M.S. THESIS

Learning based multi-agent trajectory prediction for Automated Driving

자율 주행을 위한 학습 기반의 다중 교통 참여자 경로예측 방법

BY

KIM JUNGHOON

FEBRUARY 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

Learning based multi-agent trajectory prediction for Automated Driving

자율 주행을 위한 학습 기반의 다중 교통 참여자 경로예측 방법

BY

KIM JUNGHOON

FEBRUARY 2019

DEPARTMENT OF ELECTRICAL ENGINEERING AND
COMPUTER SCIENCE
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Learning based multi-agent trajectory prediction for Automated Driving

자율 주행을 위한 학습 기반의 다중 교통 참여자 경로예측 방법

지도교수 서 승 우
이 논문을 공학석사 학위논문으로 제출함

2019년 2월

서울대학교 대학원

전기 컴퓨터 공학부

김 정 훈

김정훈의 공학석사 학위 논문을 인준함

2019년 2월

위 원 장: 최 진 영
부위원장: 서 승 우
위 원: 조 남 익

Abstract

Recent autonomous driving research has shown remarkable and promising results. However, safe, sociable driving in an urban environment still has many challenges ahead. For realizing safe, interactive driving in complex alley scenario which shares a narrow area among traffic participants, It is essential to grasp each other's intention. Even in the same road environment, safe, and sociable driving policy may differ depending on the intention of the traffic participant agents around the ego vehicle. But understanding others intention and predicting their trajectories are complicated because each one basically considers multiple factors; road environment, state of their surrounding traffic participants at the same time which realized as interaction.

In this thesis dissertation, we propose new trajectory prediction algorithm that considers all the information what each of the traffic participants would consider when they make a decision. By combining both each of history trajectories and grid map of surroundings as a latent vector representation, it predicts all the future trajectories of traffic participant agents around ego vehicle at once.

This dissertation suggests two main module that fuses spatial and temporal information effectively. We verify the effectiveness of network structure by testing on the various driving scenario comparing with some network variants through quantitative and qualitative evaluation. Also, the proposed network is verified by applying it to public pedestrian trajectory prediction dataset to verify usability as a generalized methodology and to compare it with other SOTA algorithms.

keywords: Autonomous driving, trajectory prediction, neural process

student number: 2017-21304

Contents

Abstract i

Contents ii

List of Tables iv

List of Figures v

1 INTRODUCTION 1

1.1 Background and Motivation 1

2 Related Work 4

2.1 Contributions of the Dissertation 5

3 Conditional Neural Process 7

3.1 Conditional Neural Process(CNP) Overview 8

3.2 Trajectory Prediction with Scene Information as CNP 10

3.2.1 Formulation 10

3.2.2 Loss and Training Algorithm 13

4 Efficient Network Architecture for Intention Prediction 14

4.1 Network Overview 14

4.2 Trajectory Encoder 16

4.2.1 Spatio-Temporal Representation 17

4.3 Scene Feature Extraction 18
4.3.1 Side Spatial Extraction 18
4.4 Trajectory Decoder 20

5 Experiment **21**
5.1 Driving Environment Dataset 22
5.1.1 Data Acquisition Method 22
5.1.2 Overview 23
5.1.3 Alley Scenario 23
5.1.4 Urban Scenario 27
5.2 Public Pedestrian Dataset 28

6 Conclusion **32**

Abstract (In Korean) **37**

Acknowlegement **39**

List of Tables

4.1 CNN Network architecture 19

5.1 Overall experiment result of the trajectory prediction in driving datasets 24
5.2 Experiment result of the trajectory prediction in alley driving environment 24
5.3 Experiment result of the trajectory prediction in urban driving environment 28
5.4 Experiment result of the trajectory prediction in ETH, HOTEL dataset[21] 30

List of Figures

1.1 Typical example of ”First mover conflict” 2
1.2 Three main stream approach for autonomous driving system[19] . . . 3

4.1 Network Architecture 15
4.2 Trajectory encoder module 16
4.3 (a) Table of labels, (b) Grid map representation 18
4.4 Scene feature extraction module 19
4.5 Trajectory decoder module 20

5.1 Interaction between pedestrians, (left) variance visualization, (upper-right) trajectory visualization onto grid map, (bottom-right) trajectory visualization 25
5.2 Interaction between pedestrian and vehicle, (upper-right) trajectory visualization onto grid map, (bottom-right) trajectory visualization . . . 26
5.3 Interaction between vehicles, (left) trajectory prediction per time step(number) with grid map, (right) trajectory prediction per time step 27
5.4 Various driving scenario in the urban area 29
5.5 Interaction between vehicles, (left) trajectory prediction per time step(number) with grid map, (right) trajectory prediction per time step 31

Chapter 1

INTRODUCTION

1.1 Background and Motivation

Over the past few years, researches for intelligent vehicles and autonomous driving vehicles have shown significant progress. In order for the autonomous vehicle to drive safely, high-level of the core competencies software system are required, which broadly categorized as 4 parts: perception, localization, planning, control.

Deep learning, one of the biggest recent trends in machine learning, have brought revolutionary advances, especially perception in autonomous driving category. But, there are still many challenges to utilize this effective feature learning framework into planning, decision making in highly complex situation.

When human driver make decision while driving, we consider simply call it as ”surroundings”. But, this implies consideration of multiple information, for example, road environment, traffic rule such as traffic light, road markers and traffic sign, status and intention of the traffic participants such as vehicles, pedestrian around the ego-vehicle. Also, in order for autonomous vehicle to be allowed as a substitute of the human driver, it requires to be highly generalized as well, as we don't have a limitation when considering a number of traffic participants, even in the new scene. Since all the traffic participants are interacting at the same time, existing in the same complex

scene, grasping its common understanding of surroundings among traffic participants is difficult but necessary for not to harm the traffic flow.

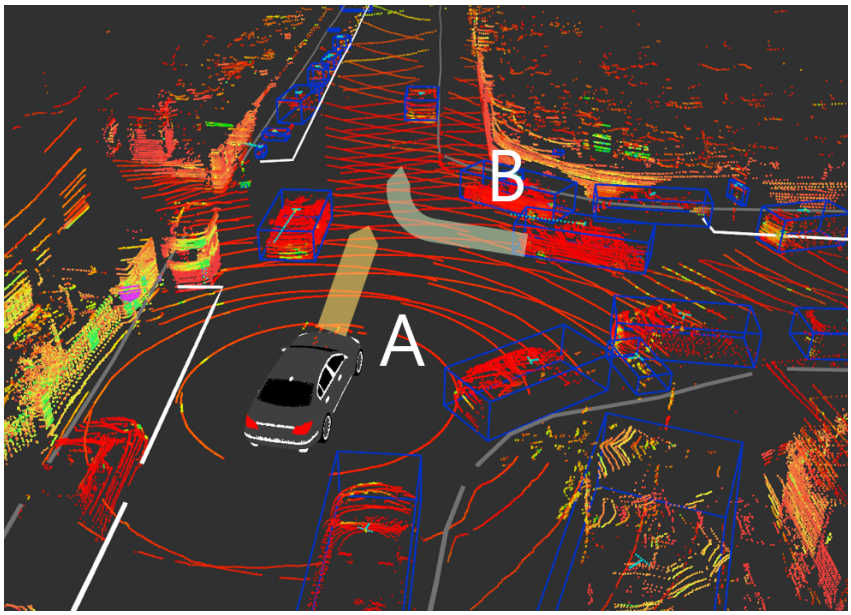


Figure 1.1: Typical example of ”First mover conflict”

Figure 1.1 shows one kind of typical situation requires autonomous vehicle to interact with traffic participants. The vehicle on the right(denoted as B) and ego vehicle(A) on the bottom tries to go to the upper side of the road in the figure. For human drivers, most drivers can decide easily which to proceed first, and the other to follow(yield). They not only have their own understanding on the current situation, but also implicitly share the ”mutual agreement” derived from each of their similar understanding. Under this assumption, people interact smoothly without harming the traffic flow. But also all the people have slightly different interpretation depending on the experience, characteristics, and so on. When people make decision conflicting or disagreement with others due to a different interpretation of understanding, it results in ”first mover conflict”, which harms the traffic flow and also cause an accident. Since interaction happens in many scenarios of driving tasks, conflict also could happen in the same

tasks such as lane changing, merging, navigating in the parking lot, proceeding in the intersection and so on.

