HUMAN BEHAVIOR UNDERSTANDING AND INTENTION PREDICTION

A Dissertation presented to

the Faculty of the Graduate School

at the University of Missouri

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

HAO SUN

Dr. Zhihai He, Dissertation Supervisor

December 2020

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

HUMAN BEHAVIOR UNDERSTANDING AND INTENTION PREDICTION

Hao Sun,

a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Zhihai He

Dr. Yunxin Zhao

Dr. Filiz Bunyak

Dr. Ming Xin

ACKNOWLEDGMENTS

First and foremost, I would to sincerely thank my advisor Dr. Zhihai He, who guided and encouraged me throughout my doctoral work. I also want to express my gratitude to my former advisor Dr. Alina Zara for her support and guidance during the first year of my doctoral research. I would also like to thank all other committee members of my dissertation, Dr. Yunxin Zhao, Dr. Filiz Bunyak and Dr. Ming Xin, for their precious advice and suggestions.

I want to thank my friends, former and current lab mates, especially to Changzhe Jiao, Zhiqun Zhao, Ke Gao, Guanghan Ning, Zhi Zhang, Chen Huang, Jianhe Yuan and Yang Li for our productive discussions during my research and life.

My major motivation to finish my doctoral studies is from my parents and wife. Without their endless love, support and encouragement, I cannot make it so far. They deserve the most sincere thanks.

TABLE OF CONTENTS

ACKNOWLEDGMENTS . . . . . **ii**

LIST OF TABLES . . . . . **vii**

LIST OF FIGURES . . . . . **ix**

ABSTRACT . . . . . **xv**

CHAPTER

1 Background and Introduction **1**

1.1 Deep Neural Networks 1

1.1.1 Convolutional Neural Networks 2

1.1.2 Long Short-Term Memory 3

1.1.3 Generative Adversarial Networks 4

1.2 Human Behavior Understanding 5

1.3 Human Trajectory Prediction 8

1.4 Summary 12

2 Automated Work Efficiency Analysis for Smart Manufacturing Using Human Pose Tracking and Temporal Action Localization **13**

2.1 Motivation 13

2.2 Related Work 14

2.2.1 Human Pose Estimation 15

2.2.2 Temporal Activity Localization 16

2.3 The Proposed Method 17

2.3.1 Human Body Pose Tracking 18
2.3.2 Body Pose Feature Extraction and Dynamic Time Warping for Action Matching 26
2.3.3 Temporal Activity Localization 30
2.4 Experimental Results 31
2.4.1 Human Pose Estimation Evaluation 31
2.4.2 Evaluating the Worker Efficiency Analysis 34
2.5 Conclusion 40
3 Reciprocal Learning Networks for Human Trajectory Prediction 41
3.1 Motivation 41
3.2 Related Work 43
3.2.1 Human-Human Models for Trajectory Prediction 44
3.2.2 Human-Scene Models for Trajectory Prediction 44
3.2.3 Recurrent Neural Networks for Sequence Prediction 45
3.2.4 Generative Networks and Cycle Consistency Learning 46
3.3 Reciprocal Networks for Human Trajectory Prediction 46
3.3.1 Problem Formulation 47
3.3.2 Method Overview 47
3.3.3 Reciprocal Network Training 49
3.3.4 Constructing the Forward and Backward Prediction Networks . . . 50
3.4 Reciprocal Attack for Matched Prediction of Human Trajectories 56
3.5 Experimental Results 60
3.5.1 Datasets 60

3.5.2 Implementation Details 61
3.5.3 Evaluation Metrics and Methods 61
3.5.4 Comparison with Existing Methods 62
3.5.5 Quantitative Results 63
3.5.6 Ablation Studies 65
3.5.7 Qualitative Results 67
3.5.8 Generalization: Evaluations on Town Centre and Grand Central Station Datasets 70
3.5.9 Backward Prediction Evaluation 71
3.6 Conclusion 73
4 Hierarchical Interactions Modeling for Human Future Trajectory Prediction 74
4.1 Motivation 74
4.2 Related Work 76
4.2.1 Human-Human Models for Trajectory Prediction 77
4.2.2 Human-Scene Models for Trajectory Prediction 78
4.2.3 Graph Neural Network 79
4.3 The Proposed Method 80
4.3.1 Problem Formulation 80
4.3.2 Method Overview 81
4.3.3 Human Trajectory Encoder 82
4.3.4 Global Scene Layout Encoder 87
4.3.5 Local Patch Context Encoder 90
4.3.6 Future Trajectory Prediction 93

4.4 Experimental Results 95
4.4.1 Datasets 95
4.4.2 Implementation Details 96
4.4.3 Evaluation Metrics and Protocol 96
4.4.4 Comparison with Existing Methods 97
4.4.5 Quantitative Results 98
4.4.6 Ablation Studies 100
4.4.7 Qualitative Results 101
4.4.8 Generalization Evaluations on Town Centre and Grand Central Station Datasets 102
4.5 Conclusion 103
5 Summary and Future Work 105
BIBLIOGRAPHY 108
VITA 130

LIST OF TABLES

Table Page

2.1 Comparisons of PCK@0.2 score on the LSP test set. 33

2.2 Comparisons of PCKh@0.5 score on the MPII test set. 34

2.3 Comparisons of worker's activity localization among surveillance videos. Error metrics reported are IoU scores. 37

2.4 Ablation experiments of using features without different components. Error metrics reported are IoU scores. 38

2.5 Cross validation of Worker's activity localization among 10 videos 38

3.1 Comparisons of different methods on ETH (Column 3 and 4) and UCY (Column 5-7) datasets. 64

3.2 Average percentage of colliding human for each scene in ETH and UCY datasets. The first column represents the ground truth. 65

3.3 Ablation experiments of our full algorithm without different components. Error metrics reported are ADE and FDE in meter scale. 66

3.4 The quantitative results (ADE and FDE) on Town Centre and Grand Central Station datasets with different prediction lengths of future trajectories. . . . 71

3.5 The quantitative results on ETH (Column 3 and 4) and UCY (Column 5-7) datasets on the task of backward prediction (predicting the trajectories of previous 8 time steps, given the trajectories of 12 future time steps). 72

4.1 Comparison results of all the baselines and our method on ETH (Column 3 and 4) and UCY (Column 5-7) datasets on the task of predicting 12 future time steps, by given the 8 previous time steps. All models takes as Error metrics reported are ADE / FDE in meter scale. 99

4.2 Average percentage of colliding pedestrians for each scene in ETH and UCY datasets. The first column represents the ground truth. 100

4.3 Ablation experiments of our full algorithm without different components. Error metrics reported are ADE and FDE in meter scale. 101

4.4 The quantitative results (ADE and FDE) on Town Centre and Grand Central Station datasets with different prediction lengths of future trajectories. . . . 103

LIST OF FIGURES

Figure Page

1.1 An example architecture of CNNs from AlexNet [1] 2

1.2 An example architecture of an LSTM unit. 3

1.3 An example architecture of GANs [2]. 4

1.4 An illustration of the task of human pose estimation. 6

1.5 An illustration of the task of temporal action localization. 7

1.6 Human's future trajectory is affected not only by other pedestrians but also by the surrounding environment. Our proposed method predicts socially and physically plausible trajectories by hierarchically modeling the influence of all pedestrians in the scene as well as the global scene layout and local context. 9

2.1 Overview of our worker efficiency evaluation system. 18

2.2 The overall framework of our proposed poseGAN for the joint human pose estimation and conditional image synthesis. It includes the following components: Generator (G), Discriminator (D), Feature Extractor (FE) and Pose Estimator (PE). M and N are the number of channels of a tensor. N is the number of keypoints to be predicted. During training, the discriminator D takes real images and their corresponding ground truth annotations P_t as positive sample pairs, whereas negative sample pairs consist of two groups. The first is real images with mismatched pose estimations P_e, while the second is synthetic images with their corresponding true annotations P_t. 21

2.3 **Implementation details of Generator (G) and Discriminator (PE)**. In each module, the top notation $[M]$ indicates the resolution of the input tensor, while the bottom $[N]$ denotes the resolution at the output tensor of this module. The up-sampling blocks consist of the nearest-neighbor up-sampling followed by a $3x3$ stride 1 convolution. Batch normalization and ReLU activation are applied after every convolution except the last one. The residual blocks consist of $3x3$ stride 1 convolutions, Batch normalization and ReLU. Two residual modules are used in $128x128$ models while four are used in $256x256$ models. The down-sampling blocks consist of $4x4$ stride 2 convolutions, Batch normalization and LeakyReLU, except that the first one does not have Batch normalization. 23

2.4 Examples of human poses obtained by our poseGAN algorithm. 26

2.5 (a) Human skeleton model (blue spheres are joints and red lines are limbs). (b) Green lines generated to extract features 27

2.6 Temporal localization of worker actions. 31

2.7 Examples of human pose estimation from the LSP test set. 35
2.8 Examples of human pose estimation from the MPII test set. 36
2.9 Example frames of our worker operation efficiency evaluation dataset. . . . 37
2.10 Time (sec) spent by each worker for the whole assembling task. 39
2.11 Time (sec) spent for each operation of the manufacturing task. 39

3.1 Imagine that time is reversed and person is traveling backwards. The human trajectory is not only *forward* predictable, but also *backward* predictable. This leads to our new approach of reciprocal coupling and learning between the forward and backward prediction networks for accurate human trajectory prediction. 42
3.2 Illustration of the proposed reciprocal learning approach. 48
3.3 Illustration of the training process of reciprocal learning. 49
3.4 Overview of our prediction model. Our model consists of two key components: (1) a feature extraction module, (2) an LSTM-based GAN module. . 51
3.5 Examples of the input (left column) and the output (right column) of the monocular depth estimation [3]. The input image is from Town Centre dataset [4]. 54
3.6 Illustration of the proposed reciprocal attack method. 56
3.7 Example of our proposed reciprocal attack performed on HOTEL dataset. X-coordinates in both figures indicate the iteration, while Y-coordinates indicate error metrics, ADE and FDE (see detailed explanation in Section 3.5.3). 59
3.8 Illustration of ADE and FDE changes with respect to different λ values on ZARA1 and UNIV dataset 68

3.9 Illustration of ADE and FDE changes with respect to different ϵ values on the ZARA1 dataset 69
3.10 Illustration of our method predicting future 12 time steps trajectories, given previous 8 time steps. The results are drawn under HOTEL, ETH, UNIV and ZARA1 and ZARA2 datasets from 1st column to 5th column, respectively. 70
3.11 Qualitative examples of our method predicting future 12 time steps trajectories, given previous 8 time steps ones on Town Centre (1st column) and Grand Central Station (2nd column) dataset. Note that, we crop and resize the original image for better visualization. 72
3.12 Illustration of backward prediction (predicting previous 8 time steps trajectories, given future 12 time steps ones). The results are drawn under HOTEL, ETH, UNIV and ZARA1 and ZARA2 datasets from 1st column to 5th column respectively. Note that, we crop and resize the original image for better visualization. 73

4.1 Human's future trajectory is affected not only by other pedestrians but also by the surrounding environment. Our proposed method predicts socially and physically plausible trajectories by hierarchically modeling the influence of all pedestrians in the scene as well as the global scene layout and local context. 75
4.2 The input information observed for our method. 76

4.3 Pipeline of our proposed system. We take input from three scale information: observed human trajectories, observed global scene images and observed local scene patches. Our whole model is built based on seq2seq model, where Encoder, Intermediate State and Decoder are included. The Encoder is designed to extract spatio-temporal information from multiple sources. The spatial and temporal information concatenated with a noise are summarized in the Intermediate State and then form the input for the Decoder to yield the predicted trajectory for each observed human. 81

4.4 The framework of Human Trajectory Encoder. The upper level LSTMs are human motion extractor to capture the hidden motion states for each pedestrian. The middle level GATs are used to model the human-human interactions. The lower level LSTMs are applied to learn the temporal correlation of the interactions. Then the hidden motion states and the spatio-temporal vectors are concatenated as the output of Human Trajectory Encoder. 82

4.5 An illustration of the complete graph we build at each time step. Each node denotes each human ($h_1, h_2, \cdots, h_n$) and the edges represent the human-human interaction. 85

4.6 An illustration of the computing mechanism of a single graph attention layer. It aggregates information from each neighboring node and following a self-attention strategy. $M_1^t, M_2^t, \cdots, M_N^t$ indicates the hidden motion states for pedestrians $1, 2, \cdots, N$ and $\alpha_{11}, \alpha_{12}, \cdots, \alpha_{1N}$ denotes the importance of the corresponding pedestrian with respect to pedestrian 1. 86

4.7 The framework of Global Scene Layout Encoder. The inputs are the observed scene images at each time step. A CNN is used to extract the scene layout features, which are then fed to LSTMs to compute the hidden state vectors. GAT is applied to model the global human-scene interactions. . . . 87

4.8 The framework of Local Patch Context Encoder. The inputs are the observed certain-sized local image patches that centered on the position of each pedestrian at each time step. A CNN is used to extract the local context features, which are then fed to LSTMs to capture the hidden state of local context. GAT is applied to model the local human-scene interactions. An extra set of LSTMs are designed to learn the temporal correlations of these interactions. The tensors from the upper level LSTMs and the lower level LSTMs are concatenated to form the output of the Local Patch Context Encoder. 90

4.9 Qualitative examples of our method predicting future 12 time steps trajectories, given previous 8 time steps ones on ETH and UCY dataset. Note that, we crop and resize the original image for better visualization. 102

4.10 Qualitative examples of our method predicting future 12 time steps trajectories, given previous 8 time steps ones on Town Centre (1st row) and Grand Central Station (2nd row) dataset. Note that, we crop and resize the original image for better visualization. 103

ABSTRACT

Human motion, behaviors, and intention are governed by human perception, reasoning, common-sense rules, social conventions, and interactions with others and the surrounding environment. Humans can effectively predict short-term body motion, behaviors, and intention of others and respond accordingly. The ability for a machine to learn, analyze, and predict human motion, behaviors, and intentions in complex environments is highly valuable with a wide range of applications in social robots, intelligent systems, smart manufacturing, autonomous driving, and smart homes. In this thesis, we propose to address the above research question by focusing on three important problems: human pose estimation, temporal action localization and informatics, human motion trajectory and intention prediction.

Specifically, in the first part of our work, we aim to develop an automatic system to track human pose, monitor and evaluate worker's efficiency for smart workforce management based on human body pose estimation and temporal activity localization. We have developed a deep learning based method to accurately detect human body joints and track human motion. We use the generative adversarial networks (GANs) for adversarial training to better learn human pose and body configurations, especially in highly cluttered environments. In the second step, we have formulated the automated worker efficiency analysis into a temporal action localization problem in which the action video performed by the worker is matched against a reference video performed by a teacher using dynamic time warping.

In the second part of our work, we have developed a new idea, called reciprocal learning, based on the following important observation: the human trajectory is not only forward

predictable, but also backward predictable. Both forward and backward trajectories follow the same social norms and obey the same physical constraints with the only difference in their time directions. Based on this unique property, we design and couple two networks, forward and backward prediction networks, satisfying the reciprocal constraint, which allows them to be jointly learned. Based on this constraint, we borrow the concept of adversarial attacks of deep neural networks, which iteratively modifies the input of the network to match the given or forced network output, and develop a new method for network prediction, called reciprocal attack for matched prediction. It further improves the prediction accuracy.

In the third part of our work, we have observed that human's future trajectory is not only affected by other pedestrians but also impacted by the surrounding objects in the scene. We propose a novel hierarchical framework based on a recurrent sequence-to-sequence architecture to model both human-human and human-scene interactions. Our experimental results on benchmark datasets demonstrate that our new method outperforms the state-of-the- art methods for human trajectory prediction.

Chapter 1

Background and Introduction

1.1 Deep Neural Networks

Deep learning is a sub-branch of a broader family of machine learning methods based on artificial neural networks which is inspired from the biological brain structure. It is a process of inducing intelligence into a machine by using complicated structure with multiple processing layers and non-linear transformations. Due to the substantial improvement of hardware computing capabilities and the development of large data sets, deep learning has played an irreplaceable role in both academia and industry. Deep neural networks such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) and Generative Adversarial Networks (GANs) have been widely applied and demostrated success in various fields such as image processing, computer vision, natural language processing. In this section, we will make a brief introduction of CNNs, LSTM, and GANs in the following.

1.1.1 Convolutional Neural Networks

CNNs is a class of Multi-layer Perceptron (MLP) which consists of an input layer, multiple hidden layers, and an output layer. The hidden layers usually contain three primary layers, such as convolutional layers, pooling layers, and fully-connected layers. An example architecture of CNNs is presented in Figure 1.1.

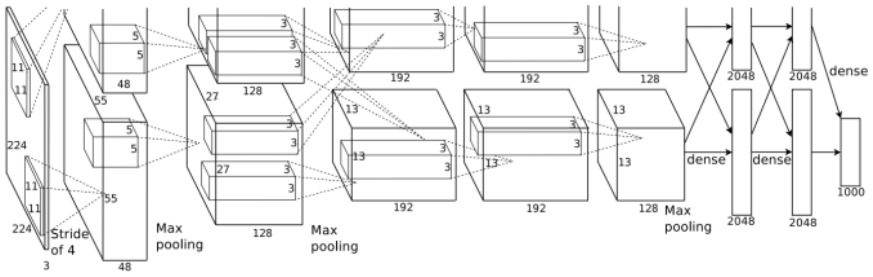

Figure 1.1: An example architecture of CNNs from AlexNet [1]

Covolutional layers are the essential building blocks of CNNs, which perform convolution operation to the input with multiple learnable filters and pass its results to the next layer. Mathematically, convolution is a sliding dot product between the entries of the filter and the input, which generates a activation map for each filter. The output is formed by stacking all activation maps along the depth dimension of the input.

Pooling layer usually follows the convolutional layer and performs a non-linear down-sampling operation to the input. It reduces the spatial dimension of the intermediate representation by combining the input of neuron clusters into a single neuron and passing to the next layer, so the amount of parameters of the network is reduced significantly. However, the important information is still retained. Also, a pooling layer introduces non-linearity in the network to bring the better learning ability for the real world data.

Fully connected layers usually comes after several convolutional and pooling layers. It

is similar with the traditional MLP, where each neuron in this layer has fully connections to the output from the previous layer. Therefore, the fully connected layer has much more parameters than the convolutional layer.

1.1.2 Long Short-Term Memory

Long short-term memory (LSTM) [5] is a special kind of recurrent neural network (RNN). It is widely used for process single data points like images and sequence data like speeches. A typical LSTM unit consists of a cell which is the memory part of the LSTM unit and three gates, such as an input gate, a forget gate and an output gate, which controls the flow of information inside the unit.

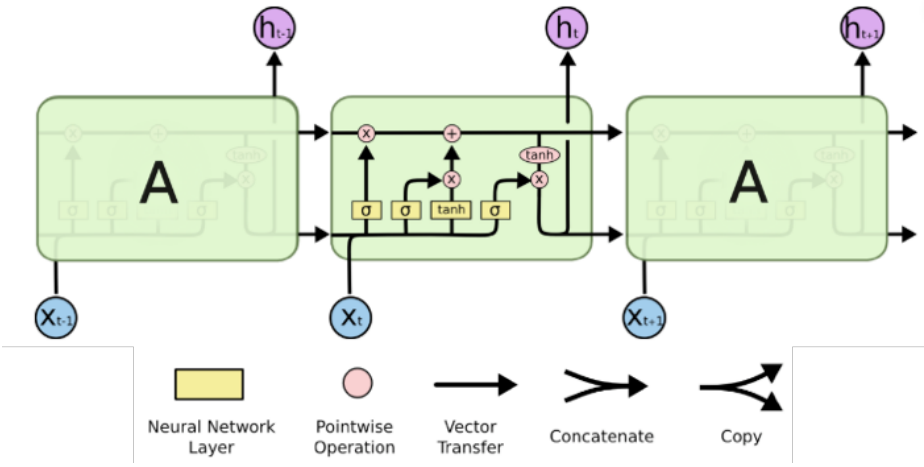

Figure 1.2: An example architecture of an LSTM unit.

The core idea of LSTM is the cell state, it keeps tracking the long and short dependencies over arbitrary time intervals in the input sequence. The gate mechanism controls the information flow to the cell state. The input gate is used to decide what information we need to store in the cell, the forget gate controls what information we need to throw

away from the cell and the output gate controls the extent of the information is used to compute the output. The extent of the information flow is often decided by using logistic sigmoid activation function. With the great ability to learn long-term dependencies, LSTM works better than standard RNN in many tasks when RNN encounters gradient vanishing problem.

1.1.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) belongs to the family of machine learning frameworks, which was introduced by Ian Goodfellow *et al.* in [6]. GANs are algorithm architectures that consist of two neural networks, contesting with each other in order to generate new, synthetic instances of new data with the same statistics as the real data. One neural network, called the generator, generates candidates while the other network, called the discriminator evaluate them for authenticity [6]. An illustrative figure of GANs are presented in Figure 1.3.

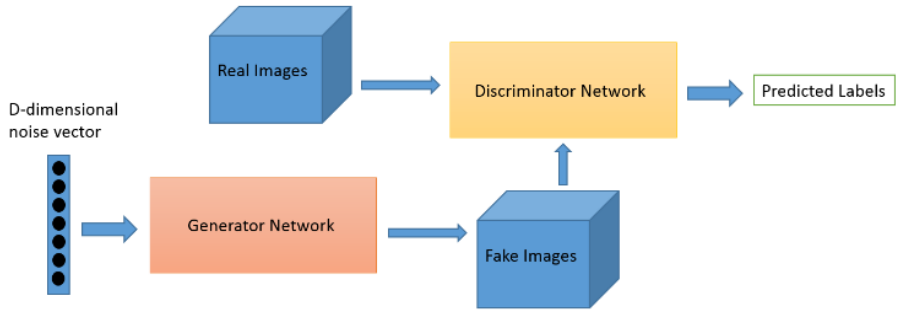

Figure 1.3: An example architecture of GANs [2].

Specifically, the goal of generator network is to learn to map from a latent space to a data distribution of interest, while the discriminator network is trained to distinguish candidates

generated by the generator from the real data distribution. Backpropagation mechanism is employed in training both networks. The generator network's training objective is to generate more realistic candidates to "fool" the discriminator network. Simultaneously, the discriminator network is trained to become more skilled at classifying the generated candidates into "fake" or real data [6, 7].

Generative Adversarial Networks (GANs) have been widely used and achieved impressive results in representation learning [8–10], image translation [11, 12] and image synthesis [13–16].

1.2 Human Behavior Understanding

Human behavior understanding is a critical part of real-world applications, such as video surveillance [17], human-machine interactions [18], smart manufacturing[19], and many others [20]. In our research, we are committed to applying the recent development of deep neural networks to help smart manufacturing. As we all know, smart manufacturing requires smart workforce management [19]. Monitoring operation efficiency of workers is the central component in workforce management. Current practice often relies on subjective visual monitoring by supervisors. For efficient management, it is highly desirable to develop an automated system for real-time worker monitoring and operation efficiency evaluations. This will provide fine-grain data for worker training, performance monitoring, efficiency optimization, and manufacturing management. We aim to develop one of the first automatic systems for real-time worker monitoring and operation efficiency evaluations in a manufacturing environment based on human pose estimation and temporal activity localization. In this following, we will briefly introduce two major research topics, human pose

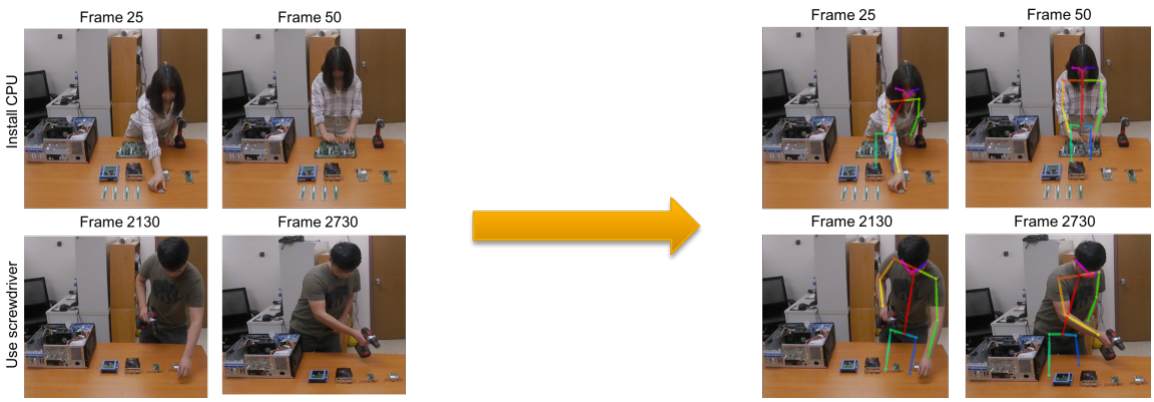

Figure 1.4: An illustration of the task of human pose estimation.

estimation and temporal activity localization.

Human pose estimation is a one of the key research topics in the field of human behavior understanding in recent years. The goal of this task is to estimate the configuration of body parts and determine the precise pixel locations of body keypoints from the input image or video sequences [21]. An illustration represents the task of human pose estimation is shown in Figure. 1.4.

Human pose estimation is often regarded as a regression problem. A considerable amount of previous studies have primariy concentrated on deep neural networks and achieved significant progress. *DeepPose* [22], opens the door to apply a deep neural network to extract features and directly regress locations of body joints from a RGB image. Also some later studies such as Tompson *et al.* [23] argued that it is more efficient to regress the input image into a set of heatmaps where each heatmap represent a body part.

To handle partially occluded body joints and limbs, existing regression-based pose estimation methods try to learn a body configuration model to infer their locations [24]. In our experiments, we recognize that they cannot efficiently handle fully occluded limbs, which

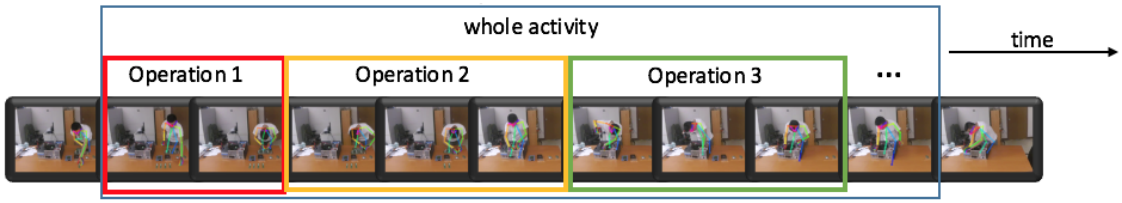

Figure 1.5: An illustration of the task of temporal action localization.

occur quite often in practical scenarios, especially when the person is moving around in the manufacturing floor. Ideally, we wish that the training data contains images of all different poses of human body, including samples with fully occluded limbs. In this way, the deep neural network can be carefully designed and trained to predict the body joints of these fully occluded limbs. However, this is a nearly impossible task in practice since persons with free-style motion will have a wide variety of body poses being occluded by different objects, especially in highly cluttered environments, such as the manufacturing floors filled with machines. In the training data, some typical body poses are dominating while difficult cases are very rare. This poses a significant challenge for learning highly efficient human pose estimation. To address this issue, we develop a deep learning-based method to accurately detect human body joints and track its motion. We use the generative adversarial networks (GANs) [6] for adversarial training to better learn human pose prior and body configurations, especially in highly cluttered environments.

Another key aspect to understand human behavior is to decompose human behavior into as set of activities. Therefore, the topic of temporal activity localization has also drawn a lots of attentions in past few years [25–28]. The task for human activity detection is to localize the temporal boundary such as start and end frame of an activity among long and untrimmed videos. An illustration of the task of human pose estimation is shown in Figure. 1.5.

