The Journal of Supercomputing (2023) 79:20923–20937
https://doi.org/10.1007/s11227-023-05467-x

STM-GCN: a spatiotemporal multi-graph convolutional network for pedestrian trajectory prediction

Taki Youssef[1] · **Elmoukhtar Zemmouri**[1] · **Anas Bouzid**[1]

Accepted: 31 May 2023 / Published online: 17 June 2023

Abstract

Pedestrian trajectory prediction has many real-world applications, such as crowd video surveillance and self-driving cars. However, this is a challenging problem due to the complexity of modeling social interactions between road agents (pedestrians and vehicles). Previous studies that addressed trajectory prediction applied traditional deep learning approaches, including CNN and RNN. Meanwhile, the applications of graph neural networks have drawn increasing interest recently, and significant progress has been made in extracting features from complex and unstructured data. In this paper, we propose STM-GCN, a spatiotemporal multi-graph convolutional network for pedestrian trajectory prediction. Our approach consists of collecting information about the social interactions between pedestrians in a crowd that are position-based and velocity-based interactions integrated into a multi-graph. Then we apply Graph Neural Network learning on the obtained multi-graph for predictions. Through experiments on the ETH and UCY benchmarking datasets, our proposed method outperforms the state of the art by 6% average displacement error (ADE) and 10% final displacement error (FDE). Our implementation is available at https://github.com/YoussefTaki/STMGCN.

Keywords Trajectory prediction · Multi-graph · Graph convolutional network

✉ Taki Youssef
youssefTaki1301@gmail.com

Elmoukhtar Zemmouri
e.zemmouri@umi.ac.ma

Anas Bouzid
bouzid.anas.1@gmail.com

[1] Moulay Ismail University, ENSAM, Meknes, Morocco

1 Introduction

Human trajectory prediction aims to predict the future trajectories of pedestrians based on historical observations. It is a challenging problem having many applications in daily life, such as in self-driving cars [1] and surveillance systems [2]. In self-driving, one of the critical tasks for an advanced driver assistance system (ADAS) is pedestrian trajectory prediction because it is a thin key for the safe navigation of autonomous vehicles on urban roads with crowded pedestrians. For this reason, it has been extensively studied for years, from the practical gaussian regression method [3] to the spatiotemporal graph convolutional neural networks [6]. However, the complexity of modeling social interactions, the numerous influencing factors that can influence pedestrian decisions, and the infinite paths that pedestrians can take, all these factors make trajectory prediction a major challenge. The social interactions between pedestrians are primarily influenced by common sense and social norms. Predicting a pedestrian's path is complicated by various social behaviors such as walking in parallel to others, within a group, collision avoidance, and merging from different directions at a given point [6].

The recent interest in self-driving cars has raised expectations, particularly in trajectory prediction problem. Especially after the failure of old methods to effectively model human social interactions, which encouraged researchers in this field to focus on investing deep learning methods in modeling these interactions between pedestrians, and this is related to several indicators, including the distance between pedestrians, their velocities and directions.

The first model applied in predicting pedestrian paths based on deep learning was the recurrent neural network, and many articles have followed this trend. The most important was the work of Alexandre Alahi et al. [4] called the Social-LSTM. The RNN-based methods have made promising progress in pedestrian trajectory prediction task. But these methods have a lot of parameters which makes them very expensive to train [5]. Besides the fact that RNN cannot model interactions well due to the nature of the data and the use of pooling layers which have two issues. Firstly, the physical meaning of feature states is difficult to interpret, and the aggregation in feature states is neither intuitive nor direct in modeling interactions between pedestrians. Secondly, the aggregation mechanisms are typically based on heuristics such as pooling, and they may fail to accurately model pedestrian interactions [6]. Therefore, many researchers have applied GNN for its effectiveness in handling complex and unstructured data and extracting the various patterns that may exist. In addition, it is not as time-consuming as RNN-based methods. One of the works that has used graphs is STGCNN on which our work is based. STGCNN is a spatiotemporal graph convolutional neural network that extracts spatiotemporal information from the graph to create the appropriate embedding for each graph node (graph nodes are the pedestrians, as we will illustrate later) [6]. However, STGCNN model uses a simple graph representing the relative locations of pedestrians.

In this paper, we propose a new framework for effective human trajectory prediction called STM-GCN, which is a spatiotemporal multi-graph convolutional

network. Our focus is on multi-graphs, which are graphs that can have multiple edges. For example, two pedestrians can be connected by edges of different types (distance, speed, direction, etc.). This enables us to model the spatiotemporal structures inherent in a real scene. We can predict more accurate trajectories by utilizing multiple relationships, which is difficult to achieve with a single relationship.

