the key SGANv2 design choices in the TrajNet++ synthetic setup. In the synthetic setup, we have access to the goals of each agent, allowing us to calculate Distance-to-Goal (Dist2Goal) [157], defined as the L2 distance between the predicted final destination and the goal of the agent.

Rationale behind Distance to Goal: It is possible that the generator predicts a socially-acceptable mode that *does not correspond* to the ground-truth mode (see Fig. 3.7). If we calculate the ADE/FDE with respect to the ground-truth for such a predicted mode (that differs from ground-truth), the numbers will be high, *misleading* us to incorrectly conclude that the generator did not learn the underlying task of trajectory forecasting. However, if the predicted destination is close to the goal of the agent, then one can assert that a different but socially acceptable mode has been predicted. The Col metric will help to validate that no collisions take place. Thus, Dist2Goal in combination with the Col metric helps to validate that a predicted mode is socially plausible.

Table 3.7 quantifies the performance of various GAN architectures trained *without* variety loss [138]. SGAN [17] performs the worst on the Col metric as the discriminator does not perform any interaction modelling, thereby not possessing the ability to learn the concept of collision avoidance. Only if the discriminator learns the collision avoidance property, can we expect

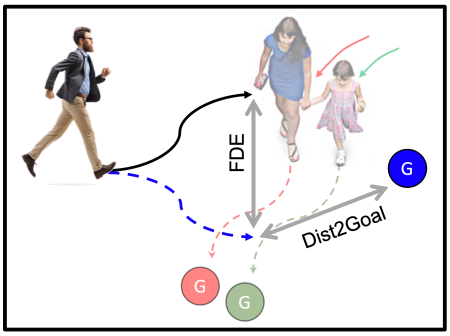


Figure 3.7: Illustration of difference between Dist2Goal and Final Displacement Error (FDE). FDE is the distance between ground-truth position and predicted position at end of prediction period. Dist2Goal is the distance between predicted position and the final position of agent in the entire dataset.

Model	Spatio-temporal Interaction Modelling in Discriminator	Discriminator Design	Col	Dist2Goal
Ground-truth	–	–	0.0	8.6
SGAN [17]	✗	LSTM	24.9	8.9
S-BiGAT [25]	✗	LSTM	8.4	8.9
SGANv2-L	✓	LSTM	0.8	8.8
SGANv2	✓	Transformer	**0.2**	**8.6**

Table 3.7: Quantitative evaluation of various GAN architectural designs on TrajNet++ synthetic dataset. Col in % and Dist2Goal in meters. SGANv2 learns to successfully predict socially acceptable outputs evidenced by lower collisions and Dist2Goal.

it to teach the generator to output collision-free trajectories. The global discriminator of S-BiGAT [25] performs spatial interaction modelling *only once*, at the end of prediction. Thus, the global discriminator is able to reason about interactions spatially but cannot model the temporal evolution of the same. SGANv2 equipped with spatio-temporal interaction modelling results in **near-zero** prediction collision. It is apparent that spatio-temporal interaction modelling within the discriminator plays a significant role in teaching the generator the concept of collision avoidance.

We now justify the design choices of sequence modelling within the discriminator using the *Dist2Goal* metric. We compare an additional design of our proposed SGANv2 architecture: SGANv2-L, an SGANv2 with an LSTM discriminator. SGANv2-L trained using LSTM discriminator shows stopping behavior, indicated by the high Dist2Goal value in the test set. In other words, SGANv2-L outputs collision-free trajectories but the predictions fail to move towards the goal of the primary agent. On the other hand, SGANv2 is able to output collision-free trajectories with a lower Dist2Goal (almost matching the ground-truth Dist2Goal value of 8.6m). In conclusion, SGANv2 is able to output socially acceptable trajectories when compared to other GAN-based designs.

3.5.6 Implementation details

The generator and the discriminator have their own spatial interaction embedding modules (SIM). Each pedestrian has his/her encoder and decoder.

Synthetic experiments. The velocity of each pedestrian is embedded into a 16-dimensional vector. The hidden-state dimension of the encoder LSTM and decoder LSTM of the generator is 64. The dimension of the interaction vector of both the generator and discriminator is fixed to 64. We utilize Directional-Grid [148] interaction module with a grid of size 12×12 and a resolution of 0.6 meters. For the LSTM discriminator, the hidden-state dimension is set to 64. For the transformer-based discriminator, we stack N=4 encoder layers together. The dimension of query vector, key vector and value vector is fixed to 64. The dimension of the feedforward hidden layer within each encoder layer is set to 64. We train using ADAM optimizer [129] with a learning rate of 0.0003 for the generator and 0.001 for the discriminator for 50 epochs. The ratio of generator steps to discriminator steps for LSTM discriminator and transformer-based discriminator is 2:1. For synthetic data experiment, we have access to the goals of each pedestrian. The direction to the goal is embedded into a 16-dimensional vector. The batch size is fixed to 32.

Real-world experiments. The velocity of each pedestrian is embedded into a 32-dimensional vector. The hidden-state dimension of the encoder LSTM and decoder LSTM of the generator is 128. The dimension of the interaction vector of both the generator and discriminator is fixed to 256. We utilize Directional-Grid [148] interaction module with a grid of size 12×12 and a resolution of 0.6 meters. For the LSTM discriminator, the hidden-state dimension is set to 128. The ratio of generator steps to discriminator steps is 2:1. For the transformer-based discriminator, we stack N=2 encoder layers together (see Fig. 2 of main text). The dimension of query vector, key vector and value vector is fixed to 128. The dimension of the feedforward hidden layer within each encoder layer is set to 1024. We train using ADAM optimizer [129] with a learning rate of 0.001 for both the generator and the discriminator for 25 epochs with a learning rate scheduler of step-size 10. The batch size is fixed to 32. The weight of variety loss is set to 0.2.

Multimodal Analysis. The velocity of each pedestrian is embedded into a 16-dimensional vector. The hidden-state dimension of the encoder LSTM and decoder LSTM of the generator is 32. We train using ADAM optimizer [129] with a learning rate of 0.0003 for the generator and 0.001 for the discriminator.

3.6 Conclusion

We presented SGANv2, an improved SGAN architecture equipped with two crucial architectural changes in order to output safety-compliant trajectories. First, SGANv2 incorporates spatio-temporal interaction modelling that can help to understand the subtle nuances of human interactions. Secondly, the transformer-based discriminator better guides the generator learning process. Furthermore, the collaborative sampling strategy helps leverage the trained discriminator during test-time to identify and refine the socially-unacceptable trajectories output by the generator. We empirically demonstrated the strength of SGANv2 to reduce the model collisions without comprising the distance-based metrics. We additionally highlighted the potential of collaborative sampling to overcome mode collapse in a challenging multimodal scenario.

The core idea behind collaborative sampling is to re-use the learned discriminator at test time to refine the bad generator predictions. As mentioned in Sec. 3.4, our sampling technique can be applied to any GAN design and is not specific to the motion forecasting setup. In our AAAI publication[36], we analyzed this technique in detail and demonstrated its applicability to a variety of vision tasks. We further showed that this technique, under mild assumptions, helps to shift the generator distribution closer to the real world distribution.

Our work in Chapter 2 and Chapter 3 aims at expanding the current horizon of trajectory forecasting models for real-world applications where human lives are at risk, such as social robots or autonomous vehicles. Accuracy, safety, and robustness are all mandatory keywords. Over the past years, researchers have focused their evaluation on distance-based metrics. Yet, if we compare the methods on the safety-critical “collision" metric, we observe a difference in performance above 50%. Hence, we believe that one should focus more on this metric and develop methods that aim for zero collisions.

In the following chapters, Chapter 4 and Chapter 5, we focus on two challenges specific to the real-world deployment of forecasting models: interpretability and adaptability.

4 Interpretable Social Anchors for Human Trajectory Forecasting

This chapter is based on the article:

Parth Kothari, Brian Sifringer and Alexandre Alahi, *Interpretable Social Anchors for Human Trajectory Forecasting in Crowds*, CVPR (2021)

4.1 Introduction

Human trajectory forecasting in crowds, at its core, is a sequence prediction problem with specific challenges of capturing inter-sequence dependencies (social interactions) and consequently predicting socially-compliant multimodal distributions. In the previous chapters, we proposed neural network designs that tackled these fundamental challenges. However, these data-driven methods still suffer from one crucial limitation: lack of interpretability. In this chapter, we focus on designing a data-driven framework that provides a high-level rationale behind its predictions without compromising the accuracy.

Early works designed hand-crafted methods based upon domain knowledge to forecast human trajectories, either with physics-based models such as Social Forces [28], or with pattern-based models such as discrete choice modelling (DCM) [158, 159, 160]. These models, based on domain knowledge, were successful in showcasing crowd phenomena like collision avoidance and leader-follower type behavior. Moreover, the hand-designed nature of these models rendered their predictions to be interpretable. However, human motion in crowds is much more complex and due to its long-term nature, these first-order methods suffer from predicting inaccurate trajectories.

Building on the success of recurrent neural network-based models in learning complex functions and long-term dependencies, Alahi *et al.* [16] proposed the first neural network (NN) based trajectory forecasting model, Social LSTM, which outperformed the hand-crafted methods on distance-based metrics. Due to the success of Social LSTM, neural networks have become the de-facto choice for designing human trajectory models [17, 85, 86, 87, 144]. However,

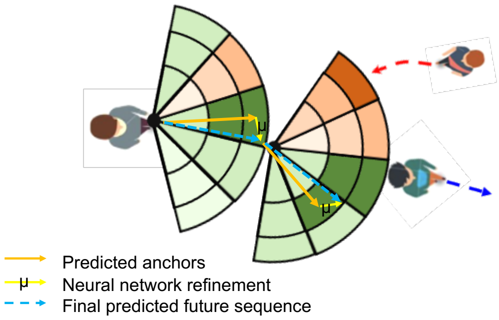


Figure 4.1: While navigating in crowds, humans display various social phenomena like collision avoidance (from red trajectory) and leader follower (towards blue trajectory). We present a model that not only outputs accurate future trajectories but also provides a high-level rationale behind its predictions, owing to the interpretability of discrete choice models. (Un)favourable anchors shown in green (red).

current NN-based trajectory forecasting models suffer from a significant limitation: lack of interpretability regarding the model's decision-making process.

In this chapter, we are interested in combining the forces of the two paradigms of human trajectory forecasting (see Fig. 4.1): the interpretability of the trajectories predicted by hand-crafted models, in particular discrete choice models [158, 161], and the high accuracy of the neural network-based predictions. With this objective, we propose a model that outputs a probability distribution over a discrete set of possible future intents. This set is designed as a function of the pedestrian's speed and direction of movement. Our model learns the probability distribution over these intents with the help of a choice model architecture, owing to its ability to output interpretable decisions. To this end, we resort to a novel hybrid and interpretable framework in DCM [162], where knowledge-based hand-crafted functions can be augmented with neural network representations, without compromising the interpretability.

Our architecture augments each predicted high-level intent with a scene-specific residual term generated by a neural network. The advantage of this is two-fold: first, the residual allows to expand the output space of the model from a discrete distribution to a continuous one. Secondly, it helps to incorporate the complex social interactions as well as the long-term dependencies that the first-order hand-crafted models fail to capture, leading to an increase in prediction accuracy. Overall, we can view our architecture as disentangling high-level coarse intents and lower-level scene-specific nuances of human motion.

We demonstrate the efficacy of our proposed architecture on TrajNet++ [148], an interaction-centric human trajectory forecasting benchmark comprising of well-sampled real-world trajectories that undergo various social phenomena. Through extensive experimentation, we demonstrate that our method performs at par with competitive baselines on both real-world and synthetic datasets, while at the same time providing a rationale behind high-level decisions, an essential component required for safety-critical applications like autonomous systems.

4.2 Related Work

Explaining model decisions. There has been a rising interest in developing methods that can explain the underlying reasons behind model outputs. One approach towards explainable AI is developing post-hoc approaches that explain the decisions of black-box neural networks [120, 163, 164]. Specific to trajectory forecasting, one black-box method is the layerwise relevance propagation (LRP) technique [54] introduced in Chapter 2. Recently, Shapley values [165], a concept based on game theory, were proposed to determine the impact of a feature (like surrounding neighbours) on the model decision [166]. The authors quantified the difference in the model behavior between including and excluding a feature.

In this chapter, we focus on the alternative approach of designing a white-box framework that can explain the model decisions based on the underlying social concepts, similar in spirit to the classical models [27, 28]. The circular distribution approach proposed by *Coscia et al.* [167] is most-closely related to ours. The authors fuse the various factors that contribute to a human motion in different physical scene contexts. On the other hand, our framework incorporates the social concepts based on human-human interactions, that influence a pedestrian's motion in crowds.

4.3 Method

Humans have mastered the ability to negotiate complicated social interactions by anticipating the movements of surrounding pedestrians, leading to social concepts such as collision avoidance and leader-follower. Current NN-based architectures, despite displaying high accuracy, are unable to provide a rationale behind their accurate predictions. Our objective is to equip these models with the ability to provide a social concept-based reason behind their decisions. In this section, we describe our proposed architecture, that outputs a high-level intent and a scene-specific residual corresponding to each intent, followed by our DCM-based component that makes the intent interpretable.

4.3.1 Problem statement

For a particular scene, we receive as input the trajectories of all people in a scene as $\mathbf{X} = [X_1, X_2, ...X_n]$, where n is the number of people in the scene. The trajectory of a person i, is defined as $X_i = (x_i^t, y_i^t)$, from time $t = 1, 2...t_{obs}$ and the future ground-truth trajectory is defined as $Y_i = (x_i^t, y_i^t)$ from time $t = t_{obs} + 1, ...t_{pred}$. The goal is to accurately and simultaneously forecast the future trajectories of all people $\hat{\mathbf{Y}} = [\hat{Y}_1, \hat{Y}_2...\hat{Y}_n]$, where $\hat{Y}_i$ is used to denote the predicted trajectory of person i. In other words, we are provided the motion states of all pedestrians at time-steps $t = 1, \dots, T_{obs}$ (input trajectory) and we need to forecast the future motion states from time-steps $t = T_{obs+1}$ to T_{pred} (output trajectory). The velocity of a pedestrian i at time-step t is denoted by $\mathbf{v}_i^t$.

4.3.2 Discrete Choice Models

The theory of discrete choice models (DCMs) is built on a strong mathematical framework, allowing high interpretability regarding the decision making process [168]. DCMs have often been applied in fields of economy [169], health [170] and transportation [171], where interpretation of parameters that capture human behavior is of utmost importance. These models are used to predict, for each person i, their choice among an available set of K options. In the most common DCM-based approach, called Random Utility Maximization (RUM), [172], each option has an associated hand designed function u_k, called utility, and each person is assumed to select the option for which their utility is maximized.

The inputs $\tilde{\mathbf{x}}$ to these utility functions are designed via expert knowledge of the given problem, and are then assigned a vector of learnable weights β. These weights are regressed on all available options with respect to the observations, and reflect the impact of each component in the utility function. It is the study of these weights and corresponding input values that lead to the high interpretability of discrete choice methods at the individual and population level. Formally, the utility for option k is calculated as:

$$U_k(\beta,\tilde{\mathbf{x}}) = u_k(\beta,\tilde{\mathbf{x}}) + \varepsilon_k \tag{4.1}$$

where ε_k is the random term. Varying assumptions on the distribution of this random term leads to different types of DCM models [173, 174].

While many works incorporate data-driven methods into the DCM framework [175, 176, 177], only recently have models been proposed that keep the knowledge-based functions and the parameters interpretable after adding the neural network [162, 178]. In this chapter, we utilize the Learning Multinomial Logit (L-MNL) [162], as our base model. From a discrete modelling perspective, we aim at correcting the underfit due to misspecification thanks to neural networks. On the other hand, from a representation learning standpoint, we incorporate inductive biases in the form of expert-designed functions within the neural network framework.

4.3.3 Social Anchor framework

As shown in Figure 4.2, at each time-step, our model outputs a distribution over a discretized set of K future intents, which we term *social anchors*, denoted by $\mathscr{A} = \{a_k\}_{k=1}^{K}$, as well as scene-specific refinements for each intent. The size of the set is defined by the number of speed levels N_s, and direction changes N_d such that $K = N_d \times N_s$. As we will see in Section 4.3.4, we choose to utilize a DCM to output the distribution over these anchors because of its ability to explain its decisions. Next, we utilize the high expressibility of neural networks to provide a refinement in the output space with respect to each anchor in $\mathscr{A}$. We call these refinements *scene residuals*. These scene-specific residuals allow us to project the coarse and discretized problem back into the continuous domain. Note that the set $\mathscr{A}$ is chosen to be rich enough to provide a desired level of coverage in the output space, so that the magnitudes of the scene-specific

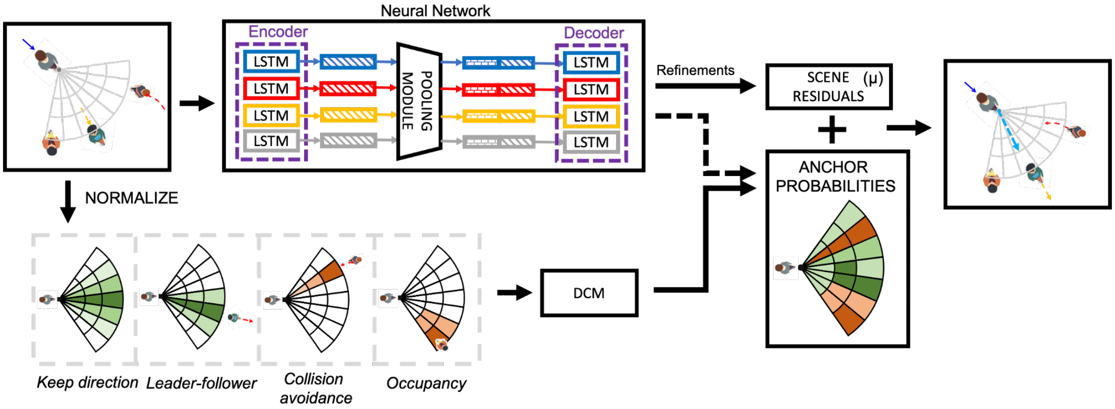


Figure 4.2: Illustration of the Social Anchors framework. At each time-step, the output space of each pedestrian is discretized into a set of possible future intents, normalized with respect to the pedestrian's speed and direction, forming a radial grid. Discrete choice modelling (DCM) is used to predict the next step probability distribution (green high, red low) in an interpretable manner by accumulating the *keep direction, leader-follower, collision avoidance* and *occupancy* rules. A neural network lifts the discrete outputs to a continuous space by providing scene-specific residuals for each intent. The neural network also provides the past motion embedding and interactions embedding which can be added to the hand-crafted DCM functions to better handle complex social interactions and long term dependencies while choosing the future intents.

residuals are minimal.

Scene Residuals: We now describe our neural network architecture that is used to output scene-specific residuals corresponding to each anchor. These residuals are used to model the long-term motion dependencies as well as the complex and often subtle social interactions that cannot be described using first-order hand-crafted rules. We first embed the velocity $\mathbf{v}_i^t$ of pedestrian i at time t using a single layer MLP to get a fixed length embedding vector $\mathbf{e}_i^t$ given as:

$$e_i^t = \phi_{emb}(\mathbf{v}_i^t; W_{emb}), \tag{4.2}$$

where ϕ_{emb} is the embedding function, W_{emb} are the weights to be learned.

Next, we utilize the directional pooling module proposed in [148] to model the social interactions and obtain the interaction vector p_i^t. We then concatenate the input embedding with the interaction embedding and provide the concatenated vector as input to the LSTM module, obtaining the following recurrence:

$$h_i^t = LSTM(h_i^{t-1}, [e_i^t; p_i^t]; W_{\text{encoder}}), \tag{4.3}$$

where h_i^t denotes the hidden state of pedestrian i at time t, W_{encoder} are the weights to be learned. The weights are shared between all pedestrians in the scene.

