Feed-forwards meet recurrent networks in vehicle trajectory prediction

Mohammadhossein Bahari
Alexandre Alahi

Ecole Polytechnique Federale de Lausanne (EPFL)

May 2019

STRC

19th Swiss Transport Research Conference
Monte Verità / Ascona, May 15 – 17, 2019

Ecole Polytechnique Federale de Lausanne (EPFL)

Feed-forwards meet recurrent networks in vehicle trajectory prediction

Mohammadhossein Bahari, Alexandre Alahi
Visual Intelligence for Transportation
Ecole Polytechnique Federale De Lausanne
Ecublens VD
phone: +41 21 69 30894 - +41 21 69 32608
fax:
{Mohammadhossein.bahari,Alexandre.alahi}@epfl.ch

May 2019

Abstract

For an autonomous car, getting surprised is the worst thing that can happen. To prevent that, plenty of studies are trying to forecast traffic participants' actions especially in an urban scene using recurrent networks. Recurrent networks are used for temporal tasks in many application domains like Natural Language processing and computer vision. Despite the tendency in the literature to use recurrent neural networks for trajectory prediction, we argue that because of small dependency in trajectory sequences of a vehicle, a feed-forward neural network can be used, instead. In this paper, we will compare these two methods in vehicle trajectory prediction while considering the vanilla models or taking the scene into account. In order to have more variations in trajectories, roundabouts are used as a case study. Our results show that the proposed feed-forward network has competitive results with a recurrent network with 6 times faster processing time.

Keywords

Keywords; in English; language

Introduction

An autonomous vehicle has to navigate through complex scenes and interact with other cars without failure. This injects the need to have a clear understanding of other agents' future decisions in different road structures. Predicting next movements of a car in a highway is a straightforward problem which has been addressed repeatedly in the past works Deo and Trivedi (2018), Altche and de La Fortelle (2017) and Kim *et al.* (2017). In an urban environment, however, the problem is more complex due to the inherent complexity and diversity of the road. In an urban scene, static features of the space are the main factors in every agents' movements. Although vehicle interaction with other traffic participants has been widely addressed, despite the essence, interaction with scene has not gotten much attention so far.

A human driver instantly predicts other agents' decisions in every scene based on the road constraints and the past positions of each agent. For instance, in an intersection, it is clear that each vehicle has a limited set of feasible paths. Inspired by this, an autonomous car should be able to predict other agents' future movements by the past positions and physical constraints of the scene. Moreover, a quick response is vital for autonomous vehicles. It should process the inputs as fast as possible so the vehicle has enough time for appropriate reactions.

Pioneering work in trajectory prediction has successfully addressed the problem with different methods. Sadeghian *et al.* (2018b) used an attention module to incorporate scene features into the Long-Short Term Memory (LSTM) model. Lee *et al.* (2017) used an inverse optimal control (IOC) ranking module that determines the most likely hypotheses while incorporating scene context and interactions. They also used Recurrent Neural Networks (RNN) to encode and decode sequences. Xue *et al.* (2018) proposed a hierarchical encoder-decoder model based on LSTMs. While all of them have been made great progress in addressing the problem, they used RNNs which are sequential and can't be parallelized so have large processing time. Moreover, LSTMs are vulnerable to parameter tuning and need task-specific engineering like clipping gradients.

The majority of studies in trajectory prediction in the literature leverage recurrent networks for modeling the trajectories as recurrent networks are designed for sequence-specific tasks. However, according to recent studies, feed-forward networks are able to have the same results as recurrent networks Miller and Hardt (2018). Feed-forwards have some attractive features which make this substitution appealing. For instance, they could be easily parallelized. Also, they are less vulnerable to vanishing gradients. More specific to our problem, vehicle trajectory prediction, trajectories do not have long dependencies which promote the substitution. On the

other hand, recurrent networks have some advantages like being able to extract long dependencies and efficiency for long sequence generation. This controversy motivated us to shed light on their pros and cons in different aspects of vehicle trajectory prediction. As they are the main building blocks of different predictors, the results could help when designing a predictor.

