Social-TGCN: Human Trajectory Prediction with Temporal Graph Convolutional Network

Vivian Wong [1]

Abstract

Being able to understand and to predict the motion of humans in a crowded space is critical to the design of space and agents that navigate around the space, such as autonomous vehicles;;. The problem is difficult, however, given the complex nature of human interactions. A machine learning model used for trajectory prediction should be able to observe both the spatial and temporal information to foresee future trajectories. To do this, we propose Social Temporal Graph Convolutional Network (Social-TGCN), which uses graph convolutional network (GCN) to learn spatial knowledge, and Gated Recurrent Unit (GRU) to learn temporal knowledge. In this work, we show that a naive implementation of Social-TGCN can achieve a performance comparable to that of a pioneering work, Social-LSTM (Alahi et al., 2016), using standard human trajectory datasets used in the field. We also conduct a qualitative analysis on the pedestrian behaviors learned by the model.

1. Introduction

Human trajectory prediction is a challenging yet important task. It can be used to predict the behavior of humans in a space, which is utterly important in the design of buildings and other infrastructures. For example, the design of evacuation routes need to carefully consider human trajectories in a crowded scenario. Furthermore, unmanned vehicles could predict pedestrian trajectories in order to avoid colliding into them, causing casualties.

However, the task itself is difficult. This is because of the fact that humans are complex in nature. Any interaction with the surrounding environment could affect their trajectories. For this reason, accurate modeling of human trajectory is difficult.

Many non-data-driven methods have been developed to model human trajectory. The most common one being the social force model (Helbing & Molnar, 1995). This method uses parameters and human experience to model trajectories. Some parameters are still tuned with ground truth trajectory data, but the demand for data is not as much as the data-driven methods. When the deep learning era began, studies have started to look at using massive amount of trajectory data to train deep neural networks in order to predict the trajectories. The most notable deep learning approach is the Social-LSTM (Alahi et al., 2016). Following the success of Social-LSTM, other deep learning based approaches such as GAN has been adopted to achieve improved results (Sadeghian et al., 2019), (Gupta et al., 2018).

The most recent state-of-the-art approach is Social-STGCNN (Mohamed et al., 2020), which constructs graphs to model the states of human trajectories, and conducts 2D convolutions on both the spatial and temporal dimensions of this sequence of graphs. It is shown that the complex spatial dynamics humans in crowd can be represented with the graphs.

Motivated by the success of this method, we propose an alternate approach, Social Temporal Graph Convolutional Neural Network (Social-TGCN), to process the graph state to predict human trajectory. This method is inspired by T-GCN, a method used to forecast traffic speeds on road networks (Zhao et al., 2019). T-GCN combined a Graph Convolutional Network (GCN) and Gated Recurrent Unit (GRU) to learn spatial and temporal information of traffic speeds on a road network.

To evaluate whether combining GCN and GRU could be useful for capturing spatial and temporal dependence in the human trajectory prediction task, we perform a simple implementation of GCN and GRU combined that adapts to the human trajectory datasets evaluated in related works in the domain. Note that since our implementation has very basic network architectures and doesn't involve hyperparameter tuning, we do not expect that our results will achieve state-of-the-art, but our method is evaluated the same way as in other papers in the field, namely the naive linear model (Alahi et al., 2016), Social-LSTM (Alahi et al.,

[1]Department of Civil and Environmental Engineering, Stanford University, Stanford, USA. Correspondence to: Vivian Wen Hui Wong <vwwong3@stanford.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

2016), SoPhie (Sadeghian et al., 2019) and Social-STGCNN (Mohamed et al., 2020). We show that with our simple implementation, we can achieve an error rate comparable to some of these other pioneering works.

2. Related Works

2.1. Recurrent Neural Networks

Recurrent neural networks (RNNs) are a class of neural networks that allow previous outputs to be used as inputs, making it useful for sequential data (Mikolov et al., 2010). The most widely known application for RNNs is within natural language processing, as RNNs can be used to process sequences of words. RNNs have also been used for time series problems such as in the human trajectory prediction problem. For instance, Social-LSTM (Alahi et al., 2016) used Long Short-Term Memory units (LSTM), a type of RNN, to model each pedestrian in order to predict their motions.