The hidden-state at time-step t of pedestrian i is then used to predict the residuals corresponding to each anchor at time-step $t+1$. Similar to [16], we characterize the residual corresponding to the k^{th} anchor as a bivariate Gaussian distribution parameterized by the mean $\mu_k^{t+1} = (\mu_x, \mu_y)_k^{t+1}$, standard deviation $\sigma_k^{t+1} = (\sigma_x, \sigma_y)_k^{t+1}$ and correlation coefficient ρ_k^{t+1}:

$$[\mu_k^t, \sigma_k^t, \rho_k^t] = \phi_{dec}(h_i^{t-1}, W_{\text{decoder}}), \tag{4.4}$$

where ϕ_{dec} is modelled using an MLP and $W_{decoder}$ is learned.

4.3.4 Anchor selection

The pedestrian's intent for the next time-step is discretized as a set of K future intentions $\mathscr{A} = \{a_k\}_{k=1}^K$. The selection of an anchor from the choice set $\mathscr{A}$ is posed as a discrete choice modelling task. This is made possible by normalizing the anchor set with respect to both a person's speed and direction. We describe the role of normalization to integrate the DCM structure in Sec. 4.4.4.

While many different rules and behaviors for human motion have been described in DCM literature, we follow the formulation described in [158, 161], which is well adapted to our problem setting. Functions modelling human motion phenomenon which we consider for anchor selection in this chapter are:

1. ***avoid occupancy***: directions containing neighbours in the vicinity are less desirable, scaled by the inverse-distance to the considered anchor.
2. ***keep direction***: pedestrians tend to maintain the same direction of motion.
3. ***leader-follower***: pedestrians have a tendency to follow the tracks of people heading in the same direction, identified as 'leaders'. The relative speed of the leader with respect to the follower entices the follower to slow down or accelerate.
4. ***collision avoidance***: when a neighbour pedestrian's trajectory is head-on towards an anchor, this anchor becomes less desirable due to the chance of a collision.

An illustration of the effects of the above chosen functions on the final anchor selection is shown in Figure 4.2. Given the chosen functions, the associated utility u_k for anchor k is written as:

$$\begin{aligned} u_k(\mathbf{X}) = {} & \underbrace{\beta_{dir} dir_k}_{\textit{keep direction}} + \underbrace{\beta_{occ} occ_k}_{\textit{avoid occupancy}} + \underbrace{\beta_C col_k}_{\textit{collision avoidance}} \\ & \underbrace{+ \beta_{acc} L_{k,acc} + \beta_{dec} L_{k,dec}}_{\textit{leader-follower}}, \end{aligned} \tag{4.5}$$

where β_j are the learnable weights of the corresponding functions. The exact mathematical formulations of the above functions are detailed in [158, 161]. Each person is assumed to select the anchor a_k for which the corresponding utility u_k is maximum.

We would like to point that the performance of the underlying DCM is determined by the hand-crafted functions of human motion that it models. The DCM framework provides the flexibility to integrate any other knowledge-driven function extensively tested in past literature.

Although the knowledge-based functions offer stable and interpretable results, they are unable to capture the heterogeneity of trajectory decisions in more nuanced situations. The future intent of a pedestrian is also dependent on long-term dependencies and subtle social interactions that these first-order hand-designed functions are unable to capture. The inclusion of neural-network-based terms helps to alleviate this issue.

Recently proposed L-MNL [162] architecture allows having both data-driven and knowledge-based terms in the utility while maintaining interpretability. We therefore utilize this framework and add an encoded map of past observations $h(\mathbf{X})$ to adjust for the lack of long term dependencies of knowledge-based equations. Similarly, we also add an encoded map of social interactions $p(\mathbf{X})$ with information from all the neighbours to help model residual interactions, otherwise not captured by hand-designed functions.

In summary, we formulate the anchor selection probabilities as follows:

$$\pi(a_k|\mathbf{X}) = \frac{e^{s_k(\mathbf{X})}}{\sum\limits_{j\in K} e^{s_j(\mathbf{X})}}, \tag{4.6}$$

where

$$s_k(\mathbf{X}) = u_k(\mathbf{X}) + h_k(\mathbf{X}) + p_k(\mathbf{X}). \tag{4.7}$$

$s_k(\mathbf{X})$ represents the anchor function containing the NN encoded terms, $h_k(\mathbf{X})$ and $p_k(\mathbf{X})$, as well as the hand-designed term $u_k(\mathbf{X})$ (Eq. 4.5), following the L-MNL framework. Note that we use DCM assumptions from L-MNL for measuring the anchor probabilities, rather than those of the cross-nested logit model in [161].

4.3.5 Training

Following the training recipe of Chapter 2, all the parameters of our model are learned with the objective of minimizing the negative log-likelihood (NLL) loss:

$$\log p(\mathbf{y}|\mathbf{X}) = \sum_t \log \left(\sum_k \pi(a_k|\mathbf{X}) \, \mathcal{N}(y^t | v_k^t, \, \Sigma_k^t) \right), \tag{4.8}$$

with

$$v_k^t = y^{t-1} + a_k + \mu_k^t, \tag{4.9}$$

and where μ_k^t and Σ_k^t are the scene-specific residuals (described in Sec. 4.3.3), a_k are the coordinates of anchor k and y^{t-1} is the last position preceding the prediction.

As mentioned earlier, given an anchor set $\mathscr{A}$ such that it sufficiently covers the output space, the magnitude of neural network refinements are minimal. During training, we choose to penalize the anchor that is closest to the ground-truth velocity at each time-step.

Therefore, we optimize the following function during training:

$$l(\theta) = \sum_t \sum_k \Big[\mathbb{1}(k^t = \hat{k}^t_m) \Big(\log \pi(a_k|\mathbf{X}) + \log \mathcal{N}(y^t|v^t_k, \Sigma^t_k) \Big) \Big], \tag{4.10}$$

where $\mathbb{1}(\cdot)$ is the indicator function, and $\hat{k}^t_m$ is the index of the anchor most closely matching the ground-truth trajectory $\mathbf{Y}$ at time t, measured as l_2-norm distance in state-sequence space.

Testing: During test time, till time-step t_{obs}, we provide the ground truth position of all the pedestrians as input to the forecasting model. From time t_{obs+1} to t_{pred}, we use the predicted position (derived from the most-probable intent combined with the corresponding residual) of each pedestrian as input to the forecasting model and predict the future trajectories of all the pedestrians.

4.4 Experiments

In this section, we highlight the ability of our proposed method to output socially-compliant interpretable predictions. We evaluate our method on the interaction-centric TrajNet++ dataset. TrajNet++ dataset consists of real-world pedestrian trajectories that are carefully sampled such that the pedestrians of interest undergo social interactions and no collisions occur in both the training and testing set.

4.4.1 Implementation details

The velocity of each pedestrian is embedded into a 64-dimensional vector. The dimension of the interaction vector is fixed to 256. We utilize directional pooling [148] as the interaction module in all the methods for a fair comparison, with a grid of size 16×16 with a resolution of 0.6 meters. We perform interaction encoding at every time-step. The dimension of the hidden state of both the encoder LSTM and decoder LSTM is 256. Each pedestrian has their encoder and decoder. The batch size is fixed to 8. We train using ADAM optimizer [129] with a learning rate of 1e-3 for 25 epochs. For the DCM-based anchor selection, all exponential parameters of the chosen hand-designed functions are set to the estimated values in [158, 161]. For synthetic data, we embed the goals in a 64-dimensional vector.

Evaluation: we consider the following metrics:

1. **Average Displacement Error (ADE)**: the average L_2 distance between ground-truth and model prediction overall predicted time steps.
2. **Final Displacement Error (FDE)**: the distance between the final predicted destination and the ground-truth destination at the end of the prediction period.
3. **Prediction collision (Col)** [148]: this metric calculates the percentage of collision between

the pedestrian of interest and the neighbours in the *predicted* scene. This metric indicates whether the predicted model trajectories collide, *i.e.*, whether the model learns the notion of collision avoidance.

4. **Top-3 ADE/FDE**: given 3 output predictions for an observed scene, this metric calculates the ADE/FDE of the prediction *closest* to the ground-truth trajectory in terms of ADE.

Baselines: we compare against the following baselines:

1. **DCM**: the discrete choice model [27] baseline that predicts the discrete choice with maximum utility.
2. **S-LSTM**: the Social LSTM [16] baseline that outputs a unimodal trajectory distribution.
3. **Winner-Takes-All (WTA)**: this architecture was proposed in [138] to encourage the network to output diverse trajectories.
4. **SGAN**: Social GAN [17], a popular generative model to tackle multimodal trajectory forecasting.
5. **CVAE**: the Conditional Variational Auto-Encoder architectures has been shown recently [76, 87] to successfully predict multi-modal trajectories by learning a sampling model given past observations.
6. **MinK**: to demonstrate the need for a fixed set of prior anchors, we construct this baseline that directly outputs the ranked neural network residuals without any prior anchors.
7. **SAnchor** [**Ours**]: our proposed method that utilizes 15 anchors (5 angle profiles and 3 speed profiles) and their residuals for continuous prediction.

4.4.2 Quantitative analysis

Table 4.1 and Table 4.2 illustrate the quantitative performance of our proposed anchor-based method on TrajNet++ synthetic and real-world dataset respectively. It is clear that our method improves the predictable power of discrete choice models on real-world dataset. Furthermore, we can observe the importance of incorporating a fixed anchor set in our framework.

On the real dataset, there are two factors responsible for the slightly-worse performance of social anchors in comparison to competitive baselines. First, due to incorporation of expert-based functions, the predictive power of the network reduces at the cost of interpretability. Improved inductive biases in the form of more-expressive choice models [161] can help to alleviate this issue. Secondly, the direction normalization step is affected by the noisy real-world measurements. This is corroborated by the fact that on the clean synthetic dataset, social anchors outperform previous methods in terms of collision metric and distance-based metrics. We discuss direction normalization in detail in Section. 4.4.4.

The fundamental advantage of our method against competitive baselines is to provide interpretable predictions which we discuss next.

Model	ADE / FDE	Col	Top-3 ADE / FDE
S-LSTM [16]	0.25/0.50	2.2	0.25/0.50*
WTA [138]	0.28/0.54	2.4	0.22/0.42
SGAN [17]	0.27/0.54	2.8	0.22/0.43
CVAE [76]	0.26/0.52	4.6	0.23/0.47
MinK	0.34/0.72	5.2	0.22/0.42
SAnchor [Ours]	**0.22/0.45**	**0.4**	**0.19/0.38**

Table 4.1: Performance on TrajNet++ synthetic data. Errors reported are ADE / FDE in meters, Col in %. We observe the trajectories for 9 times-steps (3.6 secs) and perform prediction for the next 12 (4.8 secs) time-steps. *Unimodal

Model	ADE / FDE	Col	Top-3 ADE / FDE
DCM [161]	0.67/1.42	8.7	0.67/1.42*
S-LSTM [16]	**0.57/1.24**	5.5	0.57/1.24*
WTA [138]	0.65/1.46	5.6	**0.49/1.05**
SGAN [17]	0.66/1.45	5.9	0.51/1.08
CVAE [76]	0.60/1.28	5.7	0.55/1.20
MinK	0.68/1.48	8.4	0.59/1.25
SAnchor	0.62/1.32	**4.2**	0.58/1.24

Table 4.2: Performance on TrajNet++ real data. Errors reported are ADE / FDE in meters, Col in %. We observe the trajectories for 9 times-steps (3.6 secs) and perform prediction for the next 12 (4.8 secs) time-steps. *Unimodal

4.4.3 Interpretability of the intents

The advantage of incorporating a discrete choice framework for predicting a pedestrian's next intended position is its interpretability. Our proposed architecture allows us to compare the hand-designed features extensively studied in literature along with the data-driven features to identify the most relevant factors, at a given time-step, for the anchor selection.

We demonstrate the ability of our network to output interpretable intents in Fig. 4.3. The direction of the pedestrian of interest is normalized and is facing towards the right. For each row in Fig. 4.3, in addition to the ground-truth map (leftmost), we illustrate the activation maps of: all combined factors, the neural network (NN) map, the overall DCM map and finally the dominant behavioral rules that comprise the DCM function, according to the presented scene. In the first row, we observe that the model correctly chooses to turn left while maintaining constant speed. The different activation maps help to explain the rationale behind the model's decision. Indeed, due to the increased number of potentially colliding neighbours, one can observe that the collision avoidance map along with the occupancy map exerts a strong influence on the decision-making, resulting in the network outputting desired choice of intent.

In the second and third row, we demonstrate two similar cases of leader-follower (LF) that results in different network outputs. In the former case, one of the neighbours being close to the pedestrian of interest results in the LF map exhibiting a strong affinity for slowing down. The strength of the LF map is strong enough to overturn the NN map's decision to maintain constant speed. In contrast, in the third row, the influence of the LF map is weaker. Therefore, due to the

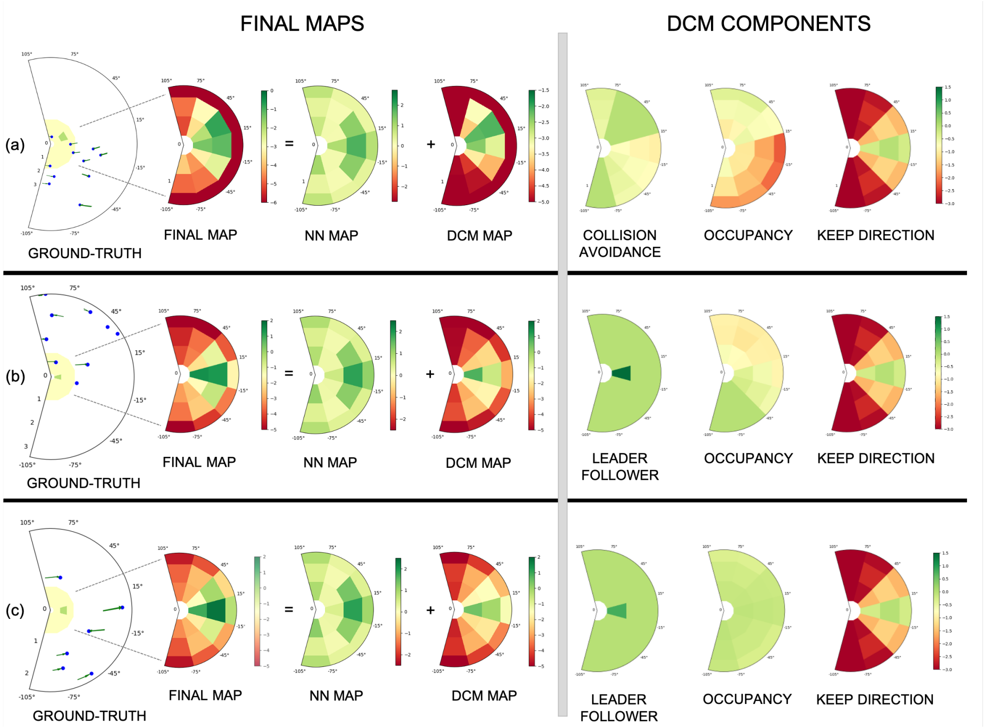


Figure 4.3: Qualitative illustration of the ability of our architecture to output high-level interpretable intents. The direction of motion of the pedestrian of interest is normalized and is facing towards the right. Current neighbour positions are shown in blue and current velocities are shown in green. The ground-truth choice is highlighted in light green. (a) In the first row, the decision of the network is strongly influenced by the collision-avoidance and occupancy map of the DCM. Consequently, the pedestrian changes the direction of motion and turns left maintaining constant speed. (b) In the second row, the leader-follower map exerts a strong influence on the final decision-making causing the model to choose the anchor corresponding to slowing-down. (c) In the third row, the leader-follower map is not strong in intensity and the neural network map guides the decision making resulting in the model maintaining constant speed.

preference of NN map, the overall network chooses to maintain constant speed and direction. Thus, we observe that the DCM maps work well in conjunction with the NN map to provide interpretable outputs.

4.4.4 Direction normalization

Direction normalization at every time-step is an necessary step to enable the integration of the DCM framework. According to the DCM framework for pedestrian forecasting [161], the anchor set A at each time-step is defined dynamically with respect to the current speed and direction of motion. The input scene needs to be rotated so that the pedestrian of interest faces the same direction at every time-step and consequently the appropriate anchor can be chosen by the model. Therefore, thanks to this normalization, we can successfully incorporate the interpretability of

Model	ADE / FDE	Col	Top-3 ADE / FDE
Unimodal methods			
NN-LSTM [148]	0.25/0.50	1.24	0.25/0.50
NN-LSTM (N)	**0.20/0.43**	**0.1**	**0.20/0.43**
Multimodal methods			
WTA [138]	0.28/0.54	4.8	0.22/0.42
WTA (N)	**0.22/0.45**	**0.6**	**0.17/0.35**
SGAN [17]	0.27/0.54	5.1	0.22/0.43
SGAN (N)	**0.24/0.50**	**1.4**	**0.19/0.37**
CVAE [76]	0.26/0.52	1.9	0.23/0.47
CVAE (N)	**0.23/0.47**	**0.5**	**0.22/0.45**

Table 4.3: Effect of normalization on synthetic data. Errors reported are ADE / FDE in meters, Col in %. (N) represents direction-normalized version of the baseline. Direction normalization improves both distance-based and collision metrics.

Model	ADE / FDE	Col	Top-3 ADE / FDE
Unimodal methods			
NN-LSTM [148]	0.58/1.24	7.5 (0.25)	0.58/1.24*
NN-LSTM (N)	0.63/1.36	**5.9**	0.63/1.36*
D-LSTM [148]	0.57/1.24	5.5 (0.19)	0.57/1.24
D-LSTM (N)	0.62/1.32	**4.5**	0.62/1.32
Multimodal methods			
WTA [138]	0.65/1.46	5.1	**0.49/1.05**
WTA (N)	0.63/1.38	**4.4**	0.54/1.15
SGAN [17]	0.66/1.45	5.9	0.51/1.08
SGAN (N)	0.64/1.38	**4.0**	0.51/1.07
CVAE [76]	0.60/1.28	5.7	0.55/1.20
CVAE (N)	0.62/1.34	**4.2**	0.58/1.23

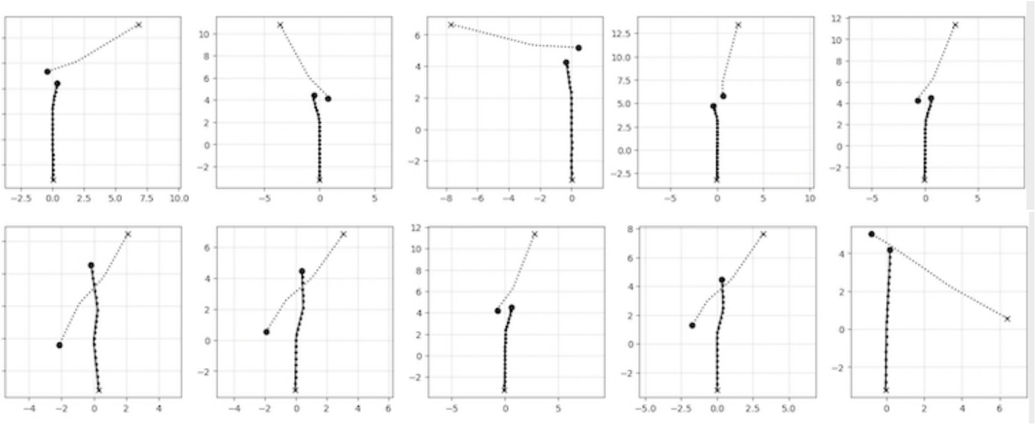
Table 4.4: Effect of normalization on real data. Errors reported are ADE / FDE in meters, Col in %. (N) represents direction-normalized version of the baseline. Direction normalization degrades the distance-based metrics potentially due to noisy real-world measurements.

DCM without compromising prediction accuracy.

