TrafficPredict: Trajectory Prediction for Heterogeneous Traffic-Agents

Yuexin Ma[1,2], Xinge Zhu[3], Sibo Zhang[1], Ruigang Yang[1], Wenping Wang[2], Dinesh Manocha[4]
Baidu Research, Baidu Inc.[1], The University of Hong Kong[2],
The Chinese University of Hong Kong[3], University of Maryland at College Park[4]
yxma@cs.hku.hk, zhuxinge123@gmail.com, sibozhang@baidu.com,
yangruigang@baidu.com, wenping@cs.hku.hk, dm@cs.umd.edu
http://gamma.cs.unc.edu/TPredict/TrafficPredict.html

Abstract

To safely and efficiently navigate in complex urban traffic, autonomous vehicles must make responsible predictions in relation to surrounding traffic-agents (vehicles, bicycles, pedestrians, etc.). A challenging and critical task is to explore the movement patterns of different traffic-agents and predict their future trajectories accurately to help the autonomous vehicle make reasonable navigation decision. To solve this problem, we propose a long short-term memory-based (LSTM-based) realtime traffic prediction algorithm, TrafficPredict. Our approach uses an instance layer to learn instances' movements and interactions and has a category layer to learn the similarities of instances belonging to the same type to refine the prediction. In order to evaluate its performance, we collected trajectory datasets in a large city consisting of varying conditions and traffic densities. The dataset includes many challenging scenarios where vehicles, bicycles, and pedestrians move among one another. We evaluate the performance of TrafficPredict on our new dataset and highlight its higher accuracy for trajectory prediction by comparing with prior prediction methods.

Introduction

Autonomous driving is a significant and difficult task that has the potential to impact peoples day-to-day lives. The goal is to make a vehicle perceive the environment and safely and efficiently navigate any traffic situation without human intervention. Some of the challenges arise in dense urban environments, where the traffic consists of different kinds of *traffic-agents*, including cars, bicycles, buses, pedestrians, etc.. These traffic-agents have different shapes, dynamics, and varying behaviors and can be regarded as an instance of a heterogeneous multi-agent system. To guarantee the safety of autonomous driving, the system should be able to analyze the motion patterns of other traffic-agents and predict their future trajectories so that the autonomous vehicle can make appropriate navigation decisions.

Driving in an urban environment is much more challenging than driving on a highway. Urban traffic is riddled with more uncertainties, complex road conditions, and diverse traffic-agents, especially on some cross-roads. Different traffic-agents have different motion patterns. At the same time, traffic-agents behaviors are deeply affected by other traffic-agents. It is necessary to consider the interaction between the agent to improve the accuracy of trajectory prediction.

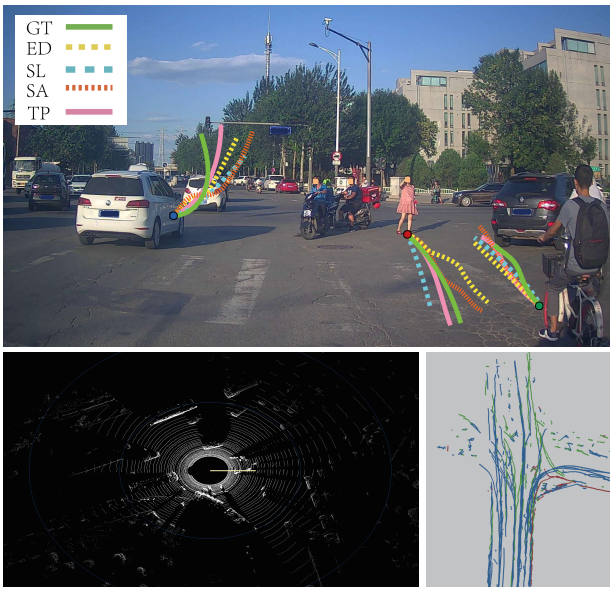

Figure 1: **Heterogeneous urban traffic scenario:** We demonstrate the improved trajectory prediction accuracy of our method over prior approaches (top). The green solid lines denote ground truth trajectories (GT), pink solid lines are for our method (TP) and dashed lines are the predicted trajectories for other methods (ED, SL, SA). We observe 20% improvement in accuracy using TP. Traffic corresponding point cloud captured by LiDAR of the acquisition car is shown on the left bottom. Original trajectories of traffic-agents in the scenario are shown on the right bottom: blue for cars, green for bicycles, and red for pedestrians.

The problem of predicting trajectories for moving agents has been studied extensively. Some traditional algorithms are based on motion models like kinematic and dynamic models (Toledo-Moreo and Zamora-Izquierdo 2009), Bayesian filters (Kalman 1960), Gaussian Processes (Rasmussen and Williams 2006), etc. These methods do not take into account interactions between the traffic-agents and the environment, making it difficult to analyze complicated sce-

narios or perform long-term predictions. With the success of LSTM networks in modeling non-linear temporal dependencies in sequence learning and generation tasks, more and more works have been using these networks to predict trajectories of human crowds(Alahi *et al.* 2016) and vehicles trajectories (Lee *et al.* 2017). The common limitation of these works is the focus on predicting one type of group (only pedestrians or cars, for example). These methods may not work in heterogeneous traffic, where different vehicles and pedestrians coexist and interact with each other(Chandra *et al.* 2018b).