Figure 1.2 shows three main approaches for autonomous vehicle systems. Either implicitly or explicitly, all three approaches contain the procedure for trajectory prediction to model the interaction around traffic participants. By forecasting the future behavior of surroundings, autonomous vehicle can drive safely in complex scenarios. Also, it is essential to be implemented for interactive high-level path planning.

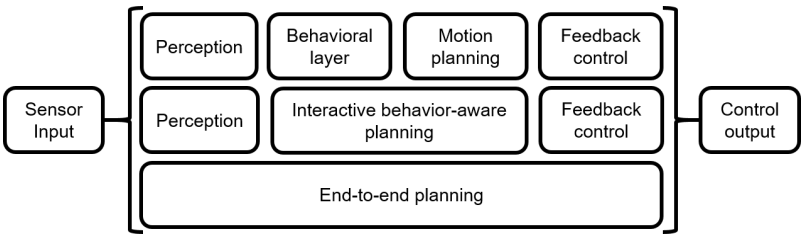


Figure 1.2: Three main stream approach for autonomous driving system[19]

The dissertation is aiming autonomous vehicle to model ”humans’ common understanding of driving scene” that enables high levels of interaction. By utilizing both histories of its traffic participants trajectory information and current scene perception from ego vehicle sensors, it predicts the future trajectory of each traffic participants at once including ego vehicle’s future trajectory, assuming ”what would ego vehicle does if it’s driven by a rational human driver”. As a form of trajectory prediction of surrounding traffic participants, it can provide useful information for safe and sociable driving strategy for autonomous driving.

Chapter 2

Related Work

A research on trajectory prediction, especially for vehicle motion prediction, can be organized as three levels of classification with an increasing degree of abstraction[3]. Physics-based motion models only depends on past behavior of the vehicle. Typically it's limited to only short-term motion prediction and unable to predict unexpected changes caused by the driver intention. Maneuver-based motion models consider intention on behalf of the physics-based motion. Assuming that the trajectory pattern implies intention of the drivers, several probabilistic models such as Gaussian Process(GP)[5], Dynamic Bayesian Network(DBNs)[6] are employed for handling uncertainty and estimating each drivers behavior. Interaction-aware motion models take into account the interaction when predicting vehicles' maneuvers. Even though it gives better interpretation of each vehicles' maneuver, most of the work have done in the only limited situation such as interaction between only two vehicles[7].

As deep learning became one of the most successful methods for machine learning, it motivated most recent work of the trajectory prediction with interaction-aware motion model. It can be categorize as two main approach Reinforcement learning(RL)/ Generative Adversarial Imitation Learning(GAIL) approach, and Long Short-Term Memory(LSTM) approach. For RL methods as an related work for interaction, Lowe, Ryan, et al.[8] suggested multi-agent actor-critic methods(MADDPG) shows the agents

can learn interaction effectively to perform the tasks requiring mixed cooperative-competitive actions. Song, Jiaming, et al.[9] extends GAIL as Multi-agent GAIL, showing that multi-agent can infer interaction from demonstration and learns policy of each role. As an application to autonomous vehicle domain, Bhattacharyya, Raunak P., et al.[10] suggested Parameter-Sharing GAIL(PS-GAIL) generates policies as a result of an interaction of surrounding vehicles simultaneously. However, the proposed PS-GAIL methods require observation features as a relative information from each of the agents. And since GAIL methods directly learn policy from the demonstration, there's a possibility that policy might fail as a failure of generalization.

One another main stream approach utilizes LSTM encoder-decoder network[13, 14, 15]. Most of the research stem from Alahi, Alexandre, et al.[11], which suggested social lstm network for pedestrian trajectory prediction effectively considers the interaction between pedestrians using social pooling layer. But their works also share the problem that LSTM requires longer computation time. In [12], as an improved version of [11], tried to alleviate the computation time using pooling module which interconnects the latent variable of each pedestrian.

2.1 Contributions of the Dissertation

In this dissertation, we propose a trajectory prediction method for all traffic participants around the ego vehicle including pedestrians, by modeling "humans' common understanding of driving scene". The basic framework regards the position of each agent as function of time, and motivated from the conditional neural process[1], represents function as a latent variable. By integrating the past trajectories of all the agents and current scene into latent variables, the network learns its representation, and predicts the future movements of each agent. This framework basically shares all the parameters of the encoder, decoder network of the trajectory, so that it does not have a limitation on the number of agents around the scene. In addition, unlike the LSTM

encoder-decoder framework, it takes the history trajectory as input at a time and predicts the entire future trajectory of each participant at once.

The dissertation is organized as follows: In chapter 3, we briefly go through the basic framework of the conditional neural process[1], including the extension to our framework. In chapter 4, we present a neural network structure that effectively learns its surroundings and each participant's past history as latent variables. In Chapter 5, we conduct an experiment comparing performance depending on the structure of the network. It also compares the performance by applying this to a set of pedestrian trajectory prediction data for comparison with other state-of-the-art pedestrian prediction algorithms.

Chapter 3

Conditional Neural Process

The theorem that the neural networks are universal approximators is one of the theoretical results to justify the use of the neural network as their applications of many fields of the area. Most of the supervised learning problem utilize neural network as function approximation given a set of observations, training it from scratch. Let a number of n dataset as $\{x_i, y_i\}$, $i = 1, \ldots, n$, denoting $x_i \in X$ as inputs, $y_i \in Y$ as outputs of the data. And there is a function mapping from input to output, $f : X \rightarrow Y$. In this perspective, the neural network learns a function $g : X \rightarrow Y$ as an approximation of f , by minimizing the loss. For example in a classification problem, the most common way to train a neural network is formulating the loss as minimizing the negative log-likelihood, parameter that best describes our dataset. To achieve the performance, equivalent as approximate f well in the generic domain, it requires a large training set.

However, the Bayesian approach, by assuming a prior distribution and updating the distribution by observing the data points, it produces posterior distribution efficiently. This approach shows the probabilistic stance that specifying a distribution over function. The typical example is GP, as one kind of stochastic processes defines any finite subset of random variables has a multivariate Gaussian Distribution. These distributions are parameterized by mean function and covariance function. Especially, the covariance function, also known as the kernel function, describes relation between data

points. But Bayesian approach requires high computation or even gets intractable as the dataset grows.

Garnelo, Marta, et al.[1] suggests CNP directly parameterize conditional stochastic process with a neural network. Similar to GP[4], it defines conditional distributions over functions given set of observations. it tries to learn set of data points' relation as a latent variable, and predicts target data points given observed data points. This chapter briefly describes the basic concepts of CNP and formulation onto trajectory prediction problem with additional assumption.

3.1 Conditional Neural Process(CNP) Overview

Let us consider a two set, O, I, as observation and target needs inference respectively. Observation set O contains pairs of inputs and outputs(labels), $O = \{(x_i, y_i)\}_{i=0}^{n-1} \in X \times Y$. Target set contains points of inputs without outputs, $I = \{x_i\}_{i=n}^{n+m-1} \in X$. Also let function f as mapping from inputs to output, $f : X \to Y$, $y_i = f(x_i)$. In stochastic process settings, P defines probability distribution over functions f, $f \sim P$. By definition, P itself defines a joint distribution over the random variables $f(x_i)_{i=0}^{n+m-1}$, including both set of points. CNP[1] learns the conditional distribution $Q_\theta(f(I)|O, I)$ approximating $P(f(I)|O, I)$. This approximation function tries to predict the output values $f(x)$ for every $x \in I$ given O.

