Social-STGCNN: A Social Spatio-Temporal Graph Convolutional Neural Network for Human Trajectory Prediction

Abduallah Mohamed[1], Kun Qian[1]
Mohamed Elhoseiny[2,3, **], Christian Claudel[1, **]
[1]The University of Texas at Austin   [2]KAUST   [3]Stanford University
{abduallah.mohamed,kunqian,christian.claudel}@utexas.edu, mohamed.elhoseiny@kaust.edu.sa

Abstract

Better machine understanding of pedestrian behaviors enables faster progress in modeling interactions between agents such as autonomous vehicles and humans. Pedestrian trajectories are not only influenced by the pedestrian itself but also by interaction with surrounding objects. Previous methods modeled these interactions by using a variety of aggregation methods that integrate different learned pedestrians states. We propose the Social Spatio-Temporal Graph Convolutional Neural Network (Social-STGCNN), which substitutes the need of aggregation methods by modeling the interactions as a graph. Our results show an improvement over the state of art by 20% on the Final Displacement Error (FDE) and an improvement on the Average Displacement Error (ADE) with 8.5 times less parameters and up to 48 times faster inference speed than previously reported methods. In addition, our model is data efficient, and exceeds previous state of the art on the ADE metric with only 20% of the training data. We propose a kernel function to embed the social interactions between pedestrians within the adjacency matrix. Through qualitative analysis, we show that our model inherited social behaviors that can be expected between pedestrians trajectories. Code is available at `https://github.com/abduallahmohamed/Social-STGCNN`.

1. Introduction

Predicting pedestrian trajectories is of major importance for several applications including autonomous driving and surveillance systems. In autonomous driving, an accurate prediction of pedestrians trajectories enables the controller to plan ahead the motion of the vehicle in an adversarial environment. For example, it is a critical component for collision avoidance systems or emergency braking systems [2, 18, 16, 22]. In surveillance systems, forecasting
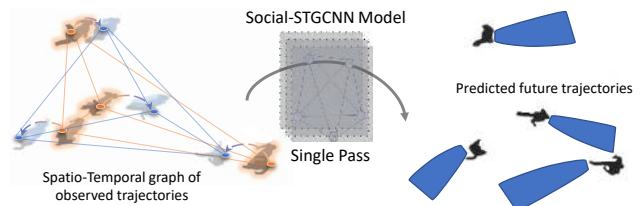

Figure 1. Pedestrian future trajectories prediction using the Social-STGCNN model. The social interactions between pedestrians and their temporal dynamics are represented by a spatio-temporal graph. We predict the future trajectories in a single pass.

pedestrian trajectories is critical in helping identifying suspicious activities [15, 28, 20].

The trajectory of a pedestrian is challenging to predict, due to the complex interactions between the pedestrian with the environment. Objects potentially influencing the trajectory of a pedestrian include physical obstacles such as trees or roads, and moving objects including vehicles and other pedestrians. According to [19], 70% of pedestrians tend to walk in groups. The interactions between pedestrians are mainly driven by common sense and social conventions. The complexity of pedestrian trajectory prediction comes from different social behaviors such as walking in parallel with others, within a group, collision avoidance and merging from different directions into a specific point. Another source of complexity is the randomness of the motion, given that the target destination and intended path of the pedestrian are unknown.

The social attributes of pedestrian motions encouraged researchers in this area to focus on inventing deep methods to model social interactions between pedestrians. In the Social-LSTM [1] article, deep learning based model is applied to predict the pedestrians trajectories by modeling each pedestrian trajectory via a recurrent deep model. The outputs of recurrent models are made to interact with each other via a pooling layer. Several articles [17, 14, 30] followed this direction. Social-LSTM [1] modeled the pedestrian trajec-

_____
** Equal advising.

tories as a bi-variate Gaussian distribution, while some of others aimed at predicting deterministic trajectories. Another direction is to use Generative Adversarial Networks (GANs) for this task, assuming that the distribution of trajectories is multi-modal. Several articles [6, 23, 13] used GANs to predict distributions of future trajectories. For these models, generators are designed using recurrent neural networks, and again, aggregation methods are relied upon to extract the social interactions between pedestrians. We argue that a limitation of earlier articles comes from the use of recurrent architectures, which are parameter inefficient and expensive in training [3]. We overcome this limitation through the use of temporal convolutional architectures.

In addition to the limitation of recurrent architectures, aggregation layers used in earlier works can also limit their performance. The aggregation layer takes the hidden states of the recurrent units as inputs. It is expected to assimilate a global representation of the scene, since each recurrent unit models a pedestrian trajectory. However, there are two issues within this type of aggregation. First, the aggregation in feature states is neither intuitive nor direct in modelling interactions between people, as the physical meaning of feature states is difficult to interpret. Second, since the aggregation mechanisms are usually based on heuristics like pooling, they could fail in modeling interactions between pedestrians correctly. For example, the pooling operation is known to be leaky in information [26]. In order to directly capture the interactions between pedestrians and predict future paths from these, the recent article social-BiGAT [10] relies on a graph representation to model social interactions. As the topology of graphs is a natural way to represent social interactions between pedestrians in a scene, we argue that it is a more direct, intuitive and efficient way to model pedestrians interactions than aggregation based methods. We also argue that social-BiGAT did not make the most of the graph representation, since they used it only as a pooling mechanism for recurrent units states. Social-STGCNN benefits more from graph representation through modeling the scene with as spatio-temporal graph and performs on it.