Main Results: For the task of trajectory prediction in heterogeneous traffic, we propose a novel LSTM-based algorithm, TrafficPredict. Given a sequence of trajectory data, we construct a *4D Graph*, where two dimensions are for instances and their interactions, one dimension is for time series, and one dimension is for high-level categorization. In this graph, all the valid instances and categories of traffic-agents are denoted as nodes, and all the relationships in spatial and temporal space is represented as edges. Sequential movement information and interaction information are stored and transferred by these nodes and edges. Our LSTM network architecture is constructed on the *4D Graph*, which can be divided into two main layers: one is the instance layer and the other is the category layer. The former is designed to capture dynamic properties and and interactions between the traffic-agents at a micro level. The latter aims to conclude the behavior similarities of instances of the same category using a macroscopic view and guide the prediction for instances in turn. We also use a self attention mechanism in the category layer to capture the historical movement patterns and highlight the category difference. Our method is the first to integrate the trajectory prediction for different kinds of traffic-agents in one unified framework.

To better expedite research progress on prediction and navigation in challenging scenarios for autonomous driving, we provide a new trajectory dataset for complex urban traffic with heterogeneous traffic-agents during rush hours. Scenario and data sample of our dataset is shown in Fig. 1. In practice, TrafficPredict takes about a fraction of a second on a single CPU core and exhibits 20% accuracy improvement over prior prediction schemes. The novel components of our work include:

- Propose a new approach for trajectory prediction in heterogeneous traffic.
- Collect a new trajectory dataset in urban traffic with much interaction between different categories of traffic-agents.
- Our method has smaller prediction error compared with other state-of-art approaches.

The rest of the paper is organized as follows. We give a brief overview of related prior work in Section 2. In Section 3, we define the problem and give details of our prediction algorithm. We introduce our new traffic dataset and show the performance of our methods in Section 4.

Related Work

Classical methods for trajectory prediction

The problem of trajectory prediction or path prediction has been extensively studied. There are many classical approaches, including Bayesian networks (Lefèvre *et al.* 2011), Monte Carlo Simulation (Danielsson *et al.* 2007), Hidden Markov Models (HMM) (Firl *et al.* 2012), Kalman Filters (Kalman 1960), linear and non-linear Gaussian Process regression models (Rasmussen and Williams 2006), etc. These methods focus on analyzing the inherent regularities of objects themselves based on their previous movements. They can be used in simple traffic scenarios in which there are few interactions among cars, but these methods may not work well when different kinds of vehicles and pedestrians appear at the same time.

Behavior modeling and interactions

There is considerable work on human behavior and interactions. The Social Force model (Helbing and Molnar 1995) presents a pedestrian motion model with attractive and repulsive forces, which has been extended by (Yamaguchi *et al.* 2011). Some similar methods have also been proposed that use continuum dynamics (Treuille *et al.* 2006), Gaussian processes (Wang *et al.* 2008), etc. Bera et al. (2016; 2017) combine an Ensemble Kalman Filter and human motion model to predict the trajectories for crowds. These methods are useful for analyzing motions of pedestrians in different scenarios, such as shopping malls, squares, and pedestrian streets. There are also some approaches to classify group emotions or identify driver behaviors (Bera *et al.* 2018; Cheung *et al.* 2018). To extend these methods to general traffic scenarios, (Ma *et al.* 2018) predicts the trajectories of multiple traffic-agents by considering kinematic and dynamic constraints. However, this model assumes perfect sensing and shape and dynamics information for all of the traffic agents.

RNN networks for sequence prediction

In recent years, the concept of the deep neural network (DNN) has received a huge amount of attention due to its good performance in many areas (Goodfellow *et al.* 2016). Recurrent neural network (RNN) is one of the DNN architectures and is widely used for sequence generation in many domains, including speech recognition (Graves and Jaitly 2014), machine translation (Chung *et al.* 2015), and image captioning (Vinyals *et al.* 2015). Many methods based on long short-term Memory (LSTM), one variant of RNN, have been proposed for maneuver classification (Khosroshahi 2017) and trajectory prediction (Altché and Fortelle 2017). Some methods (Kim *et al.* 2017; Park *et al.* 2018; Lee *et al.* 2017) produce the probabilistic information about the future locations of vehicles over an occupancy grid map or samples by making use of an encoder-decoder structure. However, these sampling-based methods suffer from inherent inaccuracies due to discretization limits. Another method (Deo and Trivedi 2018) presents a model that outputs the multi-modal distribution and then generates trajectories. Nevertheless, most of these methods require clear road lanes