Since CNP represents observation data as a fixed shape of latent variable, it has a fixed dimension. This enables CNP to achieve scalable running time complexity of $O(n + m)$, when n is the number of observations and m is the number of predictions. From this formulation, we can obtain functional flexibility and scalability but it no longer holds for the mathematical guarantees as a stochastic process. One typical example of architecture conditioned on observation set via and embedding into fixed shape latent variable can be formed as follows,

$$
\begin{aligned}
r_i &= h_\theta(x_i, y_i) \qquad \forall (x_i, y_i) \in O \\
r &= r_1 \oplus r_2 \oplus \ldots \oplus r_n \\
\phi_i &= g_\theta(x_i, r) \qquad \forall (x_i) \in I
\end{aligned}
\tag{3.1}
$$

$h_\theta(x_i, y_i) : X \times Y \to \mathbb{R}^d$ can be seen as embedding procedure of observation data, and $g_\theta(x_i, r) : X \times \mathbb{R}^d \to \mathbb{R}^e$ approximates conditional distribution from the formulation. Both h, g are neural networks. To make a varying number of observation point i as a fixed shape of latent vectors, it applied commutative operations. If the operations satisfy permutation invariant property, it can be replaced by another operation. For example, Qi, Charles R., et al.[16] utilized max operation to achieve similar functionality. [1] used mean operation.

We can train the network by minimizing the negative conditional log probability. let N as a number of samples we want to make use as an observation when training the network. Since whole observation set is $O = \{(x_i, y_i)\}_{i=0}^{n-1}$, at every iteration, N is sampled from uniform distribution 0 to $n - 1$. We can condition on the subset $O_N = (x_i, y_i)_{i=0}^{N} \in O$ as the first N elements of O. One thing to notice is that at training phase, by setting target set as total dataset $i = 0, ..n - 1$, network efficiently use the data by predicting and propagating the loss from the observation set as well.

$$
L(\theta) = -\mathbb{E}_{f \sim P}\Big[\mathbb{E}_N\big[\log Q_\theta(\{y_i\}_{i=0}^{n-1} | O_N, \{x_i\}_{i=0}^{n-1})\big]\Big] \tag{3.2}
$$

As same as [2], when calculating the gradient of this loss, expectation term is approximated using Monte Carlo estimation by sampling both f, N.

3.2 Trajectory Prediction with Scene Information as CNP

When it comes to the trajectory prediction problem considering the interaction between agents, one should take account not only a trajectory information of each agent but also the surroundings that the each of them are observing. We use CNP based network for approximating predictive distribution for the future motion of all the participants around the ego vehicle over prediction time.

3.2.1 Formulation

The objective of the trajectory prediction is to jointly reason and predict the future trajectories of all the traffic participants involved in the scene. By adopting a formulation of CNP into this problem, the network learn the probability distribution over function mapping future(unseen) time step into position of the each agent. Consider k number of the traffic participants are in the scene, and we have observed trajectories of all the agent in the scene ranging from time step $-b$ to 0 as $T_{past} = \{t|t = -b, \ldots, 0\}$. And there are future time steps we haven't observed yet in the present, ranging from 1 to e as $T_{future} = \{t|t = 1, \ldots, e\}$. Each element of a trajectory, $p_i^j = (x_i^j, y_i^j)$ is a vector in $\mathbb{R}^2$ representing the coordinate of agent i at time step j.

We formulate the objective as to learn the posterior distribution of the multiple agents' future trajectories,

$$P(\mathbf{f}(T_{future})|\mathbf{O}_{traj}, O_{scene}, T_{future}), \tag{3.3}$$

where function of future trajectories $\mathbf{f}(T_{future})$,

$$\mathbf{f}(T_{future}) = \{f_1, \ldots, f_k\}, \tag{3.4}$$

and their past trajectories on the given condition $\mathbf{O}_{traj}$,

$$\mathbf{O}_{traj} = \{O_{traj,1}, \ldots, O_{traj,k}\}, \tag{3.5}$$

and present scene observation O_{scene}.

The past trajectory of agent i, one of element in $\mathbf{O}_{traj}$, is defined as set of tuples,

$$O_{traj,i} = \{(t, p_i^t) | t \in T_{past}\}. \tag{3.6}$$

In present scene observation, we assume all the participants shares same scene observation; from ego vehicle sensing, and map information, since most of the interaction scenarios, causing the behavioral change of ego vehicle, happen around the agents nearby the ego vehicle, cause of each agent action can be enough to be fully interpreted within the ego vehicle perception algorithm due to its wide coverage. Additionally, because we have observed the past position of each agent, scene information would be enough to utilize using the only current frame as well as considering the computational efficiency.

Procedure of each traffic participants moving in the sharing scene can be regarded as a functions, we can define conditional distribution of agent i given observation, $P(f_i(T_{future})|O_{traj}, O_{scene}, T_{future})$, where $f_i : \mathbb{R}^e \rightarrow \mathbb{R}^{2 \cdot e}$ is a function that mapping from time to 2d position of all the agents.

One thing to point out is that the function f_i composing $\mathbf{f}$ shares the parameter among all f. This corresponds to modeling "humans' common understanding of driving scene", meaning that "if the same observations are given to a human, action one might choose would be mostly the same or at least similar".

By combining trajectory information and single scene information from each of observation set, we can embed fused latent vector containing all the information for trajectory prediction, and the latent vector contains the single representation of the

current contextual scene. We use the architecture as follows:

$$
\begin{aligned}
& r^{j}_{\texttt{traj},i} = h_{\theta_1}(j,\, p^{j}_{i}) && \forall (j,\, p^{j}_{i}) \in \mathbf{O}_{\texttt{traj}} \\
& r_{\texttt{traj},i} = r^{-b}_{\texttt{traj},i} \oplus r^{-b+1}_{\texttt{traj},i} \oplus \ldots \oplus r^{0}_{\texttt{traj},i} && \\
& r_{\texttt{scene},i} = c_{\theta_2}(O_{\texttt{scene}}, r_{\texttt{traj},i}) && \\
& r_{\texttt{context},i} = e_{\theta_3}(r_{\texttt{traj},i},\, r_{\texttt{scene},i}) && \\
& \phi^{t}_{i} = g_{\theta_4}(t, r_{\texttt{context},i}) && \forall (t) \in T_{future}
\end{aligned}
$$

Each functions $h_{\theta_1},\, c_{\theta_2},\, e_{\theta_3},\, g_{\theta_4}$ are neural networks parameterized each $\theta_l,\, l = 1,\, \ldots, 4$.

3.2.2 Loss and Training Algorithm

Since we have decided to use only current time scene information for efficient computation, it is not possible to utilize the observation sampling from the CNP loss, which could be helpful to alleviate Monte Carlo estimate variance. However following from [2], the network shaped as the loss function was trainable by setting enough number of minibatch. Letting $O_N = \{(j, \mathbf{p}^j)\}_{j=-b}^{0} \in O_{traj}$ where $\mathbf{p}^j = \{p_0^j, p_1^j, \ldots, p_k^j\}$ as a set of positions of all the agents at certain time step j, approximation of true f paramterized as neural network, $Q_\theta(\{\mathbf{p}^j\}_{j=1}^{e} | O_N, O_{scene}, T_{future})$, loss function defined as follows:

$$L(\theta) = -\mathbb{E}_{f\sim P}\Big[\log Q_\theta\big(\{\mathbf{p}_i\}_{i=1}^{e} | O_N, O_{scene}, \{i\}_{i=1}^{e}\big)\Big] \tag{3.7}$$

Followed by the loss function, algorithms for this network can be described similar to any other neural network.

Algorithm 1 Trajectory prediction network based on CNP training algorithm

1: **procedure** NETWORK TRAINING

Input: Dataset containing pair of scene, trajectory, prediction time information.

Output: Learned Neural network parameter θ

2: $\theta \leftarrow$ Initialize parameters

3: **for** $i = 1$ to n **do**

4: $D^N \leftarrow$ Draw N scene, trajectory data from full dataset

5: $g \leftarrow \nabla_\theta L(\theta; D^N)$

6: $\theta \leftarrow$ Update network(Q_θ) parameter using gradient based methods.