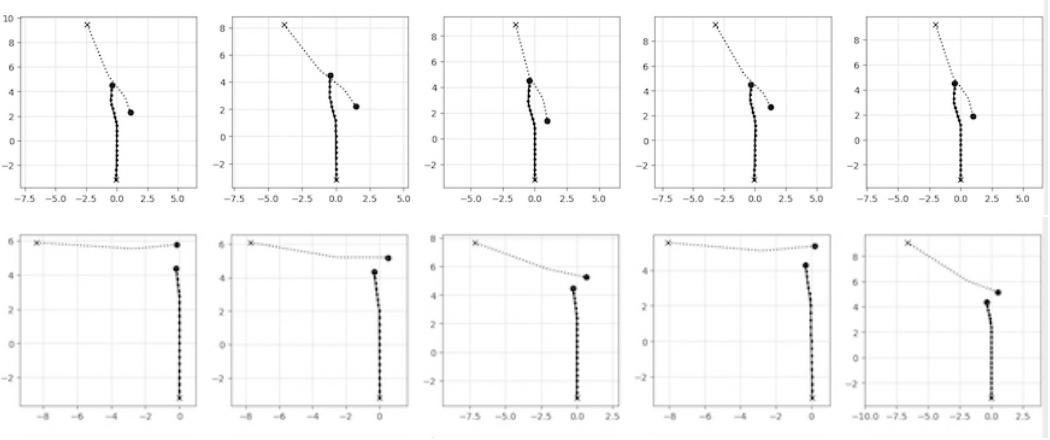
We argue that direction normalization is a general normalization scheme that provides a performance boost, in terms of avoiding collisions, when applied to many existing trajectory forecasting models. The reason behind the improvement is that direction normalization makes the forecasting model rotation-invariant at each time-step, thus allowing the model to focus explicitly on learning the social interactions. We would like to note that the direction normalization differs from the one proposed in [179], as we rotate the direction of motion of a pedestrian's model at each time-step and not just at the end of observation.

To verify the efficacy of direction normalization, we perform a comparison between various baselines and their direction-normalized versions. Table 4.3 and Table 4.4 illustrate the performance boost obtained on applying direction normalization to different trajectory prediction models on both TrajNet++ synthetic and real dataset. On the synthetic dataset, we observe that our proposed normalization scheme provides performance improvement on all the metrics across all the models. On the real dataset, we observe that direction normalization improves the model prediction collision performance.

In addition to providing a performance improvement, the latent representations obtained by a network trained using direction normalization are semantically meaningful in the aspect of



(a) Nearest Neighbours of two samples of an un-normalized baseline.



(b) Nearest Neighbours of two samples of a direction normalized baseline.

Figure 4.4: Qualitative illustration of the effect of direction normalization on model training. We demonstrate the nearest neighbours in the latent space at the output of the encoder for models trained with and without direction normalization. We observe that representations of *socially-similar* scenes are closer to each other when the model is trained using direction normalization. The synthetic dataset comprises two pedestrians interacting at various angles.

modelling social interactions. To demonstrate this, we consider a toy dataset of two pedestrians interacting with each other. The two pedestrians are initialized at different positions on the circumference of a circle with the objective of reaching the diametrically opposite position. The two pedestrians interact at different angles and positions. We train a S-LSTM [16] and direction-normalized S-LSTM model on this dataset. During testing, we obtain the representation outputted by the LSTM encoder for the particular testing scene and find the closest latent-space representations in the training set. Fig. 4.4 represents the top-4 nearest neighbours in each row, in the latent-space, from the training set. We observe that the direction-normalized representations are more semantically similar in terms of not only the trajectory of the pedestrian of interest but also the neighbourhood configuration around the pedestrian.

4.5 Conclusion

We approach the task of human trajectory forecasting by disentangling human motion into high-level discrete intents and low-level scene-specific refinements. By leveraging recent works in hybrid choice models, the discretized intents are selected using both interpretable knowledge-based functions and neural network predictions from the scene. While the former allows us to understand which human motion rules are present in predicting the next intent, the latter handles the effects of both long term dependencies and complex human interactions. Through experiments on both synthetic and real data, we highlight not only the interpretability of our method, but also the accurate predictions outputted by our model. This is made possible because of the scene-specific refinements which efficiently cast the discrete problem into the continuous domain.

In the next chapter, we tackle the adaptation aspect of real-world deployment of forecasting models. Specifically, we develop a strategy to adapt a pre-trained forecasting model to unseen, out-of-distribution scenarios as efficiently as possible.

5 Modular Low-Rank Adaptation for Trajectory Forecasting

This chapter is based on the article:

Parth Kothari, Danya Li, Yuejiang Liu and Alexandre Alahi, *Motion Style Transfer: Modular Low-Rank Adaptation for Deep Motion Forecasting*, CoRL (2022)

5.1 Introduction

Deep motion forecasting models have achieved great success when trained on a massive amount of data. Yet, they often perform poorly when they encounter novel scenarios where the training data is limited. To address this challenge, in this chapter, we propose a transfer learning approach for efficiently adapting pre-trained forecasting models to new domains, such as unseen agent types and scene contexts.

Existing deep motion forecasting models suffer from inferior performance when they encounter novel scenarios [29, 30]. For instance, a network trained with large-scale data for pedestrian forecasting struggles to directly generalize to cyclists. Some recent methods propose to incorporate strong priors robust to the underlying distribution shifts [180, 181, 182]. Yet, these priors often make strong assumptions on the distribution shifts, which may not hold in practice. This shortcoming motivates the following transfer learning paradigm: Adapting a forecasting model pretrained on one domain with sufficient data to new domains such as unseen agent types and scene contexts *as efficiently as possible*.

One common transfer learning approach is fine-tuning a pretrained model on the data collected from target domain. However, directly updating the model is often sample inefficient, as it fails to exploit the inherent structure of the distributional shifts in the motion context. In the forecasting setup, the physical laws behind motion dynamics are generally invariant across geographical locations and agent types: all agents move towards their goal and avoid collisions. As a result, the distribution shift can be largely attributed to the changes in the motion style, defined as the way an agent interacts with its surroundings. Given this decoupling of motion dynamics, it can be efficient for an adaptation algorithm to only account for the updates in the target motion style.

In this chapter, we efficiently adapt a deep forecasting model from one motion style to another. We refer to this task as *motion style transfer*. We retain the domain-invariant dynamics by freezing the pre-trained network weights. To learn the underlying shifts in style during adaptation, we introduce motion style adapters (MoSA), which are new modules inserted in parallel to the encoder layers. The style shift learned by MoSA is injected into the frozen pre-trained model. We hypothesize that the style shifts across forecasting domains often reside in a low-dimensional space. To formulate this intuition, we design MoSA as a low-dimensional bottleneck, inspired by recent works in language [183, 184]. Specifically, MoSA comprises two trainable matrices with a low rank. The first matrix is responsible for extracting the style factors to be updated, while the second enforces the updates. MoSA learns the style updates by adding and updating less than 2% of the parameters in *each layer*.

In low-resource settings, it can be difficult for MoSA to distinguish the relevant encoder layers updates from the irrelevant ones, resulting in sub-optimal performance. To facilitate an informed choice of adaptation layers, we propose a modularized adaptation strategy. Specifically, we consider forecasting architectures that disentangle the fine-grained scene context and past agent motion using two independent low-level encoders. This design allows the flexible injection of MoSA to one encoder while leaving the other unchanged. Given the style transfer setup, our modular adaptation strategy yields substantial performance gains in the low-data regime.

We empirically demonstrate the efficiency of MoSA on the state-of-the-art model Y-Net [20] on the heterogenous SDD [185] and inD [186] datasets in various style transfer setups. Next, we highlight the potential of our modularized adaptation strategy on two setups: Agent Motion Style Transfer and Scene Style Transfer. Finally, to showcase the generalizability of MoSA in self-driving applications, we adapt a large-scale model trained on one part of the city to an unseen part, on the Level 5 Dataset [187]. Through extensive experimentation, we quantitatively and qualitatively show that given just 10-30 samples in the new domain, MoSA improves the generalization error by 25% on SDD and inD. Moreover, our design outperforms standard fine-tuning techniques by 20% on the Level 5 dataset.

5.2 Related Work

Distribution shifts. The primary challenge in adapting to new domains lies in tackling the underlying distributional shifts. One ambitious approach is developing domain generalization techniques that aim to learn models that directly function well in unseen test domains [188, 189]. Negative data augmentation techniques have been applied in a limited scope to reduce collisions [181] and off-road predictions [190] on new domains. Recently, the application of distributionally robust optimization to forecasting has been explored albeit in a two pedestrian setup [191]. Domain adaptation is another line of work that allows a learning algorithm to observe a set of unlabelled test samples. While this approach has been shown effective in a variety of supervised tasks in vision [192, 193, 194, 195], it is not the ideal setup for motion forecasting setup where the crucial challenge is sample efficiency as labels in the form of future trajectories are fairly easy to acquire. Therefore, in this chapter, we take an alternate approach of transfer learning using limited data.

Transfer learning. We are interested in the inductive transfer learning setting for deep motion forecasting [196]. The standard approach of fine-tuning the entire or part of the network [197, 198] has been shown to outperform feature-based transfer strategy [199, 200]. Recently, there has been a growing interest in developing parameter-efficient fine-tuning (PET) methods in both language and vision, as they not only yield a compact model (very few extra parameters) [183, 184, 201]. but also show promising results in outperforming fine-tuning in low-resource settings [184, 202]. Similar in spirit to PET methods, we introduce additional parameters in our network but with an objective of style-conditioned motion generation. In our work, motion style adapters account for the updates in target style that are used to condition the forecasting model outputs. In the motion context, transfer learning given limited data often requires special architecture designs like external memory [203] and meta-learning objectives [204, 205], for human pose prediction in isolation. Meanwhile, our work adopts a style transfer perspective that facilitates fast adaptation for a wider variety of real-world transfer learning setups.

Style transfer. In vision, style transfer refers to the task of transforming an input image into a particular artistic style while preserving the notion of content [206, 207]. In motion context, the popular work of [185] defined *navigation style* as the way different agents interact with their surroundings. It introduced social sensitivity as two handcrafted descriptions of agent style and provided them as input to the social force model [53]. In this chapter, we model style as a latent variable that is learned in a data-driven manner. Closely related to ours, [191] decoupled domain-invariant laws and domain-specific style inside their causal forecasting framework. However, their method imposes the strong constraint of requiring access to multiple environments of varying style during training. Furthermore, we decouple motion style into scene-style components and agent-style components to favour efficient adaptation.

5.3 Method

Deep neural networks have shown remarkable performance in motion forecasting thanks to the availability of large-scale datasets. However, these models often struggle to generalize when unseen scenarios are encountered in the real world due to underlying differences in motion style. In this section, we first formally introduce the problem setting of motion style transfer. Subsequently, we introduce the design of our style adapter modules that help to effectively tackle motion style transfer. Finally, we describe the application of style adapters to a modularized architecture in order to perform style transfer in a more efficient manner.

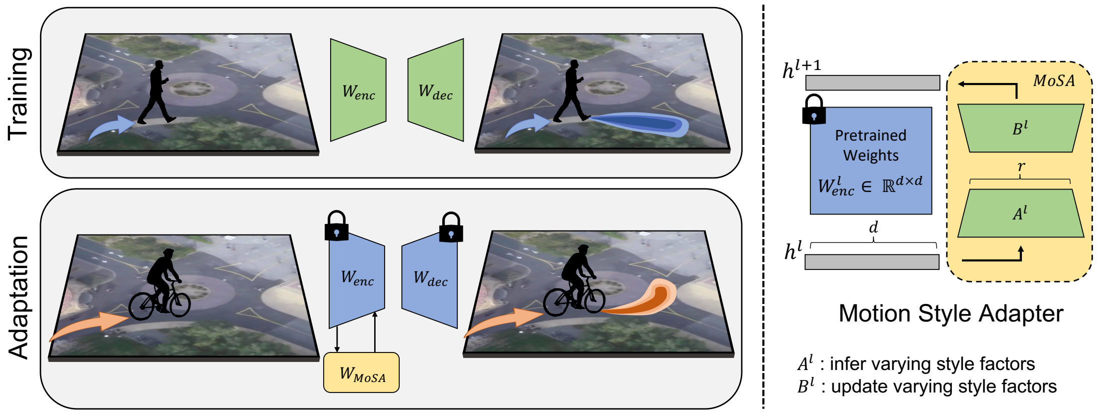


Figure 5.1: We present an efficient transfer learning technique that adapts a forecasting model trained with sufficient labeled data (e.g., pedestrians), to novel domains exhibiting different motion styles (e.g., cyclists). We freeze the pretrained model and only tune a few additional parameters that aim to learn the underlying style shifts (left). We hypothesize that the style updates across domains lie in a low-dimensional space. Therefore, we propose motion style adapters with a low-rank decomposition ($r \ll d$), designed to infer and update the few style factors that vary in the target domain (right).

5.3.1 Motion style transfer

Motion style. Modelling agent motion behavior involves learning the social norms (e.g., minimum separation distance to others, preferred speed, valid areas of traversal) that dictate the motion of the agent in its surroundings. These norms differ across agents as well as locations. For instance, the preferred speed of pedestrians differs from that of cyclists; the separation distance between pedestrians in parks differs from that in train stations. To describe these agent-specific (or scene-specific) elements that govern underlying motion behavior, we define the notion of “motion style". Motion style is the collective umbrella that models the social norms of an agent given its surroundings.

Problem statement. Consider the inductive transfer learning setting for deep motion forecasting across different motion styles. Specifically, we are provided a forecasting model trained on large quantities of data comprising a particular set of style(s) and our goal is to adapt the model to the idiosyncrasies of a target style as efficiently as possible. We denote the model input and ground-truth future trajectory of an agent i using x_i and y_i respectively. The input x_i comprises the past trajectory of the agent, surrounding neighbors, and the surrounding context map. The context can take various forms like RGB images or rasterized maps. We assume that the data corresponding to an agent type is generated by an underlying distribution $\mathscr{P}_{X,Y}(\cdot; s)$ parameterized by s, the style of the agent. As mentioned earlier, the style is dictated by both the agent type and its surroundings.

Training. The forecasting model has an encoder-decoder architecture (see Fig. 5.1) with weights W_{enc} and W_{dec} respectively. The training dataset, D_S of size N is given by $\cup_{s\in S} D_s =$

$(x_i, y_i)_{i\in\{1,\dots,N\}}$, where S is a collection of motion styles observed within the dataset. The model is trained to minimize:

$$\mathscr{L}_{train}(D_S; W_{enc}, W_{dec}) = \frac{1}{N}\sum_{i=1}^{N} \mathscr{L}(x_i, y_i; W_{enc}, W_{dec}). \tag{5.1}$$

Adaptation. When a novel scenario with style s' ($s' \notin S$) is encountered, it leads to a distribution shift and the learned model often struggles to directly generalize to the corresponding dataset $D_{s'} = (x'_i, y'_i)_{i\in\{1,\dots,N_{target}\}}$ of size N_{target}. The common approach to tackling such shifts is to fine-tune the entire or part of the pretrained model. Fine-tuning optimizes an objective similar to training, but on the new dataset:

$$\mathscr{L}_{adapt}(D_{s'}; W_{enc}, W_{dec}) = \frac{1}{N_{target}}\sum_{i=1}^{N_{target}} \mathscr{L}(x'_i, y'_i; W_{enc}, W_{dec}). \tag{5.2}$$

In this chapter, we aim to develop an adaptation strategy for efficient motion style transfer, i.e., cases where N_{target} is small ($N_{target} \ll N$). Often, motion behaviors do not change drastically across domains. Instead, most of the behavioral dynamics are governed by universal physical laws (e.g., influence of inertia, collision-avoidance). We therefore propose to freeze the weights of the pretrained forecasting model and introduce motion style adapters, termed *MoSA*, to capture the target motion style. As shown Fig. 5.1, we adapt a pre-trained forecasting model by fine-tuning W_{MoSA} with the following objective:

$$\mathscr{L}_{adapt}(D_{s'}; W_{MoSA}) = \frac{1}{N_{target}}\sum_{i=1}^{N_{target}} \mathscr{L}(x'_i, y'_i; W_{MoSA}). \tag{5.3}$$

5.3.2 MoSA: motion style adapters

Our main intuition is that the style shifts across forecasting domains are usually localized – they are often due to the changes in only a few variables of the underlying motion generation process. Therefore, during style transfer, we only need to adapt the distribution of this small portion of latent factors, while keeping the rest of the factors constant. These updates would correspond to the changes in motion style ($s \to s'$) in the target domain, as the general principles of motion dynamics (e.g., avoid collisions, move towards goal) remain the same across domains for all agents. We design motion style adapters, referred to as MoSA, to carry out these updates.

Our proposed MoSA design comprises a small number of extra parameters added to the model during adaptation (see Fig. 5.1). Each module comprises two trainable weight matrices of low rank, denoted by A and B. The first matrix A is responsible for inferring the style factors that are required to be updated to match the target style, while the second matrix B performs the desired update. The low rank r realizes our intuition that style updates reside in a low-dimensional space, by restricting the number of style factors that get updated ($r << d$ where d is the dimension size of an encoder layer). Therefore, during adaptation, the weight updates of the encoder are constrained with our low-rank decomposition $W_{MoSA} = BA$. The pretrained model is frozen and only A and B are trained.

For brevity, let us consider the adaptation of encoder layer l with input h^l and output h^{l+1}. As shown in Fig. 5.1, W^l_{enc} and W^l_{MoSA} are multiplied with the same input h^l, and their respective output vectors are summed coordinate-wise as shown below:

$$\text{(Train)}\ \ h^{l+1} = W^l_{enc} h^l, \tag{5.4}$$

$$\text{(Adapt)}\ \ h^{l+1} = W^l_{enc} h^l + W^l_{MoSA} h^l = W^l_{enc} h^l + B^l A^l h^l. \tag{5.5}$$

It has been shown that initialization plays a crucial role in parameter-efficient transfer learning [183, 201]. Therefore, following common practices, matrices A and B are initialized with a near-identity function [183], so that the original network is unaffected when training starts. Furthermore, the near-identity initialization provides flexibility to these modules to ignore certain layers during motion style updates. Despite this flexibility, the total number of extra parameters is significant and can be inconducive to efficient style transfer. Therefore, to further boost sample efficiency, we present a modular adaptation strategy which we describe next.

5.3.3 Modularized adaptation strategy

Motion style can be decoupled into scene-specific style and agent-specific style. Scene-specific style dictates the changes in motion due to physical scene structures. For instance, cyclists prefer to stay on the lanes while pedestrians move along sidewalks. The agent-specific style captures the underlying navigation preferences of different agents like distance to others and preferred speed. Modularizing the encoder into two parts that account for the physical scene and agent's past motion independently can help to decouple the functionality of our motion style adapters and further improve performance.

Consider the modularized motion encoder designs shown in Fig. 5.2. The modularized encoder models the input scene and agent's past history independently. The fusion encoder then fuses the two representations together. This design has the advantage to decouple the task of the style adapters into scene-specific updates and agent-specific updates. Given the modularized setup, the nature of the underlying distribution shifts can help guide which modules within the model are required to be updated to the target style. As demonstrated in Section. 5.4.4, given different categories of style transfer setups, decoupling style adapters can improve the adaptation performance while significantly reducing the number of updated parameters.

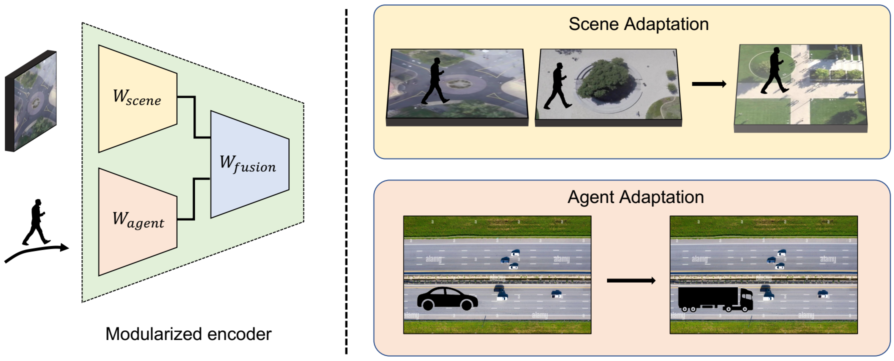


Figure 5.2: Our modular style transfer strategy updates only a subset of the encoder to account for the underlying style shifts. For instance, we adapt the scene encoder only to model scene style shifts (top right). On the other hand, for the underlying agent motion shift, we only update the agent motion encoder (bottom right). This strategy boosts performance in low-resource settings.