We designed Social-STGCNN to overcome the two aforementioned limitations. First, we model the pedestrians trajectories from the start as a spatio-temporal graph to replace the aggregation layers. The graph edges model the social interactions between the pedestrians. We propose a weighted adjacency matrix in which the kernel function quantitatively measure the influence between pedestrians. To address issues associated with recurrent units, our model manipulates over the spatio-temporal graph using a graph Convolutional Neural Networks (CNN)s and a temporal CNNs. This allows our model to predict the whole sequence in a single shot. Due to the above design, our model outperforms previous models in terms of prediction accuracy, parameters size, inference speed and data efficiency.

2. Related work

The recent interest in autonomous driving has lead to increasing focus on pedestrian trajectory prediction. Recently, new deep models are making promising progresses on this task. In this section, we give a brief review of related work. **Human trajectory prediction using deep models** Social-LSTM [1] is one of the earliest deep model focusing on pedestrian trajectory prediction. Social-LSTM uses a recurrent network to model the motion of each pedestrian, then they aggregated the recurrent outputs using a pooling mechanism and predict the trajectory afterwards. Social-LSTM assumes the pedestrian trajectory follow a bi-variate Gaussian distribution, in which we follow this assumption in our model. Later works such as Peek Into The Future (PIF) [14] and State-Refinement LSTM (SR-LSTM) [30] extends [1] with visual features and new pooling mechanisms to improve the prediction precision. It is noticeable that SR-LSTM [30] weighs the contribution of each pedestrian to others via a weighting mechanism. It is similar to the idea in Social-BiGAT [10] which uses an attention mechanism to weigh the contribution of the recurrent states that represent the trajectories of pedestrians. Based on the assumption that pedestrian trajectories follow multi-modal distributions, Social-GAN [6] extends Social LSTM [1] into a Recurrent Neural Network (RNN) based generative model. Sophie [23] used a CNNs to extract the features from the scene as a whole then a two way attention mechanism is used per pedestrian. Later on, Sophie concatenates the attention outputs with the visual CNN outputs then a Long Short Term Memory (LSTM) autoencoder based generative model is used to generate the future trajectories. The work CGNS [13] is similar to Sophie [23] in terms of the architecture but they used a Gated Recurrent Units(GRU)s instead of LSTMs. We notice that most previous works were circulating around two ideas, model each pedestrian motion using a recurrent net and combine the recurrent nets using a pooling mechanism. Recent work Social-BiGAT [10] relies on graph attention networks to model the social interactions between pedestrians. The LSTM outputs are fed to the graph in Social-BiGAT. One key difference between our model Social-STGCNN and Social-BiGAT is that we directly model pedestrian trajectories as a graph from the beginning, where we give meaningful values for vertices.

Recent Advancements in Graph CNNs Graph CNNs were introduced by [8] which extends the concept of CNNs into graphs. The Convolution operation defined over graphs is a weighted aggregation of target node attributes with the attributes of its neighbor nodes. It is similar to CNNs but the convolution operation is taken over the adjacency matrix of the graphs. The works [9, 4, 24] extend the graph CNNs to other applications such as matrix completion and Variational Auto Encoders. One of the development related to our work is the ST-GCNN [27]. ST-GCNN is a spatio-temporal Graph

CNN that was originally designed to solve skeleton-based action recognition problem. Even though the architecture itself was designed to work on a classification task, we adapt it to suit our problem. In our work, ST-GCNNs extract both spatial and temporal information from the graph creating a suitable embedding. We then operate on this embedding to predict the trajectories of pedestrians. Details are shown in section 4.

Temporal Convolutional Neural Networks (TCNs) Starting from [3], the argue between the usage of Recurrent Neural Networks (RNN)s versus the usage of temporal CNNs for sequential data modeling is highlighted. Introduced by [3], Temporal Convolutional Neural Networks(TCNs) take a stacked sequential data as input and predict a sequence as a whole. This could alleviate the problem of error accumulating in sequential predictions made by RNNs. What is more, TCNs are smaller in size compared to RNNs. We were inspired by TCNs and designed a temporal CNN model that extends the capabilities of ST-GCNNs. More details about this are in the model description section 4.

3. Problem Formulation

Given a set of N pedestrians in a scene with their corresponding observed positions $tr_o^n, n \in \{1, \ldots, N\}$ over a time period T_o, we need to predict the upcoming trajectories tr_p^n over a future time horizon T_p. For a pedestrian n, we write the corresponding trajectory to be predicted as $tr_p^n = \{ \mathbf{p}_t^n = (\mathbf{x}_t^n, \mathbf{y}_t^n) \mid t \in \{1, \ldots, T_p\}\}$, where $(\mathbf{x}_t^n, \mathbf{y}_t^n)$ are random variables describing the probability distribution of the location of pedestrian n at time t, in the 2D space. We make the assumption that $(\mathbf{x}_t^n, \mathbf{y}_t^n)$ follows bi-variate Gaussian distribution such that $\mathbf{p}_t^n \sim \mathcal{N}(\mu_t^n, \sigma_t^n, \rho_t^n)$. Besides, we denote the predicted trajectory as $\hat{\mathbf{p}}_t^n$ which follows the estimated bi-variate distribution $\mathcal{N}(\hat{\mu}_t^n, \hat{\sigma}_t^n, \hat{\rho}_t^n)$. Our model is trained to minimize the negative log-likelihood, which defined as:

$$L^n(\mathbf{W}) = -\sum_{t=1}^{T_p} \log(\mathbb{P}((\mathbf{p}_t^n | \hat{\mu}_t^n, \hat{\sigma}_t^n, \hat{\rho}_t^n)) \qquad (1)$$

in which $\mathbf{W}$ includes all the trainable parameters of the model, μ_t^n is the mean of the distribution, σ_t^n is the variances and ρ_t^n is the correlation.