The main contributions of the paper as well as the novelties of our approach compared to previous ones that used GNN are summarized as follows:

1. We propose STM-GCN, a spatiotemporal multi-graph convolution network for pedestrian trajectory prediction. The essence of the method is to build a multi-graph adjacency matrix by joining two or more simple graphs adjacency matrices, each of which is built based on a different relationship between pedestrians in the scene.
2. Experiments with the ETH [7] and UCY [8] benchmarking datasets show that STM-GCN significantly improves pedestrian trajectory prediction, with 6% in terms of the average displacement error (ADE) and 10% in terms of the final displacement error (FDE).

These results spur further work on multiple graphs on this challenging task, especially with the use of other datasets that contain other information and relationships.

The remainder of the paper is structured as follows: Sect. 2 provides an overview of related work. Section 3 discusses the problem formulation. Section 4 then defines and presents the proposed prediction model. Section 5 presents experimental results and comparisons with state of the art methods on the ETH and UCY pedestrian trajectory prediction datasets. It summarizes and concludes with some final remarks in the last section.

2 Related work

The problem of predicting pedestrian trajectory has received great attention for years. This section provides a high-level overview of the key methods and techniques used to address this issue and its various developments.

Early works that addressed this problem, including the Bayesian formulation [9], the Monte Carlo simulation [10], hidden Markov models [11], and kalman filters, [12] do not model interactions between pedestrians. Hence, these methods are less accurate than the ones that represent interactions by the social forces model [13, 14]. This model represents each pedestrian as a particle subject to forces from nearby people and obstacles. The sum of these forces influences and may determine the pedestrian's future position.

However, because they use a handcrafted function, these physics-based models can only represent a few possible paths which makes them poorly performing in the trajectory prediction task [35]. For this reason, the need arose for deep learning models that do not have this limitation.

Deep learning methods have made promising progress in a wide range of tasks, particularly in computer vision with convolutional neural networks (CNN) and natural language processing (NLP) with recurrent neural networks (RNN). This type of neural network is frequently used to model problems concerning sequence data. Long short-term memory networks, or "LSTMs," are a type of RNN that can learn long-term dependencies. Hochreiter and Schmidhuber [14] pioneered their use.

One of the first deep learning models used for pedestrian trajectory prediction was Social-LSTM [4]. Social-LSTM models each pedestrian's motion using a recurrent network and then aggregates the recurrent outputs using a pooling mechanism to predict the future trajectory. The LSTM is fed a pedestrian trajectory as well as social information used to model social interactions in this model. The social LSTM is extended into an RNN-based generative model in Social-GAN [15].

The VP-LSTM method [16] predicts pedestrian trajectories by using separate LSTM networks and social pooling layers on hidden states for heterogeneous agents. Later works continued to use social interactions, but also used more advanced techniques, such as attention mechanism. One example is the CIDNN method [17], which applied attention mechanism to hidden states of LSTM networks to model agent interactions. Due to its recurrent nature, RNN, with or without attention, is significantly more computationally expensive than other approaches. GANs (generative adversarial networks) are a method for creating new synthetic data, such as training data, which has been shown to improve pedestrian path prediction accuracy [18].

The field of graph representation learning has advanced significantly over the last years and can be roughly split into three generations: traditional graph embedding, modern graph embedding, and deep learning on graphs, also known as graph neural networks (GNNs) [36]. The previously discussed methods show limited ability in modeling interactions between pedestrians. Hence, some recent works applied GNN learning on the pedestrian trajectory prediction problem. Graph convolutional networks were introduced by Kipf and Welling [19] extending the concept of CNN to graphs that can do what convolutional neural network failed to do by encoding the trajectory in a representational graph structure to handle social interactions with implied connections.

In the context of trajectory prediction, a GNN is used to describe pedestrians and their interactions using a graph: pedestrians are represented as the graph nodes, while their interactions are represented as the graph edges. Vemula et al. [20] was among the first works to adapt GNN to pedestrian trajectory prediction, followed by others such as [21].

In sequential data modeling problems such as trajectory prediction, many methods [21, 22] used temporal CNN. the difference between RNN and temporal CNN is temporary convolutional neural networks (TCN), which begin taking stacked sequential data as input and predict the sequence. Unlike RNNs, this can eliminate the problem of error accumulation in RNN sequential predictions. Furthermore, TCNs are smaller and lighter than RNNs [6]. Another hardly used method for dealing with this problem is to use convolutional neural networks (CNNs) to predict pedestrian paths. Recent research has shown that CNN-based

methods can perform similarly to RNNs in sequential prediction while using significantly less computational resources [23, 24].