In this paper, we will compare the performance of the feed-forward network with the common LSTM network in the two tasks of modeling only trajectories and modeling trajectories taking into account scene context. Roundabouts have been chosen as a case study for the scene as they are rich in having vehicle-scene interactions.

1 Related Work

1.1 Interaction-aware Prediction

Deep learning methods have achieved a wide range of success in different applications and made the traditional handcrafted methods to be replaced by generic data-driven ones. Alahi *et al.* (2016) showed the boost over the well-known hand-crafted model, Social Forces Helbing and Molnar (1998) by a social LSTM network. LSTMs are recurrent networks designed for capturing long-term dependencies in sequences and achieved great success in sequence tasks such as machine translation Chung *et al.* (2015) and speech recognition Chorowski *et al.* (2014). Data-driven methods addressing trajectory prediction can be grouped by the interaction they attend to. Alahi *et al.* (2016) proposed a network of LSTMs with pooled hidden layers to model social interactions between pedestrians. A convolutional method is proposed in Deo and Trivedi (2018) to tackle the interactions between vehicles in a highway.

Although in pedestrian trajectory prediction, interaction among agents is an important factor, but agent-scene interaction is more influential in vehicle urban prediction as vehicles are constrained to the road. ÙDespite the crucial role of scene in trajectory prediction, few studies have addressed the problem so far. Sadeghian *et al.* (2018b,a) used a CNN block to extract scene features followed by an attention module which is in charge of deciding where to look at. Lee *et al.* (2017) proposed an encoder-decoder model based on gated recurrent units and employed pooled scene extracted features in the decoder. A hierarchical structure is used in Xue *et al.* (2018) where scene features and social interactions are encoded by LSTMs and the concatenated encoded values are decoded by another LSTM to achieve predictions. A multi-task learning approach is presented in Xu *et al.* (2017) where semantic segmentation is a side task and the

segmented context is utilized by the LSTM with the agent history. However, they use car-view as opposed to our work which uses a bird view instead. Manh and Alaghband (2018) divide the scene into cells and derive a hidden layer for each cell and use the hidden layers to incorporate static contexts, yet this information is derived with past trajectories instead of scene image which injects the need to a large amount of data and poor performance in an unseen scene. The same issue exists in Zyner *et al.* (2018), in which the driver intention in roundabouts is predicted. As they do not use scene image, the model should be used in similar sized intersections which is a big constraint.

1.2 Feed-forward and Recurrent Networks

While recurrent networks show brilliant results on sequence tasks, it has been shown that the same results could be achieved by feed-forward networks. The results on the translation task in Vaswani *et al.* (2017) and language modeling in Dauphin *et al.* (2016) are some of the examples. Miller and Hardt (2018) proved that stable recurrent networks can be approximated by feed-forward networks. Moreover, an unstable recurrent model can often be made stable without performance loss. These together mean a feed-forward network can have the same performance as the recurrent one. Apart from the success of feed-forward networks in sequential tasks, prediction of trajectories inherently doesn't have long dependencies as a language model. All the previous facts promote leveraging feed-forward networks in trajectory prediction. Nikhil and Morris (2018) used a feed-forward convolutional network in trajectory prediction problem. However, they didn't take into account the interaction with the scene. In this paper, we will employ a feed-forward network and will model agent-scene interaction.