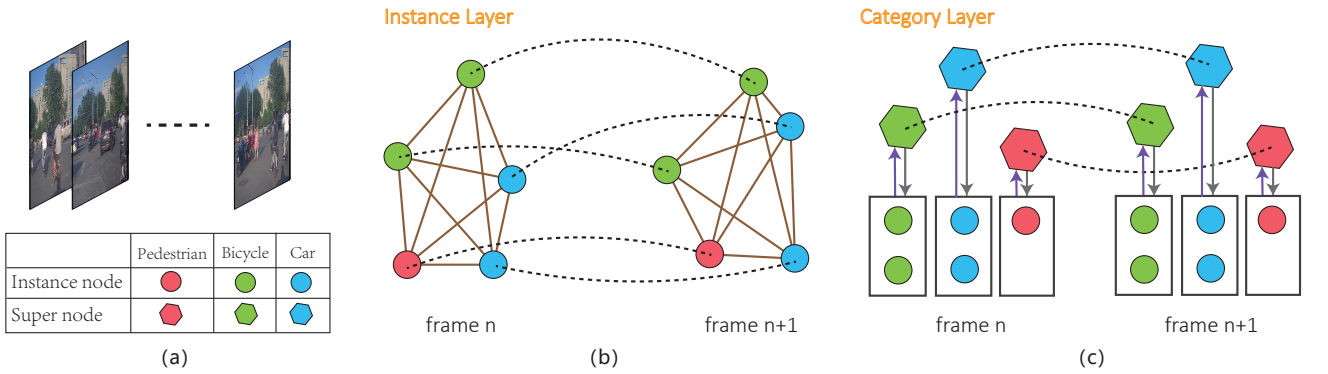

Figure 2: Our *4D Graph* for a traffic sequence. (a) Icons for instances and categories are shown on the left table. (b) The instance layer of the *4D Graph* with spatial edges as solid lines and temporal edges as dashed lines. (c) The category layer with temporal edges of super nodes drawn by dashed lines.

and simple driving scenarios without other types of traffic-agents passing through. Based on images, (Chandra *et al.* 2018a) models the interactions between different traffic-agents by a LSTM-CNN hybrid network for trajectory prediction. Taking into account the human-human interactions, some approaches (Alahi *et al.* 2016; Gupta *et al.* 2018; Vemula *et al.* 2017) use LSTM for predicting trajectories of pedestrians in a crowd and they show good performance on public crowd datasets. However, these methods are also limited in terms of trajectory prediction in complex traffic scenarios where the interactions are among not only pedestrians but also heterogeneous traffic-agents.

Traffic datasets

There are several datasets related to traffic scenes. Cityscapes (Cordts *et al.* 2016) contains 2D semantic, instance-wise, dense pixel annotations for 30 classes. ApolloScape (Huang *et al.* 2018) is a large-scale comprehensive dataset of street views that contains higher scene complexities, 2D/3D annotations and pose information, lane markings and video frames. However these two dataset do not provide trajectories information. The Simulation (NGSIM) dataset (Administration 2005) has trajectory data for cars, but the scene is limited to highways with similar simple road conditions. KITTI (Geiger *et al.* 2013) is a dataset for different computer vision tasks such as stereo, optical ow, 2D/3D object detection, and tracking. However, the total time of the dataset with tracklets is about 22 minutes. In addition, there are few intersection between vehicles, pedestrians and cyclists in KITTI, which makes it insufficient for exploring the motion patterns of traffic-agents in challenging traffic conditions. There are some pedestrian trajectory datasets like ETH (Pellegrini *et al.* 2009), UCY (Lerner *et al.* 2007), etc., but such datasets only focus on human crowds without any vehicles.

TrafficPredict

In this section, we present our novel algorithm to predict the trajectories of different traffic-agents.

Problem Definition

We assume each scene is preprocessed to get the categories and spatial coordinates of traffic-agents. At any time t, the feature of the ith traffic-agent A_i^t can be denoted as $f_i^t = (x_i^t, y_i^t, c_i^t)$, where the first two items are coordinates in the x-axis and y-axis respectively, and the last item is the category of the traffic-agent. In our dataset, we currently take into account three types of traffic-agents, $c_i \in \{1, 2, 3\}$, where 1 stands for pedestrians, 2 represents bicycles and 3 denotes cars. Our approach can be easily extended to take into account more agent types. Our task is to observe features of all the traffic-agents in the time interval $[1 : T_{obs}]$ and then predict their discrete positions at $[T_{obs}+1 : T_{pred}]$.

4D Graph Generation

In urban traffic scenarios where various traffic-agents are interacting with others, each instance has its own state in relation to the interaction with others at any time and they also have continuous information in time series. Considering traffic-agents as instance nodes and relationships as edges, we can construct a graph in the instance layer, shown in Fig.2 (b). The edge between two instance nodes in one frame is called spatial edge (Jain *et al.* 2016; Vemula *et al.* 2017), which can transfer the interaction information between two traffic-agents in spatial space. The edge between the same instance in adjacent frames is the temporal edge, which is able to pass the historic information frame by frame in temporal space. The feature of the spatial edge (A_i^t, A_j^t) for A_i^t can be computed as $f_{ij}^t = (x_{ij}^t, y_{ij}^t, c_{ij}^t)$, where $x_{ij}^t = x_j^t - x_i^t$, $y_{ij}^t = y_j^t - y_i^t$ stands for the relative position from A_j^t to A_i^t, c_{ij}^t is an unique encoding for (A_i^t, A_j^t). When traffic-agent A_j considers the spatial edge, the spatial edge is represented as (A_j^t, A_i^t). The feature of the temporal edge (A_i^t, A_i^{t+1}) is computed in the same way.

