Pedestrian Trajectory Prediction

Sreeganesh VR
Department of ECE
Boston University
Email: vrs@bu.edu

***Abstract*—Pedestrian trajectory prediction is a task that is intrinsic to the human mind. This ability helps us in a variety of ways. In the context of traffic, it helps us decide on planning our path and to look out for pedestrians whose path may intersect with ours.The two entities that are the most important in a traffic context are pedestrians and vehicles. Pedestrian motion is much more complex compared to vehicular motion owing to their ability to change directions faster than vehicles and not having to follow as many rules as vehicles.In addition to presenting the popular approaches in pedestrian trajectory prediction, we explore a GAN based approach to pedestrian trajectory prediction.**

I. Introduction

Human vision, fundamentally, evolved to enable the aims of the human body. As seen from the work of Daniel Simons [2], the human brain filters components from the eye based on what assists the current task at hand. During driving, this phenomenon presents itself as searching for entities that would force us to change their planned trajectories such as break, steer or accelerate. One of the fundamental ways that this is implemented is by predicting the trajectories of the entities. The human brain is able to do this because it has models of how vehicles and pedestrians are expected to move given their history through observations. In this paper, we shall be considering the case of pedestrians and predicting their trajectory.

Pedestrian and vehicular trajectories are fundamentally different and occupy different manifolds in the latent space. This is due to their difference in movement and freedoms. The human biped can implement trajectories that are too complex for vehicles. So when training autonomous systems, it is important that the systems has subroutines in place that predicts the trajectory of other traffic entities to decide whether or not they must be paid attention to. An example of a way that the brain does automatically is how the human brain pays more attention to the cars and pedestrians in front of the vehicle rather than the ones we already passed or are in the other lane.

This paper is divided in to the following sections. Section 2 briefly introduces some conventional approaches to predicting pedestrian trajectory. In section 3, we shall dive into a specific implementation of trajectory prediction. Section 4 quantifies improvements compared to selected models. Section 5 concludes the paper.

II. Literature Review

As presented before, we shall be focusing on the trajectory prediction of pedestrians. Trajectory prediction has been paid increased attention with the onset of autonomous vehicles. To that extent, there has been increase in the approaches used to predict pedestrian trajectories. Following are some of the popular approaches to this task.

A. Using Recurrent Neural Networks

Since the task at hand is essentially a sequence problem, using recurrent neural networks for trajectory prediction is common. Alahi et al. [3] introduced a social LSTM that extracts the state of each pedestrian movement using LSTM and then applies a social pooling layer over the LSTMs to model social interactions between pedestrians. The trajectories are then predicted assuming the pedestrians obey bivariate Gaussian distributions. [1]

B. Using Convolutional Neural Networks

CNNs have been implemented for trajectory prediction with extraction of information from images. Unlike RNN approaches which is sequential, CNN approaches can estimate all time steps simultaneously [1]. Nikhil and Morris [4] apply CNNs to predict trajectories.

C. Using Graph Neural Networks

In these approaches, the pedestrians and other entities are described using nodes and their interaction is modelled using edges. Velickovi et al [5] proposed graph attention networks (GAT), which allows the implementation of a self-attention-based architecture to any type of structured data which can be defined as a graph [1].

III. Methodology

Generative Adversarial Network's ability to produce realistic samples stems from their ability to model the manifold of real-life samples in the latent space. This means that once the network converges and learns the manifold, it can sample from said manifold in latent space and use it to produce very realistic samples. In our context, it is very useful because this ability is put to use to produce trajectories that are realistic and therefore, has a higher probability of happening compared to other trajectories that are executable by a pedestrian given their trajectory history. This is the fundamental idea behind using GANs to this task.

In the paper, the pedestrian trajectory prediction problem is formulated as a 2D world where trajectory prediction is producing sequences of coordinates corresponding to the trajectory of the pedestrian. Naturally, this also assumes discretised times across sequence samples.

The architecture is divided into three modules. They are

- Feature encoder module
- generator/decoder module
- Discriminator module

First, the interaction model of dynamic participants and fixed obstacles in the scene is established by using the spatial-temporal graph, and LSTM is used to extract nodes feature coding from the historical trajectory information of pedestrians. The extracted feature coding is input into the time attention module. This module then assigns different weights to it in each time step to get the time feature coding. The historical trajectory information of pedestrian, the location information of fixed obstacle and the time feature coding are input into the relative scaled dot product attention module to capture the relative interactive feature coding of the impact of the global scene on the pedestrian trajectory. [6]

In the generator module, we integrate random noise, time feature coding and relative interactive feature coding as the input. Based on these features, the generator generates a distribution of diverse trajectories that belong to the same manifold in the latent space of data. During training, this generator is trained with the help of a discriminator that checks whether the particular data point belongs in the manifold or not. When the discriminator is unable to say for certain, we say that the generator has been trained and that the data point generated by it is realistic. This is considered to be the trajectory that is taken by the pedestrian in this particular context.