7: **return** Network parameter θ

Chapter 4

Efficient Network Architecture for Intention Prediction

4.1 Network Overview

The main consideration of designing neural network architectures for our frameworks were as follows : 1. Fast computation compare to other LSTM based methods. 2. Efficiently joints both spatial(scene) and temporal(trajectory) information and utilize for trajectory prediction. To combine each trajectory and scene observation information, we suggest two modules, both tries to fuse spatial and temporal information; called Spatio-Temporal Representation Layer(STR), and Side Spatial Extraction Layer(SSE).

STR layer utilizes position information of each subjects' trajectory latent variable. It produce sparse feature map by positioning each of the latent variable as depth. By concatenating with CNN feature map from grid map, network efficiently learns fused features of the both information.

SSE layer takes side-output features from every downsample part of the CNN module. From the spatial feature map, by extracting the depth column where each of agents lies in the grid map, it extracts spatial information which could be lost from the global extraction.

Lastly, trajectory decoder concatenate both latent variables and pass decoding mlp, depending on the number of prediction time points, it returns the same number of mean

and variance of the predicted position of each subject. Fig. 3.1 shows overall process of the network. Each details of the module will be discussed throughout this chapter.

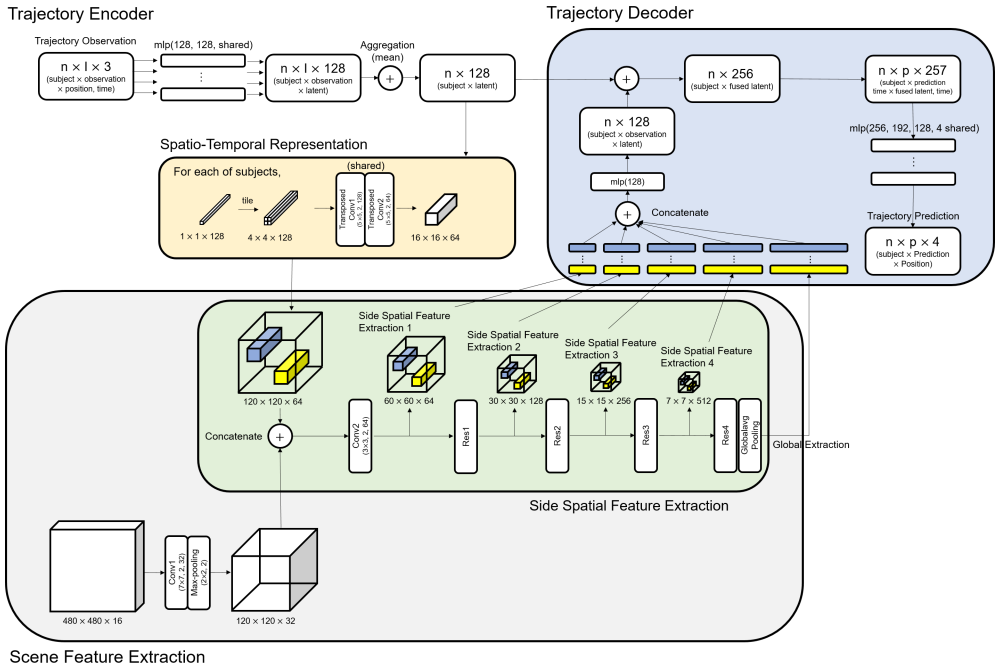


Figure 4.1: Network Architecture

4.2 Trajectory Encoder

Trajectory encoder extract information from the observation trajectory, capture the agents motion dynamics using MLP. It takes time and the history information of the each agents as $\mathbb{R}^{n \times l \times 3}$, where n is the number of agents in the scene, l is the number of observation trajectory for each of 2d position and time. The time corresponding to an each observation is counted sequentially from the lowest negative value from the furthest history to the present the most recent observation, denoted as 0. This mlp shares the parameter for each of the agents, due to its role as extracting dynamic information from the observation data. As same manner, to stabilize its training, we normalized each of the position to the center. After passing 2 layers of mlp with 128 neurons, we aggregate the number of observation dimension, so that it obtains scalability and also permutation invariance property. In this framework, we used mean as an aggregator.

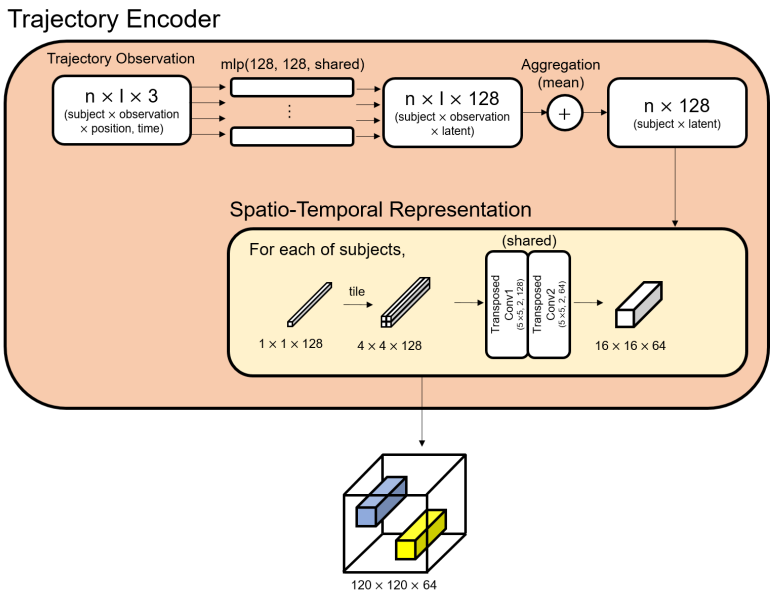


Figure 4.2: Trajectory encoder module

4.2.1 Spatio-Temporal Representation

Since trajectory information of each subject is embedded as a latent variable, Spatio-Temporal Representation module tries to matching the latent variable into a spatial feature map. By matching the latent variable onto a feature map, it feeds into the CNN network and fuses its information effectively. Figure 4.2 shows procedure for this module.

The feature map has a size of 120×120 when latent variable matches on to it. It actually occupies 1m $\times$ 1m of space for each pixel in the real world. If the latent variables are matched by a depth axis corresponding center point of the object, this spatio-temporal feature map contains a large sparsity. To mitigate this problem, latent variable with 1×1 occupancy is spatially inflated to 16×16 by utilizing transposed convolution layer. It also appears to be a reasonable assumption considering the fact that the area in which each agent actually affects each other can be assumed around 16m$\times$16m. If the result of inflated feature map overlaps in the same pixel, each activation value is just summed. As a result, agents' trajectory information can eventually be converted to a feature map of 120×120 size that is also embedding its each position.

4.3 Scene Feature Extraction

Scene Feature Extraction module takes grid map as an input. Grid map is an discrete representation of surroundings around the ego vehicle. Each grid cell contains spatial information of 25cm×25cm. In this environment, we discretize surroundings to a grid map of 480×480 pixels in total by dividing the area surrounding the vehicle into 120 meters by horizontal and vertical. 480×480 pixels of This grid map composed of 16 labels, collection of data from perception and map information. labels are as follows:

Index	Label	Index	Label
0	Background	8	Traffic light
1	Line	9	Traffic signal
2	Center lane	10	Pedestrian
3	Stop line	11	Vehicle
4	Road boundary	12	motorcycle
5	Crossroad	13	Unknown object
6	Speed bump	14	Velocity x direction
7	Crossroad2	15	Velocity y direction

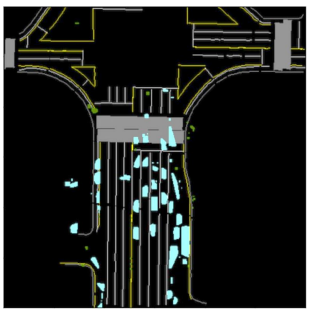


Figure 4.3: (a) Table of labels, (b) Grid map representation

Since all the road marker on the ground contains different semantic meanings, it is divided into separate layers. In addition, the velocity of each axis labels are placed which denote the velocity of occupying object on each grid to include dynamics information of the objects. As with most image classification networks, for the first raw image, the network is structured in a form that passes through a vanilla CNN structure with pooling module.