It is normally observed that the same kind of traffic-agents have similar behavior characteristics. For example, the pedestrians have not only similar velocities but also similar reactions to other nearby traffic-agents. These similarities will be directly reflected in their trajectories. We construct a super node $C_u^t, u \in \{1, 2, 3\}$ for each kind of traffic-agent

to learn the similarities of their trajectories and then utilize that super node to refine the prediction for instances. Fig.2 (c) shows the graph in the category layer. All instances of the same type are integrated into one group and each group has an edge oriented toward the corresponding super node. After summarizing the motion similarities, the super node passes the guidance through an oriented edge to the group of instances. There are also temporal edges between the same super node in sequential frames. This category layer is specially designed for heterogeneous traffic and can make full use of the data to extract valuable information to improve the prediction results. This layer is very flexible and can be easily degenerated to situations when several categories disappear in some frames.

Finally, we get the *4D Graph* for a traffic sequence with two dimensions for traffic-agents and their interactions, one dimension for time series, and one dimension for high-level categories. By this *4D Graph*, we construct an information network for the entire traffic. All the information can be delivered and utilized through the nodes and edges of the graph.

Model Architecture

Our TrafficPredict algorithm is based on the *4D Graph*, which consists of two main layers: the instance layer and the category layer. Details are given below.

Instance Layer The instance layer aims to capture the movement pattern of instances in traffic. For each instance node A_i, we have an LSTM, represented as L_i. Because different kinds of traffic-agents have different dynamic properties and motion rules, only instances of the same type share the same parameters. There are three types of traffic-agents in our dataset: vehicles, bicycles, and pedestrians. Therefore, we have three different LSTMs for instance nodes. We also distribute LSTM L_{ij} for each edge (A_i, A_j) of the graph. All the spatial edges share the same parameters and all the temporal edges are classified into three types according to corresponding node type.

For edge LSTM L_{ij} at any time t, we embed the feature f_{ij}^t into a fixed vector e_{ij}^t, which is used as the input to LSTM:

$$e_{ij}^t = \phi(f_{ij}^t; W_{spa}^e), \tag{1}$$

$$h_{ij}^t = LSTM(h_{ij}^{t-1}; e_{ij}^t; W_{spa}^r), \tag{2}$$

where $\phi(\cdot)$ is an embedding function, h_{ij}^t is the hidden state also the output of LSTM L_{ij}, and W_{spa}^e are the embedding weights, and W_{spa}^r are LSTM cell weights, which contains the movement pattern of the instance itself. LSTMs for temporal edges L_{ii} are defined in a similar way with parameters W_{tem}^e and W_{tem}^r.

Each instance node may connect with several other instance nodes via spatial edges. However, each of the other instances has different impacts on the node's behavior. We use a soft attention mechanism (Vemula *et al.* 2017) to distribute various weights for all the spatial edges of one instance node

$$w(h_{ij}^t) = softmax(\frac{m}{\sqrt{d_e}} Dot(W_i h_{ii}^t, W_{ij} h_{ij}^t)), \tag{3}$$

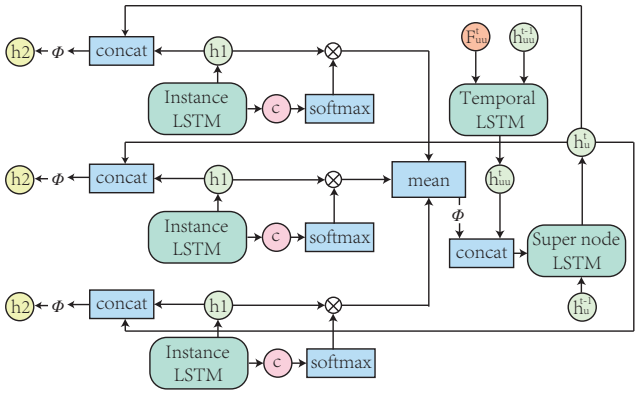

Figure 3: Architecture of the network for one super node in the category layer.

where W_i and W_{ij} are embedding weights, $Dot(\cdot)$ is the dot product, and $\frac{m}{\sqrt{d_e}}$ is a scaling factor (Vaswani *et al.* 2017). The final weights are ratios of $w(h_{ij}^t)$ to the sum. The output vector H_i^t is computed as a weighted sum of h_{ij}^t. H_i^t stands for the influence exhibited on an instance's trajectory by surrounding traffic-agents and h_{ii}^t denotes the information passing by temporal edges. We concatenate them and embed the result into a fixed vector a_i^t. The node features f_i^t and a_i^t can finally concatenate with each other to feed the instance LSTM L_i.

$$e_i^t = \phi(f_i^t; W_{ins}^e), \tag{4}$$

$$a_i^t = \phi(concat(h_{ii}^t, H_i^t); W_{ins}^a), \tag{5}$$

$$h1_i^t = LSTM(h2_i^{t-1}; concat(e_i^t, a_i^t); W_{ins}^r), \tag{6}$$

where W_{ins}^e and W_{ins}^a are the embedding weights, W_{ins}^r is the LSTM cell weight for the instance node, $h1_i^t$ is the first hidden state of the instance LSTM. $h2_i^{t-1}$ is the final hidden state of the instance LSTM in the last frame, which will be described in next section.