4. The Social-STGCNN Model

4.1. Model Description

The Social-STGCNN model consists of two main parts: the Spatio-Temporal Graph Convolution Neural Network (ST-GCNN) and the Time-Extrapolator Convolution Neural Network (TXP-CNN). The ST-GCNN conducts spatio-temporal convolution operations on the graph representation of pedestrian trajectories to extract features. These features

are a compact representation of the observed pedestrian trajectory history. TXP-CNN takes these features as inputs and predicts the future trajectories of all pedestrians as a whole. We use the name Time-Extrapolator because TXP-CNNs are expected to extrapolate future trajectories through convolution operation. Figure 2 illustrates the overview of the model.

Graph Representation of Pedestrian Trajectories We first introduce the construction of the graph representation of pedestrian trajectories. We start by constructing a set of spatial graphs G_t representing the relative locations of pedestrians in a scene at each time step t. G_t is defined as $G_t = (V_t, E_t)$, where $V_t = \{v_t^i \mid \forall i \in \{1, \ldots, N\}\}$ is the set of vertices of the graph G_t. The observed location (x_t^i, y_t^i) is the attribute of v_t^i. E_t is the set of edges within graph G_t which is expressed as $E_t = \{e_t^{ij} \mid \forall i, j \in \{1, \ldots, N\}\}$. $e_t^{ij} = 1$ if v_t^i and v_t^j are connected, $e_t^{ij} = 0$ otherwise. In order to model how strongly two nodes could influence with each other, we attach a value a_t^{ij}, which is computed by some kernel function for each e_t^{ij}. a_t^{ij}s are organized into the weighted adjacency matrix A_t. We introduce $a_{sim,t}^{ij}$ as a kernel function to be used within the adjacency matrix A_t. $a_{sim,t}^{ij}$ is defined in equation 2. We discuss the details of A_t kernel function later in section 6.1.

$$a_{sim,t}^{ij} = \begin{cases} 1/\|v_t^i - v_t^j\|_2 & , \|v_t^i - v_t^j\|_2 \neq 0 \\ 0 & , \text{Otherwise.} \end{cases} \qquad (2)$$

Graph Convolution Neural Network With the graph representation of pedestrian trajectories, we introduce the spatial convolution operation defined on graphs. For convolution operations defined on 2D grid maps or feature maps, the convolution operation is shown in equation 3.

$$z^{(l+1)} = \sigma(\sum_{h=1}^{k} \sum_{w=1}^{k} (p(z^{(l)}, h, w)).\mathbf{w}^{(l)}(h, w)) \qquad (3)$$

where k is the kernel size and $p(.)$ is the sampling function which aggregates the information of neighbors centering around z [5] and σ is an activation function and (l) indicates layer l.

The graph convolution operation is defined as:

$$v^{i(l+1)} = \sigma(\frac{1}{\Omega} \sum_{v^{j(l)} \in B(v^{i(l)})} p(v^{i(l)}, v^{j(l)}).\mathbf{w}(v^{i(l)}, v^{j(l)}))$$
$$(4)$$

where $\frac{1}{\Omega}$ is a normalization term, $B(v^i) = \{v^j | d(v^i, v^j) \leq D\}$ is the neighbor set of vertices v^i and $d(v^i, v^j)$ denotes the shortest path connecting v^i and v^j. Note that Ω is the cardinality of the neighbor set. Interested readers are referred to [8, 27] for more detailed explanations and reasoning.
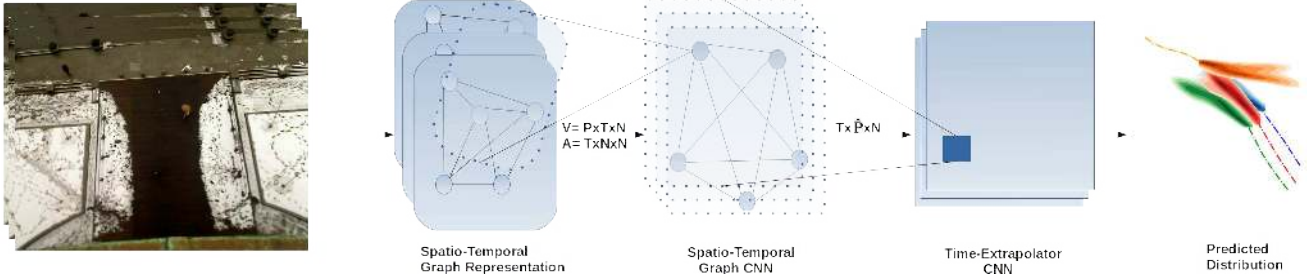
Figure 2. The Social-STGCNN Model. Given T frames, we construct the spatio-temporal graph representing $G = (V, A)$. Then G is forwarded through the Spatio-Temporal Graph Convolution Neural Networks (ST-GCNNs) creating a spatio-temporal embedding. Following this, the TXP-CNNs predicts future trajectories. P is the dimension of pedestrian position, N is the number of pedestrians, T is the number of time steps and $\hat{P}$ is the dimensions of the embedding coming from ST-GCNN.