A. Spatial-Temporal Graph Architecture

In the paper, they use a spatial-temporal graph to describe the interaction between pedestrians and the environment. The spatial-temporal graph is expressed as: $G = (v, \epsilon_S, \epsilon_T)$, v is the instance nodes set, ϵ_T is a set of time edges, $epsilons_S$ is a set of spatial edges. The instance ndoes include pedestrian nodes, P, and fixed obstacle node, O, the nodes is variable. The spatial edge connects all instance nodes while the time edge connects adjacent time steps to the same pedestrian node.[6]

B. Time Attention Module

The time attention module extracts the trajectory information in the time domain and assign different weights to it. We first use multi-layer perceptron to embed coordinate position of pedestrian, i, to obtain fixed length vector, e^t_{pi}. The LSTM unit then takes this embedded vector as input to obtain the pedestrian node feature code, h^t_i. [6]

$$e^t_{pi} = \phi(x^t_i, y^t_i; W_p) \tag{1}$$

$$h^t_i = LSTM(h^{t-1}_i, e^t_{pi}; W^p_{temporal}) \tag{2}$$

where, phi is a non-linear embedding function, W_p is the embedding weight, $W^p_{temporal}$ is the weight of the temporal-edge LSTM cell. [6]

We take the pedestrian node feature code h^t_i obtained above as the input of the time attention module to obtain the score, S_i, the score of h_i

$$S_i = FC(h_i) = tanh(w^T_a h_i + b_a) \tag{3}$$

Next, S_i is taken as the input of $S - BN$ layer, and the attention weight, a_i, of h_i is obtained.

$$a_i = softmax(BN(S_i)) \tag{4}$$

where BN is the batch normalisation function. The time feature coding vector $\hat{h}_i$ with time information is obtained by multiplying the respective node feature coding h_i and its corresponding attention weight, a_i, and summation. The time edge information of pedestrian is captured through the time attention module, improving accuracy and robustness of model.

$$\hat{h}_i = \sum_{i=1}^{N}(a_i h_i) \tag{5}$$

C. Relative Scaled Dot Product Attention Module

This module is introduced to capture the spatial information of all instance nodes. In addition to considering the relative position of the target pedestrian and its neighbours, it also considers the relative position of the pedestrian with fixed obstacles and assigns different weights. First, we calculate the relative distance, O^t_{ij} between the pedestrian and the fixed obstacle node. [6]

$$O^t_{ij} = (x^t_i - x^t_{oi}, y^t_i - y^t_{oi}) \quad \textit{Obstacles exist} \tag{6}$$

when obstacles do not exist, the the relative distance is assigned the value of (0,0).

Next, the fixed length vector, r^t_{ij} is obtained by embedding the relative distance from pedestrian i to adjacent pedestrian j and to the obstacle through MLP. The fixed length vector is then used as the input of LSTM unit to obtain the relative feature code, h^t_r, which contains the context information of the scene. When the node does not exist, the relative feature coding contains only social interaction information [6]

$$r_{ij}^t = \phi(x_i^t - x_j^t, y_i^t - y_j^t, O_{ij}^t; W_r) \quad (7)$$

$$h_r^t = LSTM(h_r^{t-1}, r_{ij}^t; W_{spatial}^r \quad (8)$$

where W_r is the embedded weight, and $W_{spatial}^R$ is the weight of the spatial edge-LSTM cell, which is shared among all instance nodes. The scaled dot product attention mechanism is used to assign influence weight to all instances nodes in the scenes. The influence weight is then multiplied by the time feature coding vector, $\hat{h}_i$ to obtain the relative interaction feature coding, I_i^t [6]

$$W_R^t = softmax\left(\frac{1}{\sqrt{d_e}} Dot(W_2 h_r^t, W_1 \hat{h}_i^t)\right) \quad (9)$$

$$I_i^t = W_R^t \cdot h_r^t \quad (10)$$

where W_1 and W_2 are weights used for linear scaling and projection of hidden states onto the d_e dimensional vector. The process of encoding relative interactive features based on spatial-temporal graph has been completed [6]

D. Generator

GANs consists of a generator/decoder and discriminator model. We use a decoder based on LSTM unit for eigenvector decoding and trajectory generation. As generator input, we use the standard normally distributed noise, z, the time feature coding vector, $\hat{h}_i$ and the relative interactive feature coding, I_i^t to obtain the mixed feature coding vector h_{gi}^t. Then, h_{gi}^t is converted to spatial coordinates through a MLP [6].

$$h_{gi}^t = LSTM(h_{gi}^{t-1}, [z, \hat{h}_i^t, I_i^t]; W_g) \quad (11)$$

$$(\hat{x}_i^{t+1}, \hat{y}_i^{t+1}) = MLP(h_{gi}^t; W_{ge}) \quad (12)$$

where, W_g and W_{ge} are the embedding weights.