Furthermore, Gated Recurrent Unit (GRU) is a special type of LSTM with fewer parameters. GRUs have shown its ability to learn temporal dependence in many applications, such as traffic flow prediction (Fu et al., 2016) and travel time prediction (Van Lint et al., 2002).

2.2. Graph Convolutional Networks

Graph convolutional networks (GCNs) are first introduced by (Kipf & Welling, 2016), which showed that GCNs is powerful in learning node representations aggregating neighboring node features in graph-structured data. GCNs have been extended and applied to several areas such as recommender systems (Ying et al., 2018), text classification (Yao et al., 2019) and knowledge graphs (Schlichtkrull et al., 2018). However, GCN alone cannot capture the dynamic change in graph data over time, hence posing the need to learn temporal dependence with the aid of GRU.

A research related to our work is the aforementioned T-GCN (Zhao et al., 2019) traffic speed data. Although basing off similar network design, the traffic speed data used in T-GCN is very different from existing human trajectory datasets: The traffic network topology is invariant, whereas with human trajectories, the number of nodes change as humans exit and enter the scene. Furthermore, human trajectories require a more complicated graph formulation with weighted edges and two-dimensional node features, whereas the traffic network can be modeled by a simple unweighted graph with single node features.

3. Problem Formulation

The pedestrian trajectory forecasting problem can be described as follows. Given a scene, as pedestrians move throughout the scene, the coordinates of each pedestrian in the scene at different time instants are observed. Given these temporal and spatial information of each pedestrian, the goal is to predict the spatial coordinates at future unobserved time instants. More formally, given a sequence of $(x_t^i, y_t^i) \forall t \in \{1, ..., T_{obs}\}$, we would like to predict the following sequence $(x_t^i, y_t^i) \forall t \in \{T_{obs}+1, ..., T_{pred}\}$ for the i^{th} person, where $i = \{1, ..., N\}$ when there are a total of N pedestrians in the scene, T_{obs} is last time instant observed, and T_{pred} is the last time instant of the prediction.

4. Social-TGCN

The Social-TGCN model can be separated into two modules, learning spatial and temporal features, respectively, to make a prediction. The first module is a GCN that embeds the spatial information of the trajectory prediction. The second module is the GRU, which uses the embedding from the GCN and solves a time series prediction problem.

Following the same assumption in (Alahi et al., 2016), we assume that pedestrian trajectories follow a bivariate Gaussian distribution. Using the observed locations and time instants as the input, for each $t \in \{T_{obs+1}, ..., T_{pred}\}$, Social-TGCN ouputs a five-dimensional vector, which are the parameters to a bivariate Gaussian distribution. The five parameters are $(\mu_x, \mu_y, \sigma_x, \sigma_y, \rho)_t$, where μ stands for the mean, σ stands for the standard deviation and ρ stands for the correlation between the two variables. The objective function to be minimized in model training is the negative log-likelihood, defined as:

$$L(\mathbf{W}) = -\sum_{t=T_{obs+1}}^{T_{pred}} \log(\mathbb{P}(x_t^i, y_t^i | \mu_x, \mu_y, \sigma_x, \sigma_y, \rho))$$

The structure of the entire workflow is presented in Figure 1.

4.1. Spatial Representation with GCN

The GCN is used to obtain a represent of the spatial structure. We first formulate the state at a given time instant as a graph. The GCN then performs graph convolution on the graph state to obtain node embeddings. In this section, we will describe the graph formulation as well as details of the GCN.

We use the same graph construction method as in (Mohamed et al., 2020): At any given time instant t, a graph $G_t = (V_t, E_t)$ can be constructed using the relative spatial coordinates, or the change in coordinates from the starting point of the trajectory, of the pedestrians in the scene, where V_t is the set of all nodes and E_t is the set of all edges. Each node $v_t^i \in V_t$ corresponds to the i^{th} pedestrian, whose relative spatial coordinates $(x_t^i - x_{t-1}^i, y_t^i - y_{t-1}^i)$ is the node feature. On the other hand, each weighted edge $e_t^{ij} \in E_t$ is