Spatio-Temporal Graph Convolution Neural Network (ST-GCNNs) ST-GCNNs extends spatial graph convolution to spatio-temporal graph convolution by defining a new graph G whose attributes are the set of the attributes of G_t. G incorporates the spatio-temporal information of pedestrian trajectories. It is worth noticing that the topology of $G_1, \ldots, G_T$ is the same, while different attributes are assigned to v_t^i when t varies. Thus, we define G as (V, E), in which $V = \{v^i \mid i \in \{1, \ldots, N\}\}$ and $E = \{e^{ij} \mid \forall i, j \in \{1, \ldots, N\}\}$. The attributes of vertex v^i in G is the set of $v_t^i, \forall t \in \{0, \ldots, T\}$. In addition, the weighted adjacency matrix A corresponding to G is the set of $\{A_1, \ldots, A_T\}$. We denote the embedding resulting from ST-GCNN as $\bar{V}$.

Time-Extrapolator Convolution Neural Network (TXP-CNN) The functionality of ST-GCNN is to extract spatio-temporal node embedding from the input graph. However, our objective is to predict further steps in the future. We also aim to be a stateless system and here where the TXP-CNN comes to play. TXP-CNN operates directly on the temporal dimension of the graph embedding $\bar{V}$ and expands it as a necessity for prediction. Because TXP-CNN depends on convolution operations on feature space, it is less in parameters size compared to recurrent units. A property to note regards TXP-CNN layer that it is not a permutation invariant as changes in the graph embedding right before TXP-CNN leads to different results. Other than this, if the order of pedestrians is permutated starting from the input to Social-STGCNN then the predictions are invariant.

Overall, there are two main differences between Social-STGCNN and ST-GCNN [27]. First, Social-STGCNN constructs the graph in a totally different way from ST-GCNN with a novel kernel function. Second, beyond the spatio-temporal graph convolution layers, we added the flexibility in manipulating the time dimension using the TXP-CNN. ST-GCNN was originally designed for classification. By using TXP-CNN, our model was able to utilize the graph embedding originating from ST-GCNN to predict the futuree

trajectories.

4.2. Implementing Social-STGCNN

Several steps are necessary to implement the model correctly. We first normalize the adjacency matrix for the ease of learning. The adjacency matrix A is a stack of $\{A_1, \ldots, A_T\}$, we symmetrically normalize each A_t using the following form [8]

$$A_t = \Lambda_t^{-\frac{1}{2}} \hat{A}_t \Lambda_t^{-\frac{1}{2}}$$

where $\hat{A}_t = A_t + I$ and Λ_t is the diagonal node degree matrix of $\hat{A}_t$. We use $\hat{A}$ and Λ to denote the stack of $\hat{A}_t$ and Λ_t respectively. The normalization of adjacency is essential for the graph CNN to work properly, as outlined in [8]. We denote the vertices values at time step t and network layer l as $V_t^{(l)}$. Suppose $V^{(l)}$ is the stack of $V_t^{(l)}$. With the above definitions, we can now implement the ST-GCNN layers defined in equation 4 as follows.:

$$f(V^{(l)}, A) = \sigma(\Lambda^{-\frac{1}{2}} \hat{A} \Lambda^{-\frac{1}{2}} V^{(l)} \mathbf{W}^{(l)}) \qquad (5)$$

where $\mathbf{W}^{(l)}$ is the matrix of trainable parameters at layer l.

After applying the ST-GCNN, we have features that compactly represent the graph. The TXP-CNN receives features $\bar{V}$ and treats the time dimension as feature channels. The TXP-CNN is made up of a series of residual connected CNNs. Only the first layer in TXP-CNN does not have a residual connection as it receives $\bar{V}$ from the ST-GCNNs, in which they differ in terms of the dimensions of the observed samples and the samples to be predicted.

5. Datasets and Evaluation Metrics

The model is trained on two human trajectory prediction datasets: ETH [21] and UCY [11]. ETH contains two scenes named ETH and HOTEL, while UCY contains three scenes named ZARA1, ZARA2 and UNIV. The trajectories in datasets are sampled every 0.4 seconds. Our method of

training follows the same strategy as Social-LSTM [1]. In Social-LSTM, the model was trained on a portion of a specific dataset and tested against the rest and validated versus the other four datasets. When being evaluated, the model observes the trajectory of 3.2 seconds which corresponds to 8 frames and predicts the trajectories for the next 4.8 seconds that are 12 frames.

Two metrics are used to evaluate model performance: the Average Displacement Error (ADE) [21] defined in equation 6 and the Final Displacement Error (FDE) [1] defined in equation 7. Intuitively, ADE measures the average prediction performance along the trajectory, while the FDE considers only the prediction precision at the end points. Since Social-STGCNN generates a bi-variate Gaussian distribution as the prediction, to compare a distribution with a certain target value, we follow the evaluation method used in Social-LSTM [1] in which 20 samples are generated based on the predicted distribution. Then the ADE and FDE are computed using the closest sample to the ground truth. This method of evaluation were adapted by several works such as Social-GAN [6] and many more.

$$\text{ADE} = \frac{\sum\limits_{n \in N} \sum\limits_{t \in T_p} \|\hat{p}_t^n - p_t^n\|_2}{N \times T_p} \tag{6}$$

$$\text{FDE} = \frac{\sum\limits_{n \in N} \|\hat{p}_t^n - p_t^n\|_2}{N}, t = T_p \tag{7}$$

6. Experiments and Results Analysis

Model configuration and training setup Social-STGCNN is composed of a series of ST-GCNN layers followed by TXP-CNN layers. We use PReLU[7] as the activation function σ across our model. We set a training batch size of 128 and the model was trained for 250 epochs using Stochastic Gradient Descent (SGD). The initial learning rate is 0.01, and changed to 0.002 after 150 epochs. According to our ablation study in table 6, the best model to use has one ST-GCNN layer and five TXP-CNN layers. Furthermore, it is noticeable that when the number of ST-GCNN layers increases, the model performance decreases. Apparently, this problem of going deep using graph CNN was noticed by the work in [12], in which they proposed a method to solve it. Unfortunately, their solution does not extend to temporal graphs.