One typical pipeline to deal with this problem is to firstly rely on deep neural network to produce a set of temporal proposals of varying duration from the input video sequence. Thereafter, a activity classifier and a completeness classifier are built to select the right boundary of a certain activity among generated proposals [20, 28].

A manufacturing task, for example, assembling a 3-D printer or a vehicle door, often consists a number of actions to be performed in a sequential manner. To measure the efficiency, we propose to compare the amount of time used by the worker to perform each action. In this work, a proficient worker, referred to as the *teacher*, is able to perform these actions accurately and efficiently. By *accurately*, we mean that each action is performed without errors, no action is missing, and all actions are performed in a correct sequential order. With this, we formulate the worker efficiency analysis problem into a cross-video matching problem: performing action-level matching between the teacher video and the worker video so as to locate the beginning and end of each action performed by the worker and compare its action time against the teacher's to measure the operation efficiency. We will describe in detail about our method in Chapter 2.

1.3 Human Trajectory Prediction

In addition to human behavior understanding, we are also very interested in predicting human future trajectory. Learning and predicting human motion trajectories in complex environments plays an important role in autonomous driving systems [29], social robots [30], human-machine interactions, and smart environments [17, 18]. Human beings have the intelligence to understand the moving patterns and intentions of surrounding persons in the environment and act appropriately to avoid collision and follow social norms. Given

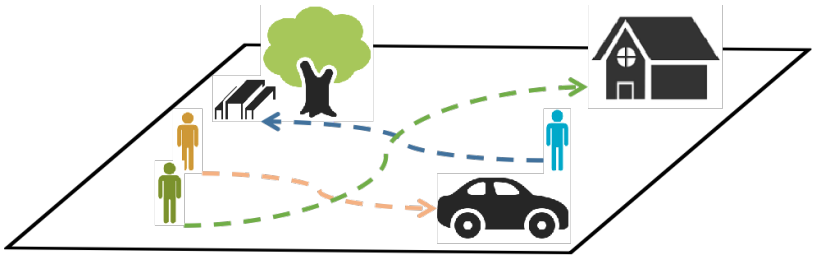

Figure 1.6: Human's future trajectory is affected not only by other pedestrians but also by the surrounding environment. Our proposed method predicts socially and physically plausible trajectories by hierarchically modeling the influence of all pedestrians in the scene as well as the global scene layout and local context.

the observed trajectories of a bunch of pedestrians involved in a crowded scenario, can a machine or robot predict their future trajectories by specifying their world coordinates on a 2D map representation?

The problem of human trajectory prediction is different from person tracking [30]. It needs to learn the human decision and behaviors in complex environments to predict future motion trajectory during the next period of time (e.g., 5 seconds), instead of the next time instance. As shown in Figure .1.6, we recognize that human motion trajectories and motion patterns are governed by human perception, behavioral reasoning, common sense rules, social conventions, and interactions with others and the surrounding environment [17, 29].

An efficient algorithm for human trajectory prediction needs to accomplish the following tasks: (1) *Obeying physical constraints of the environment.* To walk on a feasible terrain and avoid obstacles or other physical constraints, we need to analyze the local and global spatial information surrounding the person and pay attention to important elements in the environment. (2) *Anticipating movements of other persons or vehicles and their social behaviors.* Some trajectories are physically possible but socially unacceptable. Human motions are governed by social norms, such as yielding right-of-way or respecting personal

space. To capture and model them is a non-trivial task. (3) *Finding multiple feasible paths.* There are often a number of choices of moving trajectories for us to reach to the destination. This uncertainty poses significant challenges for accurate human trajectory prediction.

To handle these challenges, a bunch of traditional methods [31–33] has been focused on manually designing behavior models and energy functions. These methods usually fail in crowded scenarios with complex scene structures since they did not learn human movement behaviors from the training data. Recently, a number of methods based on deep neural networks have been developed for human trajectory prediction [34, 35]. Earlier methods have been focused on learning dynamic patterns of moving agents (human and vehicles) [34] and modeling the semantics of the navigation environment [36]. More recent approaches incorporate interactions between all agents in the scene into the analysis in order to predict the future trajectory for each agent. Methods have been developed to model human-human interactions [37], understand social acceptability using data-driven techniques based on Recurrent Neural Networks (RNNs) [35, 38, 39], and model the joint influence of all agents in the scene [40]. Methods have also been developed to predict multiple feasible paths of human [35, 39, 41].

Parallel to learn the influence of other pedestrians, modeling human-scene interactions is also quite helpful to better understand human behaviors and movements. A relatively small number of recent work have studied the impact of scene structures [42]. [41] proposed to use a hand-designed CNN to extract the scene layout features and incorporate the human-human interactions to refine the trajectory prediction. [43, 44] utilized the VGGNet-19 network [45] to capture the physical scene cues from raw scene images and applied soft attention to highlight the important regions. However, these methods either used the same scene context vectors for all pedestrians or did not model the influence of scene context

from different time steps in temporal wise.

In our research, we recognize that human trajectories, governed by social norms and constrained by physical structures of the surrounding environment, are not only forward predictable, but also backward predictable. Motivated by this observation, we develop a new approach, reciprocal twin networks, for human trajectory prediction. The forward and backward prediction networks, are tightly coupled together, which allows them to be jointly learned for accurate and robust human trajectory prediction. The detail of this methods is illustrated in Chapter 3.

All above methods model the human-human interaction by the so-called pooling mechanism that shares the latent motion dynamics of pedestrians represented with the hidden stats of LSTMs in a certain sized neighborhood. However, *Social-LSTM* just models interactions within a limited local scope. Gupta *et al.* [40], Sadeghian *et al.* [43] and our previous work [46] adopted the GANs (Generative Adversarial Networks) [6] to learn the multimodal trajectory distributions and utilized the social pooling mechanism that takes all pedestrians involved in the whole scene into consideration. These methods also have limitations. [40] and our previous method [46] lose information by using the same social vector for all pedestrians in a scene [44]. [43] manually designed a sorting operation that may not generalize to all cases.

To further improve the performance of human trajectory prediction, we propose a novel hierarchical attention-based framework based on a recurrent sequence-to-sequence architecture to jointly model human-human interactions and human-scene interactions to generate more accurate and plausible future trajectories. We leverage GAT (graph attention networks) [47] to assign the different and adaptive importance to the surrounding pedestrians and environment [48]. Because of the unequal contribution, the attention mechanism

is more helpful than the pooling mechanism to encode the relative influences and the spatial interactions [48]. We also observed that not only the spatial interactions at the same time step is important but also the temporal continuity of interactions play a critical role in predicting the future movements. Therefore, temporal correlations of these interactions are also leaned in this work. We present this novel method in detail in Chapter 4.

1.4 Summary

This chapter introduces some background knowledge of deep neural networks and high-level information about our research topics related human pose estimation, temporal action localization and human future trajectory prediction. We recognize that there is an urgent demand to develop some high-performance algorithms regards these topics in various fields, such as smart manufacturing, autonomous driving system and social robots, is emerging.‘ In the following of this dissertation, we first introduce our automatic system to monitor and analysis human's behavior for smart manufacturing based on human pose estimation and temporal activity localization in Chapter 2. Then, we present a new approach, reciprocal twin networks, for human trajectory prediction in Chapter 3. Finally, a novel attention-based approach that further improves the performance of human trajectory prediction is illustrated in Chapter 4.

Chapter 2

Automated Work Efficiency Analysis for Smart Manufacturing Using Human Pose Tracking and Temporal Action Localization

2.1 Motivation

In this chapter, we aim to develop an automatic system to monitor and evaluate worker's efficiency for smart workforce management based on human body pose estimation and temporal activity localization.

Our proposed method for automated worker efficiency analysis consists of two major steps. In the first step, we develop a deep learning-based method to accurately detect human body joints and track its motion. We use the generative adversarial networks (GANs) [6] for adversarial training to better learn human pose prior and body configurations, especially in highly cluttered environments. In the second step, we formulate the automated worker

efficiency analysis into a temporal action localization problem in which the action video performed by the worker is matched against a reference video performed by a teacher using dynamic time warping.

To measure the efficiency, we propose to compare the amount of time used by the worker to perform the action against the action time used by the teacher.

Compared to existing work, the major contributions of this work are summarized as follows: (1) We develop a generative adversarial learning method for efficient human pose estimation in highly cluttered environments. (2) We formulate the problem of operation efficiency analysis problem into action-level cross-video matching between the worker and teacher and developed an effective solution based on dynamic time warping. (3) We establish a method for automated worker efficiency analysis which provides fine-grain data for smart workforce management.

The remainder of this chapter is organized as follows. Section 2.2 reviews previous work on human pose estimation and temporal localization. Section 2.3.1 presents the details of human pose extraction based on our proposed generative adversarial training method. Section 2.3.2 explains our body pose feature extraction and dynamic time warping for action level video matching. Section 2.3.3 summarizes our algorithm for automated worker efficiency analysis. Experimental results are presented in Section 2.4. Section 2.5 concludes the paper and discusses future work.

2.2 Related Work

Our work is closely related to human pose estimation and temporal activity localization.

2.2.1 Human Pose Estimation

The task of human pose estimation is to determine the precise pixel locations of body keypoints from a single image [21]. Similar with saliency detection [49, 50], pose estimation needs to highlight and separate the the salient body joint regions from background in images [51]. Since the work of *DeepPose* [22], human pose estimation has recently achieved significant progress with deep convolutional neural networks. Human pose estimation is often formulated as a regression problem, predicting locations of body joints from deep neural network features [22]. DeepPose uses a deep neural network (DNN) to directly regress the coordinates of body joints. Tompson *et al.* [23] argued that it is more efficient to use DNNs to regress heatmap images at multiple scales. Luvizon *et al.* [52] designed a differentiable regression-based network by using a soft-argmax function to transform heatmaps to human body joint coordinates. Nibali *et al.* [24] transform human body joint positions from heatmaps by constructing a differentiable spatial to numerical transform (DSNT) layer [53]. While body models are not necessary for effective part localization, constraints between parts allow us to assemble independent detection results into an accurate body configuration. Detection-based methods are relying on powerful DNNS to detect body parts and then combine them into a human pose using a graphical model [54–58], where the input image is the raw image and the output image is a set of heatmaps that represent the estimated human pose.

In this work, we incorporate the generative adversarial networks (GAN) [6] into our human pose estimation. Instead of generating synthetic data purely from noise as in the vanilla GAN, conditional GAN [59] uses real data as condition. One example work is the image to image translation [60]. Here, the generator is fed with a pair of images A and B during training, where the conditional GAN learns to generate B from A, or vice versa. A

similar strategy has been adopted by [61] to generate synthetic human pose images, while the discriminator aims to distinguish real and synthesis pose images. Chou *et al.* [62] proposed to use two identical Hourglass networks [63] as generator and discriminator. The generator is used to generate the heatmap location of each joint, while the discriminator is used to distinguishes real heatmaps from generated heatmaps. We propose to incorporate the conditional GAN into the human pose estimation framework that can infer augmented training data from ground truth annotations during the training process. This adversarial training method will be able to better learn human pose prior and body configurations, especially in highly cluttered environments.

2.2.2 Temporal Activity Localization

Temporal activity localization in continuous videos is a challenging and interesting problem in computer vision area [64]. During the past a few years, a number of algorithms have been developed. A sliding temporal window approach with a greedy non-maximum suppression has been used to locate the action segments in long videos [64]. For spatiotemporal action localization, the number of windows can be reduced by finding action proposals [65] or optimal action tubes using branch-and-bound methods [66]. [67] developed a hidden Markov model for action segmentation and localization. [68] proposed a semi-Markov model which combines three different types of features: segment boundaries, segment content, and interactions between neighboring segments. The action localization is then formulated into a max-margin problem [68].

Motivated by the success of deep neural networks (DNNs) [1], [69] developed a two-stream DNN to learn features from still images and motion flow simultaneously. A 3-D DNN has been designed in [70] to learn spatio-temporal features for activity localization.

It has been recognized that DNN-based activity localization methods require a large number of annotations to train the action detectors and localization module, which are often hard and time-consuming to obtain in practice. To address this issue, Wang *et al.* [71] proposed to utilize auxiliary data to transfer subspace clustering knowledge from source sequences to target sequences to improve activity localization performance. [72] proposed a temporal pooling scheme which is incorporated with dynamic clustering to build a hierarchical framework to achieve accurate temporal action localization from a sequence of local features.

2.3 The Proposed Method

In this work, we develop an intelligent system to perform automated monitoring and evaluation of operation efficiency of workers for smart manufacturing. An overview of the system is illustrated in Figure. 2.1. We formulate the worker efficiency analysis problem into action-level cross-video matching between the teacher and the worker. Specifically, the teacher video for a specific manufacturing task is pre-segmented and labeled with a sequence of actions. The start time, end time, and time duration of each action performed by the teacher are extracted. We perform deep learning based human body pose estimation on both teacher and worker videos, detect body joints, and track their motion. We then extract pose features to characterize both video sequences. Using dynamic time warping, we perform feature sequence matching between these two videos. Based on this matching result, we can then obtain the start time, end time, and time duration of each action performed by the worker, compute their relative differences from the teacher action sequence, and determine the operation efficiency of the worker. In the following sections, we will

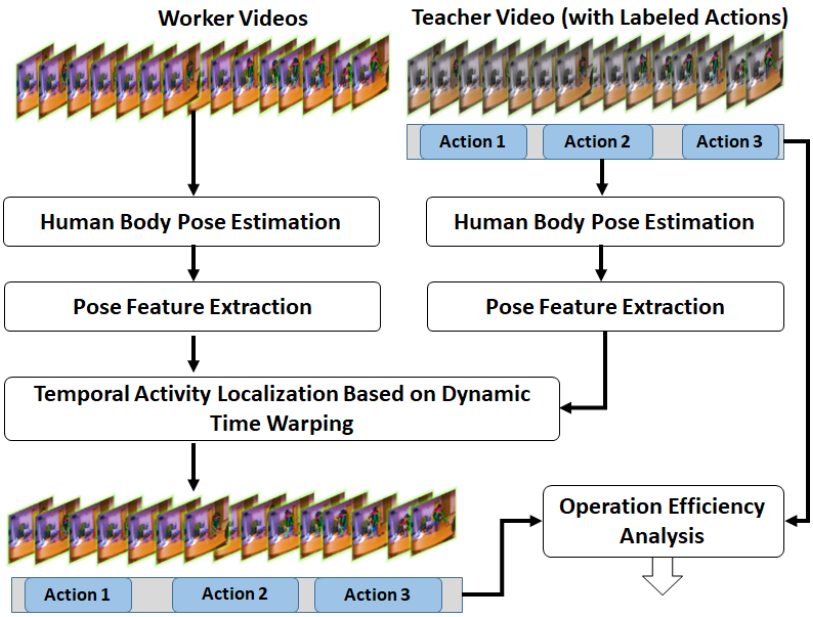

Figure 2.1: Overview of our worker efficiency evaluation system.

explain these major algorithm components in more details.

2.3.1 Human Body Pose Tracking

The first component of our proposed method is human body pose estimation and tracking. Existing regression-based pose estimation methods work well with visible limbs. To handle partially occluded body joints and limbs, they try to learn a body configuration model to infer their locations [24]. In our experiments, we recognize that they cannot efficiently handle fully occluded limbs, which occur quite often in practical scenarios, especially when the person is moving around in the manufacturing floor. Ideally, we wish that the training data contains images of all different poses of human body, including samples with fully occluded limbs. In this way, the deep neural network can be carefully designed and trained

to predict the body joints of these fully occluded limbs. However, this is a nearly impossible task in practice since persons with free-style motion will have a wide variety of body poses being occluded by different objects, especially in highly cluttered environments, such as the manufacturing floors filled with machines. In the training data, some typical body poses are dominating while difficult cases are very rare. This poses a significant challenge for learning highly efficient human pose estimation.

To address this issue, we propose to incorporate generative adversary training into human pose estimation and train the following three networks jointly: the human pose estimator, the semantic data generator, and the semantic data discriminator. Generative Adversarial Networks (GAN) [6] consist of two models that are trained in an alternative manner. The generator G is optimized to reproduce the true data distribution p_{data} by generating data that are hard for the discriminator D to differentiate them from real data. Meanwhile, D is optimized to distinguish real data from synthetic ones generated by G. The overall training procedure is similar to a two-player min-max game with the following objective function,

$$\begin{aligned} \min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \\ \mathbb{E}_{z\sim p_z}[\log(1-D(G(z)))], \end{aligned} \tag{2.1}$$

where x is an instance from the true data distribution p_{data}, and z is a noise vector sampled from distribution p_z. Conditional GAN [59] is an extension of GAN where both the generator $G(z,c)$ and discriminator $D(x,c)$ receive additional conditioning variables c, thus allowing G to generate images conditioned on variables c.

In this work, the generative (G) network augments the training data and enforces the pose estimator to estimate the joints more precisely. The discriminative (D) network evaluates the conditional pair, i.e., the training images and the estimated pose heatmaps, to

enforce the G network to generate semantically similar human pose images. An overview of the human pose estimation system is illustrated in Figure. 2.2. An RGB image is fed into the feature extraction module (FE) for visual feature extraction, and then fed into the pose estimation module to infer pose heatmaps. The visual features will be concatenated with the estimated heatmaps and the ground truth heatmaps to form an overall conditioned feature vector C, where the estimated heatmaps serve as the conditioning augmentation. Since the pose condition is naturally provided by the Gaussian heatmaps, so we donot need to have an additional Gaussian function to generate pose embedding [59]. The conditioned features C will be fed into the discriminator and generator networks to generate synthetic pose images. The discriminator is a matching-aware discriminator. The positive pair is the real image paired with the ground truth pose P_t concatenated with the extracted visual features V. The negative pairs have two groups. The first group is the real image paired with the **estimated pose** P_e concatenated with V. The second group is the synthetic image paired with P_t and V.

We use a modified version of the hourglass network [63] as our human pose estimator. The hourglass design is a state-of-the-art architecture for bottom-up and top-down inference with residual blocks. It processes input images at multiple scales with down-scaling and up-scaling. In this work, We find out that, by replacing the residual block in the hourglass network with an inception-residual [73] block, we can achieve improved accuracy in estimating occluded poses. The overall structure is shown in Figure. 2.3.

Inspired by the work of [74] which uses a conditional Variational Auto-encoder (VAE) to map the raw image into the latent space, we propose to use an hourglass network to extract latent representations of visual features in order to semantically generate image samples that follow the same distribution of the real data. The learned latent representation

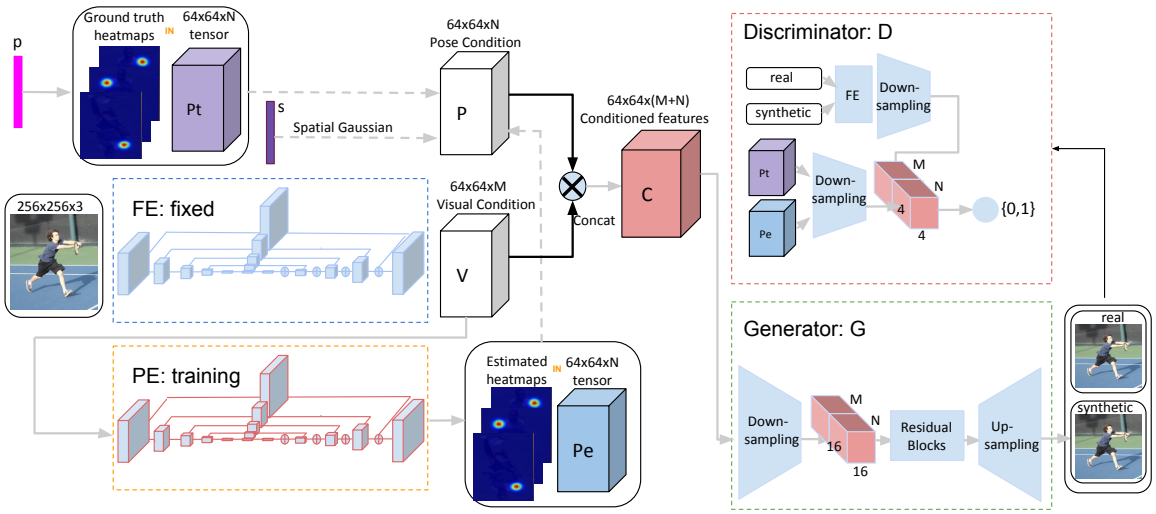

Figure 2.2: The overall framework of our proposed poseGAN for the joint human pose estimation and conditional image synthesis. It includes the following components: Generator (G), Discriminator (D), Feature Extractor (FE) and Pose Estimator (PE). M and N are the number of channels of a tensor. N is the number of keypoints to be predicted. During training, the discriminator D takes real images and their corresponding ground truth annotations P_t as positive sample pairs, whereas negative sample pairs consist of two groups. The first is real images with mismatched pose estimations P_e, while the second is synthetic images with their corresponding true annotations P_t.

V has the same resolution as the pose estimation results. The variable P follows a multivariate Gaussian distribution. It is a deterministic latent encoding of the human pose that we use to condition the high-level visual features extracted from the raw image by the Feature Extraction (FE) network. It is pre-trained jointly with the pose estimator (PE). We then fix the FE and start the training of the discriminator (D) and generator (G) networks.

During the GAN-based human pose learning process, both the generator and the pose estimator work towards the goals of generating better pose conditions and more plausible images. For the generator, it has two tasks. Specifically, it needs to generate images that are both visually reasonable and pose-wise correct. For the pose estimator, it has two tasks as well. The first one is to generate good pose estimation. Its performance is often measured by the Euclidean distance between the estimated body joint locations and the ground truth ones. The second one is to generate good poses for constructing feasible human image samples. This second task complements the first one because, in occlusion conditions, it is hard to infer the occluded body joint locations with a location-wise loss function because the these body joints are not visible. By constructing a feasible image, the pose estimator and the generator learns to cope with occluded parts.

In the following, we explain the specific loss functions to train these networks. Let I be the input RGB image and P_t be the ground truth heatmap. We use P to denote the non-linear functionality of the *Pose Estimation Network*. $P(I)$ represents the predicted heatmap of body joints from the input image I. P_e is used to simplify the denotion of $P(I)$. The *Generator* and *Discriminator* networks are denoted by G and D, respectively. The overall

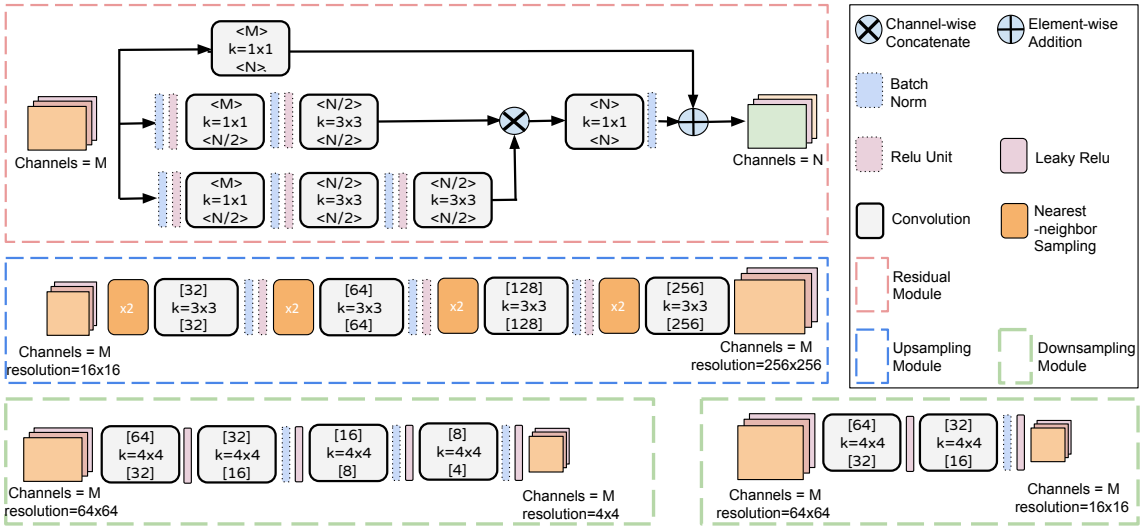

Figure 2.3: **Implementation details of Generator (G) and Discriminator (PE)**. In each module, the top notation $[M]$ indicates the resolution of the input tensor, while the bottom $[N]$ denotes the resolution at the output tensor of this module. The up-sampling blocks consist of the nearest-neighbor upsampling followed by a $3x3$ stride 1 convolution. Batch normalization and ReLU activation are applied after every convolution except the last one. The residual blocks consist of $3x3$ stride 1 convolutions, Batch normalization and ReLU. Two residual modules are used in $128x128$ models while four are used in $256x256$ models. The down-sampling blocks consist of $4x4$ stride 2 convolutions, Batch normalization and LeakyReLU, except that the first one does not have Batch normalization.

loss is the sum of GAN loss, the pose estimator loss, and L_1 loss:

$$\begin{aligned}\mathcal{L}(G, P, D; I, P_t) =& \mathcal{L}_{\text{GAN}}(G, P, D; I, P_t) \\ &+ \mathcal{L}_{\text{Pose}}(P; I, P_t) \\ &+ \lambda \mathcal{L}_1(G; I, P_t),\end{aligned} \tag{2.2}$$

where the GAN loss is the sum of the generator loss and discriminator loss:

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, P, D; I, P_t) = \mathcal{L}_{\text{G}}(G, P; I, P_t) + \\ \mathcal{L}_{\text{D}}(D, P; I, P_t).\end{aligned} \tag{2.3}$$

Following the prior work [12], the generator loss is to guide the *G* networks infer conditioned synthetic images that can hardly be distinguished by the Discriminator. That is, the fake pair $\hat{p}$ is encouraged to be real in the Discriminator output.

$$\mathcal{L}_{\text{G}}(G, P; I, P_t) = -\log(\hat{p} + \epsilon), \tag{2.4}$$

where the fake pairs are defined as:

$$\hat{p} = D(G(I, P_t \otimes V), P_t) \; + \; D(I, P_e). \tag{2.5}$$

$\otimes$ is used here to denote the concatenation operation of tensors. In practice, in order to prevent the logarithmic function to explode with input value zero, we add a constant as $\epsilon = e^{-12}$. In our case, one constituent of the pairs corresponds to the real image paired with the estimated pose P_e concatenated with V, while the other corresponds to the synthetic image paired with P_t and V. The first group of fake pairs is helpful in the training of the

pose estimator. The second group, on the other hand, provides useful gradient signals to help creating better image synthesis.

The discriminator loss consists of D outputs of both real pair input p and fake pairs input $\hat{p}$:

$$\begin{aligned}\mathcal{L}_{\mathrm{D}}(D, P; I, P_t) = & \log(p + \epsilon) + \\ & \log(1 - \hat{p} + \epsilon),\end{aligned} \tag{2.6}$$

where the fake pairs are defined in Equation 2.5 and the real pair is the real image paired with the ground truth pose P_t concatenated with the extracted visual features V which is denoted as:

$$p = D(I, P_t). \tag{2.7}$$

The pose estimator loss is the standard MSE (mean squared error) loss for heatmap regression:

$$\begin{aligned}\mathcal{L}_{\text{Pose}}(P; I, P_t) = & \|P(I) - P_t\|_2 + \\ & \|P(G(I, P_t)) - P_t\|_2.\end{aligned} \tag{2.8}$$

The L_1 loss is to enforce the synthetic image to be not only semantically congruent to pose conditions, but visually similar to the original image:

$$\mathcal{L}_1(G; I, P_t) = |I - G(I, P_t)|_1. \tag{2.9}$$

In the experimental section, we will provide comprehensive evaluation of the proposed human pose estimation method on the benchmark dataset. Figure. 2.4 shows examples of body poses estimated by the proposed poseGAN method. With these accurately detected body joints in each frame of the teacher and worker videos, we will track their motion in the temporal domain, extract features, and perform cross-video action matching.