5.4 Experiments

Datasets. We use a total of three datasets to study the performance of motion style adapters: Stanford Drone Dataset (SDD) [185], the Intersection Drone Dataset (InD) [186], and Level 5 Dataset (L5) [187]. We consider both short-term and long-term motion forecasting setups. SDD and inD with different heterogeneous agents exhibiting different motion styles provide the ideal setup to validate motion style transfer techniques. L5 dataset provides 25-second long sequences of ego vehicle to study the effects of style transfer on long-term self-driving settings.

Baselines. We compare the following methods during adaptation for the experiments reported in the paper. For a fair comparison, we use the state-of-the-art Y-Net architecture [208] for all baselines on SDD and inD. On L5, we use the Vision Transformer (ViT) architecture [209] across all methods.

- Full Model Finetuning (FT) [197]: we update the weights of the entire model. The learning rate (LR) is 5e-5, unless mentioned otherwise.
- Partial Model Finetuning (ET) [191]: we update the weights of the Y-Net encoder for SDD and inD , and the last two layers of ViT for Level 5. The LR is 5e-4, unless mentioned otherwise.
- Parallel Adapters (PA) [210]: we insert a convolutional layer with filter size of 3 in parallel to each encoder layer and update the weights of these layers. This baseline does not incorporate the low-rank constraint. The LR is 5e-5, unless mentioned otherwise.
- Adaptive Layer Normalization [211, 212]: we update the weights and biases of the layer normalization, wherever present. The LR is 1e-4, unless mentioned otherwise.
- Motion Style Adapters (MoSA) [Ours]: we insert our motion style adapters in parallel to each encoder layer in SDD and inD, and in parallel to query and value matrices of multi-headed attention in Level 5. During modularized adaptation, we add our modules across the specified encoders only, and not the entire encoder. The LR is 3e-3 and the rank r is 1, unless mentioned otherwise.

Table 5.1: Evaluation of adaptation methods for motion style transfer (pedestrians to bikers) on SDD and scene style transfer on InD using few samples $N_{target} = \{10, 20, 30\}$. Error reported is Top-20 FDE in pixels. The generalization error on SDD is 58 pixels and on inD is 33 pixels. Our proposed motion style adapters (MoSA) outperform competitive baselines and improve upon the generalization error by $> 25\%$ in both setups. Mean and standard deviation were calculated over 5 runs.

	Stanford Drone Dataset			Intersection Drone Dataset		
N_{target}	10	20	30	10	20	30
FT	57.28 ± 1.21	52.61 ± 0.87	46.31 ± 1.79	27.92 ± 1.99	25.15 ± 1.08	23.18 ± 0.64
ET [191]	51.88 ± 1.32	46.78 ± 1.78	43.13 ± 1.03	28.06 ± 0.68	23.19 ± 1.39	21.13 ± 1.00
PA [210]	52.77 ± 0.85	47.75 ± 1.83	44.70 ± 1.28	28.71 ± 1.50	26.10 ± 0.74	25.00 ± 1.08
MoSA (ours)	**49.98 ± 1.05**	**45.55 ± 0.77**	**41.69 ± 0.88**	**25.18 ± 0.72**	**21.70 ± 0.84**	**20.35 ± 1.18**

Metrics. We use the established Average Displacement Error (ADE) and Final Displacement Error (FDE) metrics for measuring the performance of model predictions. ADE is calculated as the l_2 error between the predicted future and the ground truth averaged over the entire trajectory while FDE is the l_2 error between the predicted future and ground truth for the final predicted point [16]. For multiple predictions, the final error is reported as the `min` error over all predictions [17]. Additionally, we define the generalization error as the error of the pretrained model on the target domain. The more the dissimilarity between the source domain and target domain, the higher the generalization error.

5.4.1 Motion style transfer across agents on Stanford Drone Dataset

We perform short-term prediction, in which the future trajectory is predicted for the next 4.8 seconds, given 2.5 seconds of observation. The main objective of efficient motion style transfer is to adapt the model with a limited number of samples. We choose to adapt our model on cyclists in *deathCircle_0* as there exists a clear distinction between the motion style of pedestrians and cyclists (see Fig. 5.3). We use the publicly available Y-Net model trained on pedestrian data across *all* scenes and adapt it to the cyclists' data in *deathCircle_0*. We adapt the model using $N_{target} = \{10, 20, 30\}$ samples.

Tab. 5.1 quantifies the performance of various style transfer techniques. The model trained on pedestrians does not generalize to cyclists as evidenced by the high generalization error of 58 pixels. Our MoSA design reduces this error by $\sim 30\%$ using only 30 samples. Moreover, MoSA outperforms the baselines while keeping the pretrained model frozen and updating only 0.5% additional parameters.

We qualitatively analyze the Y-Net goal decoder outputs after model adaptation using MoSA. Fig. 5.4 illustrates the updates in the goal decoder output (red means increase in focus and blue

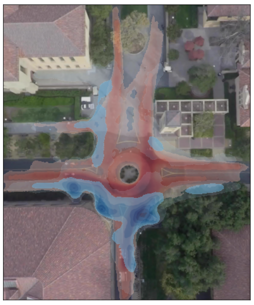


Figure 5.3: Distribution of trajectories of pedestrians (blue) and bikers (red) on *deathCircle*.

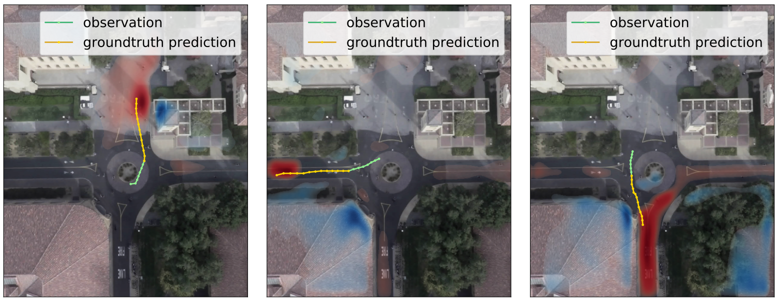


Figure 5.4: Illustration of the difference in goal decoder output of Y-Net on the adaptation of pedestrian-trained model using our proposed style-injection modules (rank=1) (Red is positive, blue is negative). During adaptation, Y-Net learns to focus on the road lanes for biker forecasting.

means decrease in focus). Adapted Y-Net successfully learns the style differences between the behavior of pedestrians and bikers: 1) it correctly infers valid areas of traversal, 2) effectively captures the multimodality of bikers, and 3) updates the motion style parameters as the new biker goal positions (red) are farther from the end of the observation position, compared to the un-adapted goal positions (blue).

5.4.2 Motion style transfer across scenes on Intersection Drone Dataset

In this setup, we perform long-term prediction, in which future trajectory in the next 30 seconds is predicted, given 5 seconds of observation. We use the publicly available Y-Net model and follow the experimental protocol of [20] in which the model is trained on pedestrians in $\{scene2, scene3, scene4\}$ and tested on unseen scene $scene1$. We adapt the model using $N_{target} = \{10, 20, 30\}$ samples.

Despite the long-term prediction setup, the generalization error, in this case, is 33 pixels that are lower compared to the previous SDD setup, as the target domain is more similar to the source domain. Tab. 5.1 quantifies the performance of scene style transfer across all methods. Once again, using just 30 samples, MoSA improves the generalization error by $\sim 40\%$ and outperforms its counterparts.

5.4.3 Motion style transfer across scenes on Lyft Level 5 dataset

To demonstrate the generalizability of our approach, we apply MoSA to L5 dataset. We divide the dataset into two splits based on the location and thereby, construct a scene style shift scenario. We train a Vision Transformer Tiny (ViT-Tiny) model on the majority route and adapt it to the smaller route that was not seen during training (see Fig. 5.6). To simulate low-resource settings, we provide the frames, sampled at different rates, that cover the unseen route only once.

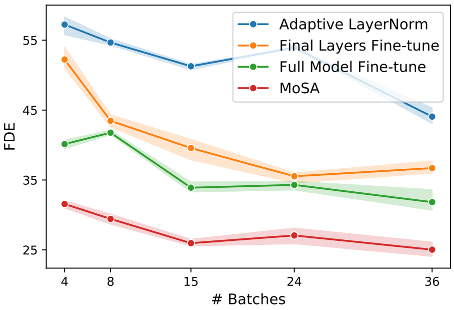


Figure 5.5: Evaluation of adaptation techniques for long-term motion prediction (25 secs) on Level 5. Error in meters for 5 seeds.

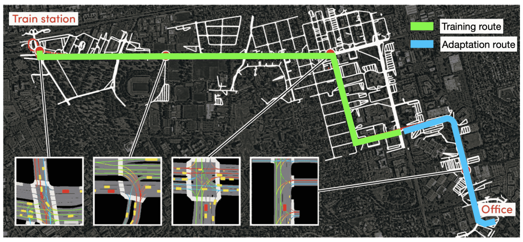


Figure 5.6: Training-Adaptation split for Level 5 prediction dataset. Model trained on the green route and adapted on the blue route. The distribution shift arises from differing styles in different scene locations. The inherent motion style parameters of the agent remain same (ego vehicle in both). Background from [187].

Fig. 5.5 quantitatively evaluates the performance of various adaptation strategies. MoSA performs superior in comparison to different baselines while adding and updating only $\sim 5\%$ of the full model parameters. It is apparent that our low-rank design is a smarter way of adapting models in the motion context as compared to fine-tuning the whole or the last few layers of the model. Fig. 5.7 qualitatively illustrates the final layer attention maps of ViT-Tiny before and after adaptation. We observe that post adaptation, the attention maps are more refined. The ego vehicle better focuses on the possible future routes and the neighbours of interest.

5.4.4 Modularized adaptation strategy

Now, we empirically demonstrate the effectiveness of applying our motion style adapters on top of a modularized architecture on two setups: Scene generalization and Agent Motion generalization.

Setup: We construct Y-Net-Mod on top of the original Y-Net architecture. The modification treats the scene context and agent motion *independently* before fusing their representations together. As shown in Fig. 5.8, the first three layers of the original encoder are decoupled into scene context and past agent motion modules in order to learn their representations independently. Subsequently, the representations are fused together using the fusion encoder, that is similar in design to the last two layers of Y-Net. The original number of channels in each encoder layer of Y-Net are evenly divided between each module in Y-Net-Mod encoder so that the latter is compatible with the Y-Net decoders.

Given Y-Net-Mod, we consider five cases given the module list on which MoSA is applied: (1) scene only [S], (2) agent motion encoder only [A], (3) scene and fusion encoder [S+F], (4) agent motion and fusion encoder [A+F], and (5) scene, agent motion and fusion encoders together [S+A+F].

Agent motion generalization: In $scene1$ of inD, cars and trucks share the same scene context

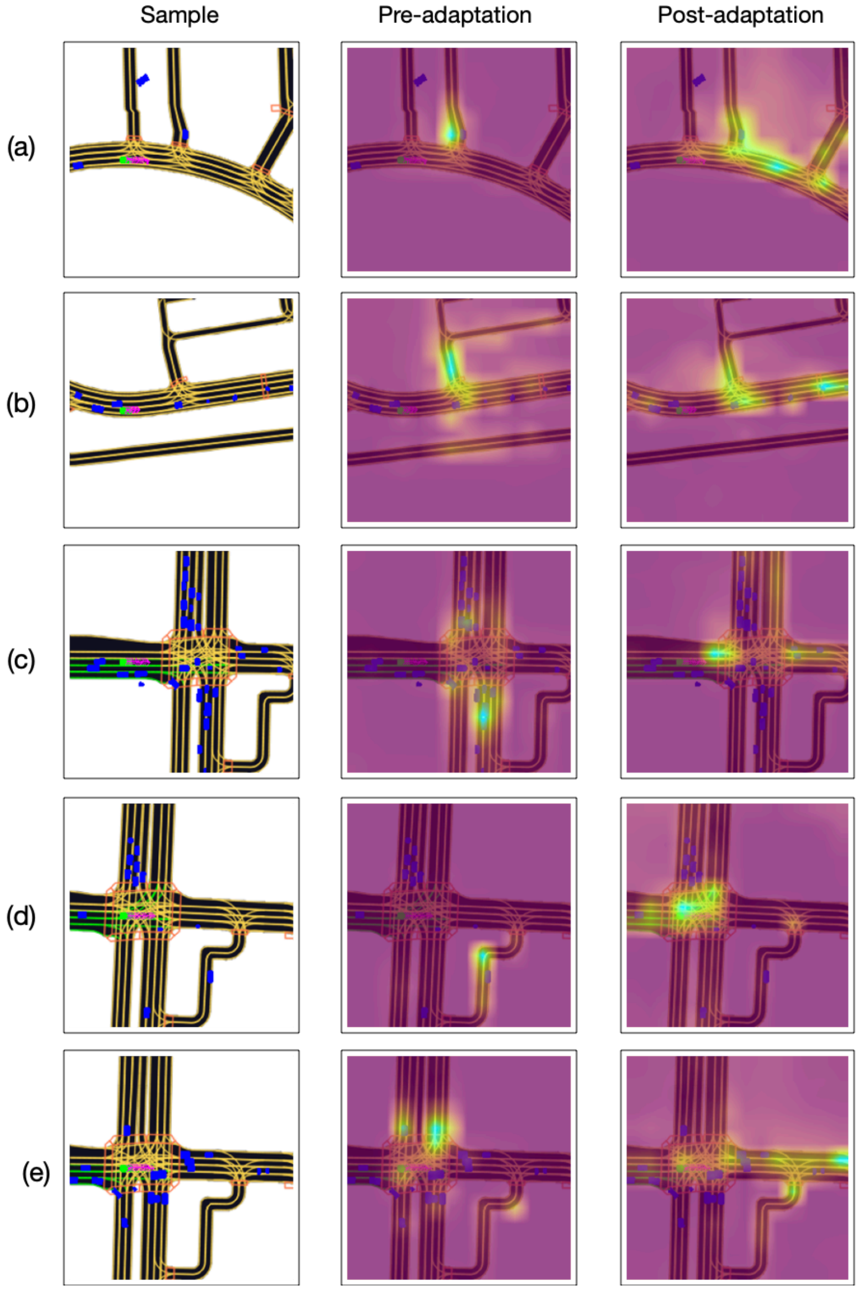


Figure 5.7: Qualitative illustration of the attention heatmaps from the last layer of the ViT before and after adapting the model using MoSA in the unseen domain on Level 5. We can observe that post adaptation, the attention maps are more refined, as the ego vehicle (in green) better focuses on the different possible future routes and the vehicles in front.

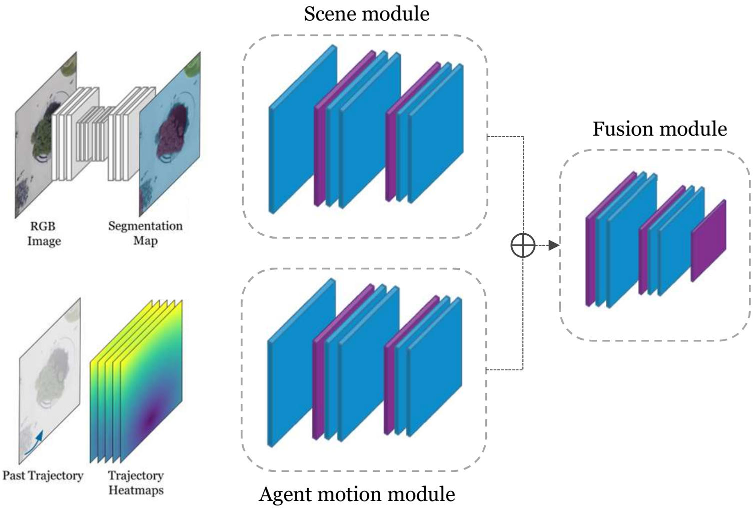


Figure 5.8: Y-Net-Mod encoder architecture

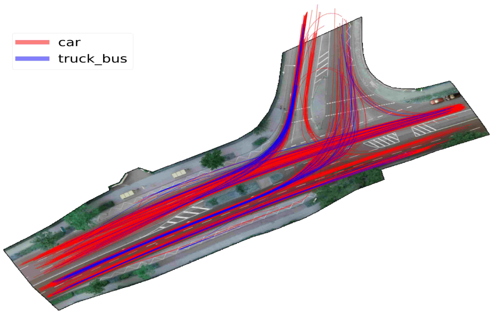


Figure 5.9: Car and truck trajectory distribution in inD dataset *scene1*. Both share the same scene and differ largely in velocity distribution.

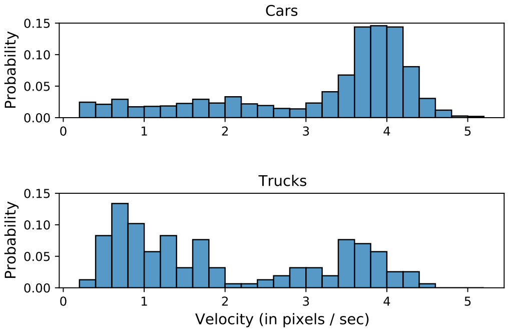


Figure 5.10: Velocity distribution of *car* and *truck* in inD scene1. Static trajectories have been filtered.

differing only in their velocity distribution (see Fig. 5.9 and Fig. 5.10). Fig. 5.11 represents the performance of style transfer from cars to trucks on 20 samples under five different adaptation cases. It is interesting to note that adapting the agent motion encoder alone [A] performs the best while including the scene encoder for adaptation deteriorates performance ([S] worse than [A], [S+A+F] worse than [A+F]).

Scene generalization: We train the Y-Net-Mod model on pedestrian data on scene ids = $\{2,3,4\}$ and adapt it on *scene*1 of inD. Fig. 5.12 represents the performance of scene style transfer on 20 samples in the five cases. Contrary to the previous setup, adapting the scene encoder [S] is clearly more important than the agent motion encoder [A]. Further, adapting the agent encoder deteriorates performance ([S+A+F] worse than [S+F]). It is clear that modularization helps to boost the performance of MoSA.

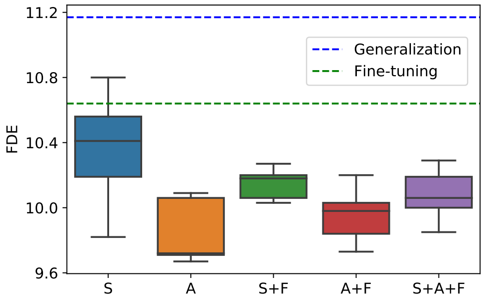


Figure 5.11: Agent Motion Transfer. Style Transfer from *cars* to *trucks* on InD scene1 with $N_{target} = 20$ samples using different MoSA configurations. [A] performs best while [S] worsens performance. Error reported is Top-20 FDE in pixels across 5 seeds.

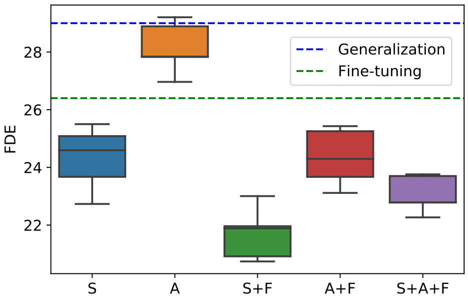


Figure 5.12: Scene Transfer. Style Transfer across scenes on InD *pedestrians* with $N_{target} = 20$ samples using different MoSA configurations.[S+F] performs best while [A] worsens performance. Error reported is Top-20 FDE in pixels across 5 seeds.