4.3.1 Side Spatial Extraction

For side spatial extraction module, we utilized Resnet-34 layers[17] as a backbone network, one of state-of-the-art network realized deep network using skip connection. As feature map goes deeper each pixel contains larger area information. Naturally, CNN

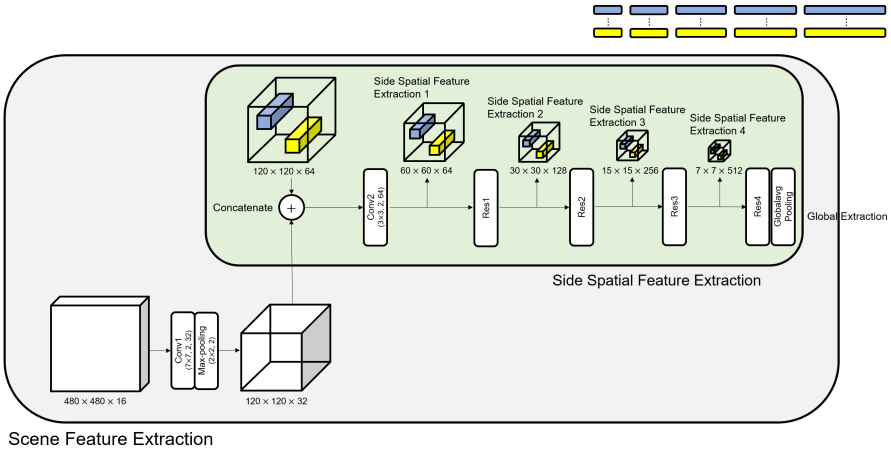


Figure 4.4: Scene feature extraction module

layer name	output size	filter shape	side-output
conv1	240×240×32	7×7, 32, stride 2	
maxpool1	120×120×32	stride 2	
conv_fusion	120×120×96		
conv2	60×60×64	3×3, 64, stride 2	side_out1
res1	30×30×128	resblock(34) × 3	side_out2
res2	15×15×256	resblock(34) × 4	side_out3
res3	8×8×512	resblock(34) × 6	side_out4
res4	8×8×512	resblock(34) × 3	
avgpool	1×1×512		
fc1	1×1×1024		global_out

Table 4.1: CNN Network architecture

module including Resnet constantly shrinking its size and getting deeper the depth, its the local details are lost and only the most significant global feature remains even though its spatial information is embedded into depth part. To alleviate this problem, similar to the opposite direction of Spatio-Temporal Representation module, we reversely extract column features from the feature map for each of subjects corresponding location. In this way, each agents' spatial latent variable contains the detailed local information and the important global information of the whole scene. The same vectors are copied and appended as for the last global feature output. Fig 4.4 depicts the procedure.

4.4 Trajectory Decoder

The trajectory decoder module decodes the fused latent variable information to perform trajectory prediction. For decoder to predict the position of certain time step of the each agent, it expands extra dimension copying latent variables p times, which represents the number of predicting time steps. A vector filled with each of the future time step is added onto the latent variables dimension starting from 1 to p.

From the perspective of autonomous driving system utilizes a trajectory prediction module, probability distribution of each prediction points must be precise. Since Gaussian distribution defined only by mean, variance, representing trajectory prediction result as Gaussian would be useful. So they are constructed in the form of means and variances for x and y coordinates respectively. Fig 4.5 depicts the decoding process.

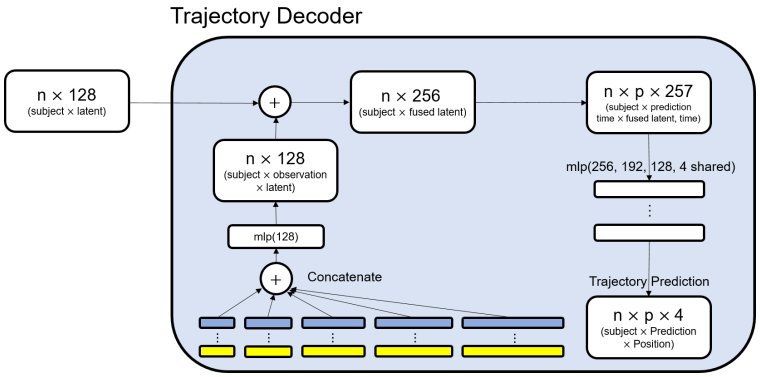


Figure 4.5: Trajectory decoder module

Chapter 5

Experiment

This chapter describes the experimental results of two datasets to evaluate network performance: driving environment dataset, public pedestrian dataset. Driving environment data is divided into two distinctive driving environments, urban roads and alleys. In the case of trajectory prediction in the driving environment, semantic information such as lane, stop sign, etc. and history information about the past trajectories of neighboring agents are should be all considered. In order to verify that the proposed network effectively utilizes the two types of information, we compared the performance by trying each module.

In the recent work of trajectory prediction for highly interactive agents, public pedestrian trajectory prediction datasets were mainly used for validation. These datasets are composed of real human trajectories data with video from a fixed camera. These are relatively less dependent on semantic information than the driving environment datasets, but also the behavior of pedestrians are more diverse even in similar situations. we evaluate our network on these datasets with state-of-the-art baseline methods[11, 12]

For training the network, we used Adam optimizer for total 50 epochs with batch size 64. The initial learning rate was 0.002 with decaying. Each layer used a dropout with a keep ratio of 0.8. We trained the network with TensorFlow and GTX1080 GPU.

We compare the method quantitatively using the commonly known metric, Average Displacement Error (ADE) and the Final Displacement Error (FDE). ADE computes the mean euclidean distance between prediction and ground truth. FDE computes the euclidean distance between last point of prediction and ground truth. ADE and FDE are defined as follows where p_i^t is the ground truth position i, $\hat{p_i^t}$ is the predicted position at time i, T is the prediction horizon, N as total number of agents.

$$\text{ADE} = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T}\left\|\sqrt{p_i^t - \hat{p_i}^t}\right\|_2}{N * T} \tag{5.1}$$

$$\text{FDE} = \frac{\sum_{i=1}^{N}\left\|\sqrt{p_i^T - \hat{p_i}^T}\right\|_2}{N}$$

5.1 Driving Environment Dataset

Since the driving pattern of the human drivers and pedestrians' moving behavior are very different for each urban road and alley scenarios, we divided the driving environment dataset into two categories according to the scenarios. For the following section, We describe the data acquisition procedure to generate the large driving environment dataset and analyze the result acquired from two driving environments qualitatively, and also quantitatively.

5.1.1 Data Acquisition Method

To gather large driving data, by incorporating all the information gathered from autonomous vehicle, such as lidar-camera perception algorithms including tracking and localization, after some filtering, we gathered dataset in semi-supervised way. The dataset collected from the above procedures contain some noise, and only moving objects and semantic information were collected to reduce the noise. We set the overall length up to 30 frames, and divided into two parts: observation and prediction(target).

The length of observation were determined randomly by sampling from uniform distribution around (5,...,25). For example, if the observation length set to be 10 frames, then the prediction length automatically set to be 20 frames, and the scene information is stored at time step 10. Through this process, 60,000 frames of data were acquired, mixed with urban and alley driving, and an additional 10,000 frames were tested to compare performance. The testset consists of 6231 frames of alleys environment and 3869 frames of urban roads. For performance evaluation we fixed prediction time as 2 seconds(20 frames), given 1 seconds of observation(10 frames).

5.1.2 Overview

In the driving scenario, we tested the variants of the network to demonstrate the effectiveness of the proposed module. Network variants are as follows : Network 1 : without using Side Spatial Extraction(SSE) Network 2 : without using Spatio-Temporal Representation(STR), Network 3 : using both SSE and STR module.

Table 5.1 shows the overall result of the experiment on the driving environment dataset combined both urban roads and alleys. Due to inefficient implementation of GPR[4], running time measure is omitted. Lack of scene information makes GPR predicts less accurate compared to network that fully utilize history information as well as scene information. Network 2, which didn't apply STR module gained big advantage on running time because scene, and trajectory latent vector can be calculated parallel. But also it's loss of performance is enormous especially FDE.