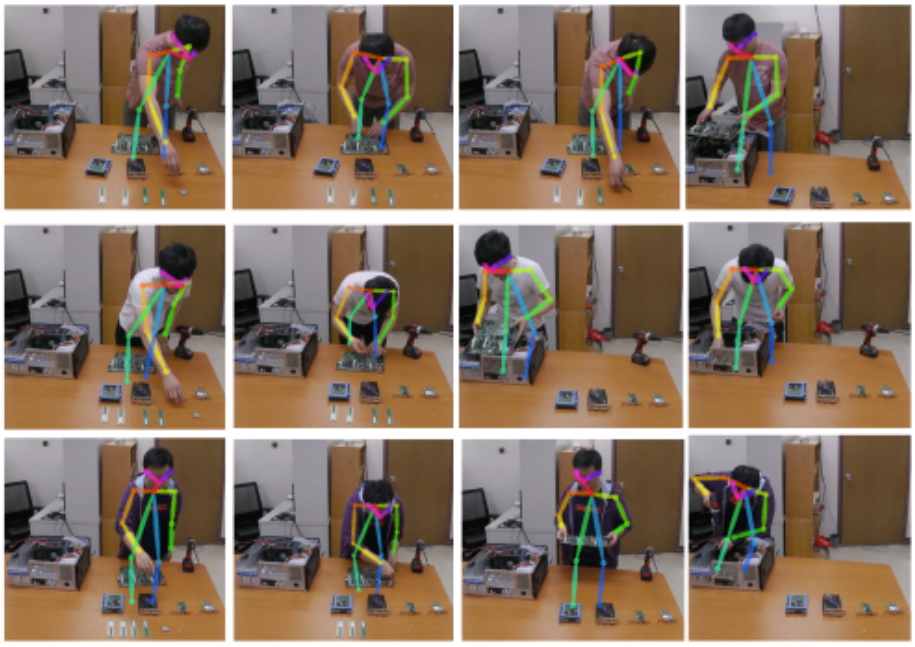

Figure 2.4: Examples of human poses obtained by our poseGAN algorithm.

2.3.2 Body Pose Feature Extraction and Dynamic Time Warping for Action Matching

In this section, we aim to characterize the action-level similarity between two segments from the teacher and worker videos. Specifically, given two video segments, what is the similarity between them from the human action perspective?

In our proposed method, we use the the poseGAN method presented in the previous section to detect the body joints in each video frame. In this section, we propose to design invariant features to represent this sequence of body joints and their motion for action-level cross-video matching. The 2-D skeleton model used in this work is shown in Figure. 2.5. It is organized in a tree structure. The joints of the lower body are denoted by J_9, J_{10}, J_{12}, and J_{13}. In manufacturing environments, these fours joints are often occluded by machines and desks. Keypoints on the head, including eyes and ears, are denoted by J_0, J_{14}, J_{15}, J_{16},

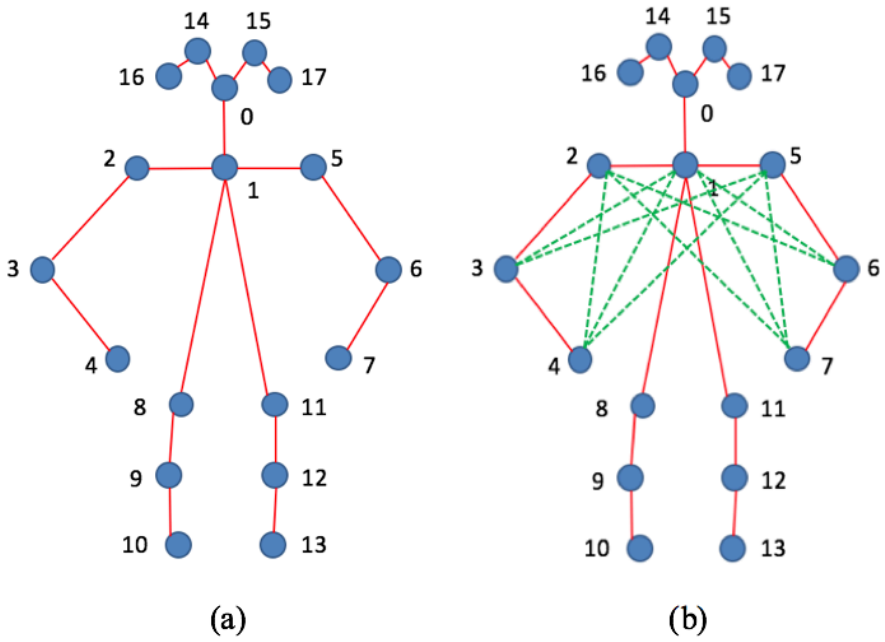

Figure 2.5: (a) Human skeleton model (blue spheres are joints and red lines are limbs). (b) Green lines generated to extract features

and J_{17}. Joints that are mostly visible in videos and important for recognize actions are head center J_0, body center J_1, arm joints (J_2, J_3, J_4, J_5, J_6, and J_7) and two knee joints J_8 and J_{11}, as illustrated in Figure. 2.5.

The following features are extracted from the sequence of detected body joints to represent the actions: (a) *(FA) normalized distance* $d(J_m, J_k)$ between joints J_m and J_k which includes J_1 and J_3, J_1 and J_4, J_2 and J_6, J_1 and J_7, J_2 and J_4, J_2 and J_6, J_2 and J_7, J_5 and J_3, J_5 and J_4, J_5 and J_7; (b) *(FB) normalized vector* between two joints J_m and J_k, which is defined by $V_{mk} = \overrightarrow{J_m J_k}/\|\overrightarrow{J_m J_k}\|$; (c) *(FC) joint moving distance* M_k^v of each joint J_k between the current frame and previous frame, where $0 \leq k \leq 17$ and v indicates the current frame; (d) *(FD) its moving direction* M_k^θ; and (e) *(FE) the relative angle* Θ_{mk} between joints J_m and J_k with respect to the body center J_1, where m and k indicate the joint index

from 2 to 7. Here, the distance vectors between joints are normalized with respect to the average should width which is defined as the Euclidean distance from J_2 to J_5.

The teacher video and the worker video are represented by sequences of joint features defined in the above, denoted by $\mathbf{F}_t(n)$ and $\mathbf{F}_w(n)$ respectively. The next step is to determine the sequence-level alignment between them. The actions performed by the teacher are denoted by A_k, $1 \leq k \leq K$. Action A_k corresponds to feature sequence segment $[\mathbf{F}_t(n_k), \mathbf{F}_t(n_k + \Delta_k)]$, where n_k is the start time (in frames) and Δ_k is the time duration of action A_k. Our task here is to identify the segments in the worker feature sequence $[\mathbf{F}_w(m_k), \mathbf{F}_w(m_k + \delta_k)]$ which match with the teacher actions $[\mathbf{F}_t(n_k), \mathbf{F}_t(n_k + \Delta_k)]$.

This is so-called temporal action localization problem which emerges as an important research problem in computer vision and machine learning [68]. This problem is challenging in our case since different workers may perform actions with different speeds from the teacher. The worker may have additional side actions between actions, causing significant interference during matching. To address this issue, we propose to explore the method of dynamic time warping (DTW). It uses a cost matrix to track the similarity between each pair of poses from these two sequences. The action-level matching and synchronization are determined by the monotonic path connecting starting and ending points of the cost matrix with the lowest accumulated distance or the highest sub-sequence similarity. Specifically, we represent the teacher and worker action video to be matched by two sequences or time series of feature vectors **X** and **Y**

$$X = [X_1, X_2, ..., X_{t_1}, ..., X_{T_1}], \tag{2.10}$$

$$Y = [Y_1, Y_2, ..., Y_{t_2}, ..., Y_{T_2}]. \tag{2.11}$$

X_{t_1} is the feature vector which combines features (FA) to (FE) for frame t_1 of the teacher

video. Y_{t_2} is the feature vector for frame t_2 in the worker video. T_1 and T_2 are the length of both videos. We first compute the local cost matrix $C_l \in \mathbb{R}^{T_1 \times T_2}$ where each entry of this matrix c_{t_1,t_2} represents the Euclidean distance between feature vectors X_{t_1} and Y_{t_2}.

$$c_{t_1,t_2} = \|X_{t_1} - Y_{t_2}\|_2. \tag{2.12}$$

From this local cost matrix, we try to find the alignment path which goes through the minimal cost positions on the matrix. We denote the alignment path to be obtained by our DTW algorithm by

$$\mathbf{W} = \{w(1), w(2), \cdots, w(K)\}. \tag{2.13}$$

The associated weight for $W(k)$ is

$$w(k) = c(x_{n_k}, y_{m_k}), n_k \in [1, T_1], m_k \in [1, T_2] \tag{2.14}$$

Then, the overall cost function is defined as:

$$C[p](X, Y) = \sum_{k=1}^{K} w(k) = \sum_{k=1}^{K} c(x_{n_k}, y_{m_k}) \tag{2.15}$$

The dynamic time warping algorithm aims to find the optimal warping path $\hat{p}$ to minimize the cost function, which can be solved with dynamic programming. We define $D(t_1, t_2)$ as the accumulated cost matrix, which is initialized with $D(1,1) = c_{t_1,t_2} = \|X_{t_1} - Y_{t_2}\|$. Then, during each stage of dynamic programming, we update the cumulative distance $D(t_1, t_2)$

as follows:

$$D(t_1, t_2) = \min\{D(t_1 - 1, t_2 - 1), D(t_1 - 1, t_2), \\ D(t_1, t_2 - 1)\} + c(X_{t_1}, Y_{t_2}). \tag{2.16}$$

Once the accumulated cost matrix is constructed, the optimal warping path p^* can be found by backtracking from the end point (T_2, T_1) to the start point $(1, 1)$. The total cost the optimal path will be used to measure the action-level similarity between two segments from the teacher and student videos.

2.3.3 Temporal Activity Localization

In the previous section, we have derive an action-level similarity measure between two video segments from the teacher and worker videos. From the optimal path, we can also identify which subset of frames achieves the best match. In this section, we will use this method to identify the matches of all teacher actions in the worker video. Specifically, the teacher has performed K actions $\{A_k\}$. Each action A_k is labeled in the teacher video for frames $[\mathbf{F}_t(n_k), \mathbf{F}_t(n_k + \Delta_k)]$ where n_k is the starting frame and $n_k + \Delta_k$ is the ending frame. During temporal activity localization, we start with the first action A_1. The input to our matching algorithm outlined in the above section will be two video segments: the first one is $[\mathbf{F}_t(n_k), \mathbf{F}_t(n_k + \Delta_k)]$ from the teacher video. The second one is from the worker video. For this worker video segment, we start with an initial guess $[\mathbf{F}_w(U_k), \mathbf{F}_w(V_k)]$. Initially, $U_1 = 1$. We can set V_1 to be a sufficiently large number which can cover the first action performed by the worker. We then apply the matching algorithm to find the optimal path based on dynamic time warping and determine the best matching frames for $[\mathbf{F}_t(n_k), \mathbf{F}_t(n_k+\Delta_k)]$, whose frame index range is denoted by $[\mathbf{F}_w(n_k), \mathbf{F}_w(n_k+\Delta_k)]$. Once

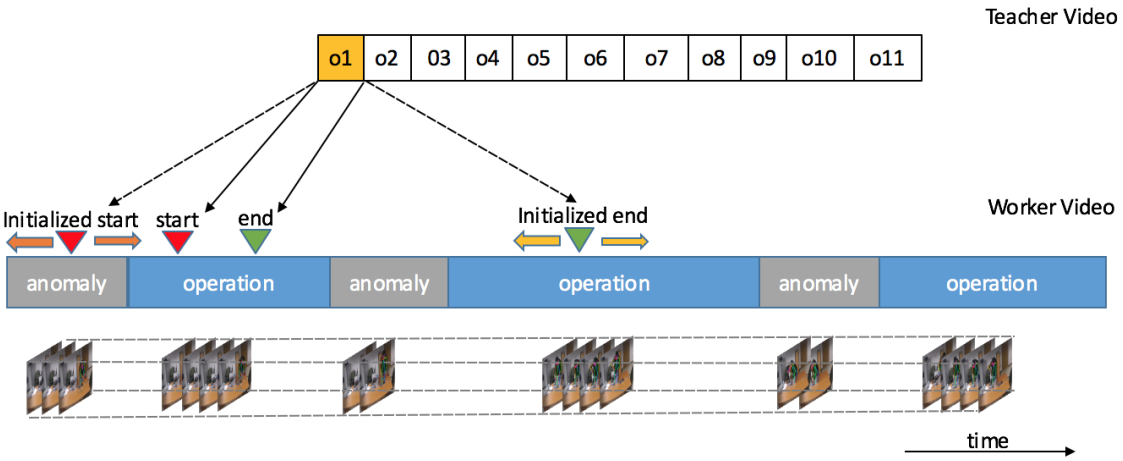

Figure 2.6: Temporal localization of worker actions.

the current action is matched, we can set $U_{k+1} = n_k + \Delta_k$ and V_{k+1} to be a number larger than U_{k+1} and being able to cover the next action. The overall procedure is illustrated in Figure. 2.6.

2.4 Experimental Results

Our experimental evaluation consists of two parts, evaluating the poseGAN method for human pose estimation on benchmark datasets and evaluating the accuracy of worker operation efficiency analysis.

2.4.1 Human Pose Estimation Evaluation

Datasets

We evaluate the proposed poseGAN on two widely used benchmark datasets: the MPII Human Pose [75] and extended Leeds Sports Poses (LSP) [76]. The MPII Human Pose

dataset has about 25K images with 40K annotated poses. The images are collected from YouTube videos covering daily human activities with highly articulated human poses. The LSP dataset consists of 11K training images and 1K testing images of sports activities.

Evaluation Criteria

Three widely used criteria are used in the literature to evaluate the performance of the proposed human pose estimation method: Percentage of Corrected Parts (PCP) [56], Percentage of Detected Joints (PDJ) [22], and Percentage of Corrected Keypoints (PCK) [56].

The PCP evaluates the localization accuracy of body parts. It requires the estimated part end points must be within half of the part length from the ground truth part end points [56]. It has been recognized that the PCP measure has the drawback of penalizing shorter limbs, such as lower arms. Thus, the PDJ measure is introduced in [22] to measure the detection rate of body joints, where a joint is considered to be detected if the distance between the detected joint and the true joint is less than a fraction of the torso diameter. The torso diameter is usually defined as the distance between two end joints on the human torso, such as left shoulder and right hip [22]. The PCK measure is very similar to the PDJ criterion. The only difference is that the torso diameter is replaced with the maximum side length of the external rectangle of ground truth body joints.

In our experiments, we follow the official benchmark evaluation protocols [75]. Official benchmark on MPII dataset adopts PCK_H (using portion of head length as reference) at 0.5, while official benchmark on LSP dataset adopts both PCP and PCK at 0.2. We use the PCK [56] metric for performance comparisons on the LSP dataset, and the PCK_H measure [75], where the error tolerance is normalized with respect to the head size, for performance comparisons on the MPII Human Pose dataset.

Table 2.1: Comparisons of PCK@0.2 score on the LSP test set.

Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
Wei *et al.* [77]	97.8	92.5	87.0	83.9	91.5	90.8	89.9	90.5
Insafutdinov *et al.* [78]	97.4	92.7	87.5	84.4	91.5	89.9	87.2	90.1
Pishchulin *et al.* [55]	97.0	91.0	83.8	78.1	91.0	86.7	82.0	87.1
Lifshitz *et al.* [79]	96.8	89.0	82.7	79.1	90.9	86.0	82.5	86.7
Belagiannis&Zisserman [80]	95.2	89.0	81.5	77.0	83.7	87.0	82.8	85.2
Yang *et al.* [81]	98.3	94.5	**92.2**	88.9	94.7	**95.0**	**93.7**	93.9
Chu *et al.* [82]	98.1	93.7	89.3	86.9	93.4	94.0	92.5	92.6
Bulat&Tzimiropoulos er al. [83]	97.2	92.1	88.1	85.2	92.2	91.4	88.7	90.7
Tang *et al.* [84]	97.1	**94.7**	91.6	89.0	93.7	94.2	93.7	93.4
Luvizon *et al.* [52]	97.4	93.8	86.8	82.3	93.7	90.9	88.3	90.5
Zhang *et al.* [85]	97.3	92.3	86.8	84.2	91.9	92.2	90.9	90.8
Ours	**98.4**	94.6	92.0	**89.5**	**95.0**	**95.0**	93.6	**94.0**

Results

Table 2.1 summarizes the PCK scores with a threshold of 0.2. Our approach outperforms several recent state-of-the-art methods with a PCK score of 94.0%. Table 2.2 shows the comparison of the PCKh performance of our method with previous state-of-the-art methods at a normalized distance of 0.5. Our algorithm achieves the state-of-the-art performance of 91.2% in the $PCK_H - 0.5$ score. Note that we test multiple scales (1 and 0.75) on both the MPII and LSP datasets. The occlusion of limbs will cause ambiguity in human pose estimation. In this work, using GAN-based image synthesis conditioned by human pose estimation and adversarial training, the pose estimator and the generator learns to cope with occluded parts. Our experiments on these two benchmarks show that this approach improves the overall performance.

Examples of pose estimation on the LSP and MPII datasets are shows in Figure. 2.7 and Figure. 2.8 respectively. Figure. 2.4 shows some examples of human poses in the manufacturing environment estimated by the proposed poseGAN method.

Table 2.2: Comparisons of PCKh@0.5 score on the MPII test set.

Method	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
Bulat&Tzimiropoulos *et al.* [83]	97.9	95.1	89.9	85.3	89.4	85.7	81.7	89.7
Wei *et al.* [77]	97.8	95.0	88.7	84.0	88.4	82.8	79.4	88.5
Insafutdinov *et al.* [78]	96.8	95.2	89.3	84.4	88.4	83.4	78.0	88.5
Rafi *et al.* [86]	97.2	93.9	86.4	81.3	86.8	80.6	73.4	86.3
Gkioxary *et al.* [87]	96.2	93.1	86.7	82.1	85.2	81.4	74.1	86.1
Lifshitz *et al.* [79]	97.8	93.3	85.7	80.4	85.3	76.6	70.2	85.0
Pishchulin *et al.* [55]	94.1	90.2	83.4	77.3	82.6	75.7	68.6	82.4
Newell *et al.* [63]	**98.2**	96.3	91.2	87.1	90.1	87.4	83.6	90.9
Tang *et al.* [84]	97.4	96.2	91.8	87.3	90.0	87.0	83.3	90.8
Yang *et al.* [88]	97.9	95.6	90.7	86.5	89.8	86.0	81.5	90.2
Zhang *et al.* [85]	98.3	**96.4**	91.5	87.4	**90.9**	87.1	**83.7**	91.1
Ours	98.1	96.3	**92.2**	**87.8**	90.6	**87.6**	82.7	**91.2**

2.4.2 Evaluating the Worker Efficiency Analysis

Dataset

In this work, we establish a dataset for worker efficiency evaluation. We choose the task of assembling a computer. The whole task consists of 11 operations or actions. We record the installation task performed by one teacher and 10 workers (students). Examples video frames are shown in Figure. 2.9. In totally, the dataset has more than 60,000 frames. We manually identify those 11 actions in these videos and label the corresponding frames.

Results

To measure the accuracy of our proposed method in localizing each operation performed by the worker, we use the Intersection over Union (IoU), which is the number of overlapped frames between the detected action video segment and the ground truth video segment divided by their union. For example, if the worker perform one operation from frame 100

Figure 2.7: Examples of human pose estimation from the LSP test set.

to frame 200. The detected video frames for this operation by our method are from 120 to 260. Then, the corresponding IoU score is $80/160 = 0.5$. We compare the performance of our method against two recent start-of-the-art methods: (1) Wang *et al.* [71] which extracted low-level HoG features [89] from each frame and performed subspace clustering by transferring knowledge from relevant labelled source sequences to unlabelled target sequences. They formulated the problem as least-square regression. (2) Zhang *et al.* [72] performed dynamic clustering to group low-level features and then use cut-based pooling to aggregate the encoded features within each time window to locate the specific human

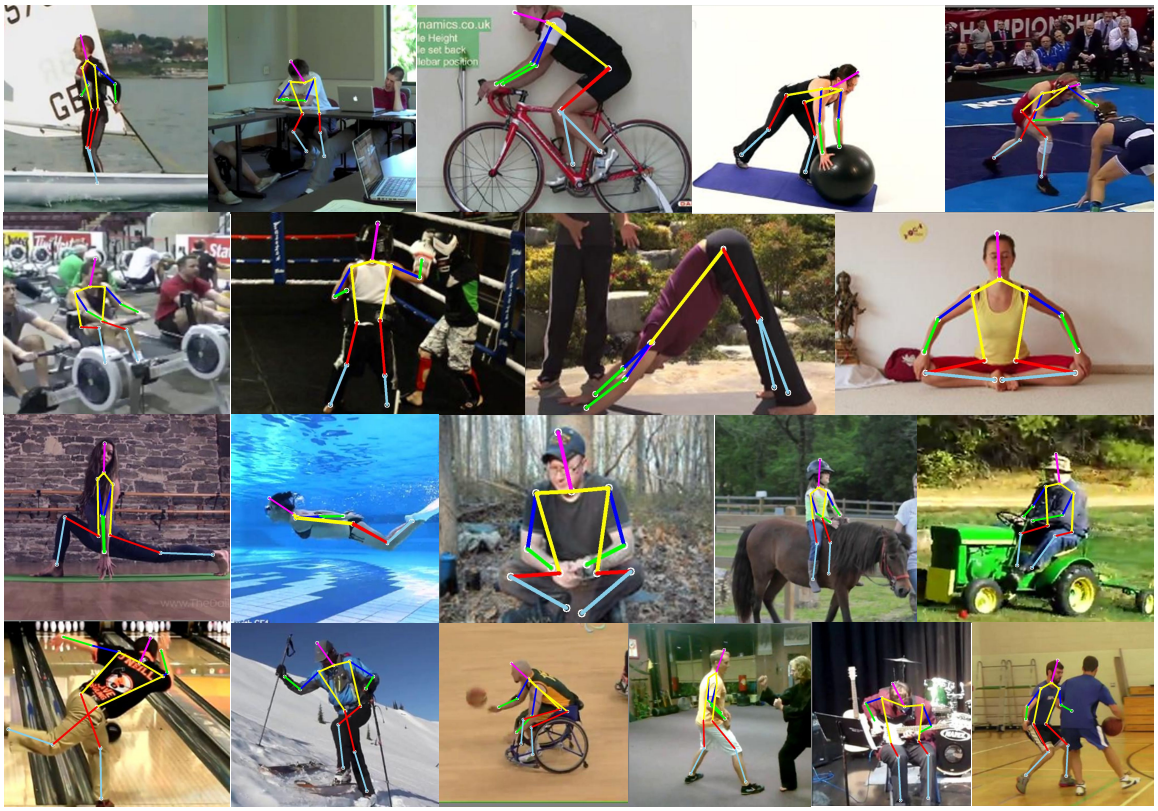

Figure 2.8: Examples of human pose estimation from the MPII test set.

action. The worker's activity localization results are presented in Table 2.3, where W indicates the worker. From the experiment results, we can clearly see that our method generally outperforms the other two methods.

In the following ablation study, we evaluate the contributions of major algorithm components. The features used in our algorithm have three major components: distance features (features FA and FC), direction features (features FB and FD), and angle features (feature FE). The results of the ablation study are summarized in Table 2.4. From the experimental results, we can clearly see that each feature component is contributing to the overall performance. The distance and direction features have relatively contributions to the overall

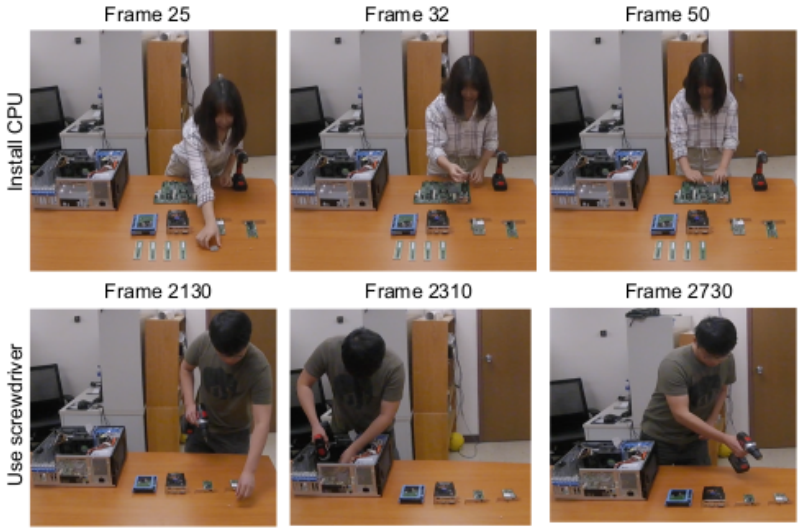

Figure 2.9: Example frames of our worker operation efficiency evaluation dataset.

Table 2.3: Comparisons of worker's activity localization among surveillance videos. Error metrics reported are IoU scores.

Method	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	Avg.
Zhang *et al.* [72]	0.9518	0.9439	0.9416	**0.9669**	0.9341	0.9413	0.9479	**0.9632**	0.9543	0.9655	0.9510
Wang *et al.* [71]	0.9244	0.9358	0.9432	0.9327	0.9419	0.9324	**0.9548**	0.9516	0.9389	0.9647	0.9420
Ours	**0.9771**	**0.9885**	**0.9624**	0.9573	**0.9608**	**0.9771**	0.9511	0.9619	**0.9681**	**0.9735**	**0.9678**

performance.

To further verify our method, we also perform the leave-one-out cross validations. We first manually extract the action video segments without empty frames for each of these 10 workers. Then we use every worker's activity video as the reference and match it with every other worker's video. The quantitative result of the cross validation result is summarized in Table 2.5. The experimental results show the average IoU score and standard deviation for each validation by leaving one worker's video out. We can see that our method can achieve the average accuracy about $94 \pm 2\%$.

Table 2.4: Ablation experiments of using features without different components. Error metrics reported are IoU scores.

Method	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	Avg.
With All Features	0.9771	0.9885	0.9624	0.9573	0.9608	0.9771	0.9511	0.9619	0.9681	0.9735	0.9678
- Without Distance Features (FA+FC)	0.8723	0.8941	0.8775	0.8813	0.9044	0.9073	0.8829	0.8936	0.9058	0.9077	0.8927
- Without Direction Features (FB+FD)	0.8962	0.9084	0.9013	0.8982	0.9061	0.9128	0.8983	0.9077	0.8909	0.9116	0.9032
- Without Angle Features (FE)	0.9244	0.9446	0.9264	0.9082	0.9239	0.9357	0.9012	0.9196	0.9212	0.9342	0.9239

Table 2.5: Cross validation of Worker's activity localization among 10 videos

Worker	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
IoU Avg	0.95	0.94	0.93	0.94	0.93	0.94	0.92	0.93	0.93	0.95
IoU Std	$\pm$0.02	$\pm$ 0.02	$\pm$ 0.02	$\pm$ 0.02	$\pm$ 0.03	$\pm$ 0.02	$\pm$ 0.01	$\pm$ 0.02	$\pm$ 0.03	$\pm$ 0.02

It is worthy to note that our proposed temporal activity localization method is specifically designed for our worker efficiency evaluation system. By "specifically", we mean that each action is performed without errors, no action is missing, and all actions are performed in a correct sequential order. For example, if a worker performs these actions in a wrong order, the localization accuracy will decrease to about 70%.