2 Method

2.1 Inputs and Outputs

To do the prediction based on the scene, we need the track histories and the scene around the car. The trajectories of the vehicle of interest (VOT) are fed to the network as

$$X_t = [\mathbf{x}_{t-t_{obs}-1}, ..., \mathbf{x}_{t-1}, \mathbf{x}_t], \ \ where \ \ \mathbf{x}_t = [x_t, y_t] \tag{1}$$

Figure 1: A pre-processed scene where the previous movements are plotted and the last one is in big circle

is the coordinates of the VOT at time t. In the real world, the decision of a driver with respect to the scene is dependent on the space around the VOT rather than the whole scene. Thus, a cropped segmented image of the scene around the VOT at the prediction time is used as an input to the network. Also, the same as some other previous works like Cui *et al.* (2018), Bansal *et al.* (2018), the image is pre-processed in a way that VOT appears in the center and the y-axis is the forward direction of the vehicle. One example of such an image can be seen in Figure 1. The output of the model will be a sequence of positions as

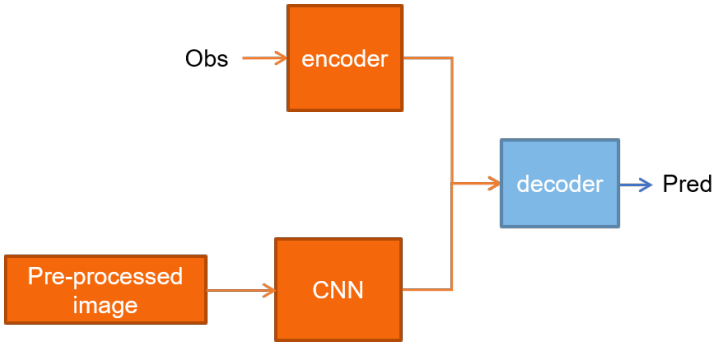
$$Y_t = [\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, ..., \mathbf{y}_{t+t_{pred}}], \ \textit{where} \ \ \mathbf{y}_t = [x_t, y_t]. \tag{2}$$

2.2 Model

To perform the predictions an encoder-decoder structure is employed as Figure 2 which is the base model used in different works in the literature like Manh and Alaghband (2018). Past positions are encoded by the encoder to acquire proper representation. The CNN network, which is in charge of detecting the boundaries of road and off-road regions, processes the scene. The two extracted features are then concatenated and decoded by a decoder block. Our comparison is based on two main tasks: modeling trajectories without any other information and modeling them taking into account scene information. It's worth mentioning that for the former, the same model as 2 is used but without the scene branch.

We will use feed-forward (FF) and recurrent neural network (RNN) as the encoder and decoder blocks of the network. For the FF, we used a multi-layer perceptron (MLP) network with 3 layers of (32, 32, 64) as the encoder and a network with 3 layers of (128, 128, 40) as the decoder. A long short-term memory (LSTM) is used for the recurrent network. The number of hidden

Figure 2: Employed encoder-decoder network



layers for the model is chosen as 128 and the embedding dimension is 64. The CNN network consists of 2 2-dimensional convolutions with kernel size of 8, stride of 4 and output channels of 16 and 32. The convolutions are followed by a 128 node fully connected layer.

2.3 Implementation Details

The models are trained with Adam optimizer Kingma and Ba (2015) with weight decay of 10^{-2} and the initial learning rate of 0.0005 which is decreased by half every 25 epochs. We used ReLU activation function. The network is trained in an end-to-end fashion with 100 epochs. The EPFLroundabout dataset is used for training and testing which will be introduced in the next section. We trained on 3 of our roundabouts and tested on two of them which consist of an unseen roundabout and a scene which tracks are divided into train and test set. The training tracks are 88K and the test tracks are 8K. In order to have the same settings for different scenes, the positions are converted to miters and the sampling rate of videos is 10. The visible scene of each car for the model is a square with sides of 50 meters with the VOT in the center. The observation length is 0.9 sec (9 frames) and the prediction length is 2 sec (20 frames) for all experiments. The model is implemented using PyTorch (Paszke *et al.,* 2017 NiPS Talk).