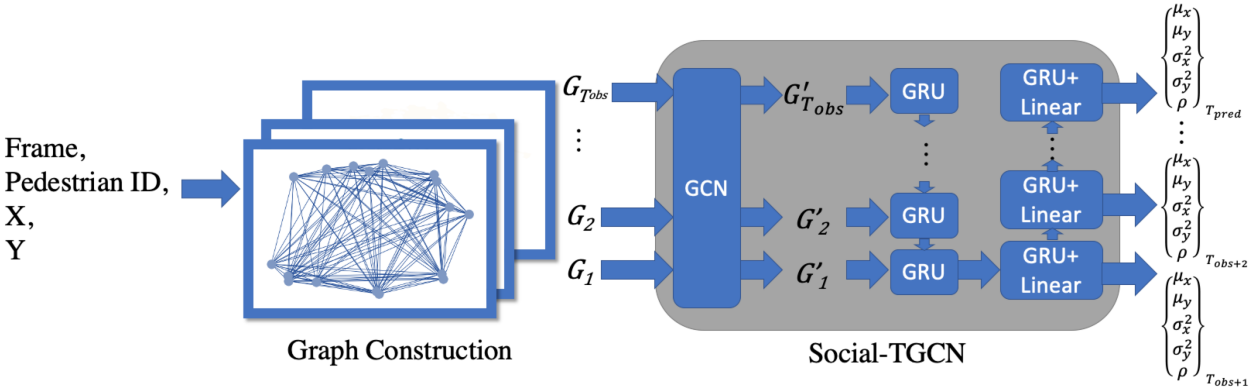


Figure 1. The workflow of Social-TGCN. We first construct a graph for every time instant capturing relative spatial locations contained in the raw data. The GCN then produces a node embedding for every graph. The GRU cells then treats the embeddings as a time series. A linear layer is used in the end to convert the GRU outputs to a 5-dimensional vector representing the predicted parameters of a bivariate Gaussian distribution.

$||v_t^i - v_t^j||_2^{-1}$ when $||v_t^i - v_t^j||_2 \neq 0$ and is 0 otherwise. The weight on the edge can be interpreted as how strongly the i^{th} and j^{th} pedestrians influence each other. If the relative changes in spatial locations of two pedestrians are similar, then the weight on their edge is large.

Graph convolution is performed on the observed graph states $G_t \forall t \in \{1, ..., T_{obs}\}$, outputting embedded node features, G'_t. To form a GCN, L layers of graph convolutional layers are stacked. A graph convolution layer of a GCN that processes G_t can be written as:

$$H^{(l+1)} = \sigma(\hat{D_t}^{-1/2} \hat{A_t} \hat{D_t}^{-1/2} H^{(l)} W^{(l)})$$

where $H^{(0)} = G_t$, $H^{(L)} = G'_t$, $\hat{A_t} = A_t + I$ with A_t being the adjacency matrix of G_t, $\hat{D_t}$ being the diagonal node degree matrix of $\hat{A_t}$, $W^{(l)}$ being the weight of layer l, and σ being a nonlinear activation function.

For derivations of the above-mentioned GCN propagation rules, we refer interested readers to (Kipf & Welling, 2016) for more details. The GCN parameters is shared across all $\{G_t | t \in \{1, ..., T_{obs}\}$. In the end, a sequence of $G'_1, ...G'_{T_{obs}}$ is obtained, which is a time series data.

4.2. time series Prediction with GRU

The GCN uses spatial information and aggregates information on the relative locations of pedestrians at the same time instant, but does not deal with information in the temporal dimension. On the other hand, GRU can be used to perform prediction on the time series data, where input sequence is of length T_{obs}, and the output sequence is of length $T_{pred} - T_{obs}$.

As shown in Figure 2, a single GRU cell consists of update gate u_t and relevance gate r_t. It takes hidden state of the previous GRU cell, h_{t-1} and an input from the time series, X_t and computes a hidden state h_t. Mathematically, we have

$$u_t = \sigma(W_u[X_t, h_{t-1}] + b_u)$$
$$r_t = \sigma(W_r[X_t, h_{t-1}] + b_r)$$
$$\tilde{h}_t = \tanh(W[r_t * h_{t-1}, x_t] + b)$$
$$h_t = (1 - u) * \tilde{h}_t + z * h$$

where W_u, W_r, W, b_u, b_r, b are learnable parameters. These parameters are shared across all GRU cells.

The first T_{obs} GRU cells take the node embeddings from the GCN module as the input $X_t = G'_t$. Another $T_{pred} - T_{obs} + 1$ GRU cells follow and take inputs as the hidden state output from the previous cell with a linear layer and activation function. The linear layer with the activation function is to transform the input to the same dimension as G'_t. The outputs of these cells are followed by a linear layer so that each of them output a vector $\in \mathbb{R}^5$, the parameters of the bivariate Gaussian distribution.