With our worker efficiency evaluation system, the amount of time spent by each worker on each operation can be automatically obtained. Figure. 2.10 shows the total amount of time spent by each worker to complete the whole assembling task with 11 operations. Figure. 2.11 shows the time of each worker spent on each operation. This data can be used directly to evaluate the performance of each worker. We can find out why a worker is performing so slowly and what operation he or she is inefficient on.

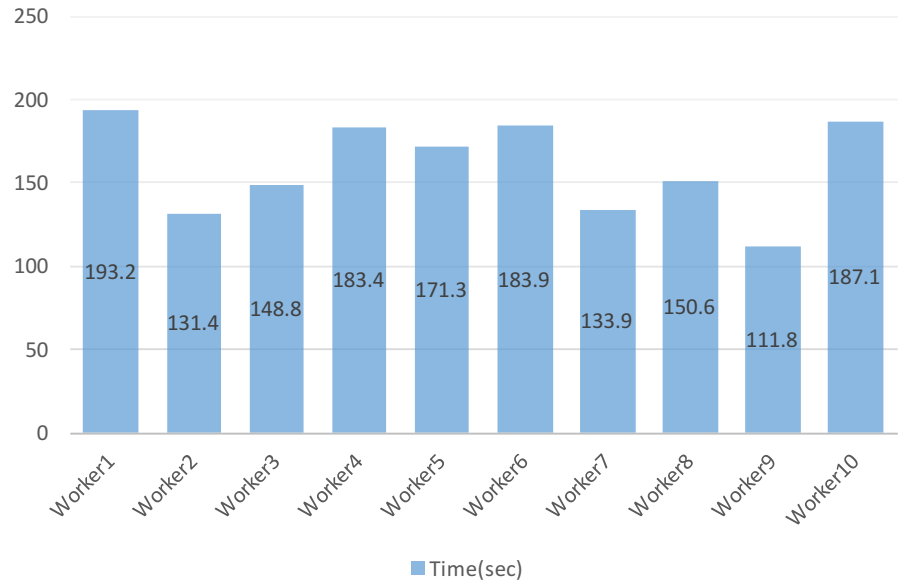

Figure 2.10: Time (sec) spent by each worker for the whole assembling task.

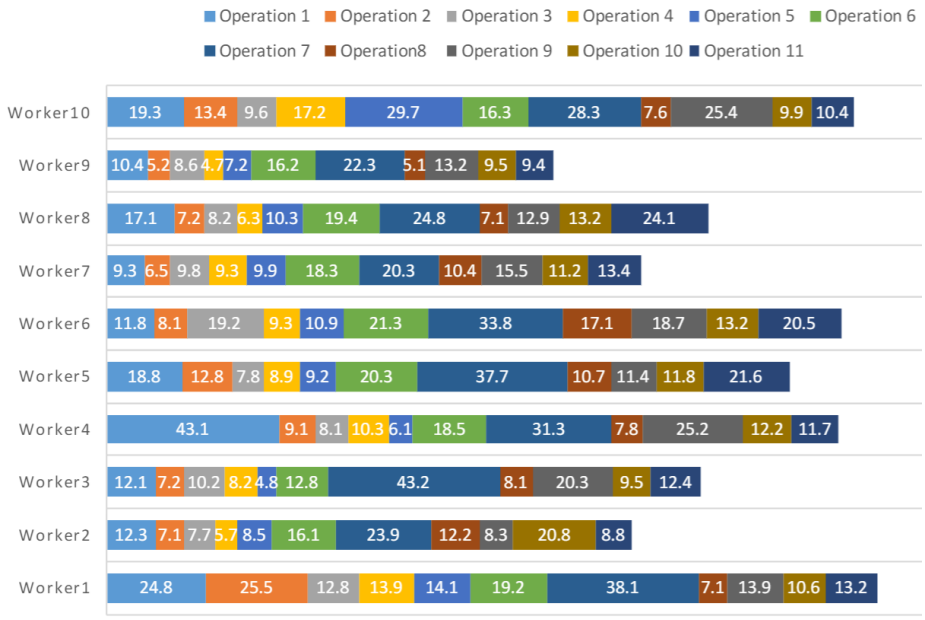

Figure 2.11: Time (sec) spent for each operation of the manufacturing task.

2.5 Conclusion

In this chapter, we have successfully developed an automatic system to monitor and evaluate worker's efficiency for smart manufacturing workforce management using human body pose estimation and temporal action localization. We first developed a novel semantic adversarial training framework with GAN networks that accurately detect human body joints, estimate its pose, and track its motion. Then, we formulated the automated worker efficiency analysis into a temporal action localization problem in which the action video performed by the worker is matched against a reference video performed by a teacher. We showed our proposed poseGAN achieves the state-of-the-art performance on benchmark dataset and our automated work efficiency analysis is able to achieve accurate action localization with an average IoU score large than 0.9. To our knowledge, this is one of the first systems to automatically evaluate worker efficiency. The resulting data can be used for performance evaluation and diagnosis of workers for smart workforce management.

Chapter 3

Reciprocal Learning Networks for Human Trajectory Prediction

3.1 Motivation

Human motion trajectory prediction in complex environments plays an important role in autonomous driving systems [29], social robots [30], and smart environments [17, 18]. In this chapter, we propose to explore the unique characteristics of human trajectories and develop a new approach, called *reciprocal learning* for human trajectory prediction. As illustrated in Figure. 3.1, we observe that the human trajectory is not only forward predictable, but also backward predictable. Imagine that the time is reversed and person is traveling backwards. As discussed in the above, the forward moving trajectories follow the social norm and obey the environmental constraints. So do the backward moving trajectories since the only difference between them is that the time is reversed. From the training data, we can train two different prediction networks, the forward prediction network $\mathbf{F}_\theta$ and

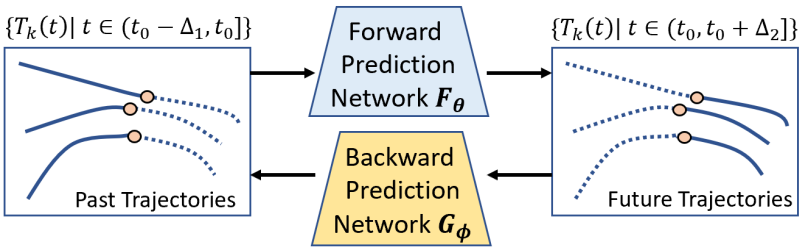

Figure 3.1: Imagine that time is reversed and person is traveling backwards. The human trajectory is not only *forward* predictable, but also *backward* predictable. This leads to our new approach of reciprocal coupling and learning between the forward and backward prediction networks for accurate human trajectory prediction.

the backward prediction network $\mathbf{G}_\phi$. These two networks are tightly coupled together, satisfying a reciprocal constraint. For example, using the forward network, we can predict the future trajectory $\mathbf{Y} = \mathbf{F}_\theta(\mathbf{X})$ from the observed or known trajectory $\mathbf{X}$. If the prediction $\mathbf{Y}$ is accurate, then $\mathbf{G}_\phi(\mathbf{Y})$ must be equal to $\mathbf{X}$.

Based on this observation and the unique reciprocal constraint, we develop a new approach called *reciprocal network learning* for accurate and robust prediction of human trajectories. We introduce the reciprocal prediction loss and establish an iterative procedure for training these two tightly coupled networks. We borrow the concept of the adversarial attacks of deep neural networks which iteratively modifies the input of the network to match a given target or forced network output. We integrate the reciprocal constraint with the adversarial attack method to develop a new matched prediction method for human trajectory prediction. Our experimental results on benchmark datasets demonstrate that our new method outperforms the state-of-the-art methods for human trajectory prediction.

The **major contributions** of this work can be summarized as follows. (1) We have established a forward and backward prediction network structure for human trajectory prediction, which satisfies the reciprocal prediction constraints. (2) Based on this constraint,

we have developed a reciprocal learning approach to jointly train these two prediction networks in an collaborative and iterative manner. (3) Once the network is successfully trained, we have developed a new approach for network inference or testing by integrating the concept of adversarial attacks with the reciprocal constraint. It is able to iteratively refine the predicted trajectory by the forward network such that the reciprocal constraint is satisfied. (4) Our ablation studies have shown that the proposed new approach is very effective with significant contributions to the overall performance of our method, which outperforms other state-of-the-art methods in the literature.

The rest of the chapter is organized as follows. Section 3.2 reviews related work on human trajectory prediction. The proposed reciprocal network learning and matched prediction are presented in Section 3.3. Section 3.5 presents the experimental results, performance comparisons, and ablation studies. Section 3.6 summarizes our major contributions and concludes the paper.

3.2 Related Work

Existing methods for human trajectory prediction mainly focus on modeling human-human interactions and human-scene interactions. Human-human models focus on learning human movements and how human interacts with others [39, 40]. Human-scene models also try to learn the dynamic contents of the background scenes to extract some visual features to help better understand human motions [34, 36, 90–95]. In this section, we review existing work, including human-human models and human-scene models for human trajectory prediction. We also discuss related work in sequence prediction using Recurrent Neural Networks (RNNs) [96]. Our work is inspired by generative models [6, 97] and the idea of

cycle consistence [98–101] in visual tracking, relevant papers in these two areas are also reviewed in this section.

3.2.1 Human-Human Models for Trajectory Prediction

A number of methods have been developed in the literature to model human social interactions and behaviors in crowded scenes, such as people attempting to avoid walking into each other. Helbing and Molnar [37] introduced the Social Force Model to characterize social interactions among people in crowded scenes using coupled Langevin equations. In recent methods based on LSTM (Long Short Term Memory) [39], social pooling was introduced to share features and hidden representations between different agents. The key idea is to merge hidden states of nearby pedestrians to make each trajectory aware of its neighbourhood. [102] found out that groups of people moving coherently in one direction should be excluded from the above pooling mechanism. [40] used a Generative Adversarial Network (GAN) to discriminate between multiple feasible paths. Their pooling mechanism relies on relative positions between all pedestrians with the target pedestrian. This model is able to capture different movement styles but does not differentiate between structured and unstructured environments. [103] predicted human trajectories using a spatio-temporal graph to model both position evolution and interactions between pedestrians.

3.2.2 Human-Scene Models for Trajectory Prediction

Another set of methods for human trajectory prediction have focused on learning the effects of physical environments. For example, human tend to walk along the sidewalk, around a tree or other physical obstacles. Sadeghian *et al.* [43] considered both traveled areas and

semantic context to predict social and context-aware positions using a GAN (Generative Adversarial Network). Liang *et al.* [104] proposed to use abstract scene semantic segmentation features and multi-scale location encoding for better predicting multiple plausible trajectories. [105] designed a probabilistic model and introduced a dynamic attention-based state encoder to encode agent interactions. [106] extracted multiple visual features, including each person's body keypoints and the scene semantic map to predict human behavior and model interaction with the surrounding environment. [38] has studied attractions towards static objects, such as artworks, which deflect straight paths in several scenarios such as museums. [34] proposed a Bayesian framework to predict unobserved paths from previously observed motions and to transfer learned motion patterns to new scenes. In [107], the dynamics and semantics for long-term trajectory predictions have been studied. Scene-LSTM [108] divided the static scene into grids and predicted pedestrian's location using LSTM. The CAR-Net method [109] integrated past observations with bird's eye view images and analyzed them using a two-levels attention mechanism.

3.2.3 Recurrent Neural Networks for Sequence Prediction

This work is also related to recurrent neural networks (RNNs) [96]. RNNs are widely used for sequence data analysis, *e.g.*, speech recognition [110–113], image captioning [114–119], machine translation [111] and video generation [120]. [121] recognizes that the drawback of RNNs model is the lack of high-level and spatio-temporal structure. [39, 122, 123] propose to learn complex interactions using multiple networks. [35] designs an RNN-based encoder-decoder framework and uses variational autoencoder (VAE) to predict the sequence. Alahi *et al.* proposes a so-called social pooling layer to capture the interactions of human within a certain range. In this work, we borrow the idea from [40] which uses a

multi-layer perceptron (MLP) followed by a max pooling layer to learn the human-human interaction.

3.2.4 Generative Networks and Cycle Consistency Learning

Generative Adversarial Networks (GANs) have been widely used and achieved impressive results in representation learning [8–10], image translation [11, 12] and image synthesis [13–16]. In this work, we adopt a GAN framework to force the generated future and past trajectories to be indistinguishable from the ground truth. Using transitivity as a way to regularize structured data has been explored. For example, in visual tracking, [99, 124] developed a forward-backward consistency constrain. In language processing, [125–127] studied human and machine translators to verify and improve translations based on back translation and reconciliation mechanisms. Cycle consistency has also been used for motion analysis [128], action prediction [129], 3D shape matching [130], dense semantic alignment [131, 132], depth estimation [133–135], and image-to-image translation [136, 137]. CycleGAN [137] introduces a cycle consistence constraint for learning a mapping to translate an image from the source domain into the target domain. Pang *et al.* [129] propose to use a bi-directional LSTM model for early actions prediction. It employs consistency learning by synthesizing future action and reconstructing observed action.

3.3 Reciprocal Networks for Human Trajectory Prediction

In this section, we present our reciprocal network learning method for human trajectory prediction.

3.3.1 Problem Formulation

We follow the standard formulation of trajectory forecasting problem in the literature [103, 106]. With observed trajectories of all moving agents in the scene, including persons and vehicles, the task is to predict the moving trajectories of all agents for the next period of time, say 10 seconds, in the near future. Specifically, let $\mathbf{X} = X_1, X_2, \cdots, X_N$ be the trajectories of all human in the scene. Our task is to predict the future trajectories of all human $\hat{\mathbf{Y}} = \hat{Y}_1, \hat{Y}_2, \cdots, \hat{Y}_N$ simultaneously. The input trajectory of human n is given by $X_n = (x_n^t, y_n^t)$ for time steps $t = 1, 2, \cdots, T_o$. The ground truth of future trajectory is given by $Y_n = (x_n^t, y_n^t)$ for time step $t = T_o + 1, \cdots, T_p$.

3.3.2 Method Overview

As illustrated in Figure. 3.2, in reciprocal learning, we are learning two coupling networks, the forward prediction network $\mathbf{F}_\theta$ which predicts the future trajectories $\mathbf{Y} = \mathbf{F}_\theta(\mathbf{X})$ from the past data $\mathbf{X}$, and the backward prediction network $\mathbf{G}_\phi$ which predicts the past trajectories $\mathbf{X} = \mathbf{G}_\phi(\mathbf{Y})$ from the future data $\mathbf{Y}$. It should be noted that, during training, both the past and future data are available. If both networks are well trained, then we should have following two reciprocal consistency constraints:

$$\mathbf{X} \approx \mathbf{G}_\phi(\mathbf{F}_\theta(\mathbf{X})), \tag{3.1}$$

$$\mathbf{Y} \approx \mathbf{F}_\theta(\mathbf{G}_\phi(\mathbf{Y})). \tag{3.2}$$

These two networks are able to help each other to improve the learning and prediction performance. Specifically, if the backward prediction network $\mathbf{G}_\phi$ is trained, we can use the reciprocal constraint (3.1) to double check the accuracy of the forward prediction network

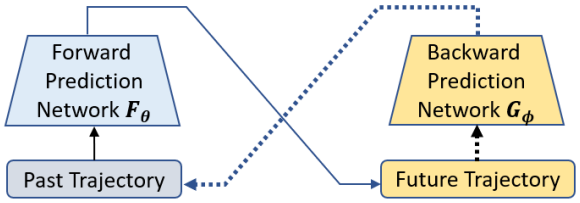

Figure 3.2: Illustration of the proposed reciprocal learning approach.

$\mathbf{F}_\theta$ and improve its performance during training. Likewise, if the forward prediction network $\mathbf{F}_\theta$ is trained, we can use (3.2) to improve the training performance of the backward prediction network $\mathbf{G}_\phi$. This results in a tightly coupled iterative learning and performance improvement process between these two prediction networks, as illustrated in Figure. 3.2. Once the forward and backward networks are successfully trained using the reciprocal learning approach, we develop a new network inference method called *reciprocal attack for matched prediction*. It borrows the concept of adversarial attacks of deep neural networks where the input is iteratively modified such that the network output matches a given target [138].

Our proposed idea echoes some thoughts in CycleGAN [137] which presents an approach for learning a mapping to translate an image from a source domain to a target domain. They also learn an inverse mapping and introduce the cycle consistence constraint. Our approach is significantly different from this CycleGAN method. We design two tightly coupled prediction networks, the forward and backward prediction networks, which are jointly learned based on the reciprocal constraint. For the testing part, our approach introduces a new reciprocal attack method for matched prediction of human trajectory.

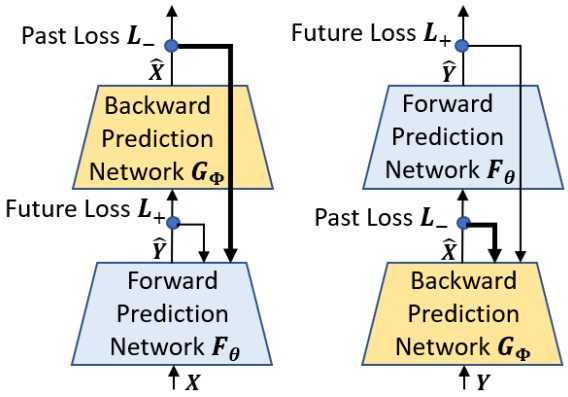

Figure 3.3: Illustration of the training process of reciprocal learning.

3.3.3 Reciprocal Network Training

To successfully train the forward and backward prediction networks, we define two loss functions, J_- and J_+, to measure the prediction accuracy of the past and future trajectories. One reasonable choice will be the L_2 between the original trajectory and its prediction. These two loss functions will be updated alternatively and combined to guide the training of each of these two networks, as illustrated in Figure. 3.3. For example, when training the forward prediction network $\mathbf{F}_\theta$, the loss function used in existing literature is the prediction error of the future trajectory L_+. In reciprocal training, we first pre-train the backward prediction network $\mathbf{G}_\phi$ using the training data with all trajectories reversed in time. We then use this network to map the prediction result of $\mathbf{F}_\theta$, $\hat{\mathbf{Y}} = \mathbf{F}_\theta(\mathbf{X})$, back to the past trajectory, which is given by

$$\hat{\mathbf{X}} = \mathbf{G}_\phi(\hat{\mathbf{Y}}) = \mathbf{G}_\phi(\mathbf{F}_\theta(\mathbf{X})). \tag{3.3}$$

The past trajectory loss is then given by $L_- = ||\mathbf{X} - \hat{\mathbf{X}}||_2$. We refer to this loss as *reciprocal loss*. It will be combined with L_+ to form the loss function for the forward prediction

network $\mathbf{F}_\theta$:

$$\begin{aligned} J_+[\theta] &= \lambda \cdot L_+ + (1-\lambda) \cdot L_- \\ &= \lambda \cdot ||\mathbf{Y} - \mathbf{F}_\theta(\mathbf{X})||_2 \\ &+ (1-\lambda) \cdot ||\mathbf{X} - \mathbf{G}_\phi(\mathbf{F}_\theta(\mathbf{X}))||_2. \end{aligned} \tag{3.4}$$

Similarly, we can derive the loss function for the backward prediction network $\mathbf{G}_\phi$:

$$\begin{aligned} J_-[\phi] &= \lambda \cdot L_- + (1-\lambda) \cdot L_+ \\ &= \lambda \cdot ||\mathbf{X} - \mathbf{G}_\phi(\mathbf{Y})||_2 \\ &+ (1-\lambda) \cdot ||\mathbf{Y} - \mathbf{F}_\theta(\mathbf{G}_\phi(\mathbf{Y}))||_2. \end{aligned} \tag{3.5}$$

In reciprocal training, we first pre-train the forward and backward prediction networks independently. Then, these two networks are jointly trained in an iterative manner based on the reciprocal constraint.

3.3.4 Constructing the Forward and Backward Prediction Networks

Both the forward and backward networks share the same network structure. In the following, we use the forward prediction network $\mathbf{F}_\theta$ as an example to explain our network design. As illustrated in Figure. 3.4, we adopt the existing Social-GAN in [40] as our baseline prediction network. Our model consists of two key components: (1) a feature extraction module and (2) an LSTM (Long Short Term Memory)-based GAN (generative adversarial network) module.

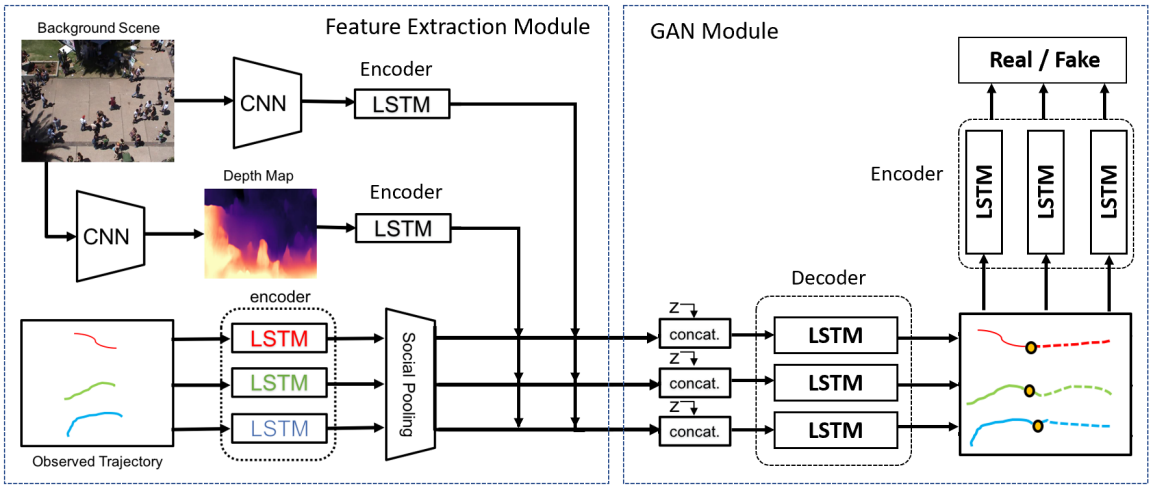

Figure 3.4: Overview of our prediction model. Our model consists of two key components: (1) a feature extraction module, (2) an LSTM-based GAN module.

Feature Extraction

In real world scenarios, human's selection of future path is affected by the surrounding environment, including other persons in the neighborhood and the physical scene. Our feature extraction module has three major components to extract human-specific, scene context and depth structure features.

(a) *Human-specific features*. The human scale feature captures the temporal pattern and dependency of each human trajectory. Given the observed trajectories $\mathbf{X} = X_1, X_2, \cdots, X_N$ of all human in the scene, the input trajectory of each human n is defined as $X_n = (x_n^t, y_n^t)$ from time steps $t = 1, 2, \cdots, T_o$. We first embed the coordinates of each human n into a fixed size vector e_n^t using a single layer MLP [139]:

$$e_n^t = \psi(x_n^t, y_n^t; W_{em}), \tag{3.6}$$

where $\psi(\cdot)$ is an embedding function with ReLU non-linearity [140] and W_{em} is the em-

bedding weight. Then, we use an LSTM module to encode them into a high-dimensional feature $F^t_{h,n}$.

$$F^t_{h,n} = LSTM_{en}(F^{t-1}_{h,n}, e^t_n; W_{en1}), \tag{3.7}$$

where W_{en1} denotes the encoding weight which can be optimized during training process. Notice that W_{en1} is shared between all human in the scene. In order to capture the joint influence of all surrounding human's movements on the prediction of the target human n, we borrow the idea from [40] to build a social pooling module (SP) which encodes the human-human interactions. The relative distances between the target person and others are calculated. These distance vectors are concatenated with the hidden state in the LSTM network for each person and then embedded by an MLP and followed by a Max-Pooling function [141] to form the joint feature $F^t_{s,n}$.

$$F^t_{s,n} = SP(F^t_{h,1}, F^t_{h,2}, \cdots, F^t_{h,N}). \tag{3.8}$$

A maximum number of moving human in the scene is set and a default value of 0 is used if the corresponding agent does not exist in the current frame.

(b) *Scene context features*. As recognized in [41, 43], the environmental context affects the decision of the human in planning its next step of movement. Features of the current scene can be incorporated into the reasoning process. Similar to prior work [43], we use the VGGNet-19 network [45] pre-trained on the ImageNet [45] to extract the visual feature f^t of background scene I^t, which is then fed into an LSTM encoder to compute the hidden state tensor F^t_v.

$$f^t = VGG(I^t), \tag{3.9}$$

$$F^t_v = LSTM_{en}(F^{t-1}_v, f^t; W_{en2}), \tag{3.10}$$

where W_{en2} is the corresponding encoding weights.

(c) *Depth structure features*. As a unique feature of our proposed method, we propose to also incorporate the 3D scene depth structure into the reasoning process, which also improves the prediction accuracy of human trajectories. This is because the human motion occurs in the original 3D environment. Therefore, its natural behavior and motion patterns are better represented in the 3D instead of 2D coordinate system. For example, the trajectory of a person walking near the camera is much different from that of a person walking far away from the camera due to the camera perspective transform. To address this issue, we propose to estimate a depth map from a single image using existing depth estimation method [3]. We use the pre-trained model *Monodepth2*, denoted by $\mathbf{D}$ to perform monocular depth estimation and obtain the depth map $D^t = \mathbf{D}(I^t)$ of scene I^t, then use an LSTM to encode it into a depth feature F_d^t.

$$F_d^t = LSTM_{en}(F_d^{t-1}, D^t; W_{en3}), \tag{3.11}$$

where W_{en3} is the associated encoding weights. Figure. 3.5 presents qualitative depth estimation examples from Town Centre dataset [4] by *Monodepth2*.

LSTM-based GAN for Trajectory Prediction

Inspired by previous work [40, 43], in this work we use an LSTM based Generative Adversarial Network (GAN) module to generate human's future path as illustrated in Figure. 3.4. The generator is constructed by a decoder LSTM. Similar to the conditional GAN [18], a white noise vector Z is sampled from a multivariate normal distribution. Then, a merge layer is used in our proposed network which concatenates all encoded features mentioned

Figure 3.5: Examples of the input (left column) and the output (right column) of the monocular depth estimation [3]. The input image is from Town Centre dataset [4].

above with the noise vector Z.

$$F_n^t = concat(F_s^t, F_v^t, F_d^t, Z), \tag{3.12}$$

We take F_n^t as the input to the LSTM decoder to generate the candidate future paths $\hat{Y}_n^t$ for each human.

$$\hat{Y}_n^t = LSTM_{de}(\hat{Y}_n^{t-1}, F_n^t; W_{de}), \tag{3.13}$$

where W_{de} is the decoding weights of LSTM.

The discriminator is built with an LSTM encoder which takes the input ${Y'}_n^t$ as randomly chosen trajectory from either ground truth Y_n^t or predicted trajectories $\hat{Y}_n^t$ and classifies them as "real" or "fake". Generally speaking, the discriminator classifies the trajectories which are not accurate as "fake" and forces the generator to generator more realistic and feasible trajectories.

$$L_n^t = LSTM_{en}({Y'}_n^t, h_{en}^t; W_{en4}), \tag{3.14}$$

where L_n^t is the predicted label from the discriminator for the chosen input trajectory to be

"real"($L_n^t = 1$) or "fake"($L_n^t = 0$). h_{en}^t denotes the hidden state of the encoding LSTM and W_{en4} is the corresponding weights.