5.4.5 Implementation Details

Motion Style Transfer across Agents in SDD. We follow the pre-processing procedure of [20]. We pretrained Y-Net network for 100 epochs with Adam optimizer with a batch size of 10 and learning rate of 5e-5. Rest of the hyper-parameters are kept the same as [20]. We adapted the pretrained model using $N_{target} = \{10, 20, 30\}$ trajectories and utilize 80 trajectories for validation. We adapt the pretrained model for 100 epochs. For MoSA, rank value is set to 3.

Motion Style Transfer across Scenes in inD. We follow the data preprocessing protocol and the pretrained model provided by Y-Net paper [20]. We adapted this model using $N_{target} = \{20\}$ trajectories and used 40 trajectories for validation. We adapted the pretrained model for 100 epochs.

Scene Style Transfer on L5. We follow the training strategy provided in [187]. To simulate low-resource settings during adaptation, the network is shown the unseen route only once, sampled at different rates. We adapt all baselines for 250 epochs using a one-cycle learning rate schedular. We apply MoSA with rank value of 8 and learning rate of 3e-3. We apply MoSA across the query and value matrices of each attention layer. We observed that applying MoSA across the feedforward layers deteriorated performance.

Modular Agent Motion Generalization. We perform style transfer from cars to trucks in *scene1* of inD. Trajectories with an average speed less than 0.2 pixels per second are filtered out. We trained a Y-Net-Mod model on cars. We adapted the model to trucks in which $N_{target} = \{20\}$ trajectories are used for adaptation and 40 trajectories for validation.

Modular Scene Generalization. We pretrained Y-Net-Mod model using pedestrian data from scene ids = $\{2,3,4\}$ and transferred to pedestrian data from *scene1* following the setup in [20]. The adaptation used $N_{target} = \{20\}$ for fine-tuning and 20 trajectories for validation.

5.5 Conclusion

In this chapter, we tackle the task of efficient motion style transfer wherein we adapt a trained forecasting model on a target domain comprising limited samples of unseen target styles. We argued that we only need to model the underlying style shift across domains, which often reside in a low-dimensional space. We formulated this intuition into our motion style adapter (MoSA) design, which is trained to infer and update the style factors of variation in the target domain while keeping the pretrained parameters frozen. Additionally, we present a modularized style transfer strategy that updates only a subset of the model given the nature of the style transfer problem. Extensive experimentation on three real-world datasets demonstrates the effectiveness of our approach.

In the chapters until now, we have tacked the important challenges in improving human trajectory forecasting in crowds. In the last chapter, we present an open-source environment that enables training driving policies on real-world data with the ability to simulate surrounding agents using any data-driven forecasting model.

6 Democratising Reinforcement Learning for Autonomous Driving

This chapter results from an applied research project in collaboration with the autonomous driving company Lyft Level 5 (now, Woven Planet).

This chapter is based on the article:

Parth Kothari, Christian Perone, Luca Bergamini, Alexandre Alahi and Peter Ondruska, *Drivergym: Democratising reinforcement learning for autonomous driving*, Machine Learning for Autonomous Driving, NeurIPS 2021 Workshop

6.1 Introduction

In the previous chapters, we focused on designing deep learning architectures for improving human trajectory forecasting. The eventual goal of developing such accurate socially-aware trajectory forecasting methods is to utilize them in improving downstream tasks like planning in autonomous robots. In this chapter, we present an open-source gym environment DriverGym that enables to train driving policies on real-world data with the additional benefit that the surrounding agents can be simulated using trajectory forecasting methods.

Recently, Reinforcement Learning (RL) has achieved great success in a variety of applications like playing Atari games [213], board games [214], manipulating sensorimotor in three-dimensions [215]. However, developing RL algorithms for real-world applications such as autonomous driving (AD) remains an open challenge [216]: with AD being an extremely safety-critical task, one cannot directly deploy a policy in the real world for data collection or policy validation.

One solution is to deploy the policy in the real world with a safety driver inside the car at all times. However, this process is time-consuming, and more importantly, not accessible to all of the research community. Therefore, to tackle this challenge, there is a dire need for an RL simulation environment that can (1) be used to easily train RL policies using real-world logs, (2) simulate

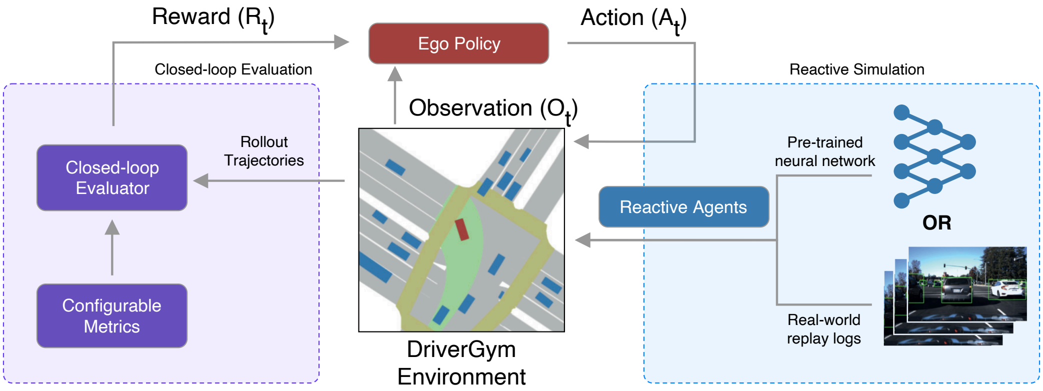


Figure 6.1: DriverGym: an open-source gym environment that enables training RL driving policies on real-world data. The RL policy can access rich semantic maps to control the ego (red). Other agents (blue) can either be simulated from the data logs or controlled using a dedicated policy pre-trained on real-world data. We provide an extensible evaluation system (purple) with easily configurable metrics to evaluate the idiosyncrasies of the trained policies.

surrounding agent behavior that is both realistic and reactive to the ego policy, (3) effectively evaluate the trained models, (4) be flexible in its design, and (5) inclusive to the entire research community.

We propose DriverGym, an open-source gym-compatible environment specifically tailored for developing and experimenting with RL algorithms for self-driving (see Fig. 6.1). DriverGym utilizes one of the largest public self-driving datasets, *Level 5 Prediction Dataset* [217] containing over 1,000 hours of data, and provides support for reactive agent behavior simulation [218] using data-driven models. Furthermore, DriverGym provides an extensive and extensible closed-loop evaluation system: it not only comprises a variety of AD-specific metrics but also can be easily extended to incorporate new metrics to evaluate idiosyncrasies of trained policies. We open-source the code and pre-trained models to stimulate development.

In this work, we provide the following contributions:

- An open-source and OpenAI gym-compatible environment for autonomous driving task;
- Support for more than 1000 hours of real-world expert data;
- Support for logged agents replay or data-driven realistic agent trajectory simulations;
- Configurable and extensible evaluation protocol;

6.2 Related Work

Autonomous Driving Simulation Environments. To replicate the success of the OpenAI Gym framework [219], many simulation environments have been developed in the context of autonomous driving [33, 34, 220, 221, 222]. Table 6.1 provides a comparison amongst commonly

Table 6.1: Comparison of various open-source RL simulation environments for autonomous driving.

Name	Gym Compatible	Evaluation Protocol	Simulator Expert Data	Real-world Expert Data	Agents Model
TORCS	✗	✗	✗	✗	Rule-based
Highway-Env	✓	✗	✗	✗	Rule-based
CARLA (Official)	✗	✓	14 hrs	✗	Rule-based
SMARTS	✓	✓	✗	✗	Rule-based *or* Data-driven
CRTS	✓	✓	✗	64 hrs	Real-world Logs
DriverGym	✓	✓	✗	1000 hrs	Data-driven *or* Real-world Logs

used RL simulation environments including DriverGym. Racing simulators like TORCS [220] offer limited scenarios of driving. Highway-Env [221] provides a collection of gym-compatible environments for autonomous driving. However, it lacks important semantic elements like traffic lights, an extensive evaluation protocol and expert data.

Traffic simulators like CARLA [33], SUMO [34] supports flexible specification of traffic conditions for training and testing. However, they are synthetic simulators that utilize hand-coded rules for surrounding agents' motion that tends to be unrealistic and display a limited variety of behaviors. Crucially, they lack access to real-world data logs. SMARTS [223] overcomes the former issue by providing *Social Agent Zoo* that supports data-driven agent models while CRTS [224] tackles the latter providing access to 64 hours of real-world logs within the CARLA simulator. DriverGym solves both these challenges: it provides access to 1000 hours of real-world logs to initialize episodes and more importantly, enables simulating reactive agents using data-driven models learned from real-world data. This allows us to utilize deep trajectory forecasting methods to facilitate downstream planning.

6.3 Method

DriverGym aims to foster the development of RL policies for self-driving by providing a flexible interface to train and evaluate RL policies. Our environment is compatible with both SB3 [225] and RLlib [226], two popular frameworks for training RL policies. The code is open-source with Apache 2 license. We describe the components of our environment below.

6.3.1 State Representation

The state representation captures the context around the ego agent, in particular, the surrounding agents' positions, their velocities, the lanes and traffic lights. We encode this information in the form of a 3D tensor that is the birds-eye-view (BEV) raster image of the current frame. DriverGym supports all the rasterization modes provided by L5Kit [217]. Compared to Atari environments, DriverGym requires more time to generate observations as the latter has to load

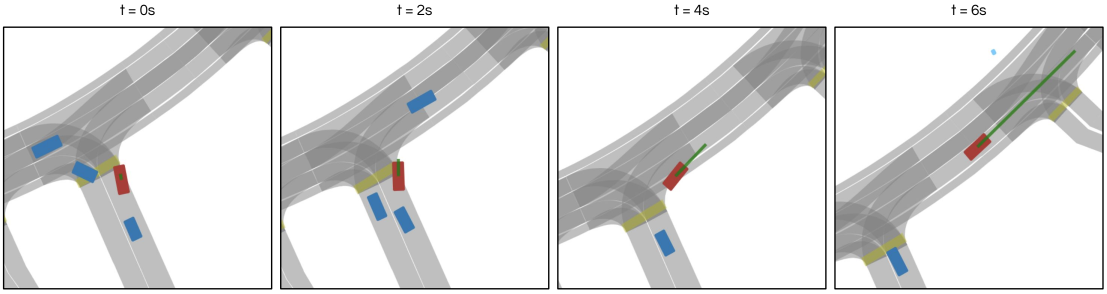


Figure 6.2: Visualization of an episode rollout (ego in red, agents in blue) in DriverGym. The policy prediction (green line) is scaled by factor of 10 and shown at 2 second intervals for better viewing.

real-world data and subsequently render high-resolution raster images (See Fig. 6.3).

6.3.2 Action Spaces

The action produced by the RL policy is used to control the motion of the ego agent. The action is propagated as (x, y, yaw) to update the state of the ego. Still, DriverGym does not make any strict assumptions on the policy itself which can, for instance, output $(acceleration, steer)$ and use a kinematic model to decode the next-step observation.

6.3.3 Reactive Agents

An important component of the DriverGym environment is to model the motion of the surrounding agents. DriverGym allows flexibility in this aspect and currently support two ways of controlling the behavior of surrounding agents: *log replay* and *reactive simulation*.

In *log replay*, during an episode rollout, the movement of surrounding agents around the ego is replayed in the exact same manner as it happened when the log was collected in the real world. In *reactive simulation*, the agent behavior is both reactive and realistic. Motivated by [218], DriverGym allows simulating agent reactivity using data-driven models that learn agent behavior from real-world data, *i.e.,* users can provide neural-network-based agent models trained on real-world data, to simulate agent behavior.

6.3.4 Rewards

The rewards in the environment quantify the performance of a driving policy during a rollout and subsequently guide the training of the policy using reinforcement learning. DriverGym, through the Closed-Loop Evaluation (CLE), supports a variety of AD-specific metrics that are computed per-frame, and can be combined to construct the reward function. This system is described in the section below.

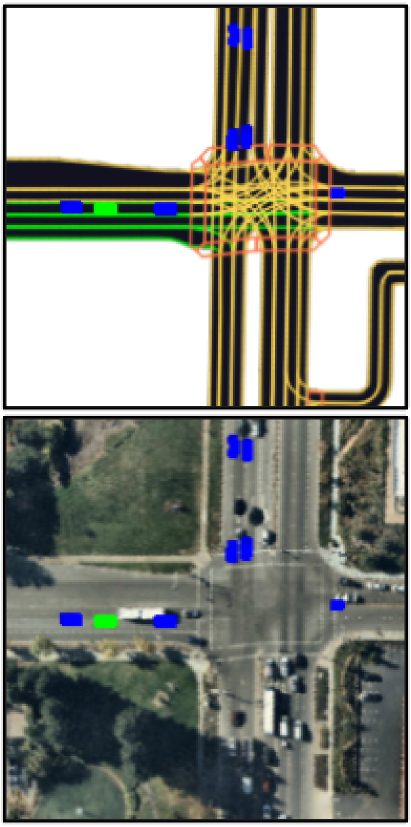


Figure 6.3: Example Rasterization Modes

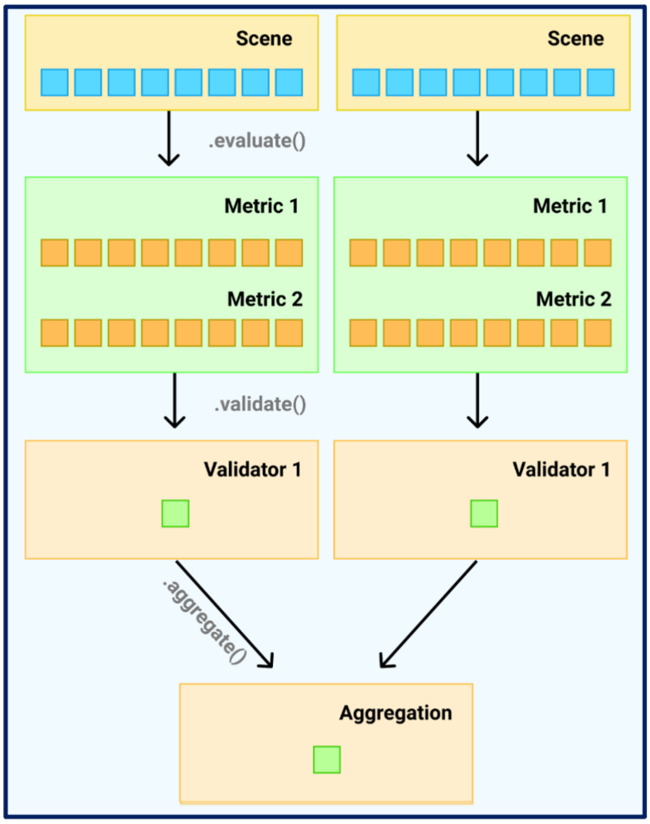


Figure 6.4: Evaluation Plan

6.3.5 Closed-Loop Evaluation Protocol

Having an extensive closed-loop evaluation (CLE) protocol is a necessity to correctly assess the performance of RL policies before deployment in safety-critical real-world scenarios. Our CLE framework comprises insightful AD-specific metrics: the first set of metrics, specific to imitation learning, are distance-based metrics. The second set of metrics, specific to safety, capture the various types of collisions that occur between ego and surrounding agents. These include front collision, side collision and rear collisions. Table. 6.2 describes the various metrics.

CLE is flexible and new metrics can be easily incorporated to target specific cases of model failures. Within CLE, the user has access to all the simulation artifacts (trajectories, maps, *log replay* data of ego and agents) while designing a new metric. We hope the DriverGym evaluation protocol facilitates researchers to diagnose targeted behaviors of their policies.

Implementation. Our CLE works on top of the simulation outputs provided by episode rollouts in DriverGym. An *evaluation plan* (Fig. 6.4) is defined that comprises (1) metrics that are computed per frame (e.g. L2 displacement error) (2) validators that enforce constraints on the metrics per scene (L2 displacement error $\leq$ 4 meters), and (3) composite metrics per scene, that

Table 6.2: Description of the various metrics provided in DriverGym closed-loop evaluation protocol.

Name	Type	Description
Average Displacement	Distance-Based	Computes the L2 distance between predicted ego centroid and ground-truth ego centroid averaged over the entire episode
Final Displacement	Distance-Based	Computes the L2 distance between predicted ego centroid and ground-truth ego centroid at the last timestep of the episode
Distance to Reference	Distance-Based	Computes the L2 distance between the predicted centroid and the closest waypoint in the reference trajectory (ground-truth ego)
Front Collision	Safety-Based	Computes whether a collision occurred between the front of ego and any another agent
Side Collision	Safety-Based	Computes whether a collision occurred between a side of ego and any another agent
Rear Collision	Safety-Based	Computes whether a collision occurred between the rear of ego and any another agent

can depend both on the output of metrics and validators (e.g. passed driven miles). Note that our CLE supports both reactive and non-reactive agents.

6.4 Experiments

We evaluate three different algorithms using DriverGym to compare the effectiveness of these training strategies. The first one is an open-loop training baseline using L2 imitation loss (**SL**). Naive behavioral cloning is known to suffer from distribution shift between training and test data [227]. We compare it with a stronger baseline, inspired by ChaufferNet [228], that alleviates distribution shift by introducing synthetic perturbations to the training trajectories (**SL + P**).

We also evaluate an RL policy, namely Proximal Policy Optimization (PPO) [229] implemented in the SB3 framework [225]. We choose PPO as it not only demonstrates remarkable performance but it is also empirically easy to tune [229]. All the experiments have been performed on 2 Tesla T4 GPUs.

6.4.1 Model Architecture

In our experiments, the backbone feature extractor is shared between the policy and the value networks. The feature extractor is composed of two convolutional networks followed by a fully connected layer, with ReLU activation. The feature extractor output is passed to both the policy and value networks composed of two fully connected layers with tanh activation. The open-loop baseline models have the same backbone architecture as above.

We perform group normalization after every convolutional layer. Empirically, we found that

Table 6.3: Evaluation of different training strategies using the CLE protocol in DriverGym. Lower is better. **SL**: Supervised learning using L2 imitation loss, **SL + P**: SL plus trajectory pertubations, **PPO**: RL using PPO, with imitation-loss based reward. Metrics and validators are in the format: average (std. deviation).

Method	**Metrics**		**Validators**				
	Average Displacement	Final Displacement	Final Displacement (>= 30.0m)	Distance To Reference (>= 4.0m)	Front Collision	Side Collision	Rear Collision
SL	32.4 ± 2.7	74.5 ± 5.5	9.5 ± 4.9	26 ± 0.0	12.5 ± 3.5	19 ± 5.6	16 ± 1.4
SL + P	**13.4** ± **1.4**	**25.5** ± **3.5**	5.3 ± 0.6	**9.7** ± **1.5**	9 ± 2.6	12.3 ± 6.8	**7** ± **0.0**
PPO	18.7 ± 2.3	46.4 ± 7.7	**4.0** ± **2.0**	12.7 ± 3.2	**4.3** ± **2.5**	**6** ± **3.5**	27 ± 5.0

group normalization performs far superior to batch normalization. This can be attributed to the fact that activation statistics change quickly in on-policy algorithms (PPO is on-policy) while batch-norm learnable parameters can be slow to update causing training issues.

6.4.2 Training

The training data comprises 100 scenes (average frame length $\sim$ 248) where the initial frame is randomly sampled. The open-loop baseline models are trained in a supervised manner where the L2 loss is calculated on the predictions of 12 future time-steps (1.2 secs).

We train the PPO policy in closed-loop (the surrounding agents are *log replayed*) for episodes of length 32 time-steps. PPO policy network predicts the mean and standard deviation values of a gaussian to represent its actions. Further, the policy is initialized such that the initial actions are independent of the observations. Therefore, we normalize the action space (zero mean) for faster training convergence. For further training stability, we incorporate a unicycle kinematic model at the policy output, *i.e.* the policy predicts the acceleration and steer.

For the PPO policy, we use an imitation loss-based reward. We define the reward as the negative of the L2 distance between the policy prediction and ego replay at every time-step. Reward clipping is performed for stability. Note that, DriverGym can also incorporate non-differentiable hand-crafted rules like collisions in the reward function to train different RL policies.