4.3. Implementation Details

Following the approach mentioned above, the Social-TGCN model used for experiments in this work is implemented according to the following: A simple one-layer GCN is used to learn the spatial information of $G_t = (V_t, E_t)$, the graph state at time t. No non-linear activation function is used. The weight in the GCN is a matrix $\in \mathbb{R}^{2\times2}$, thereby outputting node embeddings $\in \mathbb{R}^{|V_t|\times2}$. A GRU cell, on the other hand, outputs a hidden state $h_t \in \mathbb{R}^{64}$, which is linearly transformed to a vector $\in \mathbb{R}^5$ for the final output. The SGD optimizer and a learning rate of 0.1 is used.

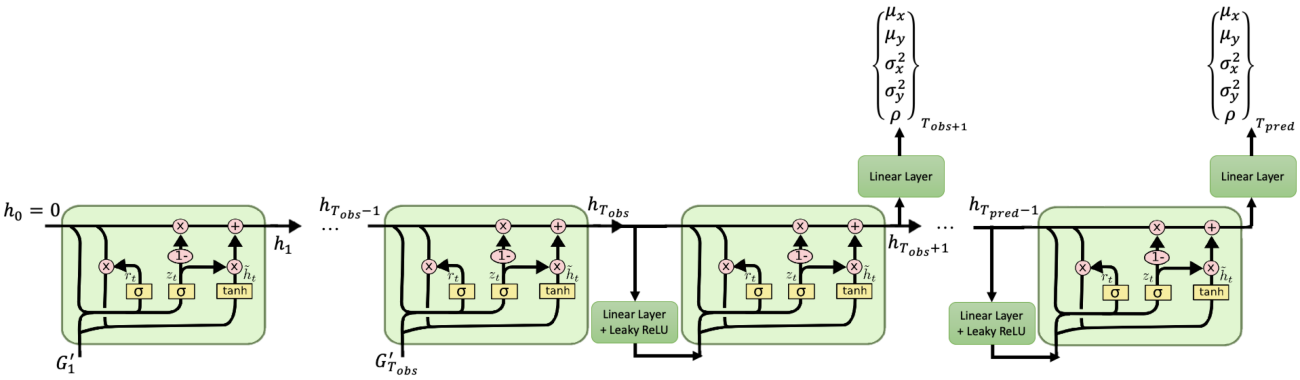


Figure 2. The architecture of the GRU part is shown. With the time series $G'_1, ...G'_{T_{obs}}$ as the input, the outputs are the bivariate Gaussian distribution parameters at each prediction time. The drawings of the GRU cells are obtained from (Rathor, 2018)

5. Experiments

5.1. Datasets and Evaluation Metrics

Two datasets are used in this work: the UCY dataset (Lerner et al., 2007) and the ETH dataset (Ess et al., 2008), which are the two datasets most commonly used for related works. The UCY dataset has three components: ZARA-01, ZARA-02 and University. The ETH dataset has two components: ETH and Hotel. In total, there are 5 sets of data, each corresponds to a different scene. We train a model on 4 sets of data and validate on the remaining set, repeating for all the 5 sets. For each data point, we observe 8 frames and predict the next 12 frames. This leave-one-out training and validating approach is chosen to be consistent with those adapted by other works in the domain. Similarly, the evaluation metrics are also chosen to be consistent with these works. Two evaluation metrics are used: Average Displacement Error (ADE) and Final Displacement Error (FDE):

- ADE - introduced in (Pellegrini et al., 2009), this is the mean squared error over all the predicted coordinates and the ground truth coordinates of a trajectory

- FDE - introduced in (Alahi et al., 2016), this is the average distance between the predicted and true coordinate at T_{pred}, or the final destination in a trajectory.

During test time, we sample 20 trajectories based on the predicted distribution and use the predicted trajectory closest to the ground truth to calculate ADE and FDE. This is the same evaluation method used by the works we compare to in the following section.

5.2. Results

5.2.1. Quantitative Evaluation

The performance of Social-TGCN is compared with other works in the field, with their ADEs and FDEs reported in (Mohamed et al., 2020) listed in Table 1.