Within the framework of our reciprocal learning for human trajectory prediction, let $G^\theta : X \to Y$ and $G^\phi : Y \to X$ be the generators of the forward prediction network $\mathbf{F}_\theta$ and the backward prediction network $\mathbf{G}_\phi$, respectively. D^θ is the discriminator for $\mathbf{F}_\theta$. Its input Y' is randomly selected from either ground truth Y or the predicted future trajectory $\hat{Y}$. Similarly, D^ϕ is discriminator for $\mathbf{G}_\phi$. To train $\mathbf{F}_\theta$ and $\mathbf{G}_\phi$, we combine the adversarial loss with the forward prediction loss $J_+[\theta]$ and the backward prediction loss $J_-[\phi]$ in Eqs. (3.4) and (3.5) together to construct the overall loss function for $\mathbf{F}_\theta$ and $\mathbf{G}_\phi$, respectively:

$$\mathcal{L}_\theta = L_{GAN}^\theta + J_+[\theta], \quad \mathcal{L}_\phi = L_{GAN}^\phi + J_-[\phi], \tag{3.15}$$

where adversarial losses L_{GAN}^θ and L_{GAN}^ϕ are defined as:

$$\begin{aligned} L_{GAN}^\theta &= \min_{G^\theta} \max_{D^\theta} \quad \mathbb{E}_{Y' \sim p(Y,\hat{Y})}[\log D^\theta(Y')] \\ &+ \quad \mathbb{E}_{X \sim p(X), Z \sim p(Z)}[\log(1 - D^\theta(G^\theta(X, Z)))], \end{aligned} \tag{3.16}$$

$$\begin{aligned} L_{GAN}^\phi &= \min_{G^\phi} \max_{D^\phi} \quad \mathbb{E}_{X' \sim p(X,\hat{X})}[\log D^\phi(X')] \\ &+ \quad \mathbb{E}_{Y \sim p(Y), Z \sim p(Z)}[\log(1 - D^\phi(G^\phi(Y, Z)))]. \end{aligned} \tag{3.17}$$

3.4 Reciprocal Attack for Matched Prediction of Human Trajectories

Once the forward and backward networks are successfully trained with the above loss functions based on the reciprocal learning approach, we are ready to perform prediction of the human trajectories. By taking advantage of the reciprocal property of the forward and backward networks, we develop a new network inference method called *reciprocal attack for matched prediction* to achieve improved performance in human trajectory prediction.

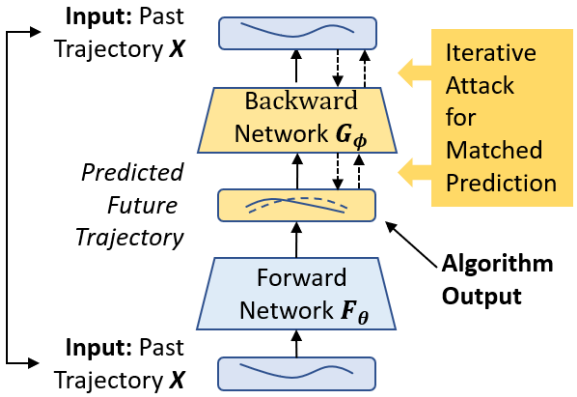

Figure 3.6: Illustration of the proposed reciprocal attack method.

As illustrated in Figure. 3.6, $\mathbf{F}_\theta$ is our trained network for human trajectory prediction. With the past trajectories $\mathbf{X}$ as input, it predicts the future trajectories $\hat{\mathbf{Y}} = \mathbf{F}_\theta(\mathbf{X})$. During network testing or actual prediction, we do not know the ground truth of the future trajectory. How do we know if this prediction $\hat{\mathbf{Y}}$ is accurate or not? How can we further improve its accuracy? Fortunately, in our reciprocal learning framework, we have another network, the backward prediction network $\mathbf{G}_\phi$, which can be used to map the estimated $\hat{\mathbf{Y}}$ back to the known input $\mathbf{X}$. Our idea is that, if $\hat{\mathbf{Y}}$ is accurate, then its backward prediction $\hat{\mathbf{X}} = \mathbf{G}_\phi(\hat{\mathbf{Y}}) = \mathbf{G}_\phi(\mathbf{F}_\theta(\mathbf{X}))$ should match the original input $\mathbf{X}$. When the prediction $\hat{\mathbf{Y}}$ is

not accurate, we can modify the prediction such that the above matching error is minimized. This leads to the following optimization problem:

$$\hat{\mathbf{Y}}^* = \arg \min_{\tilde{\mathbf{Y}}=\hat{\mathbf{Y}}+\Delta(t)} ||\mathbf{X} - \mathbf{G}_\Phi(\tilde{\mathbf{Y}})||_2. \tag{3.18}$$

Here, $\Delta(t)$ is the small perturbation or modification added to the existing prediction result $\hat{\mathbf{Y}}$. The above optimization procedure aims to find the best modification $\hat{\mathbf{Y}}^* = \hat{\mathbf{Y}} + \Delta(t)$ to minimize the matching error.

This optimization problem can be solved by adversarial attack methods recently studied in the literature of deep neural network attack and defense. In this work, we propose to borrow the idea from the famous Fast Gradient Sign method (FGSM) developed by Goodfellow *et al.* [138] to perform adversarial attacks. Essentially, it is the same error back propagation procedure as network training. The only difference is that network training modifies the network weights based on error gradients. However, the adversarial attack does not modify the network weights, it propagates the error all the way to the input layer to modify the original input image to minimize the loss.

This approach uses the sign of the gradient at each pixel to determine the direction of changing pixel value. In our case, we remove the sign function and directly use the gradient to update the input trajectory. With the matching error of human trajectories $E = ||\mathbf{X} - \mathbf{G}_\Phi(\tilde{\mathbf{Y}})||_2$, we can perform multiple iterations of the modified FGSM attack on the prediction $\hat{\mathbf{Y}}$ such that the matching error is minimized. At iteration m, the attacked trajectory (input) is given by

$$\hat{\mathbf{Y}}^m = \hat{\mathbf{Y}}^{m-1} - \epsilon \cdot \nabla_{\hat{\mathbf{Y}}} E(\mathbf{X}, \hat{\mathbf{Y}}^{m-1}), \tag{3.19}$$

with $\hat{\mathbf{Y}}^0 = \hat{\mathbf{Y}}$. ϵ is the magnitude of attacks [138]. $\nabla_{\hat{\mathbf{Y}}} E(\mathbf{X}, \hat{\mathbf{Y}}^{m-1})$ indicates the gradient of error function E with respect to the input $\hat{\mathbf{Y}}$. Intuitively, the updated trajectory $\hat{\mathbf{Y}}^m$ will minimize E. We then perform an exponential average of $\{\hat{\mathbf{Y}}^m\}$ to obtain the improved prediction

$$\hat{\mathbf{Y}}^* = \left[\sum_{m=1}^{M} e^{\alpha \cdot m} \cdot \hat{\mathbf{Y}}^m\right] / \sum_{m=1}^{M} e^{\alpha \cdot m}, \tag{3.20}$$

where M is the total iterations and α is a constant to control the relative weights between these different iterations of attacks.

Figure. 3.7 shows an example of the trajectory prediction results using reciprocal attack in each iteration performed on the HOTEL dataset. We can see that, using reciprocal attack in an iterative manner, the error metrics, ADE and FDE (see definition in Section 3.5.3) of trajectory prediction will decrease in a certain iteration, then they might increase after a few iterations. Since we do not know the ground truth, we choose to perform reciprocal attack for 20 iterations based on heuristic studies. Then an exponential average is performed on the result trajectories in each iteration to obtain the refined future trajectories prediction. The ablation studies in the following section will provide more results to demonstrate the effectiveness of this attacked-based matched prediction scheme.

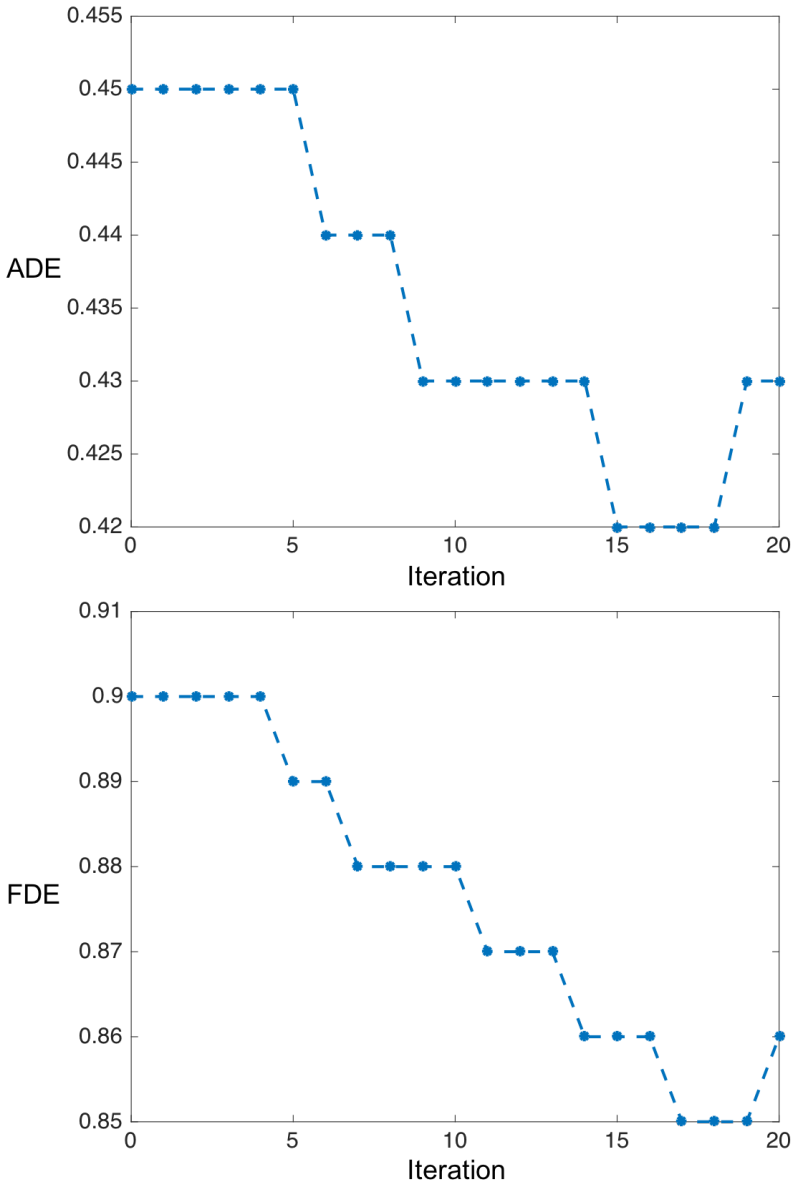

Figure 3.7: Example of our proposed reciprocal attack performed on HOTEL dataset. X-coordinates in both figures indicate the iteration, while Y-coordinates indicate error metrics, ADE and FDE (see detailed explanation in Section 3.5.3).

3.5 Experimental Results

We provide extensive performance comparisons on benchmark datasets (ETH [142] and UCY [143]) between our work and state-of-the-art methods. We also conduct ablation to demonstrate the effectiveness of each algorithm component. To further evaluate the generalization capability of our method on predicting human future trajectories, we conduct experiments on two new datasets: Town Centre [4] and Grand Central Station [144].

3.5.1 Datasets

Performance comparisons and ablation studies are performed on the ETH [142] and UCY [143] datasets, which contain real world human trajectories and various natural human-human interaction scenarios. In total, 5 sub-datasets, ETH, HOTEL, UNIV, ZARA1 and ZARA2, are included in these two datasets. Each set contains bird's-eye view images and 2D locations of each human. In total there are 1536 humans in these 5 datasets. They contain challenging situations, including human collision avoidance, human crossing each other, and dynamic group behaviors. Each scene occurs in a unconstrained outdoor environment [43].

Generalization studies are performed on the Town Centre [4] and Grand Central Station [144] datasets. The Town Center dataset contains short videos with frequent human-human and human-scene interactions. It is originally used for human tracking tasks with bounding boxes for the head and body for each human. In the experiment, we use the center of the human body bounding box as the location, as in existing methods [41, 42]. The Grand Center Station dataset contains a long-duration video (more than 32 minutes) and consists of about 12,600 pedestrians with frequent human interactions. It is originally used for

human behavior analysis.

3.5.2 Implementation Details

Our GAN model is constructed using the LSTM for the encoder and decoder. The generator and discriminator are trained iteratively with the Adam optimizer. We choose the batch size of 64 and the initial learning rate of 0.001. The whole model is trained for 200 epochs. The trajectories are embedded using a single layer MLP with dimension of 16. The encoder and decoder for the generator use an LSTM with the hidden state's dimension of 32. In the LSTM encoder for the discriminator, the hidden state's dimension is 48. In the pooling module, we follow the procedure and setting in [40]. The maximum number of human surrounding the target human is set to 32. This value is chosen since in all datasets, none of them has more than 32 human in any frame. For the feature extraction part, following the prior work [43], we use the VGG feature with a size of 512 for the background scene, which is then embedded using a single MLP to a dimension of 16. For the depth map estimation, we use the pre-trained model *Monodepth2* from [3] and the depth feature is also embedded using a single layer MLP with a dimension of 16. The weight for our loss function is $\lambda = 0.5$. We perform the reciprocal attack for 20 iterations, the perturbation ϵ is set as 0.05.

3.5.3 Evaluation Metrics and Methods

Following the standard evaluation procedure [31, 39], we use the following two error metrics for performance evaluations. (1) Average Displacement Error (ADE) is the average L_2 distance between the ground truth trajectory and our prediction over all predicted time steps

from $T_o + 1$ to T_p. (2) Final Displacement Error (FDE) is the Euclidean distance between the predicted final destination and the true final destination at end of the prediction period T_p. They are defined as:

$$\text{ADE} = \frac{\sum_{n \in \Psi} \sum_{t=T_o+1}^{T_p} \sqrt{(\hat{x}_n^t - x_n^t)^2 + (\hat{y}_n^t - y_n^t)^2}}{|\Psi| \cdot T_p}, \tag{3.21}$$

$$\text{FDE} = \frac{\sum_{n \in \Psi} \sqrt{(\hat{x}_n^{T_p} - x_n^{T_p})^2 + (\hat{y}_n^{T_p} - y_n^{T_p})^2}}{|\Psi|}, \tag{3.22}$$

where $(\hat{x}_n^t, \hat{y}_n^t)$ and (x_n^t, y_n^t) are the predicted and ground truth coordinates for human n at time t, Ψ is the set of human and $|\Psi|$ is the total number of human in the test set.

Following existing methods [39, 40, 43], we use the leave-one-out evaluation protocol on the ETH and UCY datasets. Specifically, four datasets are used for training and the remaining one is used for testing. Given the human trajectory for the past 8 time steps (3.2 seconds), our model predicts the future trajectory for next 12 time steps (4.8 seconds). In our generalization studies, following the previous work [42], we split the data of Town Centre and Grand Central Station into one half for training and the other half for testing. All location coordinates are normalized to [0, 1] for training and testing.

3.5.4 Comparison with Existing Methods

We compare our method against the following state-of-the-art methods: (1) *Linear* [40]: This method applies a linear regression to estimate linear parameters by minimizing the least square error [40]. (2) *LSTM* [39]: This is the baseline model for the LSTM-based method, which does not consider human-human interactions or background scene informa-

tion. (3) *S-LSTM* [39]: This method models each human by an LSTM and proposes a social pooling mechanism. Both *S-LSTM* and *LSTM* generate one trajectory for each observation. (4) *S-GAN* [40]: This is one of the first GAN-based methods. During the pooling stage, all human in the scene are considered. *S-GAN* and *S-GAN-P* are different only in whether the pooling mechanism is applied or not. The method chooses the best trajectory from 20 network predictions as the final test result. (5) *SoPhie* [43]: This work implements a so-called physical constrain described by background scene features. Also the attention mechanism is used in this GAN-based method. (6) *Scene-LSTM* [42]: This method imposes a two-level grid structure on the scene to incorporate the scene information with human movements. (7) *Next* [106]: This method introduces a LSTM-based predictor with pooling of multiple features. In the test part, besides using a single model, it follows [40] to train 20 different models with random initialization. In our comparison, we follow [106] to report the minimum ADE and FDE over 20 outputs.

3.5.5 Quantitative Results

Table 3.1 shows the comparison results of our method against existing methods on the above two performance metrics ADE and FDE. As illustrated in Table 3.1, our method outperforms all other methods except on the ETH dataset against *Scene-LSTM* and on the Hotel dataset against *Next*. We can see that the *Linear* method has the lowest accuracy, it can only predict the straight trajectory and have very poor performance in videos with complicated human-human and human-environment interactions. *LSTM* performs better than *Linear* since it can handle more complicated trajectories. *S-LSTM* also outperforms the Linear model since it uses the social pooling mechanism, but it performs worse than *LSTM*. According to [40], the *S-LSTM* is trained on a synthetic dataset and fine-tuned on

the real dataset to improve the accuracy. *Scene-LSTM* achieves better results than *S-LSTM* since it incorporates the scene information as well as human movements. Both *SoPhie* and *Next* outperform the *S-GAN* due to the use of background visual features and the attention module. Overall, our method achieves the best average error metrics in both ADE and FDE among all comparison methods.

Table 3.1: Comparisons of different methods on ETH (Column 3 and 4) and UCY (Column 5-7) datasets.

Metric	Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
ADE	Linear [40]	1.33	0.39	0.82	0.62	0.77	0.79
	LSTM [39]	1.09	0.86	0.61	0.41	0.52	0.70
	S-LSTM [39]	1.09	0.79	0.67	0.47	0.56	0.72
	S-GAN [40]	0.81	0.72	0.60	0.34	0.42	0.58
	S-GAN-P [40]	0.87	0.67	0.76	0.35	0.42	0.61
	SoPhie [43]	0.70	0.76	0.54	0.30	0.38	0.54
	Scene-LSTM [42]	**0.36**	0.95	0.63	0.45	0.40	0.56
	Next [106]	0.73	**0.30**	0.60	0.38	0.31	0.46
	Ours	0.69	0.43	**0.53**	**0.28**	**0.28**	**0.44**
FDE	Linear [40]	2.94	0.72	1.59	1.21	1.48	1.59
	LSTM [39]	2.14	1.91	1.31	0.88	1.11	1.52
	S-LSTM [39]	2.35	1.76	1.40	1.00	1.17	1.54
	S-GAN [40]	1.52	1.61	1.26	0.69	0.84	1.18
	S-GAN-P [40]	1.62	1.37	1.52	0.68	0.84	1.21
	SoPhie [43]	1.43	1.67	1.24	0.63	0.78	1.15
	Scene-LSTM [42]	**0.67**	1.77	1.41	1.00	0.90	1.15
	Next [106]	1.65	**0.59**	1.27	0.81	0.68	1.00
	Ours	1.24	0.87	**1.17**	**0.61**	**0.59**	**0.90**

To evaluate the performance of our method in predicting feasible paths in crowded scenes, we follow the procedure in previous papers [43] to report a new evaluation metric which is the percentage of *near-collisions* among humans. A collision is defined when the Euclidean distance between two human is smaller than 0.1m. We compute the average percentage of human near-collision in each frame of ETH and UCY datasets. The comparison results against the *Linear*, *S-GAN* and *SoPhie* are shown in Table 3.2. We can see that

Table 3.2: Average percentage of colliding human for each scene in ETH and UCY datasets. The first column represents the ground truth.

	GT	Linear [40]	S-GAN [40]	SoPhie [43]	Ours
ETH	0.000	3.137	2.509	1.757	**1.512**
HOTEL	0.092	1.568	1.752	1.936	**1.547**
UNIV	0.124	1.242	**0.559**	0.621	0.563
ZARA1	0.000	3.776	1.749	**1.027**	1.094
ZARA2	0.732	3.631	2.020	1.464	**1.252**
Avg	0.189	2.670	1.717	1.361	**1.194**

our method outperforms these three methods on the ETH, HOTEL, and ZARA2 datasets, producing less human collision in the future time. On the other two datasets, UNIV and ZARA1, *S-GAN* and *SoPhie* perform slightly better than ours. However, they suffer from significant performance degradation on other datasets.

3.5.6 Ablation Studies

To systematically evaluate our method and study the contribution of each algorithm component, we perform a number of ablation experiments in Table 3.3. Our algorithm has three major new components, the reciprocal learning, the incorporation of 3D depth map features, and the reciprocal attacks for matched prediction. In the first row of Table 3.3, we list the ADE and FDE results for our method (full algorithm). The second row shows the results for our method without reciprocal training. The third row shows results without depth map features. The last row shows results without reciprocal attacks for prediction. We can clearly see that each algorithm component is contributing to the overall performance.

With the reciprocal consistence constraints, during training, our model forces the backward predicted trajectory to be consistent with the observed past trajectory, thus the pre-

Table 3.3: Ablation experiments of our full algorithm without different components. Error metrics reported are ADE and FDE in meter scale.

Metric	Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
ADE	Our Method (Full Algorithm)	0.69	0.43	0.53	0.28	0.28	0.44
	- Without Reciprocal Learning	0.73	0.49	0.60	0.38	0.36	0.51
	- Without Depth Features	0.71	0.43	0.56	0.31	0.31	0.46
	- Without Reciprocal Attacks	0.70	0.45	0.55	0.32	0.30	0.46
FDE	Our Method (Full Algorithm)	1.24	0.87	1.17	0.61	0.59	0.90
	- Without Reciprocal Learning	1.31	0.97	1.22	0.73	0.70	0.99
	- Without Depth Features	1.30	0.88	1.19	0.63	0.62	0.92
	- Without Reciprocal Attacks	1.26	0.90	1.18	0.65	0.61	0.92

dicted future trajectory which is the input of the backward network will be forced to be closer to the ground truth. As shown in the 2nd and 6th rows of Table 3.3, the ADE increases to 0.51 from 0.44 and FDE increases to 0.99 from 0.90 on average when reciprocal consistence is excluded. By adding the depth features and reciprocal attacks, the prediction can be slightly refined to further improve the performance. Results in the 3rd and 7th rows shown in Table 3.3 shows the benefit of introducing the depth features since it can help the model to better understand human behavior and the background scene context. The reciprocal attack mechanism modifies the predicted trajectory in an iterative manner to match the original trajectory with the backward prediction network. The minor improvement of this proposed mechanism is clearly shown in the 4th and 8th rows of Table 3.3. With all these ablation experimental results, we can conclude that all three algorithm components are critical in our proposed method.

To evaluate the influence of the parameter λ in Eqn. 3.4 and 3.5, we perform ablation experiments on ZARA1 and UNIV datasets with λ value from 0.1 to 0.9. Figure. 3.8 presents how ADE and FDE changes with respect to different λ values. As we can see in

Figure. 3.8, both the forward trajectory loss and the past trajectory loss have contributions to overall performance. However, the forward trajectory loss plays a relatively more important role than the backward trajectory loss does. For example, when $\lambda = 0.1$, it indicates the weight for the forward trajectory loss is 0.1, while the weight for the backward trajectory loss is 0.9, the ADE for UNIV dataset with $\lambda = 0.1$ is greater than it with $\lambda = 0.9$.

We also perform ablation experiments to evaluate the influence of the parameter ϵ in Eqn. 3.19 which is the magnitude of reciprocal attack. As we discussed above, reciprocal attack minor refines the predicted forward trajectory to match the ground truth better. As we can see in Figure. 3.9, ϵ within a certain range has slight influence on the overall performance.

3.5.7 Qualitative Results

Figure. 3.10 shows successful and failure examples of our predicted trajectories. Following prior work *S-GAN* [40], we show the best predicted trajectory among 20 model outputs in the figure. We can see that our proposed method is able to correctly predict the future path. According to the background scene, we can see that our method can ensure that each human path follows the physical constrains of the scene, such as walking around obstacles, *e.g.* trees, and staying on sidewalks. Our method also shows the decent prediction results with human-human interactions. When persons walk in a crowded road, they can avoid each other when they merge from various directions and then walk towards a common direction.