6.1. Ablation Study of Kernel Function

In this section, our objective is to find a suitable kernel function to construct the weighted adjacency matrix. The weighted adjacency matrix A_t is a representation of the graph edges attributes. The kernel function maps attributes at v_t^i and v_t^j to a value a_t^{ij} attached to e_t^{ij}. In the implementation of Social-STGCNN , A_t weights the vertices contributions to each other in the convolution operations. The kernel function can thus be considered as a prior knowledge about the social relations between pedestrians. A straightforward idea in designing the kernel function is to use the distance measured by the L_2 norm defined in equation 8 between pedestrians to model their impacts to each other. However, this is against the intuition that the pedestrians tend to be influenced more by closer ones. To overcome this, we use similarity measure between the pedestrians. One of the proposals is to use the inverse of L_2 norm as defined in equation 10. The ϵ term is added in denominator to ensure numerical stability. Another candidate function is the Gaussian Radial Basis Function [25], shown in equation 9. We compare the performance of these kernel functions through experiments. The case that all the values in adjacency matrix between different nodes are set to one is used as a baseline.

According to results listed in table 6.1, the best performance comes from $a_{sim,t}^{ij}$ defined in function 2. The difference between functions 10 and 2 exists in the case where $\|v_t^i - v_t^j\|_2 = 0$. In function 2, we set $a_{sim,t}^{ij} = 0$ when $\|v_t^i - v_t^j\|_2 = 0$ because it is assumed that the two pedestrians can be viewed as the same person when they stay together. Without it, the model will have an ambiguity in the relationship between pedestrians. For this, we use $a_{sim,t}^{ij}$ in the definition of the adjacency matrix in all of our experiments.

$$a_{L_2,t}^{ij} = \|v_t^i - v_t^j\|_2 \tag{8}$$

$$a_{exp,t}^{ij} = \frac{\exp\left(-\|v_t^i - v_t^j\|_2\right)}{\sigma} \tag{9}$$

$$a_{sim_\epsilon,t}^{ij} = \frac{1}{\|v_t^i - v_t^j\|_2 + \epsilon} \tag{10}$$

6.2. Quantitative Analysis

The performance of Social-STGCNN is compared with other models on ADE/FDE metrics in table 2. Overall,

	1	3	5	7
1	0.47 / 0.78	0.47 / 0.84	**0.44 / 0.75**	0.48 / 0.87
3	0.59 / 1.02	0.52 / 0.92	0.54 / 0.93	0.54 / 0.92
5	0.62 / 1.07	0.57 / 0.98	0.59 / 1.02	0.59 / 0.98
7	0.75 / 1.28	0.75 / 1.27	0.62 / 1.07	0.75 /1.28

Table 1. Ablation study of the Social-STGCNN model. The first row corresponds to the number of TXP-CNN layers. The first column from the left corresponds to the number of ST-GCNN layers. We show the effect of different configurations of Social-STGCNN on the ADE/FDE metric. The best setting is to use one layer for ST-GCNN and five layers for TXP-CNN.

	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Linear * [1]	1.33 / 2.94	0.39 / 0.72	0.82 / 1.59	0.62 / 1.21	0.77 / 1.48	0.79 / 1.59
SR-LSTM-2 * [30]	0.63 / 1.25	0.37 / 0.74	0.51 / 1.10	0.41 / 0.90	0.32 / 0.70	0.45 / 0.94
S-LSTM [1]	1.09 / 2.35	0.79 / 1.76	0.67 / 1.40	0.47 / 1.00	0.56 / 1.17	0.72 / 1.54
S-GAN-P [6]	0.87 / 1.62	0.67 / 1.37	0.76 / 1.52	0.35 / 0.68	0.42 / 0.84	0.61 / 1.21
SoPhie [23]	0.70 / 1.43	0.76 / 1.67	0.54 / 1.24	0.30 / 0.63	0.38 / 0.78	0.54 / 1.15
CGNS [13]	**0.62** / 1.40	0.70 / 0.93	0.48 / 1.22	0.32 / 0.59	0.35 / 0.71	0.49 / 0.97
PIF [14]	0.73 / 1.65	**0.30 / 0.59**	0.60 / 1.27	0.38 / 0.81	0.31 / 0.68	0.46 / 1.00
STSGN [29]	0.75 / 1.63	0.63 / 1.01	0.48 / 1.08	0.30 / 0.65	**0.26** / 0.57	0.48 / 0.99
GAT [10]	0.68 / 1.29	0.68 / 1.40	0.57 / 1.29	**0.29** / 0.60	0.37 / 0.75	0.52 / 1.07
Social-BiGAT [10]	0.69 / 1.29	0.49 / 1.01	0.55 / 1.32	0.30 / 0.62	0.36 / 0.75	0.48 / 1.00
Social-STGCNN	0.64 / **1.11**	0.49 / 0.85	**0.44 / 0.79**	0.34 / **0.53**	0.30 / **0.48**	**0.44 / 0.75**