Category Layer Usually traffic-agents of the same category have similar dynamic properties, including the speed, acceleration, steering, etc., and similar reactions to other kinds of traffic-agents or the whole environment. If we can learn the movement patterns from the same category of instances, we can better predict trajectories for the entire instances. The category layer is based on the graph in Fig. 2(c). There are four important components: the super node for a specified category, the directed edge from a group of instances to the super node, the directed edge from the super node to instances, and the temporal edges for super nodes.

Taking one super node with three instances as the example, the architecture in the category layer is shown in Fig. 3. Assume there are n instances belonging to the same category in the current frame. We have already gotten the hidden state $h1$ and the cell state c from the instance LSTM, which are the input for the category layer. Because the cell state c contains the historical trajectory information of the instance, self-attention mechanism (Vaswani *et al.* 2017) is used on c by softmax operation to explore the pattern of the

(a) (b)

Figure 4: Scenarios used for data collection: (a) Normal lanes with various traffic-agents. (b) Crossroads with different traffic-agents.

internal sequence. At time t, the movement feature d for the mth instance in the category is captured as follows.

$$d_m^t = h1_m^t \bigotimes softmax(c_m^t), \tag{7}$$

Then, we obtain the feature F_u^t for the corresponding super node C_u^t by computing the average of all the instances' movement feature of the category.

$$F_u^t = \frac{1}{n}\sum_{m=1}^{n} d_m^t, \tag{8}$$

F_u^t captures valid trajectory information from instances and learn the internal movement law of the category. Equation (7)-(8) show the process of transferring information on the directed edge from a group of instances to the super node.

The feature F_{uu}^t of the temporal edge of super node is computed by $F_u^t - F_u^{t-1}$. Take W_{st}^e as embedding weights and W_{st}^r as the LSTM cell weights. The LSTM of the temporal edge between the same super node in adjacent frames can be computed as follows.

$$e_{uu}^t = \phi(F_{uu}^t; W_{st}^e), \tag{9}$$

$$h_{uu}^t = LSTM(h_{uu}^{t-1}; e_{uu}^t; W_{st}^r), \tag{10}$$

Next, we integrate the information from the group of instances and the temporal edge as the input to the super node. We embed the feature F_u^t into fixed-length vectors and then concatenate with h_{uu}^t together. The hidden state h_u^t of super node can be gotten by follows.

$$e_u^t = \phi(F_u^t; W_{sup}^e), \tag{11}$$

$$h_u^t = LSTM(h_u^{t-1}; concat(e_u^t; h_{uu}^t); W_{sup}^k), \tag{12}$$

Finally, we describe the process of transferring guidance on the directed edge from the super node to instances. For the mth instance in the group, the hidden state of the super node is concatenated with the first hidden state $h1_m^t$ and then embedded into a vector with the same length of $h1_m^t$. The second hidden state $h2_m^t$ is the final output of the instance node.

$$h2_m^t = \phi(concat((h1_m^t; h_u^t); W_s^r)), \tag{13}$$

where W_s^r is the embedding weights. By the network of the category layer, we use the similarity inside the same type of instances to refine the prediction of trajectories for instances.

Table 1: The acquisition time, total frames, total instances (count ID), average instances per frame, acquisition devices of NGSIM, KITTI (with tracklets) and our dataset.

Count		NGSIM	KITTI	Our Dataset
duration (min)		45	22	**155**
frames ($\times 10^3$)		11.2	13.1	**93.0**
total ($\times 10^3$)	pedestrian	0	0.09	**16.2**
	bicycle	0	0.04	**5.5**
	vehicle	2.91	0.93	**60.1**
average (1/f)	pedestrian	0	1.3	**1.6**
	bicycle	0	0.24	**1.9**
	vehicle	**845**	3.4	12.9
device	camera	yes	yes	yes
	lidar	no	yes	yes
	GPS	no	yes	yes

Position estimation We assume the position of the traffic-agent in next frame meets a bivariate Gaussian distribution as (Alahi *et al.* 2016) with parameters including the mean $\mu_i^t = (\mu_x, \mu_y)_i^t$, standard deviation $\sigma_i^t = (\sigma_x, \sigma_y)_i^t$ and correlation coefficient ρ_i^t. The corresponding position can be represented as follows.

$$(x_i^t, y_i^t) \sim \mathcal{N}(\mu_i^t, \sigma_i^t, \rho_i^t), \tag{14}$$

The second hidden state of traffic-agents at any time is used to to predict these parameters by linear projection.

$$[\mu_i^t, \sigma_i^t, \rho_i^t] = \phi(h2_i^{t-1}; W_f), \tag{15}$$

The loss function is defined by the negative log-Likelihood L_i.

$$\begin{aligned} &L_i(W_{spa}, W_{tem}, W_{ins}, W_{st}, W_{sup}, W_s, W_f) \\ &= -\sum_{t=T_{obs}+1}^{T_{pred}} log(P(x_i^t, y_i^t | \mu_i^t, \sigma_i^t, \rho_i^t)), \end{aligned} \tag{16}$$

We train the model by minimizing the loss for all the trajectories in the training dataset. We jointly back-propagated through instance nodes, super nodes and spatial and temporal edges to update all the parameters to minimize the loss at each time-step.