The last row in Figure. 3.10 shows some failure cases which have relatively large error rates. For example, we see human slowing down or even stops for a while, or human taking a straight path rather than making a detour around the obstacles. Nevertheless, in most

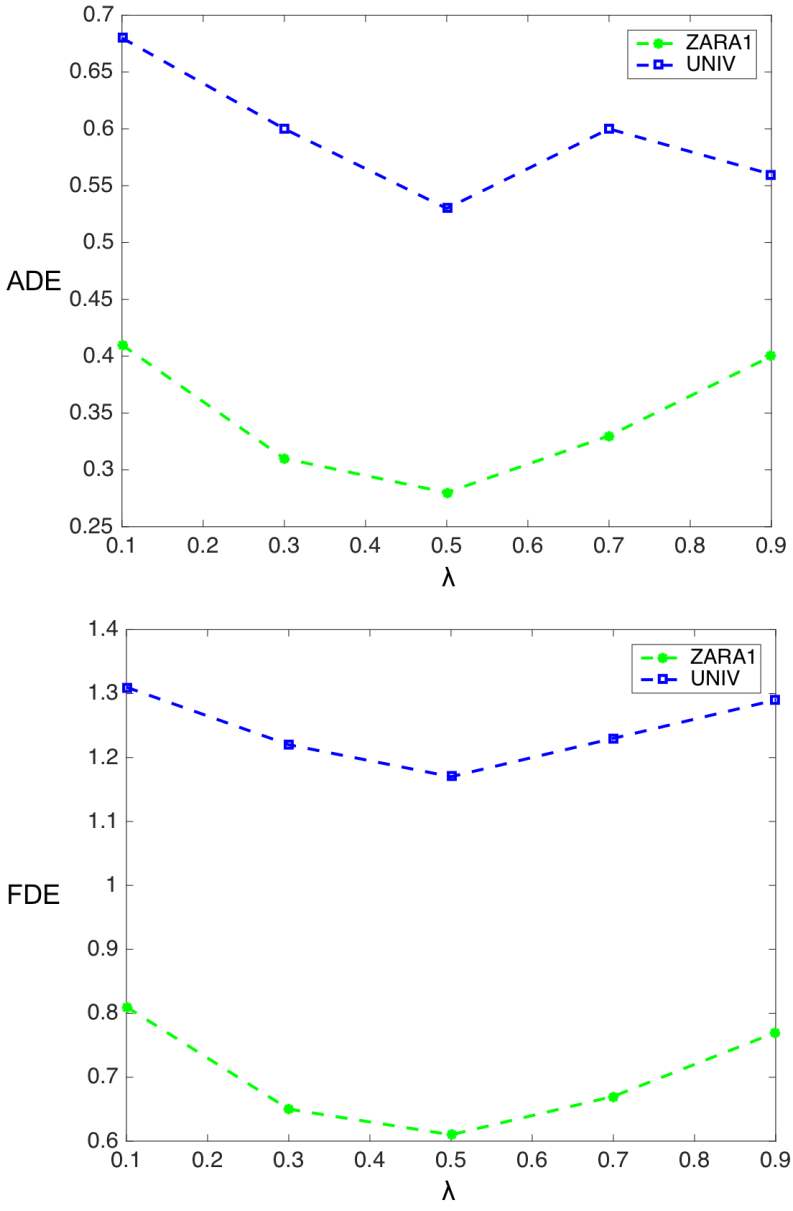

Figure 3.8: Illustration of ADE and FDE changes with respect to different λ values on ZARA1 and UNIV dataset

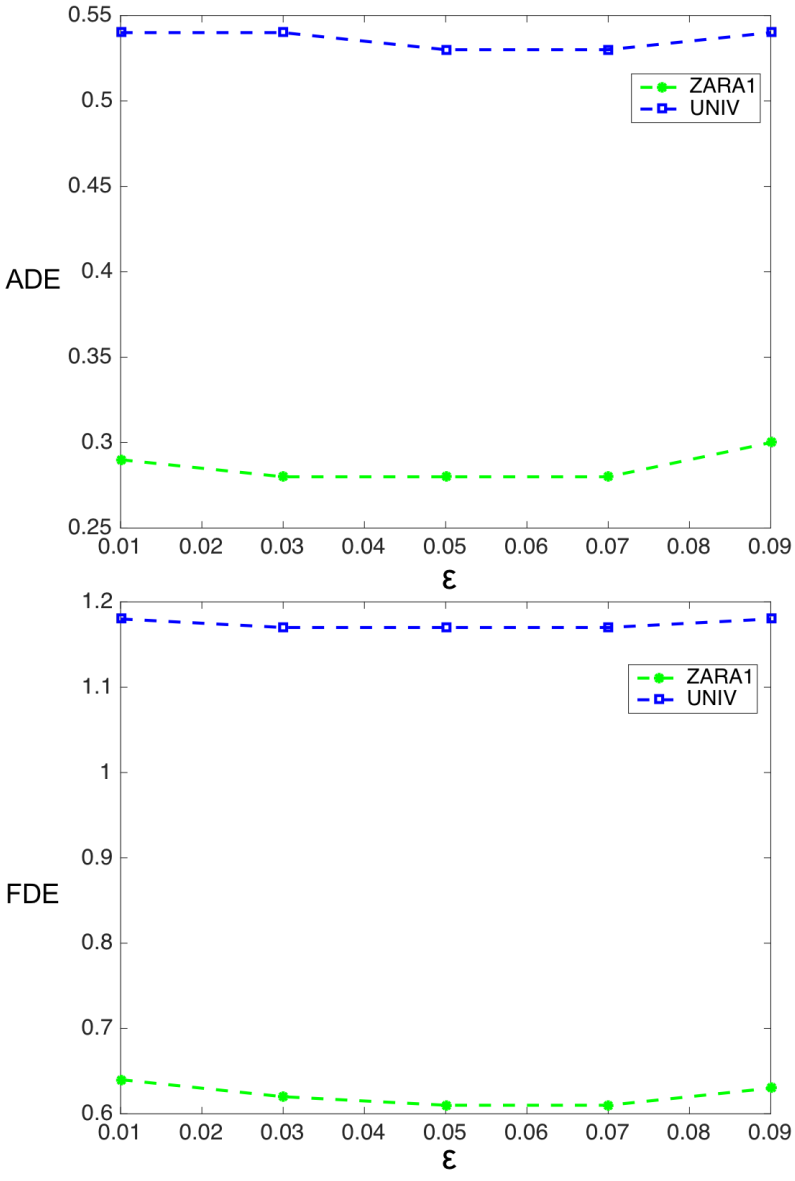

Figure 3.9: Illustration of ADE and FDE changes with respect to different ϵ values on the ZARA1 dataset

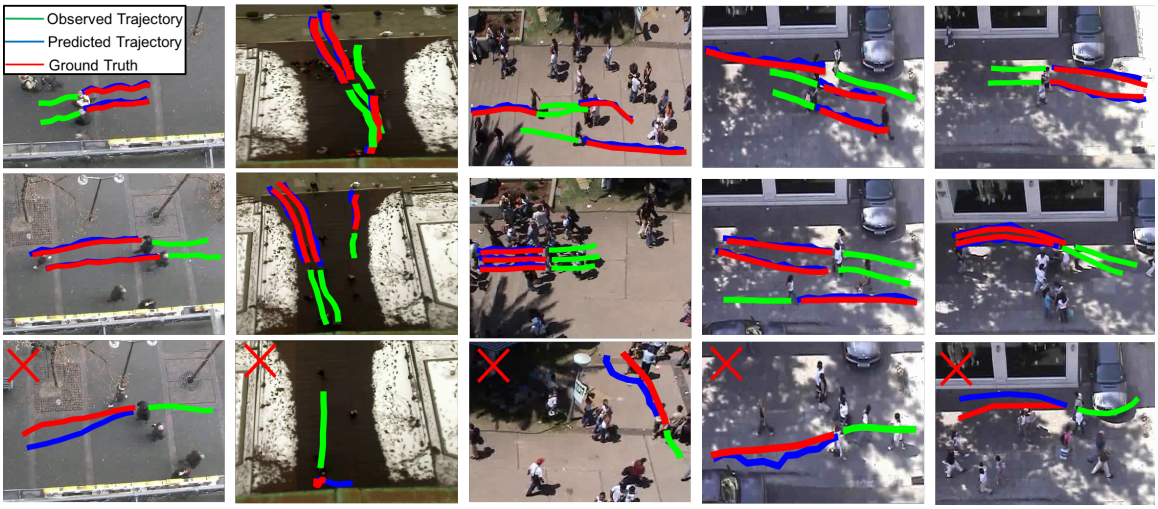

Figure 3.10: Illustration of our method predicting future 12 time steps trajectories, given previous 8 time steps. The results are drawn under HOTEL, ETH, UNIV and ZARA1 and ZARA2 datasets from 1st column to 5th column, respectively.

cases, our method still can predict the plausible path, even though the predicted path is not quite same as the ground truth. For example, for the first, third and fifth cases in the last row, in our prediction paths, the target human are trying to walk around another human or the tree in the road, which are quite reasonable in practice.

3.5.8 Generalization: Evaluations on Town Centre and Grand Central Station Datasets

To further evaluate the generalizability of our method, we perform experiments on new datasets: Town Centre [4] and Grand Central Station [144]. Following the previous work [42], for each of these two datasets, we combine the training data from ETH and UCY datasets and 50% data from this dataset for training, the remaining data is used for testing. The objective is to predict trajectories in the next 12 and 16 time steps based on the tra-

jectories of 8 previous time steps. The comparison results of our method with *S-GAN* [40] and *Scene-LSTM* [42] are shown in Table 3.4. We can clearly see that our method outperforms the existing methods in both datasets. Some qualitative examples on Town Centre and Grand Central Station datasets are presented in Figure. 3.11.

Table 3.4: The quantitative results (ADE and FDE) on Town Centre and Grand Central Station datasets with different prediction lengths of future trajectories.

Metrics	Datasets	Prediction Length	S-GAN [40]	S-GAN-P [40]	Scene-LSTM [42]	Ours
ADE	Town Center	12	0.22	0.21	0.09	**0.07**
		16	0.37	0.38	0.14	**0.09**
	Grand Central Station	12	0.21	0.40	0.11	**0.06**
		16	0.32	0.79	0.14	**0.07**
FDE	Town Center	12	0.46	0.42	0.18	**0.13**
		16	0.80	0.81	0.27	**0.18**
	Grand Central Station	12	0.45	0.74	0.17	**0.11**
		16	0.62	1.50	0.25	**0.15**

3.5.9 Backward Prediction Evaluation

We also conduct experiments of backward trajectory prediction (predict past trajectories by giving future trajectories) on the ETH and UCY datasets. We compare the ADE and FDE results with the *S-GAN* and *S-GAN-P* methods. The objective is to predict trajectories of the previous 8 time steps based on the trajectories of 12 future time steps. The prediction error results are shown in Table 3.5. We can see that, our reciprocal learning method outperforms *S-GAN* and *S-GAN-P* on both ETH and UCY datasets. The results show that our reciprocal learning is able to accurately perform both forward and backward prediction of human trajectories. Several visual examples of our backward prediction on the ETH and UCY datasets are shown in Figure. 3.12.

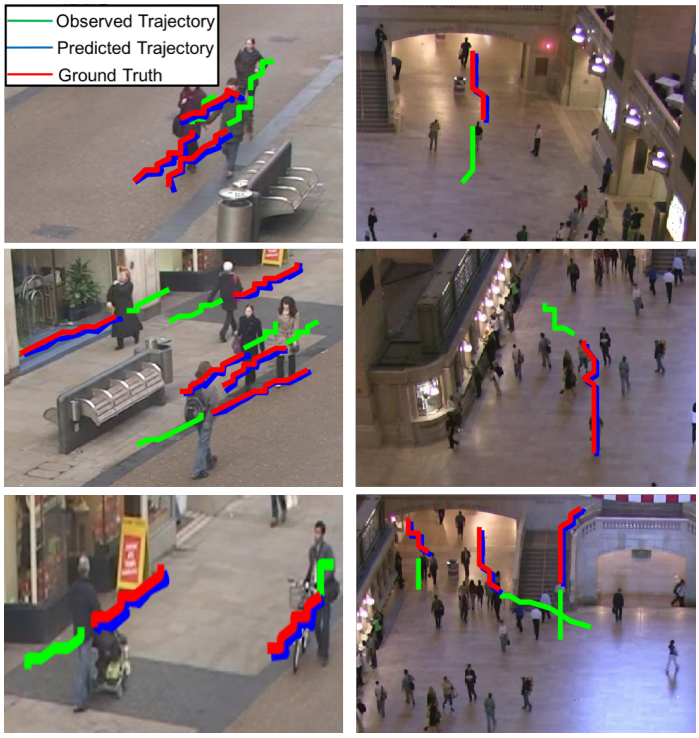

Figure 3.11: Qualitative examples of our method predicting future 12 time steps trajectories, given previous 8 time steps ones on Town Centre (1st column) and Grand Central Station (2nd column) dataset. Note that, we crop and resize the original image for better visualization.

Table 3.5: The quantitative results on ETH (Column 3 and 4) and UCY (Column 5-7) datasets on the task of backward prediction (predicting the trajectories of previous 8 time steps, given the trajectories of 12 future time steps).

Metric	Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
ADE	S-GAN [40]	0.57	0.27	0.39	0.22	0.24	0.34
	S-GAN-P [40]	0.56	0.31	0.37	0.24	0.27	0.35
	Ours	**0.50**	**0.22**	**0.31**	**0.20**	**0.18**	**0.28**
FDE	S-GAN [40]	1.05	0.68	0.74	0.42	0.43	0.67
	S-GAN-P [40]	1.07	0.72	0.71	0.43	0.49	0.68
	Ours	**0.95**	**0.44**	**0.65**	**0.40**	**0.37**	**0.56**

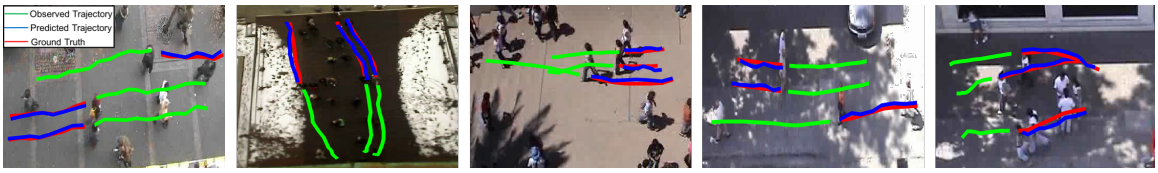

Figure 3.12: Illustration of backward prediction (predicting previous 8 time steps trajectories, given future 12 time steps ones). The results are drawn under HOTEL, ETH, UNIV and ZARA1 and ZARA2 datasets from 1st column to 5th column respectively. Note that, we crop and resize the original image for better visualization.

3.6 Conclusion

In this chapter, we have explored the unique characteristics of human trajectories and developed a new approach, reciprocal network learning, for human trajectory prediction. Two networks, the forward and backward prediction networks, are tightly coupled together, satisfying the reciprocal constraint, which allows them to be jointly learned for accurate and robust human trajectory prediction. Based on this constraint, we borrowed the concept of adversarial attacks of deep neural networks, which iteratively modifies the input of the network to match the given or forced network output, and developed a new method for network testing, called *reciprocal attack for matched prediction*. It has further improved the prediction accuracy slightly. Extensive experimental results have demonstrated our approach achieves the state-of-art performance on public benchmark datasets.

Chapter 4

Hierarchical Interactions Modeling for Human Future Trajectory Prediction

4.1 Motivation

As shown in Figure 4.1, we observe that human's future trajectory is not only affected by other pedestrians but also impacted by the surrounding objects in the scene. Human-human interactions, such as walking in groups and avoiding collisions can affect the route planning of the next time step. Also, physical constraints of the surrounding environment, such as buildings, sidewalks, trees, etc, can enable or restrict certain types of human movements.

In this chapter, we propose a novel hierarchical framework based on a recurrent sequence-to-sequence analysis architecture to model both human-human and human-scene interactions. As we can see in Figure 4.2, our method exploits three sources of information: the human trajectory information which captures each pedestrian's past trajectories, the global scene information which extracts the scene layout from the whole scene image, and

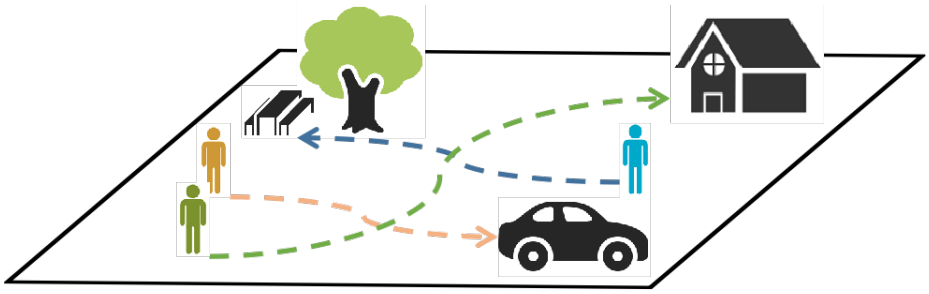

Figure 4.1: Human's future trajectory is affected not only by other pedestrians but also by the surrounding environment. Our proposed method predicts socially and physically plausible trajectories by hierarchically modeling the influence of all pedestrians in the scene as well as the global scene layout and local context.

the local image patch information which captures the scene context within a certain sized neighborhood. Our model captures the human movement patterns and learns the influence of all pedestrians involved in the scene [48]. Then, global scene layout features extracted from the whole scene image at each time step are encoded by LSTMs. The corresponding influence of these features is learned by a graph attention network (GAT). Finally, we extract local scene context features from image patches centered around each pedestrian at each time step. The context features are encoded by LSTMs and the hidden states are fed into the GAT to learn the spatial interactions of local patches. We also add an extra set of LSTMs to capture the temporal correlations of these interactions along the time steps. This novel hierarchical interactions modeling can automatically learn the influence of each pedestrian in a whole scene and the influence of the scene at multiple scales.

The rest of this chapter is organized as follows. Section 4.2 reviews and discusses related work on human trajectory prediction and GAT. Section 4.3 presents our proposed hierarchical network to model human-human and human-scene interactions. Experimental results and ablation studies are presented in Section 4.4. Section 4.5 concludes the paper.

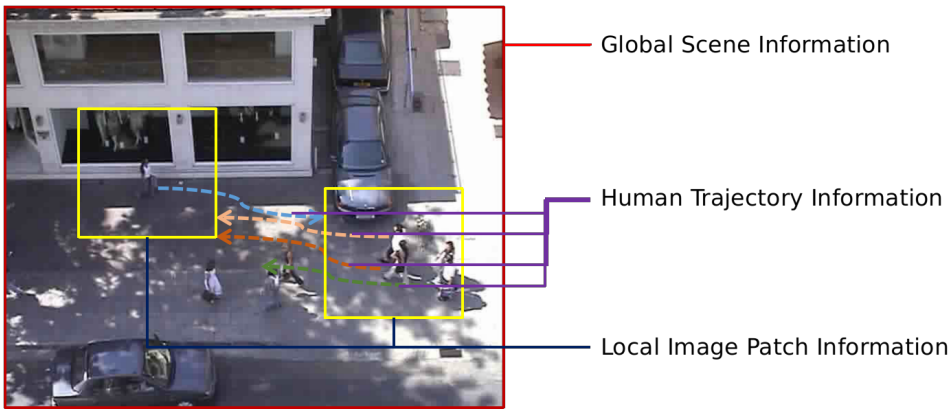

Figure 4.2: The input information observed for our method.

4.2 Related Work

Our work mainly focuses on the task of human future trajectory prediction. Researchers have designed handcrafted rules and energy parameters to capture human motions [5, 18, 31, 142, 145]. However, those methods usually fail to generalize properly [44]. Most recent works rely on utilizing Recurrent Neural Networks(RNNs) [96] to learn these parameters directly from the data or build a sequence to sequence model to predict future human trajectories by observing past human trajectories [48]. These type of methods have two main categories, modeling human and human interactions [39, 40, 48, 103] and modeling human and scene interactions [43, 106–109]. In this section, we review existing works on both human-human models and human-scene models. We also review existing work Graph Convolutional Networks (GCN) [146], which is a central component in our proposed method.

4.2.1 Human-Human Models for Trajectory Prediction

The methods of human-human models focus on learning the influence among the surrounding pedestrians. Social Force Model [37] proposed by Helbing *et al.* designed hand crafted functions to characterize pedestrians interactions in crowded scenarios to build coupled Langevin equations. Many recent works solved the problem by using LSTM networks [5]. The baseline work was introduced as Social-LSTM [39], which proposed a social pooling mechanism to share hidden representations among pedestrians. The key hypothesis of this work was that each pedestrian's moving direction and velocity are impacted by the surrounding pedestrians within a certain area. [102] observed that human commonly tend to show coherent movement patterns, thus it clustered trajectories which have similar movements into a group and propose an LSTM-based model to learn group dynamics. [40] proposed an LSTM-based Generative Adversarial Network (GAN) model to learn human-human interactions by considering all pedestrians existing in the background scene and discriminate between multiple feasible trajectories to handle the multi-modal problem of human trajectory prediction. However, the limitation of this method is that it does not differentiate between structured and unstructured environment. Social-Attention [103] argued that modeling interactions between humans as function of proximity is not necessarily true and proposed a prediction model which captures the relative importance of each surrounding pedestrian with respect to the target person in the entire scene. We observe that the above methods have not considered the impact of the nearby scene structures, such as buildings or some static obstacles, which might impact the future human movements.

4.2.2 Human-Scene Models for Trajectory Prediction

The other category of methods also take the pedestrians' interactions with their background scene structures into consideration by incorporating the extracted visual features. SS-LSTM [41] proposed to use CNN to extract the scene layout features and combined it with social interactions to learn human movements. Sadeghian *et al.* [43] explored both past trajectories of pedestrians and semantic context to learn weighted interactions between human-human and human-scene, then combine them together as the input to a Generative Adversarial Network (GAN). Liang *et al.* [104] utilized rich visual features, such each pedestrian's bounding box, keypoint information and scene semantic features for better predicting multiple plausible trajectories. Scene-LSTM [42, 108] designed a two-level grid structure to segment the static background scene into several cells and then trained two coupled LSTMs to encode both pedestrian's past movements and the scene grids. [147] proposed to encompass three pooling mechanism, such as social, navigation and semantic pooling, to capture the human-human interactions, past observations from previously crossed areas, and the scene semantics, respectively. These information are then fed into an LSTM-based model to forecast the future paths. [46] developed the reciprocal twin networks, which include a forward prediction network to predict future trajectory from past observations and a backward prediction network which performs the trajectory prediction backward in time, to form a reciprocal constraint by utilizing the property of cycle consistency. Also it combines the extracted CNN visual features and 3D depth features into encoding to better understand the environment structure.

4.2.3 Graph Neural Network

Graph neural networks (GNNs) were first introduced by [146], which are a powerful type of neural networks designed to work directly on graphs and leverage their structural information. Recently, GNNs have shown significant ability on handling the problem, such as action recognition [148, 149]. Graph Attention Networks that proposed by Velickovi *et al.* [47] were build based on the recent developments in GNNs, it can be applied to the task of a self-attention based architecture over data with any type of structured that can be represented as a graph [44]. It could assign different importance to different nodes in a graph implicitly.

A number of works applied GATs to the task of human trajectory prediction recently and have achieved state-of-the-art results. Social-BiGAT [44] proposed to formulate the human-human interactions as a graph and applied GATs as an attention mechanism to the target pedestrian to learn the influence from the surrounded pedestrians. STGAT [48] shared the same problem formulation with Social-BiGAT to model the spatial interactions between pedestrians and also designed an extra set of LSTMs to model the temporal correlations of the interactions. Social-STGCNN [150] argued that modeling human trajectories as a graph directly from the beginning is more efficient than aggregation based model like Social-BiGAT. It proposed a weighted adjacency matrix and used Temporal CNN (TCNs) [151] to quantitatively measure the influence between pedestrians.

In our work, we also formulate the complex interactions as graph structures. However, beside modeling human-human interactions using GATs like the above methods, we also try to model the human-scene interactions by building graphs for scene layout features. Two extra graphs are built, where the first one is referred to as the global scene graph and the second one is the local scene patch graph. For the global scene graph, each node

indicates the scene layout features at each time step, and the edges are the interactions of human verses the whole background scene. We also want to learn the influence of different scene configurations. So we build another graph, local scene patch graph, where nodes refer to the layout feature from a certain size image patch around each pedestrian at each time step, and edges are these interactions.

4.3 The Proposed Method

In this section, we present our hierarchical interactions modeling for human future trajectory prediction.

4.3.1 Problem Formulation

Our problem in this work is: *given the observed trajectories of a group of pedestrians in a crowded scene and the contextual information of the background environment, can we predict their future trajectories*?

Similar to prior literature [43, 44], we assume that each pedestrian's movement direction will be affected by the social impact of other pedestrians and the physical constraints of the surrounding environment. Thus, in our model, the past observed trajectories of each of the N currently visible pedestrians, $\mathbf{X} = X_1, \cdots, X_n, \cdots, X_N$, are first taken as input and referred to as the human trajectory information. The input trajectory for pedestrian n is denoted as $X_n = (x_n^t, y_n^t)$, where t indicates the time step, $t = 1, 2, \cdots, T_{obs}$. For a particular scene, the observed scene information as inputs to our model has two parts. The first one is the current scene images I^t, which is used to capture the global layout of the current environment and referred to as global scene information. While the other one

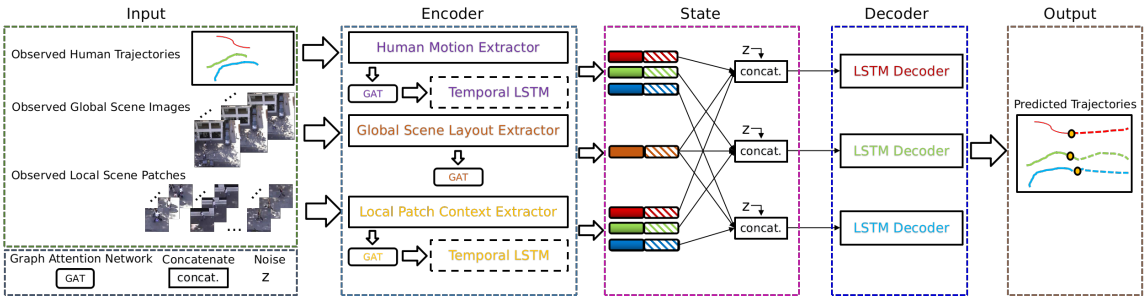

Figure 4.3: Pipeline of our proposed system. We take input from three scale information: observed human trajectories, observed global scene images and observed local scene patches. Our whole model is built based on seq2seq model, where Encoder, Intermediate State and Decoder are included. The Encoder is designed to extract spatio-temporal information from multiple sources. The spatial and temporal information concatenated with a noise are summarized in the Intermediate State and then form the input for the Decoder to yield the predicted trajectory for each observed human.

is a local image patch centered at the current position of the target pedestrian, P_n^t, which is used to capture the background context of the target pedestrian n. Given the above observed information, our goal is to predict the future trajectories of each visible pedestrian n ($\forall n \in \{1, \cdots, N\}$), $\hat{Y}_n = (\hat{x}_n^t, \hat{y}_n^t)$ for time step $t = T_{obs} + 1, \cdots, T_{pred}$. The ground truth of the future trajectories are denoted by Y_n.

4.3.2 Method Overview

Our full model starts with three feature encoding modules, where human past trajectories, observed global scene images, and local image patches are encoded to fixed-sized tensors. Then, three GAT-based modules are designed to model the human-human interactions, global human-scene interactions, and local human-scene interactions. Temporal LSTM modules are designed to capture the temporal correlations of human-human interactions and local human-scene interactions. Finally, noises are concatenated with the hidden states

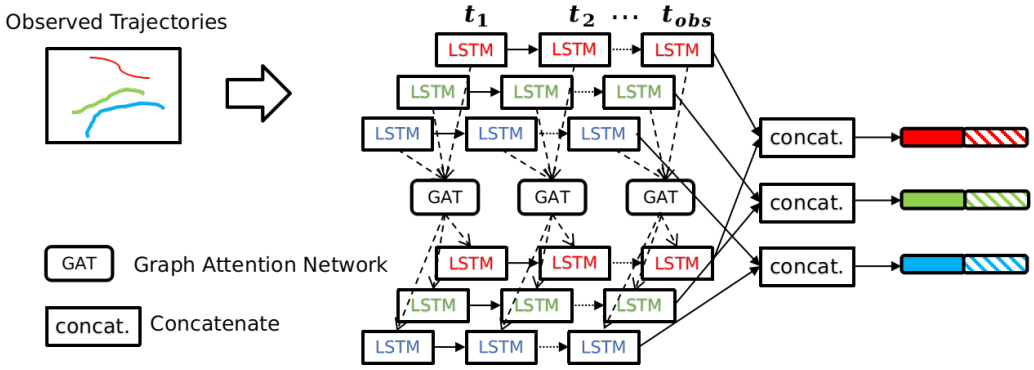

Figure 4.4: The framework of Human Trajectory Encoder. The upper level LSTMs are human motion extractor to capture the hidden motion states for each pedestrian. The middle level GATs are used to model the human-human interactions. The lower level LSTMs are applied to learn the temporal correlation of the interactions. Then the hidden motion states and the spatio-temporal vectors are concatenated as the output of Human Trajectory Encoder.

for each pedestrian and then fed into the LSTM decoder to predict the future trajectories for every pedestrians involved in the environment. An overview of our whole system is illustrated in Figure. 4.3. The details of each component will be described in the following subsections.

4.3.3 Human Trajectory Encoder

The human trajectory encoder is composed of three major components: human motion extractor, GAT-based human-human interaction modeling, and LSTM-based temporal correlation learning. The concept of human trajectory encoder is illustrated in Figure 4.4.

Human Motion Extractor

The human motion extractor is designed to capture the temporal pattern and dependency of each observed trajectory. Previous researches [39, 40] have demonstrated that the LSTM has an outstanding ability to capture the motion state of a pedestrian from the observed time-series trajectories. In our method, we also borrow the idea and build an LSTM for each pedestrian visible in a scene to get their motion states. The observed trajectory of each pedestrian n is defined as $X_n = (x_n^t, y_n^t)$ from time steps $t = 1, 2, \cdots, T_{obs}$. First, the relative position of each pedestrian to the previous time step is calculated as follows:

$$\begin{aligned} \Delta x_n^t &= x_n^t - x_n^{t-1}, \\ \Delta y_n^t &= y_n^t - y_n^{t-1}. \end{aligned} \tag{4.1}$$

Then, we use a single layer MLP [139] to embed the relative coordinates into a fixed size vector e_n^t:

$$e_n^t = \psi(\Delta x_n^t, \Delta y_n^t; W_{emb}), \tag{4.2}$$

where $\psi(\cdot)$ is an embedding function with ReLU non-linearity [140] and W_{emb} is the embedding weight. e_n^t is then fed to the LSTM cell to capture the hidden motion state M_n^t for pedestrian n at time step t:

$$M_n^t = LSTM_{en_h}(M_n^{t-1}, e_n^t; W_{en_h}), \tag{4.3}$$

where $LSTM_{en_h}$ denotes the LSTM encoder for human motion and W_{en_h} is the encoding weight for $LSTM_{en_h}$ which is shared among all pedestrians involved in the scene and can be optimized during training process.