E. Discriminator

The discriminator will evaluate the predicted future trajectories of pedestrians. We use the real future trajectories to train the discriminator to learn what real trajectories look like. We use MLP in the last hidden state of the encoder to get the classification score, L_{disi}. [6]

$$e_{disi}^t = MLP([T_i^t]; W_{e1}) \quad (13)$$

$$h_{disi}^t = LSTM(h_{disi}^{t-1}, e_{disi}^t; W_{e2}) \quad (14)$$

$$L_{disi} = MLP(h_{disi}; W_{e3}) \quad (15)$$

where T_i^t is each coordinate from $[X_i^t, \hat{Y}_i^t]$ or $[X_i^t, Y_i^t]$ at time t, h_disi, is the integration, h_{disi}^t, L_{disi} is the binary result of classification: 0 indicating false and 1 indicating real [6]

F. Loss

We define the training goals of RISTG-GAN as follows.

$$V = arg \min_G \max_D L_{GAN}(G, D) + \lambda L_{L2}(G) \quad (16)$$

where $lambda$ is the weighing coefficient and the adversarial loss $L_{GAN}(G, D)$ and $L2$ loss, $L_{L2}(G)$ are defined as follows

$$L_{GAN}(G, D) = E_{i \in RISTG}[logD(Y_i)] + E_{i \in RISTG}[log(1 - D(\hat{Y}_i))] \quad (17)$$

$$L_{L2}(G) = E_{i \in RISTG}[\left\| Y_i - \hat{Y}_i \right\|_2] \quad (18)$$

G. Implementation Details

The encoder and decoder are constructed based on LSTM units. The hidden state size of the encoder and decoder are 16 and 32 respectively, and the input coordinates are embedded into 16 dimensional vectors. The Adam optimizer is used to train the generator and discriminator. The intial learning rate is set to 0.001 and the number of training are set to 200 rounds. [6]

IV. Results

Our proposed model is compared with five existing typical models on five publicly available datasets. This system uses data within 8 seconds to evaluate the model, taking the first 3.2 seconds of each trajectory as the training value and predicting the next 4.8 seconds of trajectory. It was found that the performance of S-LSTM model is better than that of the simple LSTM model beacuse the former proposes to use the social pooling layer to capture the interaction information between local pedestrins. The SS-LSTM model was found to perform better because it not only considers the interaction of all pedestrians in the scene, but also uses the context information of the scene to predict the pedestrian trajectory. The average values of ADE and FDE in the five datasets decrease by 18% and 17% respectively, which further proves the importance of considering the context information of the scene for prediction. [6]

The previous approach that used GANs for trajectory prediction is the S-GAN model. Based on it, Sophie model takes the scene information into account and improves the prediction of the model by introducing the physical attention mechanism. In the datasets of ETH-uni and ETH-hotel, the RIST-GAN-2 model was found to perform better than SoPhie model. The mean values of FDE and ADE are reduced by 24% and 21.7%, respectively. [6]

V. Conclusion

In this paper, we briefly explained the current approaches to pedestrian trajectory detection and explored a GAN based approach to trajectory prediction. The addition of GANs is extremely smart since it helps produce trajectories that are part of the manifold of possible trajectories. Further by designing loss in the manner it is currently done means that the predicted trajectory is very well tied to the trajectory history. These two additions make this a well designed approach to pedestrian based trajectory prediction. If converted into a product, the target user would be car manufacturers since this can be implemented as the subroutine for their driver assistance systems.

References

[1] B. I. Sighencea, R. I. Stanciu and C. D. Căleanu, ”Pedestrian Trajectory Prediction in Graph Representation Using Convolutional Neural Networks,” 2022 IEEE 16th International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, 2022, pp. 000243-000248, doi: 10.1109/SACI55618.2022.9919494.

[2] Chabris, C. F., and Simons, D. J. (2010). The Invisible Gorilla: And Other Ways Our Intuitions Deceive Us. Crown Publishers.

[3] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei and S. Savarese, ”Social LSTM: Human Trajectory Prediction in Crowded Spaces”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 961-971, 2016.

[4] N. Nikhil and B. Tran Morris, ”Convolutional neural network for trajectory prediction”, Proceedings of the European Conference on Computer Vision (ECCV) Workshops, September 2018.

[5] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio and Y. Bengio, ”Graph attention networks”, 6th International Conference on Learning Representations (ICLR), 2018.

[6] D. Zhao et al., ”Multimodal Pedestrian Trajectory Prediction Based on Relative Interactive Spatial-Temporal Graph,” in IEEE Access, vol. 10, pp. 88707-88718, 2022, doi: 10.1109/ACCESS.2022.3200066.