Experiments

Dataset

We use Apollo acquisition car (BaiduApollo 2018) to collect traffic data, including camera-based images and LiDAR-based point clouds, and generate trajectories by detection and tracking.

Our new dataset is a large-scale dataset for urban streets, which focuses on trajectories of heterogeneous traffic-agents for planning, prediction and simulation tasks. Our acquisition car runs in urban areas in rush hours in those scenarios shown in Fig. 4. The data is generated from a variety of sensors, including LiDAR (Velodyne HDL-64E S3), radar (Continental ARS408-21), camera, high definition maps and

Table 2: The average displacement error and the final displacement error of the prior methods (ED, SL, SA) and variants of our method (TP) on our new dataset. For each evaluation metric, we show the values on pedestrians, bicycles, vehicles, and all the traffic-agents. We set the observation time as 2 seconds and the prediction time as 3 seconds for these measurements.

Metric	Methods	ED	SL	SA	TP-NoCL	TP-NoSA	TrafficPredict
Avg. disp. error	pedestrian	0.121	0.135	0.112	0.125	0.118	**0.091**
	bicycle	0.112	0.142	0.111	0.115	0.110	**0.083**
	vehicle	0.122	0.147	0.108	0.101	0.096	**0.080**
	total	0.120	0.145	0.110	0.113	0.108	**0.085**
Final disp. error	pedestrian	0.255	0.173	0.160	0.188	0.178	**0.150**
	bicycle	0.190	0.184	0.170	0.193	0.169	**0.139**
	vehicle	0.195	0.202	0.189	0.172	0.150	**0.131**
	total	0.214	0.198	0.178	0.187	0.165	**0.141**

a localization system at 10HZ. We provide camera images and trajectory files in the dataset. The perception output information includes the timestamp, and the traffic-agent's ID, category, position, velocity, heading angle, and bounding polygon. The dataset includes RGB videos with 100K 1920×1080 images and around 1000km trajectories for all kinds of moving traffic agents. A comparison of NGSIM, KITTI (with tracklets), and our dataset is shown in Table. 1. Because NGSIM has a very large, top-down view, it has a large number of vehicles per frame. In this paper, each period of sequential sequences of the dataset was isometrically normalized for experiments. Our new dataset has been released over the WWW (Apolloscape 2018).

Evaluation Metrics and Baselines

We use the following metrics (Pellegrini *et al.* 2009; Vemula *et al.* 2017) to measure the performance of algorithms used for predicting the trajectories of traffic-agents.

1. *Average displacement error*: The mean Euclidean distance over all the predicted positions and real positions during the prediction time.
2. *Final displacement error*: The mean Euclidean distance between the final predicted positions and the corresponding true locations.

We compare our approach with these methods below:

- *RNN ED (ED)*: An RNN encoder-decoder model, which is widely used in motion and trajectory prediction for vehicles.
- *Social LSTM (SL)*: An LSTM-based network with social pooling of hidden states (Alahi *et al.* 2016). The model performs better than traditional methods, including the linear model, the Social force model, and Interacting Gaussian Processes.
- *Social Attention (SA)*: An attention-based S-RNN architecture (Vemula *et al.* 2017), which learn the relative influence in the crowd and predict pedestrian trajectories.
- *TrafficPredict-NoCL (TP-NoCL)*: The proposed method without the category layer.
- *TrafficPredict-NoSA (TP-NoSA)*: The proposed method without the self-attention mechanism of the category layer.

Implementation Details

In our evaluation benchmarks, the dimension of hidden state of spatial and temporal edge cell is set to 128 and that of node cell is 64 (for both instance layer and category layer). We also apply the fixed input dimension of 64 and attention layer of 64. During training, Adam (Kingma and Ba 2014) optimization is applied with β_1=0.9 and β_2=0.999. Learning rate is scheduled as 0.001 and a staircase weight decay is applied. The model is trained on a single Tesla K40 GPU with a batch size of 8. For the training stability, we clip the gradients with the range -10 to 10. During the computation of predicted trajectories, we observe trajectories of 2 seconds and predict the future trajectories in next 3 seconds.

Analysis

The performance of all the prior methods and our algorithm on heterogeneous traffic datasets is shown in Table. 2. We compute the average displacement error and the final displacement error for all the instances and we also count the error for pedestrians, bicycles and vehicles, respectively. The social attention (SA) model considers the spatial relations of instances and has smaller error than RNN ED and Social LSTM. Our method without category layer (TP-NoCL) not only considers the interactions between instances but also distinguishes between instances by using different LSTMs. Its error is similar to SA. By adding the category layer without self attention, the prediction results of TP-NoSA are more accurate in terms of both metrics. The accuracy improvement becomes is more evident after we use the self-attention mechanism in the design of category layer. Our algorithm, TrafficPredict, performs better in terms of all the metrics with about 20% improvement of accuracy. It means the category layer has learned the inbuilt movement patterns for traffic-agents of the same type and provides good guidance for prediction. The combination of the instance layer and the category layer makes our algorithm more applicable in heterogeneous traffic conditions.