Recently, researchers have relied on hybrid approaches based on deep learning, which consist of networks that integrate several deep learning architectures (CNNs, GANs, and LSTMs). Each framework has benefits, and the advantages of individual architectures together can be used in many tasks, such as working with LSTMs and CNNs to traffic prediction [25]. As well, the STGAT method [26] that also uses LSTM to capture each pedestrian's path information before applying the GAT to calculate the weighted graph.

The ST-GCNN [26] is one of the works on which we have relied in this paper. ST-GCNN is a spatiotemporal graph of a CNN that was initially designed to solve the problem of skeleton-based action recognition in which Mohamed et al. [6] adapted to fit our problem.

ST-GCNN collects both spatial and temporal information from a graph in order to create an appropriate embeddings, these embeddings are then used to predict pedestrians' trajectories [6]. We try to expand this approach by creating a multi-graph to encode the different interactions and interest between pedestrians, by creating the largest number of graphs that depend on the number of relationships to be worked on. In our case, we can work by three relations (position, velocity, and direction), and these three relationships control the trajectory of the person, as well as other factors that are not present in the datasets, we are working on but in our experiment, we only used two relationships and got good results.

3 Method

3.1 Problem formulation

We can state the problem of trajectory prediction as follows: given a set of observed positions of pedestrians, how to predict their next positions with the minimum possible errors?

Formally, we are given a group of N pedestrians in a 2D scene with their corresponding observed coordinates over a period T_{obs}, the goal of pedestrian trajectory prediction is to predict the future path of each pedestrian in that scene within a period T_{pred}. The history trajectories of all pedestrians in an observed scene at time step t are given as: $P^t = \{(x_1^t, y_1^t), (x_2^t, y_2^t), \ldots, (x_N^t, y_N^t)\}$. The future predicted trajectories of these pedestrians are given then as: $\hat{P}^t = \{(\hat{x}_1^t, \hat{y}_1^t), (\hat{x}_2^t, \hat{y}_2^t), \ldots, (\hat{x}_N^t, \hat{y}_N^t)\}$ where $t \in \{T_{obs}, \ldots, T_{pred}\}$.

We assume a regular discretization of time. In the case of ETH and UCY datasets, the duration between two successive frames is $\tau = 0.4$ s. We assume also that all (x, y) coordinates are relative to the same and arbitrary fixed origin.

Generally $T_{obs} = 8$ time steps and $T_{pred} = 12$ time steps. So the prediction process for a given pedestrian is as follows: The input of the model will be 8 past positions path of that pedestrian at a given time step t:

$$P = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6), (x_7, y_7), (x_8, y_8)\}$$

and the model will predict the next 12 positions for that pedestrian in the next 12 time steps:

$$\hat{P} = \{(\hat{x}_1, \hat{y}_1), (\hat{x}_2, \hat{y}_2), (\hat{x}_3, \hat{y}_3), (\hat{x}_4, \hat{y}_4), (\hat{x}_5, \hat{y}_5), (\hat{x}_6, \hat{y}_6), (\hat{x}_7, y_7), (\hat{x}_8, \hat{y}_8), (\hat{x}_9, \hat{y}_9), (\hat{x}_{10}, \hat{y}_{10}), (\hat{x}_{11}, \hat{y}_{11}), (\hat{x}_{12}, \hat{y}_{12})\}$$

In the probabilistic models, the (x_n^t, y_n^t) are random variables describing the probability distribution of the position of pedestrian n at time t. We assume that (x_n^t, y_n^t) follows a bi-variate Gaussian distribution such that $P_n^t \sim N(\mu_n^t, \sigma_n^t, \rho_n^t)$. Besides, we denote the predicted trajectory as $\hat{P}_n^t$ which follows the estimated bi-variate distribution $\hat{P}_n^t \sim N(\hat{\mu}_n^t, \hat{\sigma}_n^t, \hat{\rho}_n^t)$.

According to the bivariate Gaussian distribution assumption, the loss function is defined as follows:

$$L^n = -\sum_{t=1}^{T_P} Log((P_n^t | \hat{\mu}_n^t, \hat{\sigma}_n^t, \hat{\rho}_n^t)))$$

where $\hat{\mu}_n^t$ is the mean, $\hat{\sigma}_n^t$ is the variances and $\hat{\rho}_n^t$ is the correlation of the distribution.

3.2 Model description

The proposed STM-GCN model is made up of three major components:

1. The multi-graph representing interactions between pedestrians.
2. The spatiotemporal graph convolution neural network (ST-GCNN).
3. The time-extrapolator convolution neural network (TXP-CNN).

To extract features, the ST-GCNN performs spatiotemporal convolutions on the multi-graph representation of pedestrian trajectories. These extracted features provide a concise representation of the observed pedestrian trajectory history.