6.4.3 Hyperparameters

We train the PPO policy for $12M$ steps in which the learning rate is fixed to 3e$-$4 for the first $8M$ steps and then decreased by a factor of 10 for the rest of the training. The discount factor is 0.80 and GAE is 0.90. 4 environments are rolled out in parallel for a total of 1024 time-steps before the model is updated for 10 epochs on the collected rollout buffer. The mini-batch size of the model update is 64 and the clipping parameter ε follows a linear decay schedule during training starting from 0.1. The reward clipping threshold is fixed to 15.

We train on 112 × 112 pixel BEV rasters centered around the ego. The raster image is generated by combining the semantic map (3 channels) and the bounding boxes of the various agents in the scene (top image in Fig 6.3). The past history and current bounding boxes of the agents (including ego) are incorporated via the channel dimension. We consider a history of 3 frames which along with the current frame leads to an additional 8 channels resulting in a raster image of size $112 \times 112 \times 11$. The raster image is transformed such that it is centered around the ego vehicle.

6.4.4 Results

The performance of three runs (different seeds) of the three models on 100 real-world test scenes is reported in Table 6.3. Based on distance-based metrics, **PPO** is similar to **SL + P** in terms of ADE, however it suffers from high FDE. **PPO** showed fewer front and side collisions, however, it showed a much higher number of rear collisions, which can be explained by the passiveness of the ego vehicle. Finally, **SL** is the worst and corroborates the expectations. We would like to point out that our goal is to provide a user-friendly environment for training RL policies and not to develop a state-of-the-art RL policy.

6.4.5 Visualizations

The DriverGym environment provides the user with the ability to visualize the output of episode rollouts (see example in Fig 6.2). The visualization is carried out using the Bokeh interaction visualization library [230].

6.5 Conclusion

We believe DriverGym is an important step towards solving the task of planning for autonomous driving. Thanks to its gym-compatible interface, it allows to easily train and evaluate RL policies for self-driving. Furthermore, surrounding agents can be controlled via a model trained on real-world data to improve their reactivity towards the ego. We hope that DriverGym will provide a common ground for training RL policies on autonomous applications, and consequently drive the improvement of the next generation of planning algorithms.

7 Concluding Remarks

This thesis examines and presents solutions for improving the task of human trajectory forecasting in crowds. In particular, we tackle four key challenges inherent to the real-world application of motion forecasting: (1) learning social interactions, (2) modelling multiple futures, (3) incorporating interpretability, and (4) sample-efficient adaptation of pre-trained models. We summarize our findings, and propose possible future directions of research.

7.1 Findings

Chapter 2 presented an in-depth analysis of existing interaction encoders and introduced our proposed interaction module and a training strategy to train socially-aware forecasting models. Despite claims in literature that specific interaction encoders better learn interactions, we observed that under *identical* conditions, all modules perform similar in terms of the distance-based metrics. The incorporation of collision metric paints a more meaningful picture of model performance. Our proposed encoder *D-Grid* outperformed competitive interaction encoders on real-world datasets, lowering the collision rate by 20%.

Chapter 3 presented a GAN-based architecture *SGANv2* equipped with spatio-temporal interaction modelling and a transformer-based discriminator. Additionally, we utilized the learned discriminator even at test-time via a collaborative sampling strategy. Our discussion on multimodal evaluation demonstrated that high-entropy predictors can easily cheat the commonly-used Top-20 metric. SGANv2 reduced the collision rates by 30% on real-world datasets. Additionally, we demonstrated that SGANv2 can prevent mode collapse on the challenging Forking Paths dataset. Ablations on synthetic datasets demonstrated the importance of spatio-temporal interaction modelling with significantly reduced the collision rates.

Chapter 4 described our interpretable *social anchors* framework for human trajectory forecasting. We leveraged the power of discrete choice models (DCMs) to learn interpretable rule-based intents, and subsequently utilise the expressibility of neural networks to model scene-specific residual. Quantitatively, we demonstrated that the neural network component improved the prediction accuracy of DCMs by $\sim 15\%$. Qualitatively, the predictions of the social anchors were explained by analyzing the underlying components. We further highlighted the potential of directional normalization to reduce the collision rates of forecasting models.

Chapter 5 presented our methodology to efficiently adapt a pre-trained forecasting model to unseen agent types and locations. We introduced two components that exploit our prior knowledge of motion style shifts: (i) a low-rank motion style adapter that projects and adjusts the style features at a low-dimensional bottleneck; (ii) a modular adapter strategy that facilitated a fine-grained choice of adaptation layers. Experimental results show that our method improves the performance of modern forecasting models by 25% using only 30 samples from the target domain on Stanford Drone and Intersection Drone dataset. Moreover, while updating less than 5% extra parameters, our method achieves 20% relative improvement compared to fully fine-tuned models on Lyft Level 5 dataset.

Chapter 6 presented our gym-compatible environment DriverGym, tailored for developing planning algorithms for autonomous driving. It provides the ability to simulate reactive agents using deep forecasting models.

7.2 Future Directions of Research

There still exist many open problems which can serve as interesting and important future research directions.

Towards collision-free trajectory forecasting. Neural networks have outperformed classical models on distance-based metrics. In Chapter 2, through our training strategy and interaction encoder design, we reduced the collision rate of the predictions. We believe that other works should also focus on this metric and develop data-driven methods that aim for zero collisions. Indeed, there have been works trying to realize this objective. Social NCE [231] built upon our training strategy and introduced contrastive learning to make the model more socially-aware. Concurrently, unlikelihood training [232] has been proposed to reduce the likelihood of unsafe predictions from the model. Developing improved training strategies and interaction encoder designs for collision-free predictions is an interesting direction of future work.

Multimodal Evaluation. In Chapter 3, we highlighted the issue with measuring the performance of multimodal forecasting models using only the Top-20 metric [17]. Concurrently, Social-Implicit [233] raised this issue and further highlighted sensitivity of the NLL metric [87] to the choice of the kernel. They proposed the Average Mahalanobis Distance (AMD) metric to measure the likelihood and Average Maximum Eigenvalue (AMV) metric to capture the variance. In addition to likelihood, we showed that it is important to measure the scene-consistency of predictions via the average collision metric. When labels for multiple plausible ground truths are available [136], the Earth Mover distance [19] and Percentage of Trajectory Usage [234] are promising metrics to capture the accuracy of the outputs. Yet, objective evaluation of multimodal frameworks on real datasets based on a single available ground-truth remains an open question.

Social Anchors based on concept neural networks. In Chapter 4, the social anchors framework incorporated the expert-based DCM utility specifications (without any changes to their parametric form) with the deep learning framework. We obtained interpretability by analyzing the contribution of each component in the choice outcome. The model would still preserve interpretability if the hand-crafted expert functions were replaced by arbitrary, continuous and differentiable functions. Therefore, an interesting extension can be to relax the assumptions on the specific parametric form of various concepts (like leader follower) suggested by DCM practitioners with neural networks. In particular, the neural networks can be used as universal function approximators for various social concepts such as leader follower, collision avoidance and grouping.

Developing automatic layer selection and training strategies for efficient modular adaptation. Chapter 5 demonstrated the effectiveness of a modular adaptation strategy, given the type of the underlying style shifts (agent motion transfer or scene transfer). Such high-level domain knowledge about the difference from one agent/environment to another is often widely available in practice. However, an interesting problem statement would be to automatically determine which modules and the layers within each module that need to be adapted, without the knowledge of underlying style shift. One strategy could be to measure the distribution shifts in the feature space per layer and selecting the ones that vary the most.

To improve the efficiency of modular adaptation, it is also interesting to develop training strategies that focus on constraining the representations learned by each encoder module. For instance, in our scene style transfer experiment, currently adapting both scene and fusion modules performs better than adapting only the scene module. A training algorithm that enforces the weights of fusion encoder to be constant across different training scenes, can potentially improve the adaptation performance of scene module.

Bringing realism of motion forecasting into planning. In Chapter 6, we present an open-source gym environment DriverGym that provides the additional benefit that the surrounding agents can be simulated using deep trajectory forecasting methods. Currently, we do not investigate the impact of data-driven forecasting models on the ego policy. As a future work, it can be interesting to study the impact of this reactivity and realism of neighbours in ego planning on DriverGym.

Improved test-time training for motion forecasting and planning. Our recent work [235] has shown the promise of robustly adapting deep classification models to new domains based on unlabelled test examples on the fly. However, applying this paradigm to motion forecasting or planning is non-trivial due to the regression nature of motion problems, the non-stationary environment changes induced by autonomous agents, label distribution shifts under the optimal policy, and many others. It can be interesting to explore solutions towards more adaptive motion forecasting and planning models in the future.

In conclusion, there exist several interesting avenues to expand the capabilities of human trajectory forecasting models and make them suitable for real-world deployment. More generally, we hope future research will continue to emphasize on tackling interaction-centric situations so that the quality of trajectory forecasting models can keep increasing, allowing us to tackle more challenging scenarios.

Bibliography

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[3] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 91–99.

[4] W. Liu, Dragomir Anguelov, D. Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. “SSD: Single Shot MultiBox Detector”. In: *ECCV*. 2016.

[5] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. “Objects as Points”. In: *ArXiv* abs/1904.07850 (2019).

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. “Mask R-CNN”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42 (2020), pp. 386–397.

[7] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 9992–10002.

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *ArXiv* abs/1505.04597 (2015).

[9] Xiaoxiang Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liang-pei Zhang, Feng Xu, and Friedrich Fraundorfer. “Deep Learning in Remote Sensing: A Comprehensive Review and List of Resources”. In: *IEEE Geoscience and Remote Sensing Magazine* 5 (2017), pp. 8–36.

[10] G. Camps-Valls, Devis Tuia, Lorenzo Bruzzone, and Jón Atli Benediktsson. “Advances in Hyperspectral Image Classification: Earth Monitoring with Statistical Learning Methods”. In: *IEEE Signal Processing Magazine* 31 (2014), pp. 45–54.

[11] Matthias Kässer Daniel Holland-Letz Benedikt Kloss and Thibaut Müllre. “Start me up: Where mobility investments are going.” In: *Tech. rep. McKinsey and Company,* (2019).

[12] The Nuro Team. “Safety @ Nuro: Our Autonomy Software”. In: *Medium* (2022). URL: https://medium.com/nuro/safety-nuro-our-autonomy-software-ad35a4f6e524.

[13] 2021. California Dept. of Motor Vehicles. “DMV approves cruise and waymo to use autonomous vehicles for commercial service in designated parts of bay area”. In: (2021). URL: https://www.dmv.ca.gov/portal/news-and-media/117199-2/.

[14] K. Habib. “PE 16-007: Tesla crash preliminary evaluation report”. In: *National Highway Traffic Safety Administration, Tech. Rep.* (2017).

[15] “Highway accident report: Collision between vehicle controlled by developmental automated driving system and pedestrian”. In: *National Transportation Safety Board, Tech. Rep.* (2018).

[16] 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), pp. 961–971.

[17] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. “Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 2255–2264.

[18] Anirudh Vemula, Katharina Muelling, and Jean Oh. “Social Attention: Modeling Attention in Human Crowds”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2017), pp. 1–7.

[19] Osama Makansi, Eddy Ilg, Özgün Çiçek, and Thomas Brox. “Overcoming Limitations of Mixture Density Networks: A Sampling and Fitting Framework for Multimodal Future Prediction”. In: *ArXiv* abs/1906.03631 (2019).

[20] Karttikeya Mangalam, Yang An, Harshayu Girase, and Jitendra Malik. “From Goals, Waypoints & Paths To Long Term Human Trajectory Forecasting”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 15213–15222.

[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. “ImageNet: A Large-Scale Hierarchical Image Database”. In: *CVPR09*. 2009.

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[23] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: *BlackboxNLP@EMNLP*. 2018.

[24] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. “Generative Adversarial Nets”. In: *NIPS*. 2014.

[25] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian D. Reid, Seyed Hamid Rezatofighi, and Silvio Savarese. “Social-BiGAT: Multimodal Trajectory Forecasting using Bicycle-GAN and Graph Attention Networks”. In: *ArXiv* abs/1907.03395 (2019).

[26] Javad Amirian, Jean-Bernard Hayet, and Julien Pettré. “Social Ways: Learning Multi-Modal Distributions of Pedestrian Trajectories with GANs”. In: *ArXiv* abs/1904.09507 (2019).

[27] Gianluca Antonini, Michel Bierlaire, and Mats Weber. “Discrete Choice Models for Pedestrian Walking Behavior”. In: *Transportation Research Part B: Methodological* 40 (Sept. 2006), pp. 667–687. DOI: 10.1016/j.trb.2005.09.006.

[28] Dirk Helbing and Peter Molnar. “Social Force Model for Pedestrian Dynamics”. In: *Physical Review E* 51 (May 1998). DOI: 10.1103/PhysRevE.51.4282.

[29] Letian Wang, Yeping Hu, Liting Sun, Wei Zhan, Masayoshi Tomizuka, and Changliu Liu. “Transferable and Adaptable Driving Behavior Prediction”. In: *arXiv preprint arXiv:2202.05140* (2022).

[30] Yi Xu, Lichen Wang, Yizhou Wang, and Yun Fu. “Adaptive Trajectory Prediction via Transferable GNN”. In: *arXiv preprint arXiv:2203.05046* (2022).

[31] Heejin Ahn, Colin Chen, Ian M. Mitchell, and Maryam Kamgarpour. “Safe Motion Planning Against Multimodal Distributions Based on a Scenario Approach”. In: *IEEE Control Systems Letters* 6 (2021), pp. 1142–1147.

[32] Kai Ren, Heejin Ahn, and Maryam Kamgarpour. “Chance-Constrained Trajectory Planning With Multimodal Environmental Uncertainty”. In: *IEEE Control Systems Letters* 7 (2022), pp. 13–18.

[33] A. Dosovitskiy, G. Ros, Felipe Codevilla, Antonio M. López, and V. Koltun. “CARLA: An Open Urban Driving Simulator”. In: *ArXiv* abs/1711.03938 (2017).

[34] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. “Microscopic Traffic Simulation using SUMO”. In: *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. URL: https://elib.dlr.de/124092/.

[35] Parth Kothari, Sven Kreiss, and Alexandre Alahi. “Human Trajectory Forecasting in Crowds: A Deep Learning Perspective”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.7 (2022), pp. 7386–7400. DOI: 10.1109/TITS.2021.3069362.

[36] Yuejiang Liu, Parth Kothari, and Alexandre Alahi. “Collaborative Sampling in Generative Adversarial Networks”. In: *AAAI*. 2020.

[37] Parth Kothari, Brian Sifringer, and Alexandre Alahi. “Interpretable Social Anchors for Human Trajectory Forecasting in Crowds”. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 15551–15561. DOI: 10.1109/CVPR46437.2021.01530.

[38] Parth Kothari, Danya Li, Yuejiang Liu, and Alexandre Alahi. “Motion Style Transfer: Modular Low-Rank Adaptation for Deep Motion Forecasting”. In: *6th Annual Conference on Robot Learning*. 2022. URL: https://openreview.net/forum?id=tVgD4METs6o.

[39] Parth Kothari, Christian Samuel Perone, Luca Bergamini, Alexandre Alahi, and Peter Ondruska. “DriverGym: Democratising Reinforcement Learning for Autonomous Driving”. In: *ArXiv* abs/2111.06889 (2021).

[40] Bin Jiang. “SimPed: simulating pedestrian flows in a virtual urban environment”. In: 1999.

[41] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. “Crowds by Example”. In: *Comput. Graph. Forum* 26 (2007), pp. 655–664.

[42] Stephen Bitgood. “An Analysis of Visitor Circulation: Movement Patterns and the General Value Principle”. In: *Curator: The Museum Journal* 49 (2006), pp. 463–475.

[43] Andreas Horni, Kai Nagel, and Kay W. Axhausen. “The Multi-Agent Transport Simulation MATSim”. In: 2016.

[44] Ramin Mehran, Alexis Oyama, and M. A. Hadi Shah. “Abnormal crowd behavior detection using social force model”. In: *CVPR*. 2009.

[45] Dirk Helbing, Illés J. Farkas, Peter Molnar, and Tamás Vicsek. “Simulation of pedestrian crowds in normal and evacuation situations”. In: 2002.

[46] Dirk Helbing, Illés J. Farkas, and Tamás Vicsek. “Simulating dynamical features of escape panic”. In: *Nature* 407 (2000), pp. 487–490.

[47] Xiaoping Zheng, Tingkuan Zhong, and Mengting Liu. “Modeling crowd evacuation of a building based on seven methodological approaches”. In: 2009.

[48] Mehdi Moussaïd, Dirk Helbing, and Guy Theraulaz. “How simple rules determine pedestrian behavior and crowd disasters”. In: *Proceedings of the National Academy of Sciences* 108 (2011), pp. 6884–6888.

[49] Hairong Dong, Min Zhou, Qianling Wang, Xiaoxia Yang, and Feiyue Wang. “State-of-the-Art Pedestrian and Evacuation Dynamics”. In: *IEEE Transactions on Intelligent Transportation Systems* 21 (2020), pp. 1849–1866.

[50] “https://uber.app.box.com/v/UberATGSafetyReport”. In.

[51] Changan Chen, Yuejiang Liu, Sven Kreiss, and Alexandre Alahi. “Crowd-Robot Interaction: Crowd-Aware Robot Navigation With Attention-Based Deep Reinforcement Learning”. In: *2019 International Conference on Robotics and Automation (ICRA)* (2019), pp. 6015–6022.

[52] Amir Rasouli and John K. Tsotsos. “Autonomous Vehicles That Interact With Pedestrians: A Survey of Theory and Practice”. In: *IEEE Transactions on Intelligent Transportation Systems* 21 (2020), pp. 900–918.

[53] Helbing and Molnár. “Social force model for pedestrian dynamics.” In: *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics* 51 5 (1995), pp. 4282–4286.

[54] S. Bach, Alexander Binder, Grégoire Montavon, F. Klauschen, K. Müller, and W. Samek. “On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation”. In: *PLoS ONE* 10 (2015).

[55] L. F. Henderson. In: *Transp. Res. 8*. 1974.

[56] D. Helbing. “An improved fluid-dynamic model for vehicular traffic”. In: *Phys. Rev. E.*

[57] J. M. Molera J. A. Cuesta F. C. Martinez and A. Sanchez. In: *Phys. Rev. E. 48, 4175*. 1993.

[58] Carsten Burstedde, Kai Klauck, Andreas Schadschneider, and Johannes Zittartz. “Simulation of pedestrian dynamics using a two-dimensional cellular automaton”. In: 2001.

[59] Adrien Treuille, Seth Cooper, and Zoran Popovi263. “Continuum crowds”. In: *SIGGRAPH ’06*. 2006.

[60] Jur P. van den Berg, Ming C. Lin, and Dinesh Manocha. “Reciprocal Velocity Obstacles for real-time multi-agent navigation”. In: *2008 IEEE International Conference on Robotics and Automation* (2008), pp. 1928–1935.

[61] Dirk Helbing. “Models for Pedestrian Behavior”. In: *arXiv: Statistical Mechanics* (1998).

[62] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. “You’ll never walk alone: Modeling social behavior for multi-target tracking”. In: *2009 IEEE 12th International Conference on Computer Vision* (2009), pp. 261–268.

[63] Peter Trautman, Jeremy Ma, Richard M. Murray, and Andreas Krause. “Robot navigation in dense human crowds: the case for cooperation”. In: *2013 IEEE International Conference on Robotics and Automation* (2013), pp. 2153–2160.

[64] Shuai Yi, Hongsheng Li, and Xiaogang Wang. “Understanding pedestrian behaviors from stationary crowd groups”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 3488–3496.

[65] Alexandre Alahi, Vignesh Ramanathan, and Li Fei-Fei. “Socially-Aware Large-Scale Crowd Forecasting”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 2211–2218.

[66] Hyun Soo Park and Jianbo Shi. “Social saliency prediction”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 4777–4785.

[67] Alex Graves. “Generating Sequences With Recurrent Neural Networks”. In: *CoRR* abs/1308.0850 (2013).