GAT-based Human-Human Interaction Modeling

Humans intuitively know to focus more on the pedestrians who might collide with their future route. Similarly, we want our model to assign different and adaptive importance to the pedestrians in the crowded scenario. A number of prior works [39, 40, 43] assume that the order of the Euclidean distance between different pedestrians plays a key role on modeling the so-called social interactions. However, this is not necessarily always true. Specifically, the faraway pedestrian may also have great impacts on the target pedestrian's movements. To tackle this problem, some recent researches [48, 103] proposed the idea of attention up. They represented the social impacts by an attention vector, which is calculated as a weighted sum of pedestrians' current states.

In order to model the human-human interactions and share information across all pedestrians in a crowded scenario, we leverage the recent work in GATs (graph attention networks) by considering each pedestrian in the scene as one node in the graph. As shown in Figure 4.5, we build a complete graph at each time step, where the nodes are the pedestrians involved in the current scene and the edges represent the human-human interactions. This mechanism does not introduce any restriction on pedestrian orders and allows pedestrians to interact with each other.

The GAT processes the graph-structured data by aggregating information from all neighboring graph nodes following a self-attention strategy. Usually, several stacked graph attention layers are applied in the networks. The computing mechanism of a single attention layer is illustrated in Figure 4.6. In our work, the input of the graph attention layer is the hidden motion states $M_n^t \in \{M_1^t, M_2^t, \cdots, M_N^t\}$ for observed pedestrians $1, 2, \cdots, N$ at time step t, where $M_n^t \in \mathbb{R}^F$, F is the feature dimension of M_n^t. The output is the aggregated feature ${M'}_n^t$, where ${M'}_n^t \in \mathbb{R}^{F'}$. Note that the input and output feature dimensions,

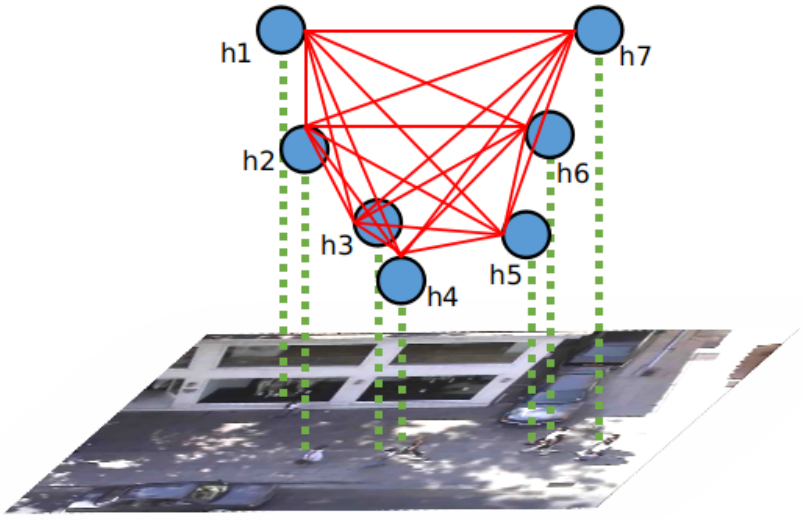

Figure 4.5: An illustration of the complete graph we build at each time step. Each node denotes each human ($h_1, h_2, \cdots, h_n$) and the edges represent the human-human interaction.

F and F', can be different. The interaction coefficients ${\alpha_{h_{nm}}}^t$ of pedestrian m to pedestrian n at time step t is given by:

$$ {\alpha_{h_{nm}}}^t = \frac{\exp(\Phi(a_h^T[W_{att_h}M_n^t \oplus W_{att_h}M_m^t]))}{\sum_{k\in\Psi_n}\exp(\Phi(a_h^T[W_{att_h}M_n^t \oplus W_{att_h}M_k^t]))}, \tag{4.4} $$

where $\oplus$ denotes the concatenation operation. $a_h \in \mathbb{R}^{2F'}$ is the weight vector of a single layer feed-forward neural network which is normalized by a softmax function with LeakyReLU denoted by $\Phi(\cdot)$. a_h^T indicates the transposition of a_h. Ψ_n represents the set of the neighboring nodes of node n on the graph. $W_{att_h} \in \mathbb{R}^{F'\times F}$ is the weight matrix of a shared linear transformation. The aggregated output of the graph attention layer for pedestrian n at time step t can be computed by:

$$ {M'}_n^t = \Theta(\sum_{m\in\Psi_n} {\alpha_{h_{nm}}}^t W_{att_h}M_m^t), \tag{4.5} $$

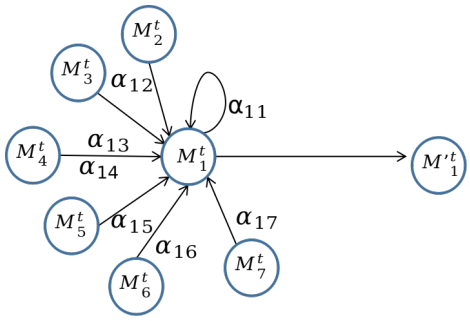

Figure 4.6: An illustration of the computing mechanism of a single graph attention layer. It aggregates information from each neighboring node and following a self-attention strategy. $M_1^t, M_2^t, \cdots, M_N^t$ indicates the hidden motion states for pedestrians $1, 2, \cdots, N$ and $\alpha_{11}, \alpha_{12}, \cdots, \alpha_{1N}$ denotes the importance of the corresponding pedestrian with respect to pedestrian 1.

where $\Theta(\cdot)$ represents a nonlinear function.

Temporal Correlation Learning for Human-Human Interactions

By far, we model human-human interactions by sharing hidden states among pedestrians at the same time step. However, these operations only capture the spatial information of human-human interactions. In our work, we design an extra set of LSTMs to learn the temporal correlations between those interactions. The input of the temporal LSTM ($LSTM_{t_h}$) is the output of the graph attention layers, M'^t_n got from Eq. (4.5), and the operation of $LSTM_{t_h}$ is given by

$$T_{h_n}^t = LSTM_{t_h}(T_{h_n}^{t-1}, M'^t_n; W_{t_h}), \tag{4.6}$$

where $T_{h_n}^t$ is the hidden temporal correlation state of human-human interactions and W_{t_h} is the weight for $LSTM_{t_h}$ and shared among all sequences.

In the last step of Human Trajectory Encoder, we concatenate the hidden motion states

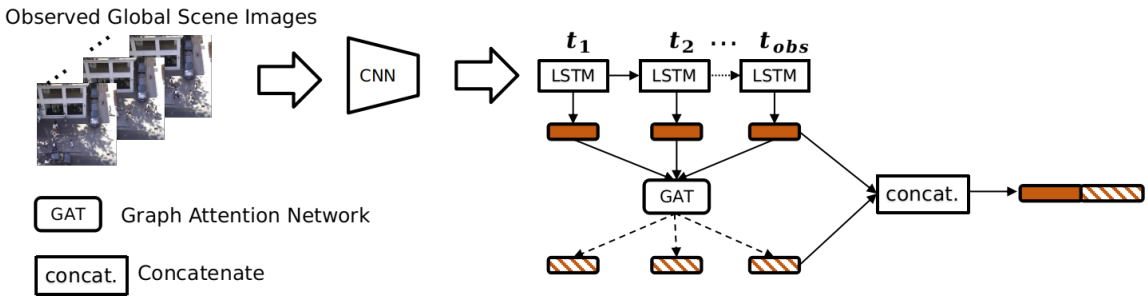

Figure 4.7: The framework of Global Scene Layout Encoder. The inputs are the observed scene images at each time step. A CNN is used to extract the scene layout features, which are then fed to LSTMs to compute the hidden state vectors. GAT is applied to model the global human-scene interactions.

M_n^t obtained from Eq. (4.3) and the temporal correlations of human-human interactions T_{hn}^t got from Eq. (4.6) to form the spatiotemporal information learned from the observed human trajectories.

$$H_{en} = M_n^t \oplus T_{hn}^t. \tag{4.7}$$

4.3.4 Global Scene Layout Encoder

As discussed in [41, 43], the chosen route of a pedestrian is also highly affected by the scene layouts, such as stationary obstacles, moving cars, entries, exits, etc. However, compared to model the influence of other pedestrians, a relatively small number of recent work paid attention to modeling the influence of the scene structures on human trajectory prediction. In our work, we propose to utilize Convolutional Neural Networks (CNNs) to extract global information of the scene and applied GAT to model the global human-scene interactions. An illustrative framework of our Global Scene Layout Encoder is presented in Figure 4.7.

Global Scene Layout Extractor

First, we use a CNN to extract the global scene layout feature l^t from scene image I^t at each time step t, which is given by

$$l^t = CNN(I^t), \tag{4.8}$$

where CNN here is the VGGNet-19 network [45] that is pre-trained on the ImageNet [45]. The extracted scene layout feature l^t is then fed to an LSTM encoder to compute the hidden state vector L^t for each observed pedestrian at time step t

$$L^t = LSTM_{en_s}(L^{t-1}, l^t; W_{en_s}), \tag{4.9}$$

where W_{en_s} is the associated encoding weight.

GAT-based Global Human-Scene Interaction Modeling

Since the camera is stationary in our task, the background scene is almost the same in most cases. However, the pedestrians in the scene are moving. The configuration of the observed scene images also changes with time due to the movement of these pedestrians. Therefore, the scene features between the observed scene images at different time steps are mainly caused by the pedestrians' movements.

In order to model these human-scene interactions, we propose to use the GAT to assign different and adaptive influence to the hidden states of the scene features at different time steps. First, we build a complete graph by considering each hidden state of the scene features at each time step as nodes and the interactions as edges, similar with Figure. 4.6.

The inputs to graph attention layers are the hidden states of the scene features at each time step, $L^1, L^2, \cdots, L^{t_{obs}}$. The interaction coefficients $\alpha_{g_{tt'}}$ of the scene hidden states $L^{t'}$ to L^t is defined as follows:

$$\alpha_{g_{tt'}} = \frac{\exp(\Phi(a_g^T[W_{att_g}L^t \oplus W_{att_g}L^{t'}]))}{\sum_{k \in \Psi_{t_{obs}}} \exp(\Phi(a_g^T[W_{att_g}L^t \oplus W_{att_g}L^k]))}. \tag{4.10}$$

Similar to Eq. (4.4), a_g is the weight vector of a single layer feed-forward neural network which is normalized by a softmax function with Leaky ReLU denoted by $\Phi(\cdot)$. a_g^T represents the transposition of a_g. $\Psi_{t_{obs}}$ represents the set of the scene feature hidden states from time step 1 to t_{obs}. W_{att_g} is the weight matrix of a shared linear transformation. The aggregated output of the graph attention layer for scene feature hidden state at time step t can be computed by

$$L'^t = \Theta(\sum_{t' \in \Psi_{t_{obs}}} \alpha_{g_{tt'}} W_{att_g} L^t), \tag{4.11}$$

where $\Theta(\cdot)$ denotes a nonlinear function. Finally, The global scene layout hidden state and global human-scene interaction are combined together to form the output of global scene layout encoder for each pedestrian,

$$G_{en} = L^t \oplus L'^t, \tag{4.12}$$

where $\oplus$ indicates the concatenation operation.

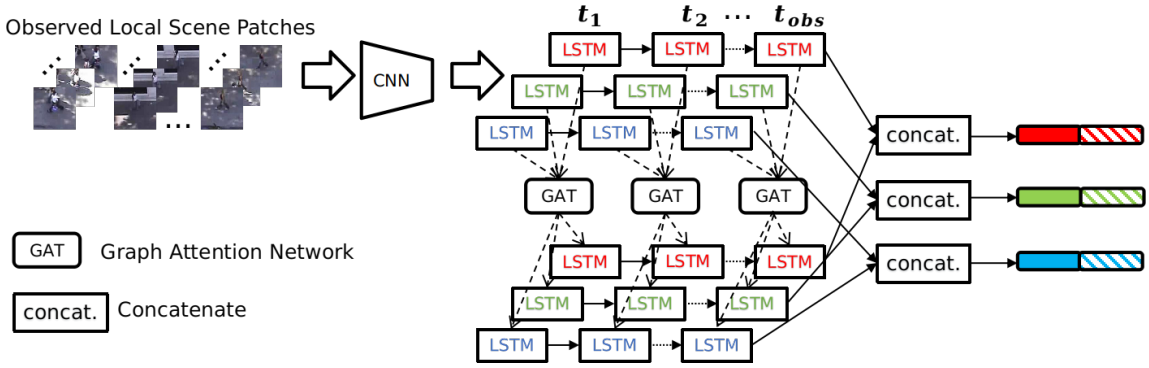

Figure 4.8: The framework of Local Patch Context Encoder. The inputs are the observed certain-sized local image patches that centered on the position of each pedestrian at each time step. A CNN is used to extract the local context features, which are then fed to LSTMs to capture the hidden state of local context. GAT is applied to model the local human-scene interactions. An extra set of LSTMs are designed to learn the temporal correlations of these interactions. The tensors from the upper level LSTMs and the lower level LSTMs are concatenated to form the output of the Local Patch Context Encoder.

4.3.5 Local Patch Context Encoder

The above mechanisms do not characterize the relationship between the past context at a specific area of a scene and the current movements of a target pedestrian. In this work, we model these relationships by learning the influence of the scene context in a local image patch centered around the pedestrian at each time step. The concept is illustrated in Figure 4.8.

Local Patch Context Extractor

The local patch for pedestrian n at time step t is denoted as P_n^t, which is segmented from the scene image with a certain size, e.g. 128 is used in our work. First, we use a pre-trained CNN to extract the context features c_n^t of the local patch P_n^t. Here the CNN is the

VGGNet-19 network [45] which is pre-trained on the ImageNet [45].

$$c_n^t = CNN(P_n^t) \tag{4.13}$$

These features are then fed into the LSTMs to capture the hidden state of the local context.

$$C_n^t = LSTM_{en_p}(C_n^{t-1}, c_n^t; W_{en_p}), \tag{4.14}$$

where W_{en_p} is the corresponding weight for $LSTM_{en_p}$.

GAT-based Local Human-Scene Interactions

In the above section, we try to model the influence of surrounding pedestrians and the scene layout of the environment the target pedestrian involved with. However, using global scene layout information to model human-scene interaction may ignore some details of scene context around pedestrians. We observe that a target pedestrian's future path planning is not only impacted by his surrounding scene context but also impacted by the surrounding scene context of the pedestrians around him. For example, the entrance of a building is included in both image patches for two pedestrians, and they are walking towards the building from different directions. If this information is observed, the target pedestrian may make some adjustments to his future path selection. We introduce this type of human-scene interactions as local human-scene interactions to further understand human movements.

Similar to modeling the interactions in the above sections, we try to model local human-scene interactions by leveraging GAT. A complete graph is constructed with nodes as the hidden state of local context for each pedestrian at same time step and edges as the interactions between the nodes. The adaptive influence ${\alpha_l}_{nm}^t$ from the local patch P_m^t to P_n^t at

time step t is computed by

$$\alpha_{l}{}_{nm}^{t} = \frac{\exp(\Phi(a_l^T[W_{att_l}C_n^t \oplus W_{att_l}C_m^t]))}{\sum_{k\in\Psi_n}\exp(\Phi(a_l^T[W_{att_l}C_n^t \oplus W_{att_l}C_k^t]))}, \tag{4.15}$$

where W_{att_l} is the weight matrix of a shared linear transformation which is applied to each node. a_l is the weight vector of a single layer feed-forward neural network which is normalized by a softmax function with Leaky ReLU denoted by $\Phi(\cdot)$. a_l^T indicates the transposition of a_l. $\oplus$ is used here to denote the concatenation operation. Ψ_n represents the set of the neighboring nodes of node n on the graph. With the normalized attention coefficients $\alpha_{l}{}_{nm}^{t}$, the aggregated output of the graph attention layer for pedestrian n at time step t is defined as follows:

$$C'^{t}_{n} = \Theta(\sum_{m\in\Psi_n}\alpha_{l}{}_{nm}^{t}W_{att_l}C_m^t), \tag{4.16}$$

where $\Theta(\cdot)$ represents a nonlinear function. Note that global scene layout feature extraction and global human-scene interaction modeling are in sync with the human trajectory information captured by Human Trajectory Encoder.

Temporal Correlation Learning for Local Human-Scene Interactions

In order to gather the spatial and temporal information, we also design an extra set of temporal LSTMs to learn temporal correlation between the local human-scene interactions modeled at the same time step.

$$T_{l_n}^{t} = LSTM_{t_l}(T_{l_n}^{t-1}, C'^{t}_{n}; W_{t_l}), \tag{4.17}$$

where $T_{l_n}^t$ is the hidden temporal correlation state of local human-scene interactions C'^t_n, and W_{t_l} is the associated weight. As we discussed above, both the local scene context of the target pedestrian and the local human-scene interactions are useful to help us to better learn human behavior. By taking advantage of the spatiotemporal information, we combine these two parts together, which is give by:

$$L_{en} = C_n^t \oplus T_{l_n}^t, \tag{4.18}$$

where $\oplus$ indicates the concatenation operation.

4.3.6 Future Trajectory Prediction

As illustrated in Figure 4.3, after the Encoder stage, the intermediate state tensor S_n^t for each pedestrian is formed by concatenating human motion hidden state H_{en} (from Eq. 4.7), global scene layout hidden state G_{en} from Eq. (4.12), and local context hidden state L_{en} from Eq. (4.18) with a randomly sampled noise Z from $\mathcal{N}(0, 1)$.

$$S_n^t = H_{en} \oplus G_{en} \oplus L_{en} \oplus Z. \tag{4.19}$$

Then, the intermediate state tensor is fed into the LSTM decoder as the initial hidden state to predict the future relative coordinates Y for each pedestrian, which is attained by

$$\begin{aligned} \hat{S}_n^t &= LSTM_{de}(\hat{S}_n^{t-1}, S_n^t; W_{de}), \\ Y_t^n &= (\hat{x}_n^t, \hat{y}_n^t) = W_o \hat{S}_n^t + b_o, \end{aligned} \tag{4.20}$$

where W_{de} is the weight matrix of the LSTM decoder, W_o and b_o are the corresponding weight and bias term of the linear output layer. The relative coordinates can be then easily converted to the real coordinates according to Eq. (4.1).

Similar to the previous works [40, 48], given limited observed information, our model tries to learn human motion patterns and generate multiple both physically and socially feasible trajectories. A number of previous works [39, 41, 108] proposed to generate the future trajectory by sampling from a Gaussian distribution, where the parameters for the distribution are trainable by minimizing the negative log-likelihood loss of the ground truth. However, as discussed in [40], the sampling process is non-differentiable, so a certain amount of difficulty will be introduced in the back-propagation stage.

Thus, in our work, we follow the similar strategy from prior work [40, 48] to model the multi-modal properties of human movements by introducing a variety loss to encourage diversity of generated trajectories from the network. For each pedestrian, k possible trajectories are generated by introducing a random noise $Z \sim \mathcal{N}(0, 1)$ before the decoder stage. Then as the final production, the estimated trajectory that is nearest to the ground truth is chosen. The variety loss is given by:

$$L_{variety} = \min_{k} ||Y_n - \hat{Y}_n^t||_2, \tag{4.21}$$

where Y_n is the ground truth of pedestrian n. k is a hyperparameter and $\hat{Y}_n^t$ is the predicted trajectory returned by our network. By only taking the best trajectory in $L2$ sense into consideration, this loss strengthens our model to generate the outputs which are consistent with the observed information.

4.4 Experimental Results

In this section, we compare the performance of our method against a number of state-of-the-art methods on two public and commonly used datasets: ETH [142] and UCY [143]. In order to evaluate the generalization ability of our method, we also perform experiments on the other two datasets, Town Centre [4] and Grand Central Station [144].

4.4.1 Datasets

ETH [142] and UCY [143] are two publicly available and commonly used datasets that consist of 2D real-world human trajectories and bird's-eye view images to evaluate the performance of human future trajectory prediction. 5 subsets, such as ETH, HOTEL, UNIV, ZARA1 and ZARA2, with 4 different background scenes are included. In total, there are 1536 pedestrians in the crowded scenarios, where contain various human movement patterns, e.g. pedestrians walking in a same direction, pedestrians crossing each other, collision avoidance, etc.

The Town Centre dataset [4] is originally collected to evaluate the perform of human tracking. It consists of a short video with hundreds of pedestrians in a real-world crowded scene. The annotation file contains bounding boxes of each pedestrian's body and head. By following some prior works [41, 42], we define the location of a pedestrian as the center position of his/her body bounding box. The trajectory data is collected in every 5th frame. Grand Central Station dataset [144] is originally collected to analyze human behaviors. A long-duration video (about 33:20 mins) is recorded in a crowded station, which contains trajectories of about 12,600 pedestrians. Both Town Centre and Grand Central Station datasets consist of considerable amounts of human-human and human-scene interactions.

4.4.2 Implementation Details

In our work,our model is constructed using LSTMs and GATs. The hidden state dimension of the LSTM encoder, temporal LSTM, and LSTM decoder is 32. Following prior work [43], the scene features with a size of 512 are extracted by the VGGNet-19 network [45] which is pre-trained on the ImageNet [45]. The local image patch size is 128×128. Two graph attention layers [152] are applied to compute the interactions. The hidden dimension and output dimensions of the graph attention layer are 16 and 32, respectively. Our model is trained with the Adam optimizer with the initial learning rate of 0.001 and batch size of 64. The hyper-parameter k in Eq. (4.21) is set as 20 for evaluation.

4.4.3 Evaluation Metrics and Protocol

We evaluate our performance by two commonly used metrics, such as Average Displacement Error (ADE) and Final Displacement Error (FDE), by following the prior works [39, 40, 43, 48]. ADE (Eq. 4.22) reports the average Euclidean distance between our predicted trajectories and the ground truth trajectories from time step $T_{obs}+1$ to T_{pred}. FDE in Eq. (4.23) reports the Euclidean distance between the final position of the predicted trajectory and ground truth trajectory at time step T_{pred}.

$$\text{ADE} = \frac{\sum_{n\in\Psi}\sum_{t=T_{obs}+1}^{T_{pred}}\sqrt{(\hat{x}_n^t - x_n^t)^2 + (\hat{y}_n^t - y_n^t)^2}}{|\Psi|\cdot T_{pred}}, \tag{4.22}$$

$$\text{FDE} = \frac{\sum_{n\in\Psi}\sqrt{(\hat{x}_n^{T_{pred}} - x_n^{T_{pred}})^2 + (\hat{y}_n^{T_{pred}} - y_n^{T_{pred}})^2}}{|\Psi|}, \tag{4.23}$$

where $(\hat{x}_n^t, \hat{y}_n^t)$ and (x_n^t, y_n^t) are the predicted and ground truth trajectory coordinates for pedestrian n at time t, Ψ represents the set of observed pedestrians and $|\Psi|$ denotes the total number of observed pedestrians.

For comparison on the ETH and UCY datasets, we follow the standard protocol in prior work [40]. The leave-one-out evaluation protocol is used on the 5 subsets, such as 4 datasets are used for training and the remaining one is used for testing. Note that, since a video of UNIV subset is not available, we use a small a smaller test set for UNIV and a smaller training set across all splits [106]. The other 4 test subsets (ETH, HOTEL, ZARA1, ZARA2) are totally same as our comparison baselines, so the results are comparable. We observe the human trajectories and the scene images for the past 8 time steps (3.2 seconds) in order to predict trajectories in next 12 time steps (4.8 seconds). For the generalization experiments on Town Centre and Grand Central Station dataset, we follow the previous work [42] to normalize the location coordinates to $[0, 1]$ and split the whole data into half for training and testing respectively.

4.4.4 Comparison with Existing Methods

We compare our experimental result on ETH and UCY datasets against the following recent state-of-the-art methods:

- *S-GAN* [40]: This is one of the first GAN-based methods. Two variants, such as *S-GAN* and *S-GAN-P*, are different in whether applying the pooling mechanism. The hyperparameter k in the variety loss is set as 20 for evaluation.

- *SoPhie* [43]: This method observes human trajectories and scene images, and applies a soft attention mechanism for both features.

- *Scene-LSTM* [42]: This method designs two coupled LSTMs to encode both pedestrian's past trajectories and the scene grids.

- *Next* [106]: This method extracts multiple visual features, such as pedestrian's bounding box, pedestrian keypoints and scene semantic features for encoding, and use a LSTM decoder to predict future trajectories. To encourage the multiple feasible paths, the authors train 20 different models with random initialization by following [40].

- *Social-BiGAT* [44]: This method leverages GAT to model the human-human interactions and applied a soft attention mechanism on the extracted visual features.

- *STGAT* [48]: This method captures the spatial interactions by GAT and also design an extra sets of LSTM to extract the temporal information along the observed time steps.

4.4.5 Quantitative Results

We present the comparison results of our method against a number of recent state-of-the-art methods on commonly used ETH and UCY datasets. Two metrics, ADE and FDE, are reported in meter scale in Table 4.1. As illustrated in Table 4.1, our method outperforms all baselines in most cases, except on the ETH dataset against *Scene-LSTM*. *S-GAN* and *S-GAN-P* perform the worst as expected, since both baselines do not utilize the scene information and directly assign the human influence by the distances with the target pedestrian. Both *Scene-LSTM* and *Sophie* performs better than *S-GAN* due to the introduction of the scene information. *Sophie* outperform *Scene-LSTM* in general since the combination of the so-called social and physical attentions applied. *Next* encodes various visual features

Table 4.1: Comparison results of all the baselines and our method on ETH (Column 3 and 4) and UCY (Column 5-7) datasets on the task of predicting 12 future time steps, by given the 8 previous time steps. All models takes as Error metrics reported are ADE / FDE in meter scale.

Metric	Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
ADE	S-GAN, CVPR'18 [40]	0.81	0.72	0.60	0.34	0.42	0.58
	S-GAN-P, CVPR'18 [40]	0.87	0.67	0.76	0.35	0.42	0.61
	SoPhie, CVPR'19 [43]	0.70	0.76	0.54	0.30	0.38	0.54
	Scene-LSTM, ISCV'19 [42]	**0.36**	0.95	0.63	0.45	0.40	0.56
	Next, CVPR'19 [106]	0.73	0.30	0.60	0.38	0.31	0.46
	STGAT, ICCV'19 [48]	0.65	0.64	0.52	0.34	0.29	0.43
	Social-BiGAT, NIPS'19 [44]	0.69	0.49	0.55	0.30	0.36	0.48
	Ours	0.59	**0.29**	**0.51**	**0.32**	**0.28**	**0.40**
FDE	S-GAN, CVPR'18 [40]	1.52	1.61	1.26	0.69	0.84	1.18
	S-GAN-P, CVPR'18 [40]	1.62	1.37	1.52	0.68	0.84	1.21
	SoPhie, CVPR'19 [43]	1.43	1.67	1.24	0.63	0.78	1.15
	Scene-LSTM, ISCV'19 [42]	**0.67**	1.77	1.41	1.00	0.90	1.15
	Next, CVPR'19 [106]	1.65	0.59	1.27	0.81	0.68	1.00
	STGAT, ICCV'19 [48]	1.12	0.66	1.10	0.69	0.60	0.83
	Social-BiGAT, NIPS'19 [44]	1.29	1.01	1.32	0.62	0.75	1.00
	Ours	1.07	**0.57**	**1.09**	**0.68**	**0.59**	**0.80**

extracted from the scene images and employs the focal attention on the encoded features to achieve a better result than *Sophie*. We share the same mechanism with *STGAT* and *Social-BiGAT* to model the human-human interactions by leveraging GAT. *STGAT* applies an extra set of LSTMs to capture the tempoal information, while *Social-BiGAT* extracts the scene image features and introduce them with a soft attention mechanism. Different from them, in addition to using GAT to model the human-human interactions, we also model the interactions between human and the scene in two scales. Overall, our method achieves the state-of-the-art performance and outperforms all comparison baselines in both ADE and FDE on average.