Table 2. ADE / FDE metrics for several methods compared to Social-STGCNN are shown. The models with * mark are non-probabilistic. The rest of models used the best amongst 20 samples for evaluation. All models takes as an input 8 frames and predicts the next 12 frames. We notice that Social-STGCNN have the best average error on both ADE and FDE metrics. The lower the better.
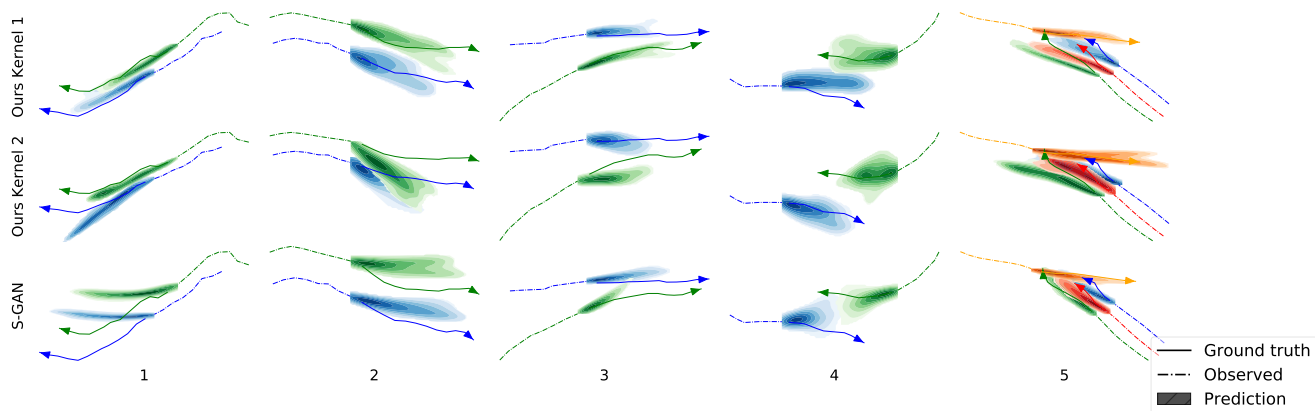

Figure 3. Qualitative analysis of Social-STGCNN . We compare models trained with different kernel functions (Kernel 1: equation 8 and Kernel 2: equation 2) versus previous models. Social-GAN [6] is taken as a baseline for the comparison. Illustration scenes are from the ETH [21] and UCY [11] datasets. We used the pre-trained Social-GAN model provided by [6]. A variety of scenarios are shown: two individuals walking in parallel (1)(2), two persons meeting from the same direction (3), two persons meeting from different directions (4) and one individual meeting another group of pedestrians from an angle (5). For each case, the dashed line is the true trajectory that the pedestrians are taking and the color density is the predicted trajectory distribution.

Social-STGCNN outperforms all previous methods on the two metrics. The previous state of art on the FDE metric is SR-LSTM [30] with an error of 0.94. Our model has an error of 0.75 on the FDE metric which is about 20% less than the state of the art. The results in qualitative analysis explains how Social-STGCNN encourages social behaviors that enhanced the FDE metric. For the ADE metric, Social-STGCNN is slightly better than the state-of-art SR-LSTM by 2%. Also, it is better than the previous generative methods with an improvement ranging in between 63% compared to S-LSTM [1] and 4% compared to PIF [14]. Interestingly, our model without the vision signal that contains scene context outperforms methods that utilized it such as SR-LSTM, PIF and Sophie.

Inference speed and model size S-GAN-P [6] previously had the smallest model size with 46.3k parameters. The size of Social-STGCNN is 7.6K parameters only which is about one sixth of the number of parameters in S-GAN-P. In terms of inference speed, S-GAN-P was previously the fastest method with an inference time of 0.0968 seconds per inference step. The inference time of our model is 0.002 seconds per inference step which is about 48 × faster than S-GAN-P. Table 6 lists out the speed comparisons between our model and publicly available models which we could bench-mark against. We achieved these results because we overcame the two limitations of previous methods which used recurrent architecture and aggregation mechanisms via the design of our model.

Data Efficiency In this section, we evaluate if the efficiency in model size leads to a better efficiency in learning from
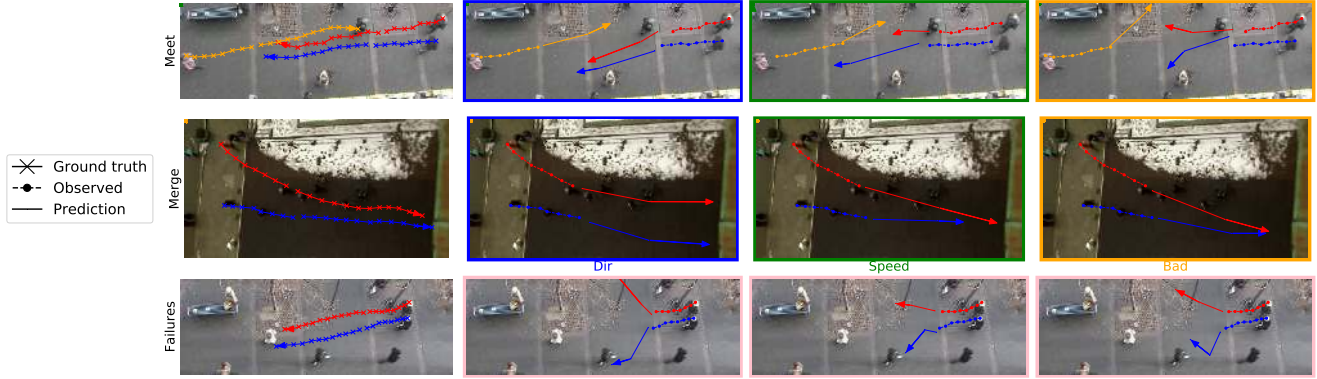
Figure 4. The first column is the ground truth, while the other columns illustrate samples from our model. The first two rows show two different scenarios where pedestrians merge into a direction or meet from opposite directions. The second and third columns show changes in speed or direction in samples from our model. The last column shows undesired behaviors. The last row show failed samples.