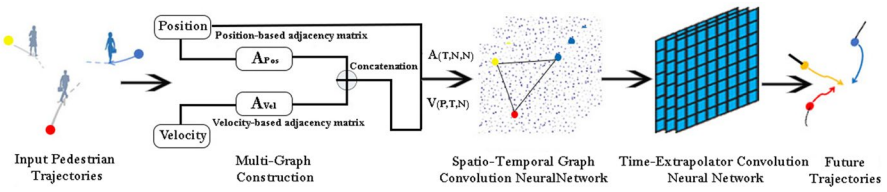

Fig. 1 General overview of the STM-GCN model. For a sequence of T_{obs} frames with N pedestrians, we create a position-based adjacency matrix A_{pos} using the relative coordinates, and a velocity-based adjacency matrix A_{vel} using the real coordinates of pedestrians. These two adjacency matrices are then combined using element-wise sum fusion to define a multi-graph with adjacency matrix A. The multi-graph is then passed through the spatiotemporal graph convolution neural network (ST-GCNN) to generate spatiotemporal embeddings, which are used by the TXP-CNN to predict future trajectories. P is the dimension of the pedestrian position, N is the number of pedestrians, and T is the time step count

TXP-CNN uses the outputs of the ST-GCNN phase as inputs to predict all pedestrians' future trajectories. Figure 1 depicts a general overview of the proposed model.

3.3 Multi-graph construction

We consider $G = (V, E)$ an undirected multi-graph, where $V = \{p_i | i \in \{1, \ldots, N\}\}$ represents the set of N pedestrians. Vertices can indeed store variations of information such as position, velocity, and direction. We define the set of edges E of the graph G using multiple adjacency matrices, each representing a particular interaction between pedestrians (vertices). To simulate how two humans can impact each other, we assign a value $A_{i,j}$ to each $e_{i,j} \in E$ that is computed by some kernel function. We note that the graph G will be created for each time step t.

In our model, we used two types of interactions: the first interaction is position-based, and the second is velocity-based.

3.3.1 Position-based interactions

We define the position-based interactions using the relative positions of pedestrians to each other. This representation aggregates the geometric interconnections of the scene by assuming that the motion of a pedestrian is more affected by nearby neighbors. As in [6], to begin, we create a set of spatial graphs G^t_{pos} representing the relative positions of pedestrians in a scene at each time step t. G^t_{pos} is defined as $G^t_{pos} = (V^t, E^t_{pos})$ where $V^t = V = \{p_i \mid \text{i} \in \{1,...,\text{N}\}\}$ is the set of pedestrians at time step t, and E^t_{pos} is the set of edges defined by the adjacency matrix A^t_{pos}.

To formulate the position-based interactions between pedestrians and so how strongly two pedestrians influence each other at time step t, we use the following kernel function:

$$A^t_{pos}(i,j) = \begin{cases} \dfrac{1}{\|P^t_i - P^t_j\|_2} & if \quad P^t_i \neq P^t_j \\ 0 & otherwise \end{cases} \tag{1}$$

where i and j are indices of two pedestrians in the observed scene; $P^t_i = (x^t_i, y^t_i)$ is the position of pedestrian i at time step t; $P^t_j = (x^t_j, y^t_j)$ is the position of pedestrian j at time step t; $\| \cdot \|_2$ is the L^2 norm.

Figure 2 shows an example of a real-world scene from the ETH dataset with the obtained position-based graph G^t_{pos}. The size of edges in this graph is proportional to position-based interactions between represented pedestrians.

3.3.2 Velocity-based interactions

We define the velocity-based interactions using the instant velocity of pedestrians in a scene at time step t. Pedestrian velocity is valuable information for knowing whether or not a group of pedestrians are walking together, and it can also tell us if two pedestrians

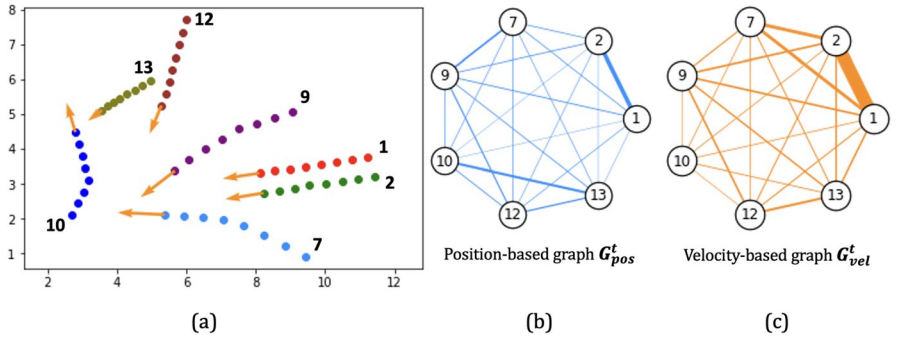

Fig. 2 Illustration of multi-graph construction. **a** depicts a real-world scene from ETH dataset with 7 selected pedestrians. The points represent their positions, and the vectors represent their instantaneous velocities. **b** is the obtained position-based graph using Eq. (1). **c** is the obtained velocity-based graph using Eq. (2). In the two graphs, the sizes of edges are proportional to their weights