[68] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Curran Associates, Inc., 2014, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf (visited on 08/16/2018).

[69] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. "Speech recognition with deep recurrent neural networks". In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (2013), pp. 6645–6649.

[70] Fandong Meng. "Neural Machine Translation by Jointly Learning to Align and Translate". In: 2014.

[71] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. "Show and tell: A neural image caption generator". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 3156–3164.

[72] Chunshui Cao et al. "Look and Think Twice: Capturing Top-Down Visual Attention with Feedback Convolutional Neural Networks". In: *2015 IEEE International Conference on Computer Vision (ICCV)* (2015), pp. 2956–2964.

[73] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9 (1997), pp. 1735–1780.

[74] Pasquale Coscia, Francesco Castaldo, Francesco AN Palmieri, Lamberto Ballan, Alexandre Alahi, and Silvio Savarese. "Point-based path prediction from polar histograms". In: *2016 19th International Conference on Information Fusion (FUSION)*. IEEE. 2016, pp. 1961–1967.

[75] Daksh Varshneya and G. Srinivasaraghavan. "Human Trajectory Prediction using Spatially aware Deep Attention Models". In: *ArXiv* abs/1705.09436 (2017).

[76] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher Bongsoo Choy, Philip H. S. Torr, and Manmohan Krishna Chandraker. "DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 2165–2174.

[77] Federico Bartoli, Giuseppe Lisanti, Lamberto Ballan, and Alberto Del Bimbo. "Context-Aware Trajectory Prediction". In: *2018 24th International Conference on Pattern Recognition (ICPR)* (2018), pp. 1941–1946.

[78] Hao Xue, Du Q. Huynh, and Mark Reynolds. "SS-LSTM: A Hierarchical LSTM Model for Pedestrian Trajectory Prediction". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 1186–1194.

[79] Tianyang Zhao, Yifei Xu, Mathew Monfort, Wongun Choi, Chris Baker, Yibiao Zhao, Yizhou Wang, and Ying Nian Wu. "Multi-Agent Tensor Fusion for Contextual Trajectory Prediction". In: *CVPR*. 2019.

[80] Matteo Lisotto, Pasquale Coscia, and Lamberto Ballan. "Social and Scene-Aware Trajectory Prediction in Crowded Spaces". In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (2019), pp. 2567–2574.

[81] Mark Pfeiffer, Giuseppe Paolo, Hannes Sommer, Juan I. Nieto, Roland Siegwart, and Cesar Cadena. "A Data-driven Model for Interaction-Aware Pedestrian Motion Prediction in Object Cluttered Environments". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2017), pp. 1–8.

[82] Alexandre Alahi, Vignesh Ramanathan, Kratarth Goel, Alexandre Robicquet, Amir A Sadeghian, Li Fei-Fei, and Silvio Savarese. "Learning to predict human behavior in crowded scenes". In: *Group and Crowd Behavior for Computer Vision*. Elsevier, 2017, pp. 183–207.

[83] Xiaodan Shi, Xiaowei Shao, Zhiling Guo, Guangming Wu, Haoran Zhang, and Ryosuke Shibasaki. "Pedestrian Trajectory Prediction in Extremely Crowded Scenarios". In: *Sensors*. 2019.

[84] Niccoló Bisagno, B. O. Zhang, and Nicola Conci. "Group LSTM: Group Trajectory Prediction in Crowded Scenarios". In: *ECCV Workshops*. 2018.

[85] Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. "SR-LSTM: State Refinement for LSTM Towards Pedestrian Trajectory Prediction". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 12077–12086. DOI: 10.1109/CVPR.2019.01236.

[86] Yanliang Zhu, Deheng Qian, Dongchun Ren, and Huaxia Xia. "StarNet: Pedestrian Trajectory Prediction using Deep Neural Network in Star Topology". In: *ArXiv* abs/1906.01797 (2019).

[87] Boris Ivanovic and Marco Pavone. "The Trajectron: Probabilistic Multi-Agent Trajectory Modeling with Dynamic Spatiotemporal Graphs". In: 2018.

[88] Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G. Hauptmann, and Li Fei-Fei. "Peeking Into the Future: Predicting Future Person Activities and Locations in Videos". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 5718–5727.

[89] Antoine Tordeux, Mohcine Chraibi, Armin Seyfried, and Andreas Schadschneider. "Prediction of pedestrian dynamics in complex architectures with artificial neural networks". In: *Journal of Intelligent Transportation Systems* (2019).

[90] Yi Ma, Eric Wai Ming Lee, and Richard Kwok Kit Yuen. "An Artificial Intelligence-Based Approach for Simulating Pedestrian Movement". In: *IEEE Transactions on Intelligent Transportation Systems* 17 (2016), pp. 3159–3170.

[91] Irtiza Hasan, Francesco Setti, Theodore Tsesmelis, Alessio Del Bue, Fabio Galasso, and Marco Cristani. "MX-LSTM: Mixing Tracklets and Vislets to Jointly Forecast Trajectories and Head Poses". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 6067–6076.

[92] I. Hasan, F. Setti, Theodore Tsesmelis, Vasileios Belagiannis, S. Amin, A. D. Bue, Marco Cristani, and Fabio Galasso. “Forecasting People Trajectories and Head Poses by Jointly Reasoning on Tracklets and Vislets”. In: *IEEE transactions on pattern analysis and machine intelligence* (2019).

[93] Tim Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone. “Trajectron++: Multi-Agent Generative Trajectory Forecasting With Heterogeneous Data for Control”. In: *ArXiv* abs/2001.03093 (2020).

[94] Irtiza Hasan, Francesco Setti, Theodore Tsesmelis, Alessio Del Bue, Marco Cristani, and Fabio Galasso. “"Seeing is Believing": Pedestrian Trajectory Forecasting Using Visual Frustum of Attention”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018), pp. 1178–1185.

[95] Yanyu Xu, Zhixin Piao, and Shenghua Gao. “Encoding Crowd Interaction with Deep Neural Network for Pedestrian Trajectory Prediction”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 5275–5284.

[96] Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. “Soft + Hardwired attention: An LSTM framework for human trajectory prediction and abnormal event detection”. In: *Neural networks : the official journal of the International Neural Network Society* 108 (2018), pp. 466–478.

[97] Jiachen Li, Hengbo Ma, Zhihao Zhang, and Masayoshi Tomizuka. “Social-WaGDAT: Interaction-aware Trajectory Prediction via Wasserstein Graph Double-Attention Network”. In: *ArXiv* abs/2002.06241 (2020).

[98] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, and Silvio Savarese. “SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints”. In: *CoRR* abs/1806.01482 (2018).

[99] Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. “GD-GAN: Generative Adversarial Networks for Trajectory Prediction and Group Detection in Crowds”. In: *ArXiv* abs/1812.07667 (2018).

[100] Sirin Haddad, Meiqing Wu, He Wei, and Siew Kei Lam. “Situation-Aware Pedestrian Trajectory Prediction with Spatio-Temporal Attention Model”. In: *ArXiv* abs/1902.05437 (2019).

[101] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. “STGAT: Modeling Spatial-Temporal Interactions for Human Trajectory Prediction”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), pp. 6271–6280.

[102] Abduallah A. Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian G. Claudel. “Social-STGCNN: A Social Spatio-Temporal Graph Convolutional Neural Network for Human Trajectory Prediction”. In: *ArXiv* abs/2002.11927 (2020).

[103] Jiachen Li, Fan Yang, Masayoshi Tomizuka, and Chiho Choi. “EvolveGraph: Heterogeneous Multi-Agent Multi-Modal Trajectory Prediction with Evolving Interaction Graphs”. In: *ArXiv* abs/2003.13924 (2020).

[104] Jianhua Sun, Qinhong Jiang, and Cewu Lu. “Recursive Social Behavior Graph for Trajectory Prediction”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 657–666.

[105] Cunjun Yu, Xiao Ma, J. Ren, H. Zhao, and Shuai Yi. “Spatio-Temporal Graph Transformer Networks for Pedestrian Trajectory Prediction”. In: *ECCV*. 2020.

[106] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, E. Adeli, J. Malik, and Adrien Gaidon. “It Is Not the Journey but the Destination: Endpoint Conditioned Trajectory Prediction”. In: *ECCV*. 2020.

[107] Chaofan Tao, Qinhong Jiang, Lixin Duan, and Ping Luo. “Dynamic and Static Context-aware LSTM for Multi-agent Motion Prediction”. In: *ECCV*. 2020.

[108] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention Is All You Need”. In: *NIPS*. 2017.

[109] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural Machine Translation by Jointly Learning to Align and Translate”. In: *CoRR* abs/1409.0473 (2014).

[110] Andrey Rudenko, Luigi Palmieri, Serge Herman, Kris M. Kitani, Dariu M. Gavrila, and Kai Oliver Arras. “Human Motion Trajectory Prediction: A Survey”. In: *ArXiv* abs/1905.06113 (2019).

[111] Mehdi Moussaid, Niriaska Perozo, Simon Garnier, Dirk Helbing, and Guy Theraulaz. “The Walking Behaviour of Pedestrian Social Groups and Its Impact on Crowd Dynamics”. In: *PLoS ONE* 5 (2010).

[112] Julian F. P. Kooij, Nicolas Schneider, Fabian Flohr, and Dariu Gavrila. “Context-Based Pedestrian Path Prediction”. In: *ECCV*. 2014.

[113] Joan Bruna, W. Zaremba, Arthur Szlam, and Y. LeCun. “Spectral Networks and Locally Connected Networks on Graphs”. In: *CoRR* abs/1312.6203 (2014).

[114] Alex Kendall and Yarin Gal. “What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?” In: *NIPS*. 2017.

[115] *Github Issue: https://github.com/Majiker/STAR/issues/16*. Accessed: 2021-11-10.

[116] Osama Makansi, Özgün Çiçek, Yassine Marrakchi, and Thomas Brox. “On Exposing the Challenging Long Tail in Future Prediction of Traffic Actors”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 13127–13137.

[117] Petar Velickovic, Guillem Cucurull, A. Casanova, A. Romero, P. Lio, and Yoshua Bengio. “Graph Attention Networks”. In: *ArXiv* abs/1710.10903 (2018).

[118] K. Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *CoRR* abs/1409.1556 (2015).

[119] Jost Tobias Springenberg, A. Dosovitskiy, T. Brox, and Martin A. Riedmiller. “Striving for Simplicity: The All Convolutional Net”. In: *CoRR* abs/1412.6806 (2015).

[120] M. Sundararajan, Ankur Taly, and Qiqi Yan. “Axiomatic Attribution for Deep Networks”. In: *ICML*. 2017.

[121] Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. “iNNvestigate Neural Networks!” In: *Journal of Machine Learning Research* 20.93 (2019), pp. 1–8. URL: http://jmlr.org/papers/v20/18-540.html.

[122] L. Arras, Jose A. Arjona-Medina, Michael Widrich, Grégoire Montavon, Michael Gillhofer, K. Müller, S. Hochreiter, and W. Samek. “Explaining and Interpreting LSTMs”. In: *ArXiv* abs/1909.12114 (2019).

[123] Emanuel Parzen. “ON ESTIMATION OF A PROBABILITY DENSITY FUNCTION AND MODE”. In: 1962.

[124] Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. “Stochastic Trajectory Prediction via Motion Indeterminacy Diffusion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 17113–17122.

[125] Stefano Pellegrini, Andreas Ess, and Luc Van Gool. “Improving Data Association by Joint Modeling of Pedestrian Trajectories and Groupings”. In: *ECCV*. 2010.

[126] Tatjana Chavdarova, Pierre Baqué, Stéphane Bouquet, Andrii Maksai, Cijo Jose, Timur M. Bagautdinov, Louis Lettry, Pascal Fua, Luc Van Gool, and François Fleuret. “WILDTRACK: A Multi-camera HD Dataset for Dense Unscripted Pedestrian Detection”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 5030–5039.

[127] Li Sun, Zhi Yan, Sergi Molina Mellado, Marc Hanheide, and Tom Duckett. “3DOF Pedestrian Trajectory Prediction Learned from Long-Term Autonomous Mobile Robot Deployment Data”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2017), pp. 1–7.

[128] N Edward. “Does the Flap of a Butterfly’s Wings in Brazil Set Off a Tornado in Texas”. In: 1972.

[129] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR* abs/1412.6980 (2015).

[130] Travis Williams and Robert Y. Li. “Wavelet Pooling for Convolutional Neural Networks”. In: *ICLR*. 2018.

[131] Hao Cheng, Wentong Liao, Michael Ying Yang, Bodo Rosenhahn, and Monika Sester. “AMENet: Attentive Maps Encoder Network for Trajectory Prediction”. In: 2020.

[132] Yanliang Zhu, Dongchun Ren, Mingyu Fan, Deheng Qian, Xin Li, and Huaxia Xia. “Robust Trajectory Forecasting for Multiple Intelligent Agents in Dynamic Scene”. In: *ArXiv* abs/2005.13133 (2020).

[133] Christoph Schöller, Vincent Aravantinos, Florian Lay, and Alois Knoll. “What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction”. In: *IEEE Robotics and Automation Letters* 5 (2020), pp. 1696–1703.

[134] Amir Sadeghian, Vineet Kosaraju, Agrim Gupta, Silvio Savarese, and Alexandre Alahi. “TrajNet: Towards a Benchmark for Human Trajectory Prediction”. In: *arXiv preprint* (2018).

[135] Yuejiang Liu, Parth Kothari, and Alexandre Alahi. “Collaborative GAN Sampling”. In: *ArXiv* abs/1902.00813 (2019).

[136] Junwei Liang, Lu Jiang, Kevin Murphy, Ting Yu, and Alexander Hauptmann. “The Garden of Forking Paths: Towards Multi-Future Trajectory Prediction”. In: *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[137] Abner Guzmán-Rivera, Dhruv Batra, and Pushmeet Kohli. “Multiple Choice Learning: Learning to Produce Multiple Structured Outputs”. In: *NIPS*. 2012.

[138] C. Rupprecht, Iro Laina, Robert S. DiPietro, Maximilian Baust, Federico Tombari, Nassir Navab, and Gregory D. Hager. “Learning in an Uncertain World: Representing Ambiguity Through Multiple Hypotheses”. In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2016), pp. 3611–3620.

[139] Lei Huang, Jihui Zhuang, Xiaoming Cheng, Riming Xu, and Hongjie Ma. “STI-GAN: Multimodal Pedestrian Trajectory Prediction Using Spatiotemporal Interactions and a Generative Adversarial Network”. In: *IEEE Access* 9 (2021), pp. 50846–50856.

[140] Alexey Dosovitskiy and Thomas Brox. “Generating Images with Perceptual Similarity Metrics based on Deep Networks”. In: *NIPS*. 2016.

[141] Yuke Li. “Which Way Are You Going? Imitative Decision Learning for Path Forecasting in Dynamic Scenes”. In: *CVPR*. 2019.

[142] Debaditya Roy, Tetsuhiro Ishizaka, Chalavadi Krishna Mohan, and Atsushi Fukuda. “Vehicle Trajectory Prediction at Intersections using Interaction based Generative Adversarial Networks”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)* (2019), pp. 2318–2323.

[143] Junchen Jin, Dingding Rong, Tong Zhang, Qingyuan Ji, Haifeng Guo, Yisheng Lv, Xiaoliang Ma, and Fei Wang. “A GAN-Based Short-Term Link Traffic Prediction Approach for Urban Road Networks Under a Parallel Learning Framework”. In: *IEEE Transactions on Intelligent Transportation Systems* (2022).

[144] Francesco Giuliari, Irtiza Hasan, Marco Cristani, and Fabio Galasso. “Transformer Networks for Trajectory Forecasting”. In: *ArXiv* abs/2003.08111 (2020).

[145] Bin Yu, Ke Zhu, Kaiteng Wu, and Michael Zhang. “Improved OpenCL-Based Implementation of Social Field Pedestrian Model”. In: *IEEE Transactions on Intelligent Transportation Systems* 21 (2020), pp. 2828–2839.

[146] Hao Sun, Zhiqun Zhao, and Z. He. “Reciprocal Learning Networks for Human Trajectory Prediction”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 7414–7423.

[147] Javad Amirian, Wouter van Toll, Jean-Bernard Hayet, and Julien Pettré. “Data-Driven Crowd Simulation with Generative Adversarial Networks”. In: *CASA*. 2019.

[148] Parth Kothari, S. Kreiss, and Alexandre Alahi. “Human Trajectory Forecasting in Crowds: A Deep Learning Perspective”. In: *IEEE Transactions on Intelligent Transportation Systems* (2021).

[149] Xudong Mao, Q. Li, Haoran Xie, Raymond Y. K. Lau, Z. Wang, and Stephen Paul Smolley. “Least Squares Generative Adversarial Networks”. In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 2813–2821.

[150] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. “Generalization and equilibrium in generative adversarial nets (gans)”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 224–232.

[151] Gareth O. Roberts and Richard L. Tweedie. “Exponential convergence of Langevin distributions and their discrete approximations”. EN. In: *Bernoulli* 2.4 (Dec. 1996), pp. 341–363. ISSN: 1350-7265. URL: https://projecteuclid.org/euclid.bj/1178291835 (visited on 08/28/2019).

[152] Radford M. Neal. *Bayesian Learning for Neural Networks*. Berlin, Heidelberg: Springer-Verlag, 1996. ISBN: 0387947248.

[153] Nati Daniel et al. “PECNet: A Deep Multi-Label Segmentation Network for Eosinophilic Esophagitis Biopsy Diagnostics”. In: *ArXiv* abs/2103.02015 (2021).

[154] Hamid Eghbal-zadeh and Gerhard Widmer. “Likelihood Estimation for Generative Adversarial Networks”. In: *ArXiv* abs/1707.07530 (2017).

[155] Yuejiang Liu, Qi Yan, and Alexandre Alahi. “Social NCE: Contrastive Learning of Socially-aware Motion Representations”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 15098–15109.

[156] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett. Curran Associates, Inc., 2016, pp. 2172–2180. URL: http://papers.nips.cc/paper/6399-infogan-interpretable-representation-learning-by-information-maximizing-generative-adversarial-nets.pdf (visited on 10/14/2018).

[157] Wei-Chiu Ma, De-An Huang, Namhoon Lee, and Kris M. Kitani. “Forecasting Interactive Dynamics of Pedestrians with Fictitious Play”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 4636–4644.

[158] Gianluca Antonini, Michel Bierlaire, and Mats Weber. “Discrete choice models of pedestrian walking behavior”. In: *Transportation Research Part B: Methodological* 40.8 (2006), pp. 667–687.

[159] R.Y. Guo and H.J. Huang. “A mobile lattice gas model for simulating pedestrian evacuation”. In: *Physica A: Statistical Mechanics and its Applications* 387.2 (2008), pp. 580–586. ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2007.10.001. URL: http://www.sciencedirect.com/science/article/pii/S0378437107010333.

[160] Miho Asano, Takamasa Iryo, and Masao Kuwahara. “Microscopic pedestrian simulation model combined with a tactical model for route choice behaviour”. In: *Transportation Research Part C: Emerging Technologies* 18.6 (2010), pp. 842–855.

[161] Th. Robin, G. Antonini, M. Bierlaire, and J. Cruz. “Specification, estimation and validation of a pedestrian walking behavior model”. In: *Transportation Research Part B: Methodological* 43.1 (2009), pp. 36–56. ISSN: 0191-2615. DOI: https://doi.org/10.1016/j.trb.2008.06.010. URL: http://www.sciencedirect.com/science/article/pii/S0191261508000763.

[162] Brian Sifringer, Virginie Lurkin, and Alexandre Alahi. “Enhancing discrete choice models with representation learning”. In: *Transportation Research Part B: Methodological* 140 (2020), pp. 236–261.

[163] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: *International Journal of Computer Vision* 128 (2017), pp. 336–359.

[164] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning Important Features Through Propagating Activation Differences”. In: *ICML*. 2017.

[165] Lloyd S. Shapley. “17. A Value for n-Person Games”. In: 1953.