5.1.3 Alley Scenario

Compared to driving in urban scenario, driving in alleys is a bit more detailed and careful. Pedestrians move more freely, and vehicles move more carefully because they do not have sufficient safety distance from the surrounding pedestrians and other vehicles on the narrower road. Since the narrow space is shared by many actors, each participant's choice becomes narrow, but the process of decision making is very interactive

Metric	Road Scenario	GPR[4]	Network 1 (STR only)		Network 2 (SSE only)		**Network 3 (STR+SSE)**	
			STR	SSE	STR	SSE	STR	SSE
			O	X	X	O	O	O
ADE	Urban	0.931	0.718		0.622		**0.523**	
	Alley	1.301	1.111		0.979		**0.772**	
	Combined	1.171	0.970		0.851		**0.683**	
FDE	Urban	1.379	1.113		1.143		**0.914**	
	Alley	2.115	1.778		1.994		**1.631**	
	Combined	1.851	1.539		1.685		**1.370**	
runtime(s)		-	∼0.024		∼**0.013**		∼0.030	

Table 5.1: Overall experiment result of the trajectory prediction in driving datasets

and complex. Table 5.2 shows the results of an experiment in an alley. Compared to the overall results in table 5.1, the performance has declined, which is due to the results of implicit decision making between participants' in the alley situations.

Metric	axis	GPR[4]	Network 1		Network 2		**Network 3 (ours)**	
			STR	SSE	STR	SSE	STR	SSE
			O	X	X	O	O	O
ADE	x	1.517	1.246		1.094		**0.929**	
	y	1.084	0.976		0.864		**0.615**	
FDE	x	2.398	1.970		2.115		**1.818**	
	y	1.831	1.587		1.873		**1.445**	

Table 5.2: Experiment result of the trajectory prediction in alley driving environment

As mentioned above, since pedestrians and vehicles share spaces in narrow areas, many interactions occur in alley scenarios. In this section, we analyze the predictions the our network made in three scenarios ; vehicle to vehicle, pedestrian to vehicle, pedestrian to pedestrian.

The figure in 5.1 shows pedestrians passing through narrow streets caused by vehicles. Network models the behavior between pedestrians keeping their own distance. Conventional methods were modeled in the form of potential energy or distance func-

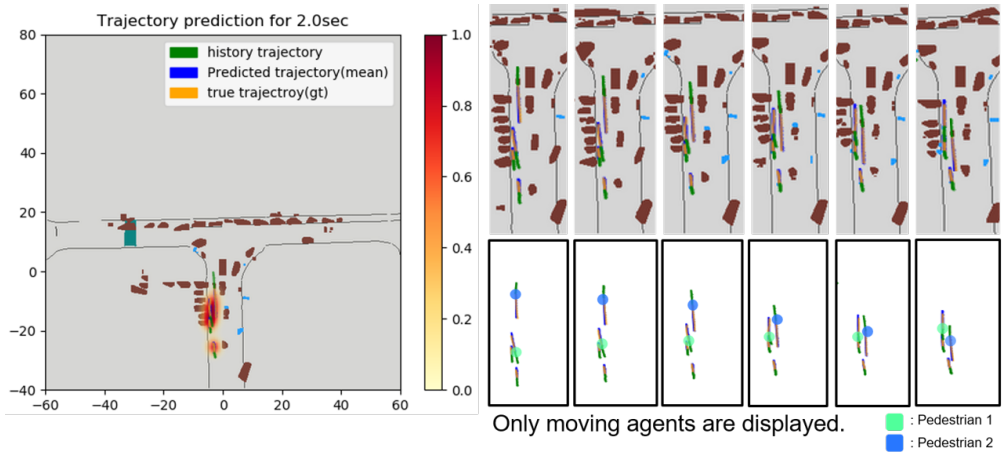


Figure 5.1: Interaction between pedestrians, (left) variance visualization, (upper-right) trajectory visualization onto grid map, (bottom-right) trajectory visualization

tions. However, without knowledge about the social force or any model that describes interaction, result shows that network can model interaction between pedestrian only given a data.

For situations where interaction between pedestrians and vehicles occurs, in figure 5.2, network expected pedestrian agent to yield the car behind(cyan) to move on and pass the street. First few scene where trajectory prediction is done in narrow areas with limited direction is accurate to some extent, but there is a relatively large error when the pedestrian reaches a space that relatively more free to move according to the goal the agent is heading for(last 2 time step in the figure 5.2). Network expected the pedestrian to keep more distance to the vehicle agent, however, because everyone has a slightly different boundary of safety distance, this caused error. This is also related to mode collapse, when multiple decisions are available, the network converges only one mode in the most generic form.

Intersection between vehicles is also the most frequent occurrence of alley driving. In the case of figure 5.3, This is when A(yellow) and B(gray) are both moving straight

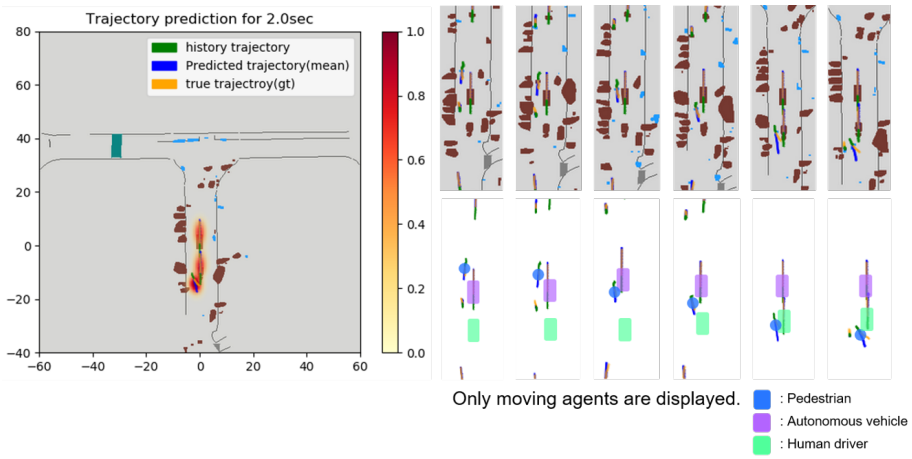


Figure 5.2: Interaction between pedestrian and vehicle, (upper-right) trajectory visualization onto grid map, (bottom-right) trajectory visualization

ahead and B rapidly accelerates. In this situation, it is difficult for A to respond to existing history-based trajectory prediction. However, network contextually recognized the situation as 'first mover conflict', and predict the agent A to stop.

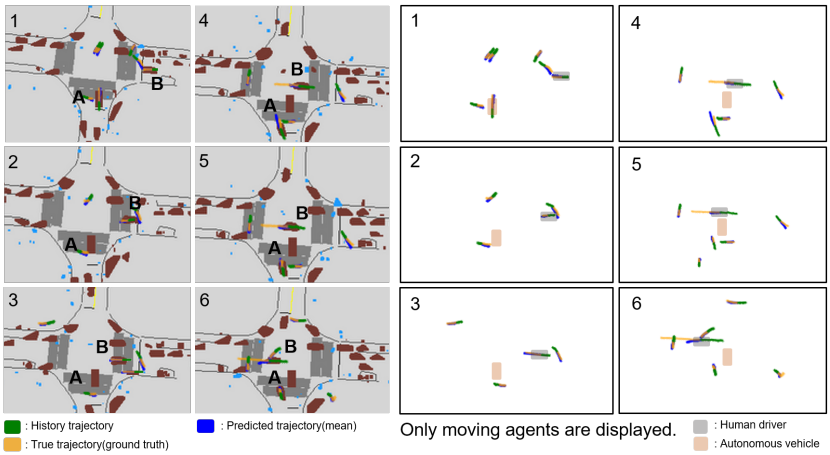


Figure 5.3: Interaction between vehicles, (left) trajectory prediction per time step(number) with grid map, (right) trajectory prediction per time step

5.1.4 Urban Scenario

Driving scene in the urban traffic is mostly following the lane with less pedestrians. In this urban environment, driver interaction is almost unnecessary. However, in order for the network to operate as a generalized risk assessment module with both trajectory prediction and intention understanding, it must be able to accurately predict the movements of each object. Table 5.3 shows quantitative evaluation in urban driving scenarios. Compared to the previous quantitative assessment results, the performance is relatively high, due to the fact that most urban driving situations are achieved most of the time only within simple regulations.