will collide at a specific point in the future. This feature is very important in predicting the future trajectories of pedestrians. So we use this feature to construct our second graph $G^t_{vel} = (V^t, E^t_{vel})$ where $V^t = V = \{p_i | i \in \{1, \ldots, N\}\}$ is the set of pedestrians at time step t, E^t_{vel} is the set of edges that represent the velocity correlations between graph nodes and are defined by the adjacency matrix A^t_{vel}.

The kernel function we used to formulate the velocity-based interactions is as follows:

$$A^t_{vel}(i,j) = \begin{cases} \dfrac{1}{\|v^t_i - v^t_j\|_2} & if \quad v^t_i \neq v^t_j, Or \quad P^t_i \neq P^t_j \\ 0 & otherwise \end{cases} \tag{2}$$

where $v^t_i = (x^t_i - x^{t-1}_i, y^t_i - y^{t-1}_i)/\tau$ is the instant velocity of pedestrian i; $v^t_j = (x^t_j - x^{t-1}_j, y^t_j - y^{t-1}_j)/\tau$ is the instant velocity of pedestrian j;

Figure 2 illustrates the obtained velocity-based graph G^t_{vel} from an example scene of the ETH dataset.

3.4 Spatiotemporal graph convolution neural network

After the graphs construction stage, a spatiotemporal graph convolution is used to integrate the graph representation of pedestrians using the spatiotemporal convolution process described in [6]. This operation generalizes the array-to-graph convolution process by using a kernel to aggregate features across a graph's local neighborhood in both spatiotemporal dimensions, similar to Li et al. [18] and Rainbow et al. [27]. We also normalize the adjacency matrix A for each time step t, as in previous work [18], to allow for accurate feature extraction:

$$\hat{A}^t = (D^t)^{-1/2}(A^t + I)(D^t)^{-1/2} \tag{3}$$

where D^t denotes as the diagonal node degree of $(A^t + \mathrm{I})$ at time step t and I is the identity matrix. $\hat{A}^t$ presents the normalized weight adjacency matrix at time t. This normalization step ensures the adjacency to be positive semi-definite for spectral decomposition in the GCN [28]. The message passing process for each layer of ST-GCNN is performed using: $H^{l+1}=\sigma(\hat{A}\ H^l\ W^l)$, where $\hat{A}$ denote the concatenation of the $\hat{A}^t$ along the time dimension, for all $t \in T_{obs}$, and W^l is weight matrix for the l^{th} layer and σ is the activation function.

3.5 Time extrapolator convolution neural network

The functionality of the ST-GCNN is to extract spatiotemporal nodes (that are pedestrians) embeddings from the input multi-graph. However, our objective is to predict further steps in the future for each pedestrian. The goal of the time extrapolator convolution neural network (TXP-CNN) is to decode the future trajectory by performing time convolutions on the final embedded output (past trajectory) from the ST-GCNN using temporal convolution neural networks (TCNs) inspired from [5].

In the TXP-CNN phase, we stack five convolutional layers with kernel sizes (3×1) and PReLU activation function is performed along every convolutional layer with each layer connected residually to its previous layer that is usually adopted to boost the network capacity [22]. Next, a convolutional layer with a kernel size of (3×1) is used to produce the output feature T_{out}.

4 Experiments

4.1 Datasets

There are many datasets designed for the human trajectory prediction problem, but the most common and benchmarking datasets are:

1. ETH [7] dataset contains two scenes: ETH-Univ is taken from a university, and ETH-Hotel is taken from a city street.
2. UCY [8] dataset includes three urban scenes UCY-Zara01, UCY-Zara01, and UCY-Hotel.

The two datasets are publicly available and widely used in literature. Concatenated, they contain more than 1600 pedestrian trajectories that are sampled every 0.4 s.

Our proposed model is trained on these two human trajectory prediction datasets. In order to make a fair comparison with existing methods, our method follows the experimental setups that Social-LSTM [4] and other works do. In Social-LSTM, the model was trained on a subset of a specific dataset, then tested and validated against the remaining datasets. When evaluated, the model observes the

3.2 s trajectory, which corresponds to 8 frames, and predicts the 4.8 s trajectory, which corresponds to 12 frames [6].