[166] Osama Makansi, Julius von Kügelgen, Francesco Locatello, Peter V. Gehler, Dominik Janzing, Thomas Brox, and Bernhard Scholkopf. “You Mostly Walk Alone: Analyzing Feature Attribution in Trajectory Prediction”. In: *ArXiv* abs/2110.05304 (2022).

[167] Pasquale Coscia, Francesco Castaldo, Francesco A. N. Palmieri, Alexandre Alahi, Silvio Savarese, and Lamberto Ballan. “Long-term path prediction in urban scenarios using circular distributions”. In: *Image Vis. Comput.* 69 (2018), pp. 81–91.

[168] Daniel McFadden et al. “Conditional logit analysis of qualitative choice behavior”. In: (1973).

[169] Victor Aguirregabiria and Pedro Mira. “Dynamic discrete choice structural models: A survey”. In: *Journal of Econometrics* 156.1 (2010), pp. 38–67.

[170] Mandy Ryan, Karen Gerard, and Mabel Amaya-Amaya. *Using discrete choice experiments to value health and health care*. Vol. 11. Springer Science & Business Media, 2007.

[171] Chandra R Bhat, Naveen Eluru, and Rachel B Copperman. “Flexible model structures for discrete choice analysis”. In: *Handbook of transport modelling*. Emerald Group Publishing Limited, 2007.

[172] Charles F Manski. “The structure of random utility models”. In: *Theory and decision* 8.3 (1977), p. 229.

[173] Huw CWL Williams. “On the formation of travel demand models and economic evaluation measures of user benefit”. In: *Environment and planning A* 9.3 (1977), pp. 285–344.

[174] Daniel McFadden. “Modeling the choice of residential location”. In: *Transportation Research Record* 673 (1978).

[175] Yves Bentz and Dwight Merunka. “Neural networks and the multinomial logit for brand choice modelling: a hybrid approach”. In: *Journal of Forecasting* 19.3 (2000), pp. 177–200.

[176] Harald Hruschka, Werner Fettes, and Markus Probst. “An empirical comparison of the validity of a neural net based multinomial logit choice model to alternative model specifications”. In: *European Journal of Operational Research* 159.1 (2004), pp. 166–180.

[177] Melvin Wong and Bilal Farooq. “A bi-partite generative model framework for analyzing and simulating large scale multiple discrete-continuous travel behaviour data”. In: *Transportation Research Part C: Emerging Technologies* 110 (2020), pp. 247–268.

[178] Yafei Han, Christopher Zegras, Francisco Camara Pereira, and Moshe Ben-Akiva. “A Neural-embedded Choice Model: TasteNet-MNL Modeling Taste Heterogeneity with Flexibility and Interpretability”. In: *arXiv preprint arXiv:2002.00922* (2020).

[179] Yuning Chai, B. Sapp, Mayank Bansal, and Dragomir Anguelov. “MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction”. In: *CoRL*. 2019.

[180] Junwei Liang, Lu Jiang, and Alexander Hauptmann. “Simaug: Learning robust representations from simulation for trajectory prediction”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 275–292.

[181] Yuejiang Liu, Qi Yan, and Alexandre Alahi. “Social nce: Contrastive learning of socially-aware motion representations”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15118–15129.

[182] Prarthana Bhattacharyya, Chengjie Huang, and Krzysztof Czarnecki. “SSL-Lanes: Self-Supervised Learning for Motion Forecasting in Autonomous Driving”. In: *arXiv preprint arXiv:2206.14116* (2022).

[183] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. “Lora: Low-rank adaptation of large language models”. In: *arXiv preprint arXiv:2106.09685* (2021).

[184] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. “Compacter: Efficient Low-Rank Hypercomplex Adapter Layers”. In: *NeurIPS*. 2021.

[185] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. “Learning Social Etiquette: Human Trajectory Understanding In Crowded Scenes”. In: *ECCV*. 2016.

[186] 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)* (2020), pp. 1929–1934.

[187] Johnny L. Houston, Guido C. A. Zuidhof, Luca Bergamini, Yawei Ye, Ashesh Jain, Sammy Omari, Vladimir I. Iglovikov, and Peter Ondruska. “One Thousand and One Hours: Self-driving Motion Prediction Dataset”. In: *CoRL*. 2020.

[188] Ishaan Gulrajani and David Lopez-Paz. “In Search of Lost Domain Generalization”. In: *ArXiv* abs/2007.01434 (2021).

[189] Gilles Blanchard, Gyemin Lee, and Clayton D. Scott. “Generalizing from Several Related Classification Tasks to a New Unlabeled Sample”. In: *NIPS*. 2011.

[190] Deyao Zhu, Mohamed Zahran, Li Erran Li, and Mohamed Elhoseiny. “Motion Forecasting with Unlikelihood Training in Continuous Space”. In: *Conference on Robot Learning*. PMLR. 2021, pp. 1003–1012.

[191] Yuejiang Liu, Riccardo Cadei, Jonas Schweizer, Sherwin Bahmani, and Alexandre Alahi. “Towards Robust and Adaptive Motion Forecasting: A Causal Representation Perspective”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 17081–17092.

[192] Gabriela Csurka. “Deep Visual Domain Adaptation”. In: *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)* (2020), pp. 1–8.

[193] Mei Wang and Weihong Deng. “Deep Visual Domain Adaptation: A Survey”. In: *Neurocomputing* 312 (2018), pp. 135–153.

[194] Sicheng Zhao, Bo Li, Colorado Reed, Pengfei Xu, and Kurt Keutzer. “Multi-source Domain Adaptation in the Deep Learning Era: A Systematic Survey”. In: *ArXiv* abs/2002.12169 (2020).

[195] Devis Tuia, Claudio Persello, and Lorenzo Bruzzone. “Domain Adaptation for the Classification of Remote Sensing Data: An Overview of Recent Advances”. In: *IEEE Geoscience and Remote Sensing Magazine* 4 (2016), pp. 41–57.

[196] Sinno Jialin Pan and Qiang Yang. “A Survey on Transfer Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22 (2010), pp. 1345–1359.

[197] Jeremy Howard and Sebastian Ruder. “Universal Language Model Fine-tuning for Text Classification”. In: *ACL*. 2018.

[198] Alec Radford and Karthik Narasimhan. “Improving Language Understanding by Generative Pre-Training”. In: 2018.

[199] Daniel Matthew Cer et al. “Universal Sentence Encoder for English”. In: *EMNLP*. 2018.

[200] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. "Distributed Representations of Words and Phrases and their Compositionality". In: *NIPS*. 2013.

[201] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. "Parameter-Efficient Transfer Learning for NLP". In: *ICML*. 2019.

[202] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. "Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning". In: *ArXiv* abs/2205.05638 (2022).

[203] Chuanqi Zang, Mingtao Pei, and Yu Kong. "Few-shot Human Motion Prediction via Learning Novel Motion Dynamics". In: *IJCAI*. 2020.

[204] Liangyan Gui, Yu-Xiong Wang, Deva Ramanan, and José M. F. Moura. "Few-Shot Human Motion Prediction via Meta-learning". In: *ECCV*. 2018.

[205] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". en. In: *arXiv:1703.03400 [cs]* (Mar. 2017). arXiv: 1703.03400. URL: http://arxiv.org/abs/1703.03400 (visited on 11/23/2018).

[206] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. "A Learned Representation For Artistic Style". In: *ArXiv* abs/1610.07629 (2017).

[207] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual Losses for Real-Time Style Transfer and Super-Resolution". In: *ECCV*. 2016.

[208] Karttikeya Mangalam, Yang An, Harshayu Girase, and Jitendra Malik. "From goals, waypoints & paths to long term human trajectory forecasting". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15233–15242.

[209] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ArXiv* abs/2010.11929 (2021).

[210] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. "Efficient parametrization of multi-domain deep neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8119–8127.

[211] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. "Revisiting Batch Normalization For Practical Domain Adaptation". In: *ArXiv* abs/1603.04779 (2017).

[212] Harm de Vries, Florian Strub, Jérémie Mary, H. Larochelle, Olivier Pietquin, and Aaron C. Courville. "Modulating early visual processing by language". In: *ArXiv* abs/1707.00683 (2017).

[213] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. "Playing Atari with Deep Reinforcement Learning". In: *ArXiv* abs/1312.5602 (2013).

[214] D. Silver et al. "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm". In: *ArXiv* abs/1712.01815 (2017).

[215] Roberto Martín-Martín, Michelle A. Lee, Rachel Gardner, S. Savarese, Jeannette Bohg, and Animesh Garg. "Variable Impedance Control in End-Effector Space: An Action Space for Reinforcement Learning in Contact-Rich Tasks". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019), pp. 1010–1017.

[216] B. R. Kiran, Ibrahim Sobh, V. Talpaert, P. Mannion, A. A. Sallab, S. Yogamani, and P. P'erez. "Deep Reinforcement Learning for Autonomous Driving: A Survey". In: *ArXiv* abs/2002.00444 (2020).

[217] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska. *One Thousand and One Hours: Self-driving Motion Prediction Dataset*. https://level-5.global/level5/data/. 2020.

[218] Luca Bergamini, Y. Ye, Oliver Scheel, Long Chen, Chih Hu, Luca Del Pero, Blazej Osinski, Hugo Grimmett, and Peter Ondruska. "SimNet: Learning Reactive Self-driving Simulations from Real-world Observations". In: *ArXiv* abs/2105.12332 (2021).

[219] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, J. Schulman, Jie Tang, and Wojciech Zaremba. "OpenAI Gym". In: *ArXiv* abs/1606.01540 (2016).

[220] E. Espié, Christophe Guionneau, Bernhard Wymann, Christos Dimitrakakis, Rémi Coulom, and Andrew Sumner. "TORCS, The Open Racing Car Simulator". In: 2005.

[221] Edouard Leurent. *An Environment for Autonomous Driving Decision-Making*. https://github.com/eleurent/highway-env. 2018.

[222] Craig Quiter and Maik Ernst. *deepdrive/deepdrive: 2.0 (2.0)*. https://doi.org/10.5281/zenodo.1248998. 2018.

[223] Ming Zhou et al. "SMARTS: Scalable Multi-Agent Reinforcement Learning Training School for Autonomous Driving". In: *ArXiv* abs/2010.09776 (2020).

[224] Blazej Osinski, Piotr Milos, Adam Jakubowski, Pawel Ziecina, Michal Martyniak, Christopher Galias, Antonia Breuer, Silviu Homoceanu, and Henryk Michalewski. "CARLA Real Traffic Scenarios - novel training ground and benchmark for autonomous driving". In: *ArXiv* abs/2012.11329 (2020).

[225] Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. *Stable Baselines3*. https://github.com/DLR-RM/stable-baselines3. 2019.

[226] Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph E. Gonzalez, Michael I. Jordan, and Ion Stoica. "RLlib: Abstractions for Distributed Reinforcement Learning". In: *ICML*. 2018.

[227] Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning". In: *AISTATS*. 2011.

[228] Mayank Bansal, Alex Krizhevsky, and Abhijit S. Ogale. "ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst". In: *ArXiv* abs/1812.03079 (2019).

[229] J. Schulman, F. Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. "Proximal Policy Optimization Algorithms". In: *ArXiv* abs/1707.06347 (2017).

[230] Bokeh Development Team. *Bokeh: Python library for interactive visualization*. 2018. URL: https://bokeh.pydata.org/en/latest/.

[231] Yuejiang Liu, Qi Yan, and Alexandre Alahi. "Social NCE: Contrastive Learning of Socially-Aware Motion Representations". en. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 15118–15129. URL: https://openaccess.thecvf.com/content/ICCV2021/html/Liu_Social_NCE_Contrastive_Learning_of_Socially-Aware_Motion_Representations_ICCV_2021_paper.html (visited on 10/25/2021).

[232] Deyao Zhu, Mohamed A. Zahran, Li Erran Li, and Mohamed Elhoseiny. "Motion Forecasting with Unlikelihood Training in Continuous Space". In: 2021.

[233] Abduallah A. Mohamed, Deyao Zhu, Warren Vu, Mohamed Elhoseiny, and Christian G. Claudel. "Social-Implicit: Rethinking Trajectory Prediction Evaluation and The Effectiveness of Implicit Maximum Likelihood Estimation". In: *ArXiv* abs/2203.03057 (2022).

[234] Lihuan Li, Maurice Pagnucco, and Yang Song. "Graph-based Spatial Transformer with Memory Replay for Multi-future Pedestrian Trajectory Prediction". In: *ArXiv* abs/2206.05712 (2022).

[235] Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. "TTT++: When Does Self-Supervised Test-Time Training Fail or Thrive?" In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 21808–21820. URL: https://proceedings.neurips.cc/paper/2021/file/b618c3210e934362ac261db280128c22-Paper.pdf.

Rue de la Blancherie 17
Chavannes-près-Renens 1022
Lausanne
☎ *+41 78 238 96 36*
✉ *parth.kothari@epfl.ch*
🖱 *thedebugger811.github.io*

Parth Kothari

Education

08/2018 – 10/2022
Ph.D in Electrical Engineering, *EPFL*, *5.9/6.0*.
Advised by Prof. Alexandre Alahi
Affiliated with the Visual Intelligence For Transportation (VITA) Labratory
Conducting research in Human Motion Forecasting, building Socially-Aware Models

07/2014 – 07/2018
B.Tech in Electrical Engineering, *IIT Bombay*, *9.74/10*.
Completed with Minor in Computer Science and Honors in Electrical Engineering
Department Rank 2 in a batch of 66 students
Institute Rank 7 in a batch of 880 students

Internship

Title: **DriverGym: Democratising RL for Autonomous Driving** [Paper][Code]
L5 Research, Woven Planet (Previously, Lyft) [06/2021 – 11/2021]

Description: DriverGym is an open-source OpenAI Gym-compatible environment specifically tailored for developing RL algorithms for autonomous driving. It provides access to more than 1000 hours of expert logged data and also supports reactive and data-driven agent behavior. Further, we provide an extensive and flexible closed-loop evaluation protocol. The DriverGym code, as well as all the accompanying baselines are publicly available to further stimulate development from the community.

Publications

- Modular Low-Rank Adaptation for Deep Motion Forecasting, **CoRL 2022**
- Safety-compliant GANs for Human Trajectory Forecasting, **IEEE ITS 2022**
- DriverGym: Democratising RL for Autonomous Driving, **ML4AD Workshop, NeurIPS 2021**
- TTT++: Improved Test-Time Training, **NeurIPS 2021**
- Interpretable Social Anchors for Human Trajectory Forecasting in Crowds, **CVPR 2021**
- Human Trajectory Forecasting in Crowds: A Deep Learning Perspective, **IEEE ITS 2020**
- Collaborative Sampling in Generative Adversarial Networks, **AAAI 2020**

Research Projects

Title: **TrajNet++: Human Trajectory Forecasting in Crowds** [Paper] [Code] [2020]

Description: We present an in-depth analysis of existing deep learning-based methods for modelling social interactions. To objectively compare the performance of these trajectory forecasting models, we develop a large scale interaction-centric benchmark TrajNet++. We propose a domain-knowledge inspired data-driven method to provide safer, socially compliant predictions and validate its efficacy on TrajNet++. Finally, we apply layer-wise relevance propagation to explain the decision-making of current models.

Title	**Modular Low-Rank Adaptation for Deep Motion Forecasting** [Paper][Code] [2022]
Description	Deep motion forecasting models have achieved great success when trained on a massive amount of data. Yet, they often perform poorly when training data is limited. To address this challenge, we propose a transfer learning approach for efficiently adapting pre-trained forecasting models to new domains, such as unseen agent types and scene contexts. Unlike the conventional fine-tuning approach that updates the whole encoder, our main idea is to reduce the amount of tunable parameters that can precisely account for the target domain-specific motion style. Experiments on SDD and Lyft Level 5 dataset show demonstrate the effectiveness of our proposed method.
Title	**Interpretable Social Anchors for Human Trajectory Forecasting** [Paper] [2021]
Description	Current neural network based forecasting models suffer from one crucial limitation: lack of interpretability. To overcome this, we leverage the power of discrete choice models to learn interpretable rule-based intents, and subsequently utilise the expressibility of neural networks to model scene-specific residual. Experiments on TrajNet++ demonstrates the efficacy of our method to explain its predictions without compromising the accuracy.
Title	**TTT++: Improved Test-Time Training** [Paper] [2021]
Description	Test-Time Training (TTT) is a promising paradigm that leverages an auxiliary Self-Supervised Learning (SSL) task at test-time to improve generalization. In this work, we improve upon TTT, by introducing *TTT++*, an online feature alignment strategy by utilizing offline feature summarization. Further, we incorporate a suitable SSL task in the form of contrastive learning and empirically demonstrate that our proposed strategy outperforms state-of-the-art methods on multiple benchmarks.
Title	**Safety-compliant GANs for Human Trajectory Forecasting** [2021]
Description	We introduce SGANv2: an improved safety-compliant SGAN architecture equipped with spatio-temporal interaction modelling and a transformer-based discriminator. Additionally, SGANv2 utilizes the learned discriminator even at test-time via a collaborative sampling strategy that refines the colliding trajectories. Through extensive experimentation on multiple real-world and synthetic datasets, we demonstrate the efficacy of SGANv2 to provide socially-compliant multimodal trajectories.
Title	**Collaborative Sampling in Generative Adversarial Networks** [Paper] [Code] [2019]
Description	Developed a collaborative sampling scheme between the generator and the discriminator for improved data generation during sampling. Proposed a practical discriminator shaping method for effective sample refinement. Experiments on synthetic and image datasets demonstrate the efficacy of our method to improve generated samples both quantitatively and qualitatively, offering a new degree of freedom in GAN sampling.
Title	**Adversarial Loss for Human Trajectory Prediction** [Paper] [Code] [2019]
Description	Highlighted an unexpected pitfall in the state-of-the-art architecture for multimodal human prediction via controlled experiments. Proposed a modification to the architecture leveraging the progress in the GAN community. Demonstrate the efficacy of the proposed modification on real world datasets, indicating room for improvement on state-of-the-art.

Workshops and Challenges

Title	**TrajNet++: Human Trajectory Forecasting Benchmark** [Challenge] [Code] [2020] *Appearing in Multi-Agent Interaction and Reasoning Workshop,* ***ICCV 2021*** *Appeared in Long-term Human Motion Prediction Workshop,* ***ICRA 2021*** *Appeared in Benchmarking Trajectory Forecasting Models,* ***ECCV 2020***

Technical Strengths

Languages	Python, C++, C, MATLAB
Softwares	Pytorch, Tensorflow, OpenCV

Academic Achievements

- Recipient of the **Institute Academic Prize** for securing **First Rank** out of 66 students of the Electrical Engineering Department in the second academic year 2015-16.
- Secured **State Rank 1** and **All India Rank 11** in JEE-Mains-2014, national level engineering entrance examination, out of **1.5 million** candidates
- Secured **All India Rank 458** in JEE-Advanced-2014 out of **150,000** candidates
- Awarded **Gold medal** in Indian National Physics Olympiad-2014 and Indian National Chemistry Olympiad-2014 for being among the top 35 students in India

Relevant Courses

CS Courses	Machine Learning, Computer Vision, Digital Image Processing, Advanced Image Processing, Medical Image Processing, Data Structures and Algorithms
EE Courses	Signals and Systems, Digital Signal Processing, Advanced Topics in Signal Processing, Markov Chains, Control Systems, Probability and Random Processes
Math Courses	Linear Algebra, Applied Mathematical Analysis, Complex Analysis, Calculus

Leadership

2020-2022	**Teaching Assistant for Deep Learning for Autonomous Vehicles**, *EPFL*, [Page].
Autumn 2018	**Teaching Assistant for Calculus**, *IIT Bombay*.
2016–2017	**Manager, Electronics Club**, *IIT Bombay*. Conducted workshops, institute-wide events, hackathons and group discussions to promote Electronics among the student community Awarded Technical Organizational Special Mention for exemplary contribution
2017-2018	**Student Mentor, Institute Student Mentorship Programme**. Responsible for mentoring a group of 12 freshmen to help adjust to the new environment, academically and socially and guide them towards a holistic development ensuring a smooth transition to college life