Observing the average ADE and FDE across all datasets, the performance of our model is comparable with the Social-LSTM model and outperforms the naive linear model. Although our model does not outperform the more recent state-of-the-art methods, we show that the GCN+GRU combination has the potential to predict trajectories with reasonable accuracy.

We also note that our model is considerably smaller than most other methods. The parameters count of Social-LSTM, SGAN-P and Social-STGCNN is reported in Table 2. Since the GCN weights and GRU weights are all shared for all graph state inputs, the parameters count is invariant to the number of frames observed and predicted. The parameters count mainly depends on the number of features in the GCN's node embedding and the GRU's hidden size, which we showed that can be quite small but still yield reasonable results.

5.2.2. Qualitative Evaluation

In Figure 3, several trajectory predictions from the ETH dataset is shown. The 20 sample trajectories obtained from the predicted bivariate Gaussian distribution are plotted. We can observe that our model generally predicts trajectories going towards the right direction, and output similar predictions for pedestrians moving together in parallel. However, we note the following scenarios:

Parallel Behavior Pedestrians often cluster in pairs or groups of more and walk in parallel or have similar tra-

Table 1. The ADE and FDE of several models compared to Social-TGCN are presented. All methods use 8 frames to predict the trajectories of the next 12 frames. The lower the better. The models being compared to are: Naive linear model (Alahi et al., 2016), Social-LSTM (Alahi et al., 2016), Social GAN 20VP-20 (SGAN-P) (Gupta et al., 2018), SoPhie (Sadeghian et al., 2019), Social-STGCNN (Mohamed et al., 2020).

METRIC	DATASET	LINEAR	SOCIAL-LSTM	SGAN-P	SOPHIE	SOCIAL-STGCNN	SOCIAL-TGCN (OURS)
ADE	ETH	1.33	1.09	0.87	0.70	0.64	1.16
	HOTEL	0.39	0.79	0.67	0.76	0.49	0.76
	UNIV	0.82	0.67	0.76	0.54	0.44	0.74
	ZARA1	0.62	0.47	0.35	0.30	0.34	0.61
	ZARA2	0.77	0.56	0.42	0.38	0.30	0.61
	AVERAGE	0.79	0.72	0.61	0.54	0.44	0.77
FDE	ETH	2.94	2.35	1.62	1.43	1.11	1.76
	HOTEL	0.72	1.76	1.37	1.67	0.85	1.25
	UNIV	1.59	1.40	1.52	1.24	0.79	1.14
	ZARA1	1.21	1.00	0.68	0.63	0.53	1.03
	ZARA2	1.48	1.17	0.84	0.78	0.48	0.94
	AVERAGE	1.59	1.54	1.21	1.15	0.75	1.22

Table 2. The number of parameters of several models compared to Social-TGCN are shown. The lower the better.

MODEL	PARAMETERS COUNT
SOCIAL-LSTM	264K
SGAN-P	46.3K
SOCIAL-STGCNN	7.6K
SOCIAL-TGCN (OURS)	13.5K

jectories. From the parallel trajectories in the plots in Figure 3(a,b,c), we observe that the model's prediction generally maintains the parallel motion towards the future.

Leader-follower Behavior Another common behavior is when one or more pedestrians follow a leader in a group. 3(d) shows two pedestrians marked in orange and green following this behavior. Even though the green observed trajectory has a tendency to point downward, the prediction preserves the leader-follower behavior, making the green predictions motioning towards the same direction as the orange ones.

Change in Direction As expected, it is difficult to predict a sudden change in direction. In Figure 3(a), the pedestrian marked in blue has almost stationary observed data, but started motioning upward at some point. This causes the model to make a wrong prediction in the motion's direction. In Figure 3(d), we can see that the pedestrians made a sharp turn after the last observed frame. Our prediction assumes that the pedestrians are still going forward, when in reality they have turned upward. Nonlinear changes in directions is also difficult to predict. Figure 3(b) shows three pedestrians that meet each other, but their observed trajectories do not show this trend.

Collision Figure 3(c) shows an interesting scenario where there is potentially a collision expected between the pedestrian marked in green and the one in orange. The model seems to predict a yielding behavior to avoid the collision. This could possibly mean that the GCN module is successfully capturing spatial information of neighboring nodes. However, we also observe that the model generally lacks the ability to predict very long trajectories accurately, as seen also in Figure 3(d). More uncertainties exist at frames more distant in the future. This could mean that the GRU module, which is responsible for learning temporal features, needs further improvement to understand temporal dependence better.