We illustrate some prediction results on corresponding 2D images in Fig. 5. The scenario in the image captured by the front-facing camera does not show the entire scenario. However, it is more intrinsic to project the trajectory results on the image. In most heterogeneous traffic scenarios, our algorithm computes a reasonably accurate predicted trajectory

Figure 5: Illustration of our TrafficPredict (TP) method on camera-based images. There are six scenarios with different road conditions and traffic situations. We only show the trajectories of several instances in each scenario. The ground truth (GT) is drawn in green and the prediction results of other methods (ED,SL,SA) are shown with different dashed lines. The prediction trajectories of our TP algorithm (pink lines) are the closest to ground truth in most of the cases.

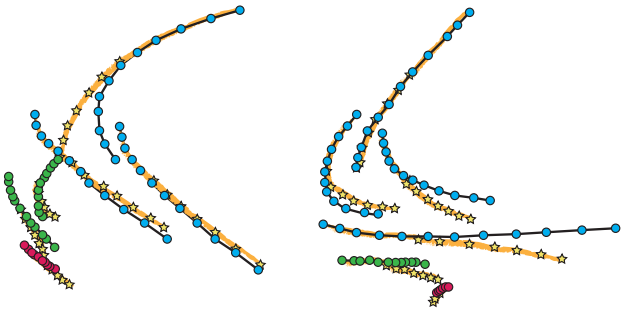

Figure 6: Illustration of some prediction results by our method. The ground truth of trajectories of vehicles, bicycles and pedestrians are drawn by blue, green and red respectively. Predicted locations are all represented by yellow stars. For each instance, first five discrete points are observed positions, but there are some overlaps in the illustration of pedestrian trajectories.

and is close to the ground truth. If we have prior trajectories over a longer duration, the prediction accuracy increases.

When traffic-agents are moving on straight lanes, it is easy to predict their trajectories because almost all the traffic-agents are moving in straight direction. It is more challenging to provide accurate prediction in cross roads, as the agents are turning. Fig. 5 shows 2D experimental results of two sequences in cross areas. There are some overlaps on trajectories. In these scenarios, there are many curves with high curvature because of left turn. Our algorithm can compute accurate predicted trajectories in these cases.

Conclusion

In this paper, we have presented a novel LSTM-based algorithm, TrafficPredict, for predicting trajectories for heterogeneous traffic-agents in urban environment. We use a instance layer to capture the trajectories and interactions for instances and use a category layer to summarize the similarities of movement pattern of instances belong to the same type and guide the prediction algorithm. All the information in spatial and temporal space can be leveraged and transferred in our designed *4D Graph*. Our method outperforms previous state-of-the-art approaches in improving the accuracy of trajectory prediction on our new collected dataset for heterogeneous traffic. We have evaluated our algorithm in traffic datasets corresponding to urban dense scenarios and observe good accuracy. Our algorithm is realtime and makes no assumption about the traffic conditions or the number of agents.

Our approach has some limitations. Its accuracy varies based on traffic conditions and the duration of past trajectories. In the future, we will consider more constraints, like the lane direction, the traffic signals and traffic rules, etc. to further improve the accuracy of trajectory prediction. Furthermore, we would like to evaluate the performance in more dense scenarios.

Acknowledgement

Dinesh Manocha is supported in part by ARO Contract W911NF16-1-0085, and Intel. We appreciate all the people who offered help for collecting the dataset.

References

[Administration 2005] U.S. Federal Highway Administration. Us highway 101 dataset. 2005. 3

[Alahi *et al.* 2016] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, F.F. Li, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *CVPR*, pages 961–971, 2016. 2, 5, 6

[Altché and Fortelle 2017] F. Altché and A. De La Fortelle. An lstm network for highway trajectory prediction. In *ITSC*, pages 353–359. IEEE, 2017. 2

[Apolloscape 2018] Apolloscape. Trajectory dataset for urban traffic. 2018. http://apolloscape.auto/trajectory.html. 5

[BaiduApollo 2018] BaiduApollo. 2018. https://github.com/ApolloAuto/apollo. 5

[Bera *et al.* 2016] A. Bera, S. Kim, T. Randhavane, S. Pratapa, and D. Manocha. Glmp-realtime pedestrian path prediction using global and local movement patterns. In *ICRA*, pages 5528–5535. IEEE, 2016. 2

[Bera *et al.* 2017] A. Bera, T. Randhavane, and D. Manocha. Aggressive, tense, or shy? identifying personality traits from crowd videos. In *IJCAI*, volume 17, pages 112–118, 2017. 2

[Bera *et al.* 2018] A. Bera, T. Randhavane, E. Kubin, A. Wang, D. Manocha, and K. Gray. Classifying group emotions for socially-aware autonomous vehicle navigation. In *CVPRW*, pages 1039–1047, 2018. 2