The following figure shows the results of trajectory prediction for different driving situations encountered in typical urban driving situations. Since network learns its rule that vehicle normally keep driving in lane, it predicts trajectory of vehicles well, also same as pedestrian crossing the crossroads. Figure shows that the network has learned that the driver follows the rule that vehicle should be driven between lanes. One thing to point out is that in the case of two vehicles predicted on the left side of the im-

Metric	axis	GPR[4]	Network 1		Network 2		**Network 3 (ours)**	
			STF	SSE	STF	SSE	STF	SSE
			O	X	X	O	O	O
ADE	x	1.298	0.980		0.861		**0.770**	
	y	0.564	0.455		0.383		**0.276**	
FDE	x	2.077	1.648		1.795		**1.397**	
	y	0.682	0.578		0.491		**0.431**	

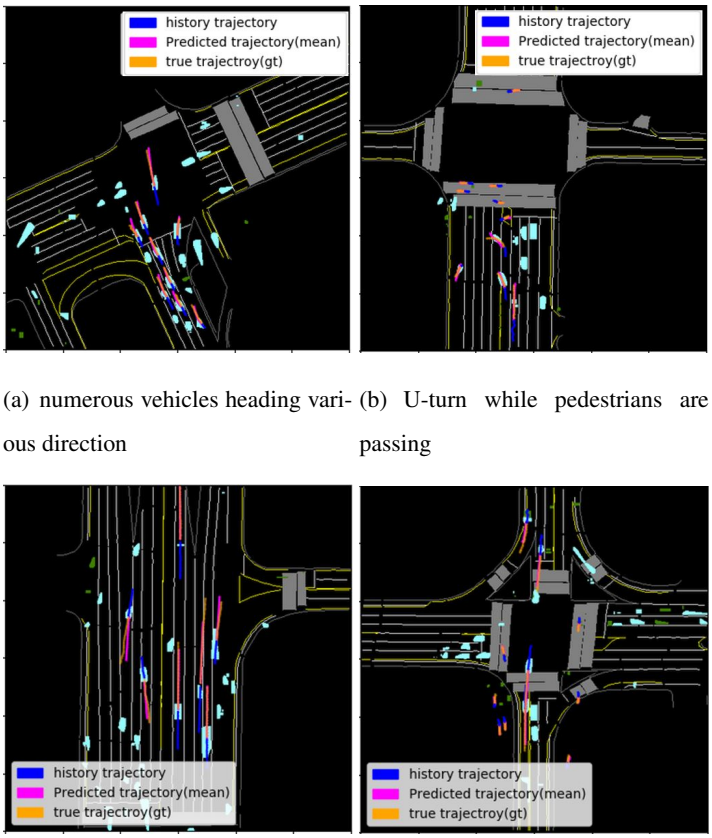
Table 5.3: Experiment result of the trajectory prediction in urban driving environment

age (c), they are actually moving several lanes at once, moving to turn right and left, respectively. In most case, the network has trained or seen the data only one lane to change, so in both cases the actual movement appears to the outside of the lane while the prediction is directed toward the center of the lane. In this regard, the network trained some traffic laws without any other applications. However, some results are not predictable for drivers who does not comply with the law.

This shows the limitations of the current framework. Similar to the Variational Inference, when we try to approximate the complex distributions with already known simple functions, because the model fit into only some part of actual complex distribution. In fact, this is less problematic in alleys where people's interaction is more active because there are only a limited number of cases people can choose from in narrow alleys. However, if we directly learn the true distribution, there is no way to utilize it without a post process such as sampling. Therefore, it is necessary to research for a future work to complement the two opposing views on this part.

5.2 Public Pedestrian Dataset

For the purpose of the relative comparison of the network between SOTA algorithm and verification in general other data, experiments were conducted on the public pedestrian trajectory prediction[21] which contains homography matrix for utilizing camera



(a) numerous vehicles heading various direction

(b) U-turn while pedestrians are passing

(c) multiple lane change

(d) wide intersection

Figure 5.4: Various driving scenario in the urban area

data. We set a random interval of 20% for each data to the test set, training each network to the rest of the training set, and verified the performance.

To easily adapt to the original framework, we simply resized the image to 480×480, and calculated the pixel position of each agents using homography matrix. And instead of using grid map image, We simply used raw image data. For training, similar to the procedure on driving scenario data described at the beginning of chapter 5, we sampled random number from uniform[5,...,15], the total length of data as 30.

For testing, we follow a similar methodology as [11]. The network makes the prediction for steps of 8(3.2 seconds) and 12(4.8 seconds) given observation steps of 8(3.2

seconds). Table 5.4 shows the result of ADE, FDE of the each data set. The predicted results of 8 seconds and 12 seconds are separated by parentheses. $t_{pred} = 8(12)$.

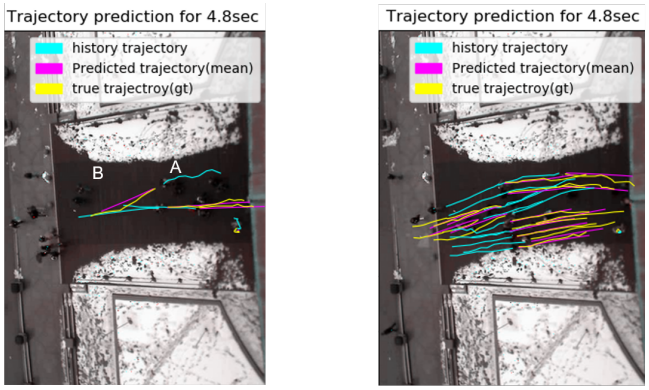
Metric	Dataset	S-LSTM[10]	S-GAN[11]	Ours
ADE	ETH	0.79(1.20)	0.68(**0.94**)	**0.57**(0.103)
	HOTEL	0.57(0.92)	0.51(**0.84**)	**0.48**(0.94)
FDE	ETH	1.61(2.52)	1.24(1.77)	**1.12(1.58)**
	HOTEL	1.16(1.85)	1.01(**1.63**)	**0.94**(1.71)
runtime(s)		1.16(1.97)	0.07(0.11)	**0.031(0.034)**

Table 5.4: Experiment result of the trajectory prediction in ETH, HOTEL dataset[21]

Compared to other networks utilizing the history information only, our network showed better performance by utilizing the scene information presented as an image still with faster inference time. Also, our network can increase the length of prediction time by a small change of computation time, while LSTM-based network shows a significant change in computing times depending on the length of prediction time.

But when it comes to longer prediction(12 seconds) as in the parentheses in the table, S-GAN showed better performance. When the prediction time gets longer, the influence of the latent variables at the starting time step is reduced. In lstm-based networks, latent variables are updated together as prediction proceeds by dealing with sequential output data. In the case of S-GAN, it is expected that the module considers interactions such as social pooling can be repeatedly applied to the updated latent variable as a sequence of prediction(output) and the accuracy is improved in the prediction for longer time.

We consider two common scenarios where many people walk in the opposite direction. Figure 5.5 depicts the scenarios with past trajectory for 3.2 seconds, and future trajectory prediction for 4.8 seconds with the ground truth. Road in front of the door of the building or in roads people walking towards different destination are typical example where similar scenarios frequently happens. When people recognize others coming from in front, people takes avoiding action while walking unless opponent



Predicting evasive behavior of A towards B

Predicting group walking

Figure 5.5: Interaction between vehicles, (left) trajectory prediction per time step(number) with grid map, (right) trajectory prediction per time step

gets too close. It could be change of the walking direction in slightly detouring way, or slow down and waits for others to detour. People choose the action depending on the context or intention of the people around them. In the left side of the figure 5.5 shows the one example, where B and A are walking towards each other. Our model is able to predict evasive behavior of B(purple), slightly different from ground truth(yellow) still one of valid action. As mentioned in the last part of the 5.1.4, our model tries to learn the generalized behavior of the agents. Compared to the driving dataset, behavior of pedestrian diverse with more various ways. Due to its limitation of representation, predicted trajectories tend to be smoothed with low curvature. Another common scenario is when multiple people walking across by others opposite direction. The model predicts the dividing behavior of the groups from the past trajectory information and their final destination from the scene information.