We also compute an evaluation metrics called *near-collisions* that is proposed in [43] to further evaluate the ability of our method in predicting reasonable and feasible paths in the crowded scenarios. *near-collisions* represents the percentage of two pedestrians getting

Table 4.2: Average percentage of colliding pedestrians for each scene in ETH and UCY datasets. The first column represents the ground truth.

	GT	S-GAN [40]	SoPhie [43]	Ours
ETH	0.000	2.509	1.757	**1.447**
HOTEL	0.092	1.752	1.936	**1.326**
UNIV	0.124	0.559	0.621	**0.514**
ZARA1	0.000	1.749	1.094	**1.172**
ZARA2	0.732	2.020	1.464	**1.315**
Avg	0.189	1.717	1.361	**1.155**

closer than 0.1m. The average percentages of near-collisions across all frames in ETH and UCY datasets are reported in Table 4.2, the results of *S-GAN* and *Sophie* are cited from [43]. As we can see in Table 4.2, our method outperforms both *S-GAN* and *Sophie*, which indicates our method generates more reasonable paths to prevent pedestrian collisions.

4.4.6 Ablation Studies

Compared with the baselines, our method has two major new components, the global scene layout encoder and local scene context encoder.To evaluate the importance and the contribution of each new component, several ablation experiments are performed and the ADE and FDE are reported in Table 4.3. As illustrated in the second and fifth rows of Table 4.3, the error metrics ADE and FDE both increase when we exclude the global scene layout encoder. With the global scene layout encoder, our model can learn the influence of the entire layout of the scene to generate better physical reasonable future trajectories. In the third and sixth rows of Table 4.3, we list the results without the local scene context encoder. Both ADE and FDE increase since the model lacks the knowledge of the scene context in a local neighborhood. From the last column of Table 4.3, we can see that both our two new components have contributions to the overall performance. While the global scene lay-

Table 4.3: Ablation experiments of our full algorithm without different components. Error metrics reported are ADE and FDE in meter scale.

Metric	Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
ADE	Our Method (Full Algorithm)	0.59	0.29	0.51	0.32	0.28	0.40
	- Without Global Scene Layout Encoder	0.66	0.47	0.52	0.30	0.30	0.45
	- Without Local Scene Context Encoder	0.63	0.45	0.52	0.31	0.29	0.44
FDE	Our Method (Full Algorithm)	1.07	0.57	1.09	0.68	0.59	0.80
	- Without Global Scene Layout Encoder	1.25	0.72	1.10	0.69	0.60	0.87
	- Without Local Scene Context Encoder	1.28	0.78	1.11	0.68	0.61	0.89

out encoder may potentially be slightly more helpful in ADE computation, the local scene context encoder may generate a slightly more precise destination.

4.4.7 Qualitative Results

As we discussed before, the task of human trajectory prediction is challenging and our goal is to predicted both socially acceptable and physical acceptable future paths. Given a number of pedestrians in a crowded scene, many complicated interactions may occur, such as group moving, walking towards each other, change directions to avoid collisions with other pedestrian or stationary obstacles, etc. We present some qualitative results that represent several interactions in Figure. 4.9. As shown in the first figure in the first row, a pedestrian is walking out of the building and three other pedestrians are walking towards a same direction without collisions. In the second figure in the first row and the last figure in the second row, our model can learn the interactions between the pedestrian and the obstacles like cars and trees, therefore generate the trajectories with a changing direction. From the last figure in the first row and the first figure in the second row, we can see that our model can generate reasonable and feasible paths for multiple pedestrians walking towards

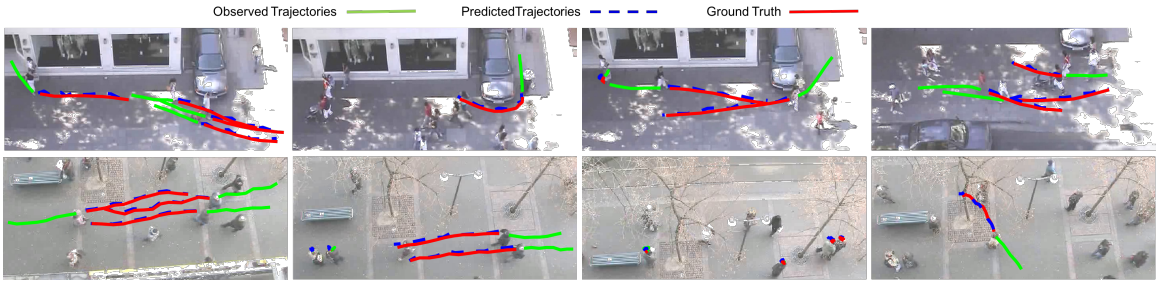

Figure 4.9: Qualitative examples of our method predicting future 12 time steps trajectories, given previous 8 time steps ones on ETH and UCY dataset. Note that, we crop and resize the original image for better visualization.

each other. Our model also performs well on the pedestrians that almost stay still. As shown in the second and third figures in the second row, our model can learn that pedestrians may stand in the same place waiting for the train or chatting with each other. These successful qualitative results can demonstrate that our method can predict reasonable and feasible future trajectories in a complex and crowded scene.

4.4.8 Generalization Evaluations on Town Centre and Grand Central Station Datasets

We conduct experiments on two new datasets, Town Centre [4] and Grand Central Station [144], to further evaluate the generalization capability of our method. The experiment setting is exactly same as [42]. For both datasets, the training data is formed by combining the training data from ETH and UCY datasets and 50% data from this dataset. While the remaining data is used for testing. We compare the results of predicting trajectories in next 12 and 16 time steps against *S-GAN* [40] and *Scene-LSTM* [42]. As illustrated in Table 4.4, our method clearly outperforms these two state-of-the-art baselines. Some qualitative examples on both datasets are presented in Figure. 4.10.

Table 4.4: The quantitative results (ADE and FDE) on Town Centre and Grand Central Station datasets with different prediction lengths of future trajectories.

Metrics	Datasets	Prediction Length	S-GAN [40]	S-GAN-P [40]	Scene-LSTM [42]	Ours
ADE	Town Center	12	0.22	0.21	0.09	**0.07**
		16	0.37	0.38	0.14	**0.10**
	Grand Central Station	12	0.21	0.40	0.11	**0.08**
		16	0.32	0.79	0.14	**0.11**
FDE	Town Center	12	0.46	0.42	0.18	**0.11**
		16	0.80	0.81	0.27	**0.19**
	Grand Central Station	12	0.45	0.74	0.17	**0.14**
		16	0.62	1.50	0.25	**0.17**

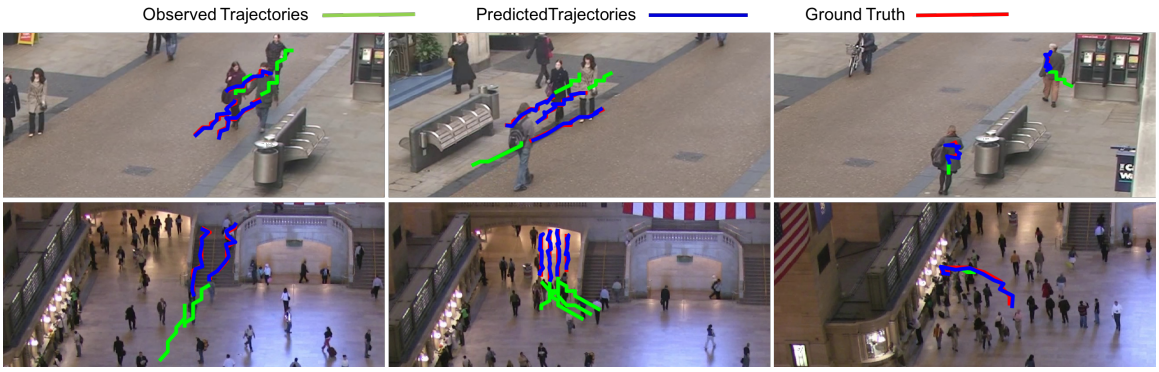

Figure 4.10: Qualitative examples of our method predicting future 12 time steps trajectories, given previous 8 time steps ones on Town Centre (1st row) and Grand Central Station (2nd row) dataset. Note that, we crop and resize the original image for better visualization.

4.5 Conclusion

In this chapter, we have developed a novel hierarchical framework based on a recurrent sequence-to-sequence architecture to jointly predict the future trajectories of all pedestrians involved in a crowded scenario. We use LSTMs to encode the past trajectory information, global scene layout information and local scene context information. Then graph attention networks are utilized to model the interactions between human-human and human-scene

at each time step. Moreover, we design extra LSTMs to learn the temporal correlations of these interactions. Experimental results on several benchmark datasets demonstrate that by jointly modeling the interactions between all pedestrians and the physical environment, our method outperforms prior state-of-the-art methods and generates more accurate and plausible trajectories for pedestrians.

Chapter 5

Summary and Future Work

In this dissertation, my research during my Ph.D study that covering human pose estimation, temporal action localization and human trajectory prediction has been summarized.

In Chapter 2, we have successfully developed an automatic system to monitor and evaluate worker's efficiency for smart manufacturing workforce management using human body pose estimation and temporal action localization. We first developed a novel semantic adversarial training framework with GAN networks that accurately detect human body joints, estimate its pose, and track its motion. Then, we formulated the automated worker efficiency analysis into a temporal action localization problem in which the action video performed by the worker is matched against a reference video performed by a teacher. We showed our proposed poseGAN achieves the state-of-the-art performance on benchmark dataset and our automated work efficiency analysis is able to achieve accurate action localization with an average IoU score large than 0.9.

In Chapter 3, we have explored the unique characteristics of human trajectories and developed a new approach, reciprocal network learning, for human trajectory prediction.

Two networks, the forward and backward prediction networks, are tightly coupled together, satisfying the reciprocal constraint, which allows them to be jointly learned for accurate and robust human trajectory prediction. Based on this constraint, we borrowed the concept of adversarial attacks of deep neural networks, which iteratively modifies the input of the network to match the given or forced network output, and developed a new method for network testing, called reciprocal attack for matched prediction. It has further improved the prediction accuracy. Extensive experimental results have demonstrated our approach achieves the state-of-art performance on public benchmark datasets.

In Chapter 4, we propose a novel hierarchical framework based on a recurrent sequence-to-sequence architecture to jointly predict the future trajectories of all pedestrians involved in a crowded scenario. We use LSTMs to encode the past trajectory information, global scene layout information and local scene context information. Then graph attention networks are utilized to model the interactions between human-human and human-scene at each time step. Moreover, we design extra LSTMs to learn the temporal correlations of these interactions. Experimental results on several benchmark datasets demonstrate that by jointly modeling the interactions between all pedestrians and the physical environment, our method outperforms prior state-of-the-art methods and generates more accurate and plausible trajectories for pedestrians.

To better under human behaviors, in Chapter 2, we focus on human pose estimation and temporal action localization. These topics are our initial step towards analyzing human actions. There are more steps waiting to be explored and studied, such as human activity recognition, 3D human pose estimation, etc. Also, we cover the topic of human future trajectory prediction in Chapter 3 and 4. Another topic, future activity prediction, also plays a critical role in applications, such as autonomous driving, social robots and smart

surveillance system. Future work will include developing a method that jointly predicting human's future path and their future activities.

Bibliography

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

[2] O'Reilly. A beginner's guide to generative adversarial networks (gans), 2020. [Online; accessed May 27, 2020].

[3] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3838, 2019.

[4] Ben Benfold and Ian Reid. Stable multi-target tracking in real-time surveillance video. In *CVPR 2011*, pages 3457–3464. IEEE, 2011.

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

[6] 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*, pages 2672–2680, 2014.

[7] Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408*, 2016.

[8] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[9] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[10] Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5040–5048, 2016.

[11] Lei Chen, Le Wu, Zhenzhen Hu, and Meng Wang. Quality-aware unpaired image-to-image translation. *IEEE Transactions on Multimedia*, 2019.

[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[13] Yong Guo, Qi Chen, Jian Chen, Qingyao Wu, Qinfeng Shi, and Mingkui Tan. Auto-embedding generative adversarial networks for high resolution image synthesis. *IEEE Transactions on Multimedia*, 2019.

[14] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan

Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[15] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.

[16] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.

[17] Matthias Luber, Johannes A Stork, Gian Diego Tipaldi, and Kai O Arras. People tracking with human motion predictions from social forces. In *2010 IEEE International Conference on Robotics and Automation*, pages 464–469. IEEE, 2010.

[18] Ramin Mehran, Alexis Oyama, and Mubarak Shah. Abnormal crowd behavior detection using social force model. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 935–942. IEEE, 2009.

[19] Sabina Jeschke, Christian Brecher, Tobias Meisen, Denis Özdemir, and Tim Eschert. Industrial internet of things and cyber manufacturing systems. In *Industrial Internet of Things*, pages 3–19. Springer, 2017.

[20] Yuxiang Zhou, Jiankang Deng, and Stefanos Zafeiriou. Improve accurate pose alignment and action localization by dense pose estimation. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 480–484. IEEE, 2018.

[21] Lianrui Fu, Junge Zhang, and Kaiqi Huang. Orgm: Occlusion relational graphical model for human pose estimation. *IEEE Transactions on Image Processing*, 26(2):927–941, 2017.

[22] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[23] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.

[24] Aiden Nibali, Zhen He, Stuart Morgan, and Luke Prendergast. Numerical coordinate regression with convolutional neural networks. *arXiv preprint arXiv:1801.07372*, 2018.

[25] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5734–5743, 2017.

[26] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016.

[27] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *Proceedings of the IEEE international conference on computer vision*, pages 5783–5792, 2017.

[28] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2914–2923, 2017.

[29] Shashank Srikanth, Junaid Ahmed Ansari, Sarthak Sharma, et al. Infer: Intermediate representations for future prediction. *arXiv preprint arXiv:1903.10641*, 2019.

[30] Xuan-Tung Truong and Trung Dung Ngo. Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model. *IEEE Transactions on Automation Science and Engineering*, 14(4):1743–1760, 2017.

[31] 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*, pages 261–268. IEEE, 2009.

[32] Peter Trautman and Andreas Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 797–803. IEEE, 2010.

[33] Kota Yamaguchi, Alexander C Berg, Luis E Ortiz, and Tamara L Berg. Who are you with and where are you going? In *CVPR 2011*, pages 1345–1352. IEEE, 2011.

[34] Lamberto Ballan, Francesco Castaldo, Alexandre Alahi, Francesco Palmieri, and Silvio Savarese. Knowledge transfer for scene-specific motion prediction. In *European Conference on Computer Vision*, pages 697–713. Springer, 2016.

[35] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes

with interacting agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 336–345, 2017.

[36] Kris M Kitani, Brian D Ziebart, and J Andrew. Bagnell, and martial hebert. activity forecasting. In *European Conference on Computer Vision. Springer*, volume 59, page 88, 2012.

[37] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[38] Federico Bartoli, Giuseppe Lisanti, Lamberto Ballan, and Alberto Del Bimbo. Context-aware trajectory prediction. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1941–1946. IEEE, 2018.

[39] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[40] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018.

[41] 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)*, pages 1186–1194. IEEE, 2018.

[42] Manh Huynh and Gita Alaghband. Trajectory prediction by coupling scene-lstm with human movement lstm. In *International Symposium on Visual Computing*, pages 244–259. Springer, 2019.

[43] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2019.

[44] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *Advances in Neural Information Processing Systems*, pages 137–146, 2019.

[45] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[46] Hao Sun, Zhiqun Zhao, and Zhihai He. Reciprocal learning networks for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7416–7425, 2020.

[47] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[48] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Mod-

eling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6272–6281, 2019.

[49] Muwei Jian, Kin-Man Lam, Junyu Dong, and Linlin Shen. Visual-patch-attention-aware saliency detection. *IEEE transactions on cybernetics*, 45(8):1575–1586, 2014.

[50] Muwei Jian, Kin-Man Lam, and Junyu Dong. Facial-feature detection and localization based on a hierarchical scheme. *Information Sciences*, 262:1–14, 2014.

[51] Muwei Jian, Wenyin Zhang, Hui Yu, Chaoran Cui, Xiushan Nie, Huaxiang Zhang, and Yilong Yin. Saliency detection based on directional patches extraction and principal local color contrast. *Journal of Visual Communication and Image Representation*, 57:1–11, 2018.

[52] Diogo C Luvizon, Hedi Tabia, and David Picard. Human pose regression by combining indirect part detection and contextual information. *Computers & Graphics*, 2019.

[53] Yucheng Chen, Yingli Tian, and Mingyi He. Monocular human pose estimation: A survey of deep learning-based methods. *Computer Vision and Image Understanding*, page 102897, 2020.

[54] Xianjie Chen and Alan L Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *Advances in neural information processing systems*, pages 1736–1744, 2014.

[55] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, and Bernt Schiele. Deepcut: Joint subset partition and

labeling for multi person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4929–4937, 2016.

[56] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *TPAMI*, 35(12):2878–2890, 2013.

[57] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5693–5703, 2019.

[58] Wei Tang and Ying Wu. Does learning specific features for related parts help human pose estimation? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1107–1116, 2019.

[59] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[60] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[61] Yu Chen, Chunhua Shen, Xiu-Shen Wei, Lingqiao Liu, and Jian Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation. *arXiv preprint arXiv:1705.00389*, 2017.

[62] Chia-Jung Chou, Jui-Ting Chien, and Hwann-Tzong Chen. Self adversarial training for human pose estimation. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 17–30. IEEE, 2018.

[63] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[64] Svebor Karaman, Lorenzo Seidenari, and Alberto Del Bimbo. Fast saliency based pooling of fisher encoded dense trajectories. In *ECCV THUMOS Workshop*, volume 1, page 7, 2014.

[65] Jan C Van Gemert, Mihir Jain, Ella Gati, Cees GM Snoek, et al. Apt: Action localization proposals from dense trajectories. In *BMVC*, volume 2, page 4, 2015.

[66] Junsong Yuan, Zicheng Liu, and Ying Wu. Discriminative video pattern search for efficient action detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1728–1743, 2011.

[67] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer vision and image understanding*, 115(2):224–241, 2011.

[68] Qinfeng Shi, Li Wang, Li Cheng, and Alex Smola. Discriminative human action segmentation and recognition using semi-markov model. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[69] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[70] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[71] Lichen Wang, Zhengming Ding, and Yun Fu. Learning transferable subspace for human motion segmentation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, page 269, 2018.

[72] Yan Zhang, Siyu Tang, He Sun, and Heiko Neumann. Human motion parsing by hierarchical dynamic clustering. In *BMVC*, page 269, 2018.

[73] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.

[74] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. *ICCV*, 2017.

[75] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[76] Sam Johnson and Mark Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *bmvc*, number 4, page 5. Citeseer, 2010.

[77] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[78] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer, 2016.

[79] Ita Lifshitz, Ethan Fetaya, and Shimon Ullman. Human pose estimation using deep consensus voting. In *European Conference on Computer Vision*, pages 246–260. Springer, 2016.

[80] Vasileios Belagiannis and Andrew Zisserman. Recurrent human pose estimation. *arXiv preprint arXiv:1605.02914*, 2016.

[81] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Learning feature pyramids for human pose estimation. *ICCV*, 2017.

[82] Xiao Chu, Wei Yang, Wanli Ouyang, Cheng Ma, Alan L Yuille, and Xiaogang Wang. Multi-context attention for human pose estimation. *CVPR*, 2017.

[83] Adrian Bulat and Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *European Conference on Computer Vision*, pages 717–732. Springer, 2016.

[84] Zhiqiang Tang, Xi Peng, Shijie Geng, Yizhe Zhu, and Dimitris N Metaxas. Cu-net: coupled u-nets. *arXiv preprint arXiv:1808.06521*, 2018.

[85] Feng Zhang, Xiatian Zhu, and Mao Ye. Fast human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3517–3526, 2019.

[86] Umer Rafi, Bastian Leibe, Juergen Gall, and Ilya Kostrikov. An efficient convolutional network for human pose estimation. In *BMVC*, volume 1, page 2, 2016.

[87] Georgia Gkioxari, Alexander Toshev, and Navdeep Jaitly. Chained predictions using convolutional neural networks. In *European Conference on Computer Vision*, pages 728–743. Springer, 2016.

[88] Sen Yang, Wankou Yang, and Zhen Cui. Pose neural fabrics search. *arXiv preprint arXiv:1909.07068*, 2019.

[89] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[90] Bolei Zhou, Xiaogang Wang, and Xiaoou Tang. Random field topic model for semantic region analysis in crowded scenes from tracklets. In *CVPR 2011*, pages 3441–3448. IEEE, 2011.

[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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6067–6076, 2018.

[92] 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)*, pages 1961–1967. IEEE, 2016.

[93] Kihwan Kim, Dongryeol Lee, and Irfan Essa. Gaussian process regression flow for analysis of motion trajectories. In *2011 International Conference on Computer Vision*, pages 1164–1171. IEEE, 2011.

[94] Weiming Hu, Dan Xie, Zhouyu Fu, Wenrong Zeng, and Steve Maybank. Semantic-based surveillance video retrieval. *IEEE Transactions on image processing*, 16(4):1168–1181, 2007.

[95] Brendan Tran Morris and Mohan Manubhai Trivedi. Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. *IEEE transactions on pattern analysis and machine intelligence*, 33(11):2287–2301, 2011.

[96] Larry R Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5, 2001.

[97] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[98] Pan Pan, Fatih Porikli, and Dan Schonfeld. Recurrent tracking using multifold consistency. In *Proceedings of the Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009.

[99] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *2010 20th International Conference on Pattern Recognition*, pages 2756–2759. IEEE, 2010.

[100] Ishwar K Sethi and Ramesh Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on pattern analysis and machine intelligence*, (1):56–73, 1987.

[101] Hao Wu, Aswin C Sankaranarayanan, and Rama Chellappa. In situ evaluation of tracking algorithms using time reversed chains. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[102] Niccoló Bisagno, Bo Zhang, and Nicola Conci. Group lstm: Group trajectory prediction in crowded scenarios. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[103] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2018.

[104] Junwei Liang, Lu Jiang, Kevin Murphy, Ting Yu, and Alexander Hauptmann. The garden of forking paths: Towards multi-future trajectory prediction. *arXiv preprint arXiv:1912.06445*, 2019.

[105] Charlie Tang and Russ R Salakhutdinov. Multiple futures prediction. In *Advances in Neural Information Processing Systems*, pages 15398–15408, 2019.

[106] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5725–5734, 2019.

[107] Pasquale Coscia, Francesco Castaldo, Francesco AN Palmieri, Alexandre Alahi, Silvio Savarese, and Lamberto Ballan. Long-term path prediction in urban scenarios using circular distributions. *Image and Vision Computing*, 69:81–91, 2018.

[108] Huynh Manh and Gita Alaghband. Scene-lstm: A model for human trajectory prediction. *arXiv preprint arXiv:1808.04018*, 2018.

[109] Amir Sadeghian, Ferdinand Legros, Maxime Voisin, Ricky Vesel, Alexandre Alahi, and Silvio Savarese. Car-net: Clairvoyant attentive recurrent network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 151–167, 2018.

[110] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based recurrent nn: First results. *arXiv preprint arXiv:1412.1602*, 2014.

[111] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988, 2015.

[112] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*, pages 1764–1772, 2014.

[113] Lingyun Yu, Jun Yu, and Qiang Ling. Bltrcnn based 3d articulatory movement prediction: Learning articulatory synchronicity from both text and audio inputs. *IEEE Transactions on Multimedia*, 2018.

[114] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. In *Advances in neural information processing systems*, pages 1889–1897, 2014.

[115] Min Yang, Wei Zhao, Wei Xu, Yabing Feng, Zhou Zhao, Xiaojun Chen, and Kai

Lei. Multitask learning for cross-domain image captioning. *IEEE Transactions on Multimedia*, 21(4):1047–1061, 2018.

[116] Tianmin Shu, Sinisa Todorovic, and Song-Chun Zhu. Cern: confidence-energy recurrent network for group activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5523–5531, 2017.

[117] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[118] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[119] Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11):1875–1886, 2015.

[120] Yichao Yan, Bingbing Ni, Wendong Zhang, Jingwei Xu, and Xiaokang Yang. Structure-constrained motion sequence generation. *IEEE Transactions on Multimedia*, 2018.

[121] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European Conference on Computer Vision*, pages 816–833. Springer, 2016.

[122] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.

[123] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852, 2015.

[124] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *European conference on computer vision*, pages 438–451. Springer, 2010.

[125] Richard W Brislin. Back-translation for cross-cultural research. *Journal of cross-cultural psychology*, 1(3):185–216, 1970.

[126] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pages 820–828, 2016.

[127] Mark Twain. *The jumping frog: in English, then in French, then clawed back into a civilized language once more by patient, unremunerated toil*. Courier Corporation, 1971.

[128] Christopher Zach, Manfred Klopschitz, and Marc Pollefeys. Disambiguating visual relations using loop constraints. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1426–1433. IEEE, 2010.

[129] Guoliang Pang, Xionghui Wang, Jian-Fang Hu, Qing Zhang, and Wei-Shi Zheng. Dbdnet: learning bi-directional dynamics for early action prediction. In *Proceedings*

of the 28th International Joint Conference on Artificial Intelligence, pages 897–903. AAAI Press, 2019.

[130] Qi-Xing Huang and Leonidas Guibas. Consistent shape maps via semidefinite programming. In *Computer Graphics Forum*, volume 32, pages 177–186. Wiley Online Library, 2013.

[131] Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. Learning dense correspondence via 3d-guided cycle consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 117–126, 2016.

[132] Tinghui Zhou, Yong Jae Lee, Stella X Yu, and Alyosha A Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1191–1200, 2015.

[133] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017.

[134] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1983–1992, 2018.

[135] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017.

[136] Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-gan: Unsupervised video retargeting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–135, 2018.

[137] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[138] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[139] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[140] Sachin S Talathi and Aniket Vartak. Improving performance of recurrent neural network with relu nonlinearity. *arXiv preprint arXiv:1511.03771*, 2015.

[141] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International conference on artificial neural networks*, pages 92–101. Springer, 2010.

[142] Stefano Pellegrini, Andreas Ess, and Luc Van Gool. Improving data association by joint modeling of pedestrian trajectories and groupings. In *European conference on computer vision*, pages 452–465. Springer, 2010.

[143] Laura Leal-Taixé, Michele Fenzi, Alina Kuznetsova, Bodo Rosenhahn, and Silvio Savarese. Learning an image-based motion context for multiple people tracking. In

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3542–3549, 2014.

[144] Bolei Zhou, Xiaogang Wang, and Xiaoou Tang. Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2871–2878. IEEE, 2012.

[145] Alexandre Alahi, Vignesh Ramanathan, and Li Fei-Fei. Socially-aware large-scale crowd forecasting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2203–2210, 2014.

[146] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[147] Matteo Lisotto, Pasquale Coscia, and Lamberto Ballan. Social and scene-aware trajectory prediction in crowded spaces. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[148] Chenyang Si, Wentao Chen, Wei Wang, Liang Wang, and Tieniu Tan. An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1227–1236, 2019.

[149] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[150] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14424–14432, 2020.

[151] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[152] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18)*, 2018.

VITA

Hao Sun was born in April 1992, in Anhui, China. He received the Bachelor degrees in Electrical Engineering from Nanjing University of Science and Technology in July 2013. He joined the Department of Electrical Engineering and Computer Science at the University of Missouri in August 2013 as a PhD student. His research interests include image processing, computer vision, machine learning, and deep learning.