	Parameters count	Inference time
S-LSTM [1]	264K (35x)	1.1789 (589x)
SR-LSTM-2 [30]	64.9K (8.5x)	0.1578 (78.9x)
S-GAN-P [6]	46.3K (6.1x)	0.0968 (48.4x)
PIF [14]	360.3K (47x)	0.1145 (57.3x)
Social-STGCNN	**7.6K**	**0.0020**

Table 3. Parameters size and inference time of different models compared to ours. The lower the better. Models were bench-marked using Nvidia GTX1080Ti GPU. The inference time is the average of several single inference steps. We notice that Social-STGCNN has the least parameters size compared and the least inference time compared to others. The text in blue show how many times our model is faster than others.

shows the data learning efficiency experiments results with mean and error. We notice that our model exceeds the state of the art on the FDE metric when only 20% of training data is used. Also, Social-STGCNN exceeds the performance of Social-GAN on the ADE metric when trained only on with 20% of the training data. The results also show that S-GAN-P did not improve much in performance with more training data, unlike the present model. It is an interesting phenomenon that S-GAN-P does not absorb more training data. We assume that this behavior is due to the fact that GANs are data efficient because they can learn a distribution from few training samples. However, the training of GANs can easily fall into the problem of mode collapse. In comparison, the data efficiency of our model comes from the parameter efficiency.
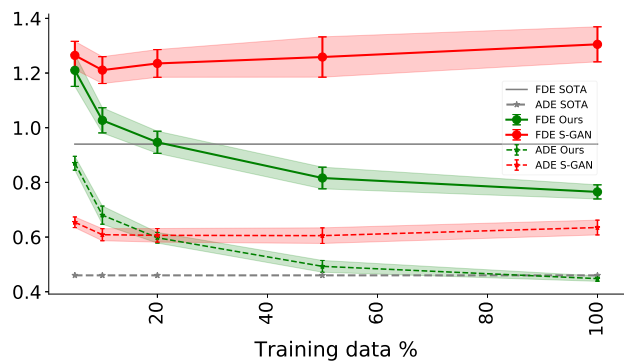
Kernel function	ADE / FDE
$a^{ij}_{L_2,t}$	0.48 / 0.84
$a^{ij}_{exp,t}$	0.50 / 0.84
$a^{ij}_{sim_\epsilon,t}$	0.48 / 0.88
Just ones	0.49 / 0.79
$a^{ij}_{sim,t}$	**0.44 / 0.75**

Table 4. The effect of different kernel functions for the adjacency matrix A_t over the Social-STGCNN performance.

Figure 5. Model performance versus shrinked training dataset. The x-axis shows several randomly samples shrink percentages. The shade represents errors. The same shrinked data were used across the models. The figure shows our performance versus Social-GAN which is the closest model in terms of parameter size to ours.

fewer samples of the data. We ran a series of experiments where 5%, 10%, 20% and 50% of the training data. The training data were randomly selected. Once selected, we fed the same data to train different models. Social-GAN is employed as a comparison baseline because it has least trainable parameters amongst previous deep models. Figure 6.2

6.3. Qualitative Analysis

The quantitative analysis section shows that Social-STGCNN outperforms previous state-of-art in terms of ADE/FDE metrics. We now qualitatively analyze how Social-STGCNN captures the social interactions between pedestrians and takes that into consideration when predicting the distributions. We show cases in which Social-STGCNN successfully predicts collision free trajectories between pedestrians coming from different angles, maintains parallel walking, and correctly predicts the outcome of situations where person meets with a group of pedestrians.

We qualitatively compare the prediction results between Social-GAN [6], Social-STGCNN with L_2 norm (equation 8) as the kernel function and Social-STGCNN with inverse L_2 norm (equation 2) as the kernel function.

Parallel walking In scenarios one and two in figure 3, two pedestrians are walking in parallel. Usually, when people are walking in parallel they are tightly connected to each other and their momentum will be preserved in the future. The predictions by Social-STGCNN and Social-GAN all show that these two pedestrians will keep walking in parallel in the future. However, the predicted density by Social-STGCNN closely matches with the ground truth trajectory unlike the deviation we see in Social-GAN.

Using our proposed kernel function $a_{sim,t}$ defined in equation 2 for weighted adjacency matrix helps us model the social influences between pedestrians better than using the regular L_2 norm kernel function defined in equation 8. It is shown in scenes one and two that the model with $a_{sim,t}$ preforms much better in maintaining the relative location between people walking side by side. In scene five, similar behavior is observed.

Collision avoidance Scenario three and Scenario four in figure 3 are scenarios in which two pedestrians are heading towards similar or opposite directions. A collision could happen if they maintain their momentum. In scenario 3, two pedestrians are walking towards a similar direction. The forecast by Social-GAN acts linearly based on the momentum of the pedestrians and may lead to a collision. In the forecast of Social-STGCNN , we notice that the trajectories are adjusted slightly such that they both avoid collision and align well with the observed momentum of pedestrians. As a result, Social-STGCNN matches better with ground truth. In scenario four, Social-GAN fails to avoid the collision, while ours shows a realistic collision free path prediction.