Chapter 6

Conclusion

This dissertation suggests neural network structure for multi-agent trajectory prediction. In order to obtain a high-performance trajectory prediction result, the ability to consider interactions among surrounding agents must be considered together with spatial information in addition to temporal dynamics information.

To achieve this, we proposed the Spatio-Temporal Representation layer to match a latent variable that extracted from trajectory encoder onto feature map extracted from the grid map. It matches each agent's temporal latent variable onto the feature map of the actual position of each agent by concatenating a depth direction. This process contains the problem that activation maps are extremely sparse. To alleviate this problem, before it concatenated with a depth direction, we applied transposed convolutional network to stretch the feature into width, height direction considering the actual distance each agents would affect.

Also, to resolve the known problem that losing local features as CNN network goes deeper, we construct side output. It is opposite process from spatio-temporal representation layer. Every time a downsample of feature map is made, by extracting features from the feature map for each of agents corresponding location. When the flow reaches the last layer of 1×1 FCN, its extracted neurons were concatenated. So that it extracts local and global feature efficiently. In this way, we fused the latent variable for space

and the latent variable for dynamics. Lastly, the decoder predicts the mean and variance of path of all objects around the time step user requested.

By modeling the process of handling data similar to a human driver, we tried to model the ”humans' common understanding of driving scene”. As a result, network achieved good performance where the driving scenario requires a high level of interaction. However, this network possess structural limitation in which it can only predict one option when multiple decisions are available. This is due to having output in the form of a Gaussian distribution to facilitate the practical use of trajectory prediction. Future work would focus on solving multi-modal problem.

Bibliography

[1] Garnelo, Marta, Dan Rosenbaum, Chris J. Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J. Rezende, and S. M. Eslami, "Conditional neural processes," in *arXiv preprint arXiv:1807.01613* (2018).

[2] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes," in *arXiv preprint arXiv:1312.6114* (2013).

[3] Lefèvre, Stéphanie, Dizan Vasquez, and Christian Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *Robomech Journal* 1.1 (2014): 1.

[4] Rasmussen, Carl Edward, and Christopher KI Williams. *"Gaussian Processes for Machine Learning."* Gaussian Processes for Machine Learning, by CE Rasmussen and CKI Williams, MIT Press, ISBN-13 978-0-262-18253-9 (2006).

[5] Tran, Quan, and Jonas Firl. "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression." *Intelligent Vehicles Symposium Proceedings,* 2014 IEEE. IEEE, 2014.

[6] Gindele, Tobias, Sebastian Brechtel, and Rüdiger Dillmann, "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments." *Intelligent Transportation Systems (ITSC),* 2010 13th International IEEE Conference on. IEEE, 2010.

[7] Stéphanie Lefèvre, Christian Laugier, Javier Ibañez-Guzmán, "Intention-Aware Risk Estimation for General Traffic Situations, and Application to Intersection Safety." *[Research Report] RR-8379, INRIA*. 2013.

[8] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., Mordatch, I. "Multi-agent actor-critic for mixed cooperative-competitive environments." *In Advances in Neural Information Processing Systems (pp. 6379-6390).* (2017).

[9] Song, Jiaming, et al. "Multi-agent generative adversarial imitation learning." (2018).

[10] Bhattacharyya, Raunak P., et al. "Multi-Agent Imitation Learning for Driving Simulation." arXiv preprint arXiv:1803.01044 (2018).

[11] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S. "Social lstm: Human trajectory prediction in crowded spaces." *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 961-971).* (2016).

[12] Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A. "Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks." *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (No. CONF).* (2018).

[13] Su, S., Muelling, K., Dolan, J., Palanisamy, P., Mudalige, P. "Learning Vehicle Surrounding-aware Lane-changing Behavior from Observed Trajectories." *In IEEE Intelligent Vehicles Symposium (IV) (pp. 1412-1417). IEEE.* (2018).

[14] Altché, F., De La Fortelle, A. "An LSTM network for highway trajectory prediction." *In Intelligent Transportation Systems (ITSC), IEEE 20th International Conference on (pp. 353-359). IEEE.* (2017).

[15] Deo, Nachiket, and Mohan M. Trivedi. "Convolutional Social Pooling for Vehicle Trajectory Prediction." *arXiv preprint arXiv:1805.06771* (2018).

[16] Qi, Charles R., Hao Su, Kaichun Mo, and Leonidas J. Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation." *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE 1, no. 2* (2017): 4.

[17] He, K., Zhang, X., Ren, S., Sun, J. "Deep residual learning for image recognition." *In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).* (2016).

[18] Held, D., Levinson, J., Thrun, S., Savarese, S. (2014, July). Combining 3D Shape, Color, and Motion for Robust Anytime Tracking. *In Robotics: science and systems.*, June 2014

[19] Schwarting, Wilko, Javier Alonso-Mora, and Daniela Rus. "Planning and decision-making for autonomous vehicles." *Annual Review of Control, Robotics, and Autonomous Systems.* (2018).

[20] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).

[21] Pellegrini, Stefano, Andreas Ess, and Luc Van Gool. "Improving data association by joint modeling of pedestrian trajectories and groupings." *European conference on computer vision. Springer, Berlin, Heidelberg,* (2010).

[22] Bhattacharyya, R. P., Phillips, D. J., Wulfe, B., Morton, J., Kuefler, A., Kochenderfer, M. J. (2018). "Multi-Agent Imitation Learning for Driving Simulation." *arXiv preprint arXiv:1803.01044* (2018).

초 록

자율주행차량이 안전하면서도, 교통 흐름을 방해하지 않는 인간 수준의 interactive한 주행을 실현하기 위해서는 주변 운전자와 보행자를 포함한 교통 참여자들의 의도를 파악하는 것이 필수적이다. 동일한 도로 환경에서도 주변의 교통 참여자들이 어떤 의도를 가지고 이동하고 있는가에 따라서 적합한 주행 전략은 매번 달라진다. 특히 좁은 공간을 많은 에이전트들이 공유하고 있는 골목길 상황에서는 각 에이전트들의 선택은 제한되지만, 각 에이전트들의 의사결정과정은 그들간의 상호작용이 고려되어 매우 복잡하다. 이런 상황에 맞는 경로예측은 과거의 궤적, 현재 인식하고 있는 도로 환경, 주변 교통 참여자의 상태 등을 고려하여 수행되어야 한다. 또한 일반화된 대부분의 환경에서 수행되려면, 빠른 러닝 타임과 동시에 고려하는 교통 참여자의 숫자에 대한 제약이 적어야 한다.

이 논문에서는 각 교통 참여자들의 과거 궤적과 현재 위치 상황, 도로 상황을 동시에 고려하여 모든 참여자들의 상호작용이 고려된 경로예측 방법을 제안한다. 각 물체들의 과거 궤적과 동시에 자율주행 차량이 인식하고 있는 도로와 주변 물체를 입력으로하고, 이 두개의 임베딩 결과를 혼합하여 모든 물체들에 대한 경로 예측을 동시에 수행한다. 이 과정에서 네트워크 구조 내부에서 각각 물체들의 임베딩 결과를 위치 정보에 매칭시킴으로써 효과적으로 주변 상황과 과거의 궤적을 동시에 고려하는 경로 예측을 학습할 수 있는 구조를 제안한다. 몇가지 네트워크 구조에 따른 성능 비교와 함께, 다양한 주행 환경에서 정량적인 평가와 정성적인 평가로 유효성을 입증한다. 또한 보행자 경로예측 데이터에 테스트를 진행함으로써 타 알고리즘과의 성능을 비교한다.

주요어: 자율 주행, 경로 예측, 뉴럴 프로세스
학번: 2017-21304

ACKNOWLEGEMENT