3 Experiments

3.1 Baselines and Evaluation

A set of different methods have been used as baselines as follows:

Table 1: Qualitative results of different baselines on two roundabouts, Route Cantonale which was used in training data (of course with separated tracks for training and test) and Morges avenue which is kept unseen to the model. The metrics are ADE and FDE in parenthesis both in meters. The FF model does the same as the RNN in the two models.

	Kalman filter	Vanilla FF	Vanilla RNN	Scene-FF	Scene-RNN
Route Cantonale:	1.22 (2.72)	0.85 (1.82)	0.83 (1.74)	0.61 (1.21)	0.59 (1.16)
Morges avenue(unseen):	1.36 (3.07)	1.20 (2.75)	1.16(2.59)	0.98 (2.16)	0.97 (2.14)

- **Kalman filter**. Kalman filter forecasts by extrapolating past trajectories without any other information.
- **Vanilla LSTM**. This method utilizes past trajectories to do the prediction. LSTM models are used as the encoder and decoder. The model takes into account only trajectories.
- **Vanilla FF**. An MLP network is used as the encoder and decoder to model trajectories. The model takes into account only trajectories.
- **Scene RNN**. A hierarchical network according to Figure 2 where LSTM models with the mentioned settings at section 2.2 is used. The model takes into account both trajectories and scene.
- **Scene FF** . The MLP is used as the encoder and the decoder with the settings explained at section 2.2. The model takes into account both trajectories and scene.

The prediction error is calculated by the two common metrics in the literature, as in Pellegrini *et al.* (2009):

1. *Average displacement error (ADE)*. The average Euclidean distance between the predicted points and the ground truth over all predicted time steps and vehicles.
2. *Final displacement error (FDE)*. The displacement error between the final predicted point at the end of the prediction horizon and the actual destination.

3.2 EPFL-Roundabout Dataset

Datasets are one of the key elements in machine learning research. There are plenty of previous public datasets on vehicle trajectory. However, they are more suitable for interactions among the agents rather than the static scene context. Noisy annotations, lack of interactions with the scene and inappropriate camera view injects the need for a dataset rich in scene interactions with accurate annotations and available video. To answer the essence of a dataset with vehicle-

Figure 3: One scene of EPFLRoundabout dataset which is Route de la Pierre roundabout

scene interaction, we captured the EPFL-Roundabout dataset of 4 roundabouts in Lausanne, Switzerland. We have chosen roundabouts as a case study since they are more complex than highways or other structured environments. The EPFL-Roundabout dataset is comprised of drone videos from 4 different roundabouts around EPFL in Lausanne, Switzerland. Each video provides a top-down view of the cars entering the roundabouts. One example of the roundabouts can be seen in Figure 3. The dataset is over 2 hours videos of 25 frame per seconds capturing more than 4500 vehicles which are sufficiently large. The videos are stabilized and traffic participants are detected and tracked using state-of-the-art methods. The annotations comprise of the location of the objects, bounding boxes around them and types of the objects in 5 categories (car, truck, bus, pedestrian, cyclist) for every frame.

3.3 Results

We aimed to compare FF and RNN networks. Table 1 shows the quantitative results for different baselines on two scenes. As we expect, the FF could achieve the same results as the LSTM both in the vanilla model and the scene model. Also, the scene shows it's major impact and improved the results. The results show that our model is generalizable and can perform well on a new unseen scene.