6. Conclusion

In this work, we presented a model combining GCN and GRU to predict human trajectories learning from both spatial and temporal information. We model each time step of the trajectory prediction problem as a weighted graph that allows the GCN module to aggregate neighboring node features, and use the graph data from all time steps to form a time series data, allowing the GRU module to predict a probability distribution of future trajectories. Our proposed method performs at about the same error rate as the Social-LSTM model (Alahi et al., 2016) with a simple implementation and no hyperparameter tuning.

7. Future Work

One way to potentially improve the performance of this approach is to more carefully design the network architectures of the GCN and the GRU. Given the performance with simple architectures, the performance will likely improve with more finetuned architectures and hyperparameters. The learning rate of 0.1 also seems very large compared to most deep learning models. A reduction in the learning rate will likely improve robustness and convergence.

A possible future direction is to attempt the method proposed in (Pareja et al., 2019), which suggests giving GCN temporal information by useing GRU or LSTM cells to propagate the weights of the GCN (instead of the outputs of GCN, as in this work).

It would also be interesting to extend the Social-TGCN approach onto other datasets or other problems that require analysis in both spatial and temporal dimensions. For example, the model can be adapted to perform multi-class trajectory datasets such as the Stanford Drone Dataset (Robicquet et al., 2016).

References

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and 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.

Ess, A., Leibe, B., Schindler, K., , and van Gool, L. A mobile vision system for robust multi-person tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. IEEE Press, June 2008.

Fu, R., Zhang, Z., and Li, L. Using lstm and gru neural network methods for traffic flow prediction. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 324–328. IEEE, 2016.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2255–2264, 2018.

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

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Lerner, A., Chrysanthou, Y., and Lischinski, D. Crowds by example. *Computer Graphics Forum*, 26(3):655–664, 2007.

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

Mohamed, A., Qian, K., Elhoseiny, M., and Claudel, C. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. *arXiv preprint arXiv:2002.11927*, 2020.

Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., and Leisersen, C. E. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. *arXiv preprint arXiv:1902.10191*, 2019.

Pellegrini, S., Ess, A., Schindler, K., and Van Gool, L. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pp. 261–268. IEEE, 2009.

Rathor, S. Simple rnn vs gru vs lstm : Difference lies in more flexible control, Jun 2018. URL `https://medium.com`.

Robicquet, A., Sadeghian, A., Alahi, A., and Savarese, S. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision*, pp. 549–565. Springer, 2016.

Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, H., and Savarese, S. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1349–1358, 2019.

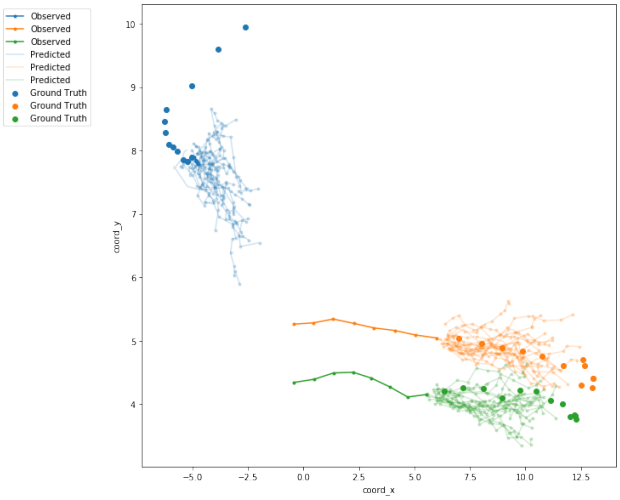
Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pp. 593–607. Springer, 2018.

Van Lint, J., Hoogendoorn, S., and van Zuylen, H. J. Freeway travel time prediction with state-space neural networks: modeling state-space dynamics with recurrent neural networks. *Transportation Research Record*, 1811 (1):30–39, 2002.