[Chandra *et al.* 2018a] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha. Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions. *arXiv:1812.04767*, 2018. 2

[Chandra *et al.* 2018b] R. Chandra, T. Randhavane, U. Bhattacharya, A. Bera, , and D. Manocha. Deeptagent: Realtime tracking of highly heterogeneous agents in very dense traffic. 2018. http://gamma.cs.unc.edu/HTI/. 2

[Cheung *et al.* 2018] E. Cheung, A. Bera, E. Kubin, K. Gray, and Dinesh D. Manocha. Identifying driver behaviors using trajectory features for vehicle navigation. *IROS*, 2018. 2

[Chung *et al.* 2015] J. Chung, K. Kastner, L. Dinh, K. Goel, A.C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In *NIPS*, pages 2980–2988, 2015. 2

[Cordts *et al.* 2016] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016. 3

[Danielsson *et al.* 2007] S. Danielsson, L. Petersson, and A. Eidehall. Monte carlo based threat assessment: Analysis and improvements. In *IV*, pages 233–238. IEEE, 2007. 2

[Deo and Trivedi 2018] N. Deo and M.M. Trivedi. Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. *arXiv:1805.05499*, 2018. 2

[Firl *et al.* 2012] J. Firl, H. Stübing, S.A. Huss, and C. Stiller. Predictive maneuver evaluation for enhancement of car-to-x mobility data. In *IV*, pages 558–564. IEEE, 2012. 2

[Geiger *et al.* 2013] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 32(11):1231–1237, 2013. 3

[Goodfellow *et al.* 2016] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016. 2

[Graves and Jaitly 2014] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *ICML*, pages 1764–1772, 2014. 2

[Gupta *et al.* 2018] A. Gupta, J. Johnson, F.F. Li, S. Savarese, and A. Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *CVPR*, number CONF, 2018. 2

[Helbing and Molnar 1995] D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995. 2

[Huang *et al.* 2018] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. *CVPR Workshop*, 2018. 3

[Jain *et al.* 2016] A. Jain, A.R. Zamir, S. Savarese, and A. Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *CVPR*, pages 5308–5317, 2016. 3

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

[Khosroshahi 2017] A. Khosroshahi. *Learning, Classification and Prediction of Maneuvers of Surround Vehicles at Intersections using LSTMs*. PhD thesis, UC San Diego, 2017. 2

[Kim *et al.* 2017] B. Kim, C.M. Kang, S.H. Lee, H. Chae, J. Kim, C.C. Chung, and J.W. Choi. Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. *ITSC*, 2017. 2

[Kingma and Ba 2014] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 6

[Lee *et al.* 2017] N. Lee, W. Choi, P. Vernaza, C.B. Choy, P.H. Torr, and M. Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *CVPR*, pages 336–345, 2017. 2

[Lefèvre *et al.* 2011] S. Lefèvre, C. Laugier, and J.Ibañez-Guzmán. Exploiting map information for driver intention estimation at road intersections. In *IV*, pages 583–588. IEEE, 2011. 2

[Lerner *et al.* 2007] A. Lerner, Y. Chrysanthou, and D. Lischinski. Crowds by example. In *CGF*, volume 26, pages 655–664. Wiley, 2007. 3

[Ma *et al.* 2018] Y. Ma, D. Manocha, and W. Wang. Autorvo: Local navigation with dynamic constraints in dense heterogeneous traffic. *CSCS*, 2018. 2

[Park *et al.* 2018] S. Park, B. Kim, C.M. Kang, C.C. Chung, and J.W. Choi. Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture. *arXiv:1802.06338*, 2018. 2

[Pellegrini *et al.* 2009] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, pages 261–268. IEEE, 2009. 3, 6

[Rasmussen and Williams 2006] C.E. Rasmussen and C. K. Williams. Gaussian processes for machine learning. 2006. *The MIT Press, Cambridge, MA, USA*, 38:715–719, 2006. 1, 2

[Toledo-Moreo and Zamora-Izquierdo 2009] R. Toledo-Moreo and M.A. Zamora-Izquierdo. Imm-based lane-change prediction in highways with low-cost gps/ins. *ITS*, 10(1):180–185, 2009. 1

[Treuille *et al.* 2006] A. Treuille, S. Cooper, and Z. Popović. Continuum crowds. *TOG*, 25(3):1160–1168, 2006. 2

[Vaswani *et al.* 2017] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 4

[Vemula *et al.* 2017] A. Vemula, K. Muelling, and J. Oh. Social attention: Modeling attention in human crowds. *arXiv:1710.04689*, 2017. 2, 3, 4, 6

[Vinyals *et al.* 2015] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015. 2

[Wang *et al.* 2008] J.M. Wang, D.J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *TPAMI*, 30(2):283–298, 2008. 2

[Yamaguchi *et al.* 2011] K. Yamaguchi, A.C. Berg, L.E. Ortiz, and T.L. Berg. Who are you with and where are you going? In *CVPR*, pages 1345–1352. IEEE, 2011. 2