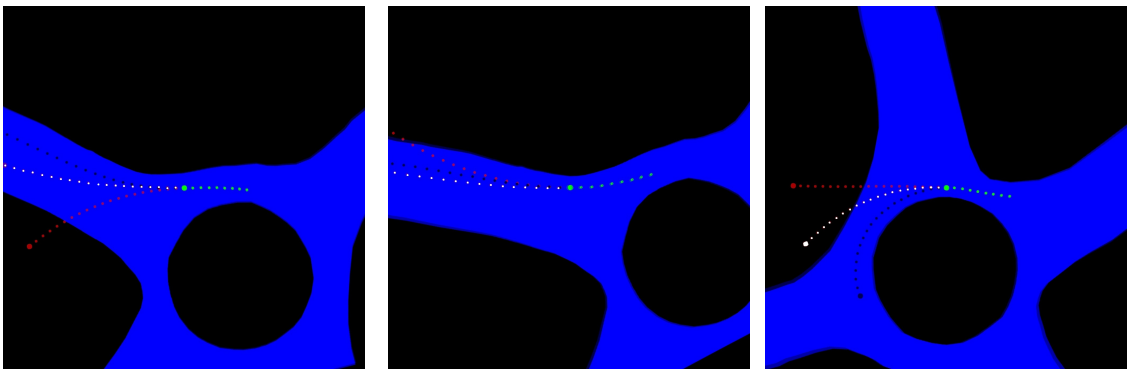
To visually see the effects of adding the scene to the model, let's have a look at the qualitative results. Figure 4 shows the predicted positions for the vanilla and the scene model. Figures 4(a) and 4(b) show how adding the scene changed predictions. Without the scene, the model only extrapolates past positions. However, the scene model observes the road and the prediction turns due to the road shape. The predicted points in Figure 4(c) end in the off-road part of the image

Figure 4: Qualitative analysis: The observation (in green), the vanilla prediction (in red), the scene model prediction (in white) and the ground truth (in black) are shown.

(a) By extrapolating the past positions, the vanilla model predicts a continues turn but the scene model detects the road and enters it.

(b) The vehicle is having a small turn to exit roundabout but the vanilla takes it as a turn. However that is not the case for the scene model.

(c) Although the scene model predicts better than the vanilla one, the prediction is off-the-road which shows the model should still be better.



which shows the model should be improved. The wrong predictions could be due to having only one output trajectory rather than multiple plausible ones which causes the output to be the mean of them. We will count some ideas to improve the performance in the next section.

3.4 Response time

FF networks are able to be parallelized, however, RRN networks are inherently sequential and thus, slow to train and test. For an LSTM network, input sequence should be given to the model one by one and the output is generated sequentially. However, in the FFs, the outputs can be calculated independently. We assessed the two networks with respect to the test time which is the average time to predict each input sequence. Due to Table 2, MLP is more than 11 times faster than LSTM because of the parallelized calculations. Note that the reported time includes the pre-processing part and is measured leveraging Nvidia GTX 1080 GPU. The MLP result is promising and suitable for real-time applications.

Table 2: Average time of the scene MLP and scene LSTM models to calculate the prediction for each input.

	Test time (sec)
Scene LSTM	0.016
Scene MLP	0.0014

3.5 conclusion and Future work

In this paper, we studied the performance of a FF and an RNN network on the task of vehicle trajectory prediction. We showed that the FF can achieve the same results as the RNN in both vanilla and scene models with 11 times faster test time. This encourages using FF models in previous RNN based solutions which probably results in the same performance with a faster response time.

Regarding modeling the scene, which is the most important part of vehicle prediction, we showed that we can predict vehicle trajectories using the scene with noticeable improvements over vanilla models. However, the model needs to better perceive the scene. Injecting human knowledge to the model could help its understanding a lot. As an example Bansal *et al.* (2018) add a loss function to avoid off-road predictions. We will add such information to the model to improve the performance. Also, predicting vehicle trajectory doesn't have a unique answer most of the times as it is a probabilistic problem. We will add multi-modality to the model to have multiple plausible predictions with their probabilities. The next step after modeling the scene is to take into account other vehicles. Due to our results, FF has better features than recurrent networks in vehicle trajectory prediction. Based on that, we will study their ability to acquire joint distribution of vehicles in the scene.

Acknowledgement

The research reported in this publication was partially supported by European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant.