4.2 Implementation details and evaluation metrics

Following the same implementation details of Mohamed et al. [6], the models are trained in the PyTorch deep learning framework. We used PReLU [31] as the activation function across our model, and the stochastic gradient descent (SGD) algorithm is used as the optimizer. The model is trained for 250 epochs with a batch size of 128. The initial learning rate is set to $\alpha = 0.01$ and the decay is set to 0.002 after 150 epochs. STM-GCN is composed of a series of ST-GCNN layers followed by TXP-CNN layers, according to our experiments, the best model to use has one ST-GCNN layer and five TXP-CNN layers.

Like previous works [6, 15, 22, 29], we used average displacement error (ADE) [30] defined in Eq. 4, and final displacement error (FDE) [4] defined in Eq. 5, to evaluate our model:

$$ADE = \frac{\sum_{n \in N} \sum_{t \in T_{obs}} \| \hat{P}^t_n - P^t_n \|_2}{N * T_{obs}} \tag{4}$$

$$FDE = \frac{\sum_{n \in N} \| \hat{P}^{T_{obs}}_n - P^{T_{obs}}_n \|_2}{N * T_{}} \tag{5}$$

4.3 Results analysis

We have conducted several experiments to validate our proposed model, using the two real datasets we mentioned above, we compared the performance of our model on the ADE/FDE measures in Table 1 with several baseline models.

STM-GCN outperforms all previous methods on both metrics. The previous state of the art on the FDE scale is Social-STGCNN [6] with an error of 0.75. Our model has an error of 0.67 in termes of the FDE scale which is about 10% lower than the Social-STGCNN. For the ADE scale, STM-GCN is 6% better than the Social-STGCNN.

4.4 Data efficiency

In order to test if the model size effectiveness leads to better learning effectiveness from very few sample data, we conducted a series of experiments for which 5, 10, 20, and 50% of the training data have been used. The training data was selected at random. After this selection step, we used the same data to train different models.

The results of mean and error data learning efficiency experiments are depicted in Fig. 3. Even when only 20% of the training data are used, our model outperforms the state of the art in terms of the FDE, outperforming Social-STGCNN

Table 1 Experimental results comparison of several methods to our proposed model STM-GCN are shown

Model	Category	Type	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
LINEAR [4]	CNN	Deterministic	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 [22]	RNN	Deterministic	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 [4]	RNN	Probabilistic	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 [15]	RNN	Probabilistic	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 [32]	RNN	Probabilistic	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 [34]	GNN	Probabilistic	**0.62**/1.40	0.70/0.93	0.48/1.22	**0.32**/0.59	0.35/0.71	0.49/0.97
STSGN [21]	GNN	Probabilistic	0.75/1.63	0.63/1.01	0.48/1.08	0.30/0.65	**0.26**/0.57	0.48/0.99
Social-BiGAT [21]	GNN	Probabilistic	0.69/1.29	0.49/1.01	0.55/1.32	030/0.62	0.36/0.75	0.48/1.00
Next [33]	GNN	Probabilistic	0.73/7.65	0.30/0.59	0.60/1.27	0.38/0.81	0.31/0.68	0.46/1.00
STGAT [26]	GNN	Probabilistic	0.70/1.35	0.37/0.67	0.59/1.23	0.35/0.69	0.31/0.64	0.47/0.92
Social-STGCNN [6]	GNN	Probabilistic	0.64/**1.11**	0.49/0.85	**0.44/0.79**	0.34 /**0.53**	0.30/0.48	0.44/0.75
STM-GCN	GNN	Probabilistic	0.73/1.13	**0.31/0.42**	0.45/0.85	0.33/**0.53**	0.29/**0.46**	**0.42/0.67**

All models take as an input 8 frames and predict the next 12 frames. Results are in the format ADE/FDE and the best results are in bold

[6] and S-GAN-P [15]. Furthermore, on the ADE scale, STM-GCN outperforms Social-STGCNN when only 20% of the training data is used. The results also show that S-GAN-P did not improve significantly with more training data when compared to current GCNN models. It is an intriguing phenomenon that S-GAN-P does not absorb more training data. We believe this behavior is due to GANs' data efficiency, as they can learn the distribution from a small number of training samples. GANs for training, on the other hand, are prone to mode breakdown. Our model data's efficiency, on the other hand, stems from parameter efficiency (see Fig. 3).