Individual meeting a group A more complex scenario is case five in figure 3, in which one person meets a group of parallel walking individuals. Our model suggests that the group of people still walk in parallel while adjusting their heading direction to avoid collision. In this case, although neither our model nor Social-GAN capture the ground truth trajectory very well, the predicted distribution by our model still makes sense from the social interaction point of view.

Diversity in samples and social behaviors In order to understand in detail how Social-STGCNN generates samples, we plot the samples generated from predicted bi-variate Gaussian distributions. There are two different scenarios in figure 4. In the first scene, three people meet from opposite directions. In the other scene, two people merge at an angle. Several patterns of samples could be generated by the predicted distributions. In column two in figure 4, the generated samples adjusts the advancing direction to avoid possible collisions in both scenes. Another social attribute of pedestrians is to slow down or accelerate to avoid crash. Samples in the third column in figure 4 capture this attribute. This analysis shows that our samples encode different expected social behaviors of pedestrians. However, some samples show undesired behaviors such as collision or divergence in the last column. More cases of these undesired behaviors are in the last row of figure4.

7. Conclusion

In this article, we showed that a proper graph-based spatio-temporal setup for pedestrian trajectory prediction improves over previous methods on several key aspects, including prediction error, computational time and number of parameters. By applying a specific kernel function in the weighted adjacency matrix together with our model design, Social-STGCNN outperforms state-of-art models over a number of publicly available datasets. We also showed that our configuration results in a data-efficient model and can learn from few data samples. We also qualitatively analyze the performance of Social-STGCNN under situations such as collision avoidance, parallel walking and individual meeting a group. In these situations, Social-STGCNN tend to provide more realistic path forecasts than several other reported methods. Furthermore, Social-STGCNN is extremely efficient computationally, dividing the number of required parameters by a factor of 8.5, and boosting the inference speed by up to 48 × comparing to previous models. In the future, we intend to extend Social-STGCNN to multi-modal settings that involve other moving objects including bicycles, cars and pedestrians.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[2] Haoyu Bai, Shaojun Cai, Nan Ye, David Hsu, and Wee Sun Lee. Intention-aware online pomdp planning for autonomous

driving in a crowd. In *2015 ieee international conference on robotics and automation (icra)*, pages 454–460. IEEE, 2015.

[3] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[4] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[5] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[6] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[8] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[9] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[10] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, S Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *arXiv preprint arXiv:1907.03395*, 2019.

[11] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007.

[12] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9267–9276, 2019.

[13] Jiachen Li, Hengbo Ma, and Masayoshi Tomizuka. Conditional generative neural system for probabilistic trajectory prediction. *arXiv preprint arXiv:1905.01631*, 2019.

[14] Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G Hauptmann, and Li Fei-Fei. Peeking into the future: Predicting future person activities and locations in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5725–5734, 2019.

[15] Matthias Luber, Johannes A Stork, Gian Diego Tipaldi, and Kai O Arras. People tracking with human motion predictions from social forces. In *2010 IEEE International Conference on Robotics and Automation*, pages 464–469. IEEE, 2010.

[16] Yuanfu Luo, Panpan Cai, Aniket Bera, David Hsu, Wee Sun Lee, and Dinesh Manocha. Porca: Modeling and planning for autonomous driving among many pedestrians. *IEEE Robotics and Automation Letters*, 3(4):3418–3425, 2018.

[17] Huynh Manh and Gita Alaghband. Scene-lstm: A model for human trajectory prediction. *arXiv preprint arXiv:1808.04018*, 2018.

[18] Kohei Morotomi, Masayuki Katoh, and Hideaki Hayashi. Collision position predicting device, Sept. 30 2014. US Patent 8,849,558.

[19] Mehdi Moussaïd, Niriaska Perozo, Simon Garnier, Dirk Helbing, and Guy Theraulaz. The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PloS one*, 5(4):e10047, 2010.

[20] Basam Musleh, Fernando García, Javier Otamendi, José Mª Armingol, and Arturo De la Escalera. Identifying and tracking pedestrians based on sensor fusion and motion stability predictions. *Sensors*, 10(9):8028–8053, 2010.

[21] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268. IEEE, 2009.

[22] Pongsathorn Raksincharoensak, Takahiro Hasegawa, and Masao Nagai. Motion planning and control of autonomous driving intelligence system based on risk potential optimization framework. *International Journal of Automotive Engineering*, 7(AVEC14):53–60, 2016.

[23] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. 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*, pages 1349–1358, 2019.

[24] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[25] Jean-Philippe Vert, Koji Tsuda, and Bernhard Schölkopf. A primer on kernel methods. *Kernel methods in computational biology*, 47:35–70, 2004.

[26] Travis Williams and Robert Li. Wavelet pooling for convolutional neural networks. In *International Conference on Learning Representations*, 2018.

[27] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[28] Masahiro Yasuno, Noboru Yasuda, and Masayoshi Aoki. Pedestrian detection and tracking in far infrared images. In *2004 Conference on Computer Vision and Pattern Recognition Workshop*, pages 125–125. IEEE, 2004.

[29] Lidan Zhang, Qi She, and Ping Guo. Stochastic trajectory prediction with social graph network. *arXiv preprint arXiv:1907.10233*, 2019.

[30] Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12085–12094, 2019.