4 References

Alahi, A., K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei and S. Savarese (2016) Social lstm: Human trajectory prediction in crowded spaces, paper presented at the *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Altche, F. and A. de La Fortelle (2017) An lstm network for highway trajectory prediction, paper presented at the *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 353–359, Oct 2017, ISSN 2153-0017.

Bansal, M., A. Krizhevsky and A. S. Ogale (2018) Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst, *CoRR*, **abs/1812.03079**.

Chorowski, J., D. Bahdanau, K. Cho and Y. Bengio (2014) End-to-end continuous speech recognition using attention-based recurrent nn: First results, *CoRR*, **abs/1412.1602**.

Chung, J., K. Kastner, L. Dinh, K. Goel, A. C. Courville and Y. Bengio (2015) A recurrent latent variable model for sequential data, in C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama and R. Garnett (eds.) *Advances in Neural Information Processing Systems 28*, 2980–2988, Curran Associates, Inc.

Cui, H., V. Radosavljevic, F. Chou, T. Lin, T. Nguyen, T. Huang, J. Schneider and N. Djuric (2018) Multimodal trajectory predictions for autonomous driving using deep convolutional networks, *CoRR*, **abs/1809.10732**.

Dauphin, Y., A. Fan, M. Auli and D. Grangier (2016) Language modeling with gated convolutional networks, paper presented at the *ICML*.

Deo, N. and M. M. Trivedi (2018) Convolutional social pooling for vehicle trajectory prediction, *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1549–15498.

Helbing, D. and P. Molnar (1998) Social force model for pedestrian dynamics, *Physical Review E*, **51**, 05 1998.

Kim, B., C. M. Kang, S.-H. Lee, H. Chae, J. Kim, C. C. Chung and J. W. Choi (2017) Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network, *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 399–404.

Kingma, D. P. and J. Ba (2015) Adam: A method for stochastic optimization, *CoRR*, **abs/1412.6980**.

Lee, N., W. Choi, P. Vernaza, C. Choy, P. H. S. Torr and M. Chandraker (2017) Desire: Distant future prediction in dynamic scenes with interacting agents, 2165–2174, 07 2017.

Manh, H. and G. Alaghband (2018) Scene-lstm: A model for human trajectory prediction, *CoRR*, **abs/1808.04018**.

Miller, J. and M. Hardt (2018) When recurrent models don't need to be recurrent, *CoRR*, **abs/1805.10369**.

Nikhil, N. and B. T. Morris (2018) Convolutional neural network for trajectory prediction, paper presented at the *ECCV Workshops*.

Paszke, A., S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga and A. Lerer (2017 NiPS Talk) Automatic differentiation in pytorch.

Pellegrini, S., A. Ess, K. Schindler and L. van Gool (2009) You'll never walk alone: Modeling social behavior for multi-target tracking, paper presented at the *2009 IEEE 12th International Conference on Computer Vision*, 261–268, Sep. 2009, ISSN 2380-7504.

Sadeghian, A., V. Kosaraju, A. Sadeghian, N. Hirose and S. Savarese (2018a) Sophie: An attentive gan for predicting paths compliant to social and physical constraints, *CoRR*, **abs/1806.01482**.

Sadeghian, A., F. Legros, M. Voisin, R. Vesel, A. Alahi and S. Savarese (2018b) Car-net: Clairvoyant attentive recurrent network, paper presented at the *Computer Vision – ECCV 2018*, 162–180, Cham, ISBN 978-3-030-01252-6.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin (2017) Attention is all you need, paper presented at the *NIPS*.

Xu, H., Y. Gao, F. Yu and T. Darrell (2017) End-to-end learning of driving models from large-scale video datasets, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3530–3538.

Xue, H., D. Q. Huynh and M. Reynolds (2018) Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction, paper presented at the *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1186–1194, March 2018.

Zyner, A., S. Worrall and E. M. Nebot (2018) Naturalistic driver intention and path prediction using recurrent neural networks, *CoRR*, **abs/1807.09995**.