4.5 Parameters count and inference time

The size of the model STGCNN that we used as a starting point of our approach is 7.6 K parameters. After using the multi graph application, our model's parameters

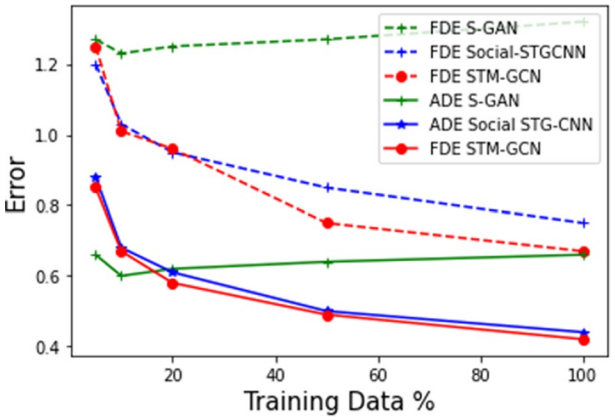

Fig. 3 Model effectiveness against a contracted training dataset. The *x*-axis depicts several randomly sampled training data in percentages, the lines represent the errors. The models all used the same shrinkage data. The figure illustrates the effectiveness of our STM-GCN compared to other methods

Table 2 Parameters count and inference time of previous models compared to ours

Model	Parameters count (K)	Inference time	Paper Refs.
S-LSTM	264	1.1789	[4]
SR-LSTM	64.9	0.1578	[22]
Social-STGCNN	7.6	0.0020	[6]
STM-GCN	10.1	0.0033	

The inference time is measured in seconds per time step. The duration of a time step is 0.4 s, which mean that our model can be used in real-time applications

count is around 10 K but still a small number if we compare it to the number of parameters in S-GAN-P and SR-LSTM. Regarding the inference speed, the inference time of our model is 0.0033 s per time step which is the same as the original work and is the fastest in SOTA, because GNN is used and it is fast compared to RNN or CNN. Table 2 lists speed comparisons between our model and the publicly available models against which we can measure it.

5 Conclusion

In this paper, we presented the STM-GCN framework to improve the predictive accuracy of the pedestrian trajectory problem by integrating two relationships into a multi-graph. The first is position-based interactions extracted from relative pedestrian coordinates, while the second is velocity-based interactions computed using instant velocities. The obtained results compared with related work show that the more information extracted from the trajectories data, the more realistic the model will get performance results. This is what experienced in this work, but using only

two types of information (position and velocity). However, we got good results compared to the latest stat-of-the-art methods. As mentioned before, our work is built upon previous approaches that extract social interactions through a spatiotemporal graph representing pedestrians as the graph nodes. One of our future works will be to use other types of graphs, for example using a pedestrian graph based on joints as nodes. The other perspective to deal with this problem is to merge graphs, as we can see in our research paper merging two graphs of the same type gives us good results, so how about merging two different types of graphs into one graph (pedestrian graph, spatiotemporal graph), we can get great results.

Author Contributions The authors confirm their contribution to the paper as follows: study conception and design: Taki Youssef and Elmoukhtar Zemmouri; implementation: Taki Youssef1, Elmoukhtar Zemmouri, and Anas Bouzid; analysis and interpretation of results: Taki Youssef and Elmoukhtar Zemmouri; draft manuscript preparation: Taki Youssef and Elmoukhtar Zemmouri;. All authors reviewed the results and approved the final version of the manuscript.

Funding Not applicable.

Availability of data and materials In this work, we used two publicly available datasets widely used in the literature.

Code Availability Applicable.

Declarations

Conflict of interest The authors declare no competing interests.

Ethics approval Not applicable.

References

1. Houenou A, Bonnifait P, Cherfaoui V, Yao W (2013) Vehicle trajectory prediction based on motion model and maneuver recognition. In: IEEE International Conference on Intelligent Robots and Systems, pp 4363–4369
2. Zhou B, Tang X, Wang X (2015) Learning collective crowd behaviors with dynamic pedestrian-agents. Int J Comput Vis 111(1):50–68
3. Rasmussen CE, Williams CKI (2005) Gaussian processes for machine learning (adaptive computation and machine learning). The MIT Press, Cambridge
4. Alahi A, Goel K, Ramanathan V, Robicquet A, Fei-Fei L, Savarese S (2016) Social lstm: human trajectory prediction in crowded spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 961–971
5. Bai S, Kolter JZ, Koltun V (2018) An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271
6. Mohamed A, Qian K, Elhoseiny M, Claudel C (2020) Social-STGCNN: a social spatio-temporal graph convolutional neural network for human trajectory prediction. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 14424–14432
7. Pellegrini S, Ess A, Schindler K, van Gool L (2009) You'll never walk alone: modeling social behavior for multi-target tracking. In: 2009 IEEE 12th International Conference on Computer Vision, pp 261–268
8. Lerner A, Chrysanthou Y, Lischinski D (2007) Crowds by example. Comput Graph Forum 26(3):655–664