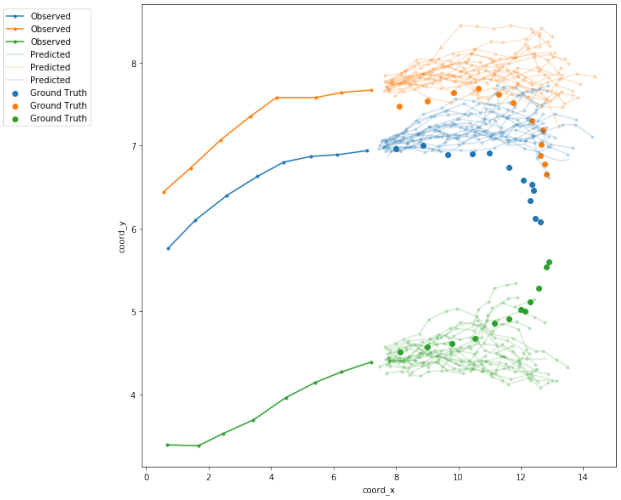
Yao, L., Mao, C., and Luo, Y. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 7370–7377, 2019.

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., and Leskovec, J. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 974–983, 2018.

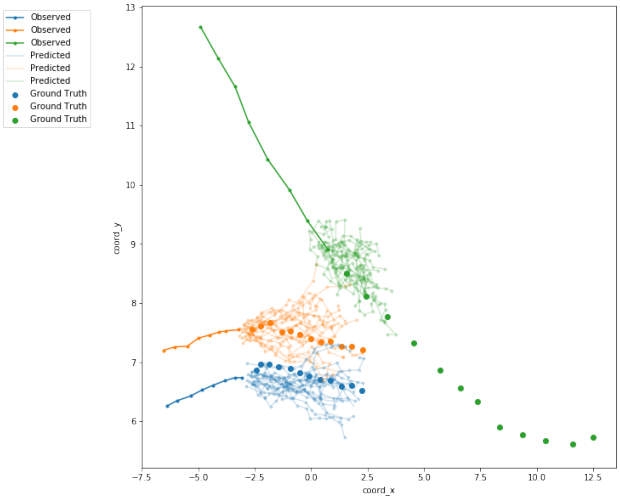
Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., and Li, H. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2019.



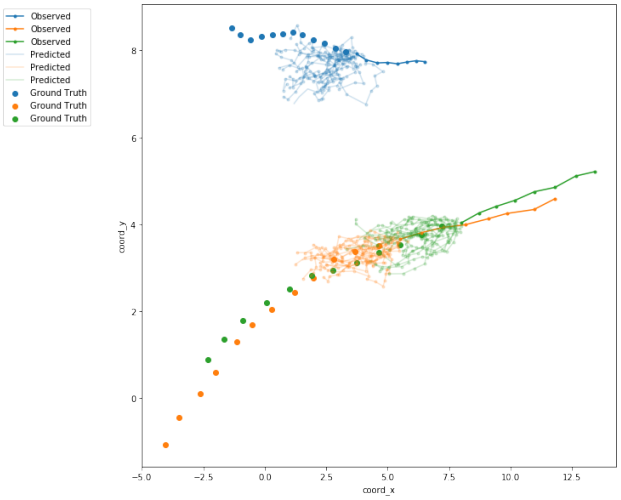
(a)



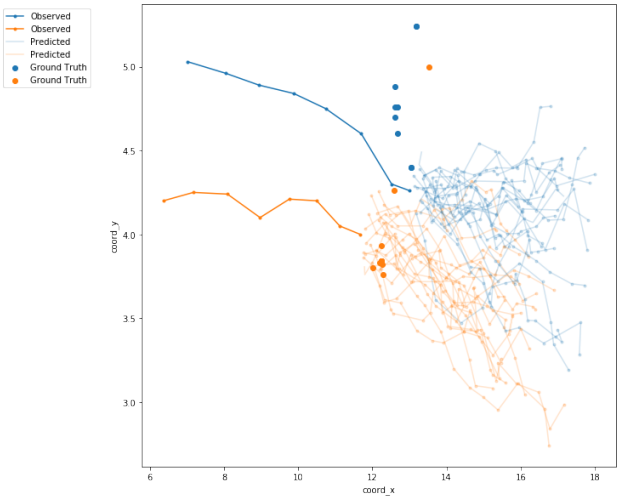
(b)



(c)



(d)



(e)

Figure 3. Observed trajectories, ground truth and the 20 sampled trajectories predicted with Social-TGCN. Results are on the ETH dataset(Ess et al., 2008).