9. Lefèvre S, Laugier C, Ibañez-Guzmán J (2011) Exploiting map information for driver intention estimation at road intersections. In: Intelligent Vehicles Symposium (IV), 2011 IEEE. IEEE, pp 583–588
10. Danielsson S, Petersson L, Eidehall A (2007) Monte carlo based threat assessment: analysis and improvements. In: Intelligent Vehicles Symposium, 2007 IEEE. IEEE, pp 233–238
11. Firl J, Stübing H, Huss SA, Stiller C (2012) Predictive maneuver evaluation for enhancement of car-to-x mobility data. In: Intelligent Vehicles Symposium (IV), 2012 IEEE. IEEE, pp 558–564
12. Kalman RE (1960) A new approach to linear filtering and prediction problems. Trans ASME-J Basic Eng 82(Series D):35–45
13. Helbing D, Molnr P (1995) Social force model for pedestrian dynamics. Phys Rev E 51:4282–4286
14. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
15. Gupta A, Johnson J, Fei-Fei L, Savarese S, Alahi A (2018) Social gan: socially acceptable trajectories with generative adversarial networks. IEEE/CVF Confer Comput Vis Pattern Recogn 2018:2255–2264. https://doi.org/10.1109/CVPR.2018.00240
16. Bi H, Fang Z, Mao T, Wang Z, Deng Z (2019) Joint prediction for kinematic trajectories in vehicle-pedestrian-mixed scenes. In: The IEEE International Conference on Computer Vision (ICCV)
17. Xu Y, Piao Z, Gao S (2018) Encoding crowd interaction with deep neural network for pedestrian trajectory prediction. In: The IEEE Conference on Computer Vision, and Pattern Recognition (CVPR)
18. Li K, Eiffert S, Shan M, Gomez-Donoso2 F (2021) Attentional-GCNN: adaptive pedestrian trajectory prediction towards generic autonomous vehicle use cases. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE
19. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907
20. Vemula A, Muelling K, Oh J (2018) Social attention: modeling attention in human crowds. In: IEEE International Conference on Robotics and Automation (ICRA) 2017, pp 1–7
21. Zhang L, She Q, Guo P (2019) Stochastic trajectory prediction with social graph network. arXiv preprint arXiv:1907.10233
22. Zhang P, Ouyang W, Zhang P, Xue J, Zheng N (2019) Sr-lstm: state refinement for lstm towards pedestrian trajectory prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 12085–12094
23. Nikhil N, Morris BT (2018) Convolutional neural networks for trajectory prediction. In: Proceedings of the European Conference on Computer Vision (ECCV)
24. Zamboni S, Tilahun Kefato Z (2022) Pedestrian trajectory prediction with convolutional neural networks. Pattern Recogn 121:108252
25. Chandra R, Bhattacharya U, Bera A (2021) TraPHic: trajectory prediction in dense and heterogeneous traffic using weighted interactions. arXiv:1812.04767v4 [cs.RO]
26. Huang Y, Bi H, Li Z, Mao T, Wang Z (2019) Stgat: modeling spatial-temporal interactions for human trajectory prediction. In: The IEEE International Conference on Computer Vision (ICCV)
27. Rainbow BA, Men Q, Shum HP (2021) Semantics-STGCNN: a semantics-guided spatial-temporal graph convolutional network for multi-class trajectory prediction. arXiv:2108.04740v1 [cs.CV]
28. Kipf TN, Welling M, Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907
29. Eiffert S, Li K, Shan M, Worrall S, Sukkarieh S, Nebot E (2020) Probabilistic crowd gan: multimodal pedestrian trajectory prediction using a graph vehicle-pedestrian attention network. IEEE Robot Autom Lett 5(4):5026–5033
30. Pellegrini S, Ess A, Schindler K, Van Gool L (2009) You'll never walk alone: modeling social behavior for multi-target tracking. In: 2009 IEEE 12th International Conference on Computer Vision. IEEE, pp 261–268
31. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1026–1034
32. Sadeghian A, Kosaraju V, Sadeghian A, Hirose N, Rezatofighi H, Savarese S (2019) 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
33. Liang J, Jiang L, Niebles JC, Hauptmann AG, Fei- Fei L (2019) Peeking into the future: predicting future person activities and locations in videos. In: CVPR

34. Li J, Ma H, Tomizuka M (2019) Conditional generative neural system for probabilistic trajectory prediction. arXiv preprint arXiv:1905.01631
35. Zhou H, Ren D, Xia H, Fan M, Yang X, Huang H (2021) AST-GNN: an attention-based spatio-temporal graph neural network for Interaction-aware pedestrian trajectory prediction. Neurocomputing 445:298–308
36. Yao M, Tang J (2021) ma2021deep, deep learning on graphs. Cambridge University Press