PREDICTING PEDESTRIAN TRAJECTORIES IN ARCHITECTURAL SPACES: A GRAPH NEURAL NETWORK APPROACH

RUNYU YANG[1], WEILI WANG[2] and PENG GUI[3]
[1,2,3]*Glodon Company Limited.*
[1]*691302245@qq.com, 0009-0002-7646-5376*
[2]*2280918070@qq.com, 0009-0005-5059-7903*
[3]*gui.ruipeng@outlook.com, 0009-0001-5820-4423*

Abstract. This paper introduces a graph neural network-based model for predicting pedestrian trajectories in architectural spaces. Compared to traditional simulations based on physics-based models, this data-driven model has a stronger ability to learn and predict pedestrian behaviour patterns from real-world data. The model is pre-trained based on Hongqiao Railway Station Dataset, then trained and tested based on the ETH Dataset and the Stanford Drone Dataset, enabling comparisons with other AI models. By creating a more intelligent model, we can establish a digital replica of the real world that can predict pedestrian flow with higher accuracy in daily life or extreme situations such as sudden fires. Our results underscore the critical role of such models in comprehending how architectural spaces are utilized, and thus in improving architectural design and urban planning.

Keywords. Multi-agent Simulation, Trajectory Prediction, Graph Neural Network, Conditional Variational Autoencoder, Path-finding.

1. Introduction

The prediction of pedestrian trajectories involves using pedestrian features or historical trajectories to anticipate future pedestrian movements. This is a pivotal topic in computer science and social science, and has gained widespread attention and application in areas like industrial robots, autonomous driving, spatial analysis, and design (Bendali-Braham et al., 2021).

Pedestrians are intelligent agents who can naturally navigate and avoid obstacles in complex environments. They possess an inherent theory of mind, which allows them to reason about other people's actions in terms of their mental states (Gweon et al., 2013). Therefore, an effective pedestrian trajectory prediction model must not only perceive the environment but also perceive and anticipate the behaviors of neighboring agents.

There are numerous multi-agent behavior prediction methods, ranging from traditional cellular automata models and social force models to neural network-based regression models, generative models, and more. However, many of these methods mechanistically simplify agent behavior, failing to model agents as entities capable of perception, prediction, and decision-making. Our goal is to implement a model where each agent can perceive its surrounding environment, including obstacles and neighboring agents, and combine temporal features to predict the behavior of other agents before making its own decisions. The model parameters should be learnable, allowing for continuous optimization using real-world data.

2. Related Work

2.1. PHYSICS-BASED APPROACHES

The Social Force Model is a classic physics-based model that describes pedestrian movement. Based on Newtonian mechanics, this model assumes that pedestrians move under the influence of social forces (Helbing et al., 1995). Pedestrians are subjected to three types of forces: self-driven force, interpersonal force, and the force between individuals and obstacles.

Physics-based models align with the intuitive understanding of individual movement, with straightforward modelling and computation processes. However, they simplify pedestrians into particles, ignoring the individual's comprehensive perception capabilities, decision-making abilities, and randomness. Additionally, because the formulas involve many empirical parameters such as the avoidance range, these models lack generalization ability. The simulation of specific scenarios depends on subjective parameter adjustments or experiments, thus the practical application and validation present considerable challenges due to their higher usage threshold.

2.2. CLASSIC MACHINE LEARNING-BASED MODELS

In contrast to physics-based models, classic machine learning-based models analyze historical data to predict future trajectories. Specific methods include Kalman filtering, support vector regression models, hidden Markov models, and more.

The Kalman filtering method can account for the randomness of the motion process. The individual's uncertainty can be modelled by the Gaussian distribution (Narayanan et al., 2021). In each time step, the hidden states are first calculated and updated, and the mean and covariance matrix of the states in the next time step are predicted to represent the uncertain pedestrian movement.

Support Vector Regression (SVR) can learn the behaviour of agents in complex environments. It uses a kernel function to project input data into high-dimensional space, conducts regression prediction, finds the most probable movement strategy, and thus predicts future trajectories.

2.3. DEEP LEARNING-BASED MODELS

Classic machine learning-based methods are suitable for simple prediction scenarios and short-term prediction tasks. However, in recent years, deep learning-based trajectory prediction methods have become increasingly popular. These models consider various factors, such as physical factors, environmental factors, and the influence of neighbouring agents, and can be continually optimized through training. By using transfer learning, they can adapt to diverse scenarios.

Sequential models are capable of extracting historical trajectory features. It stores and updates information in hidden layers to predict the output. Alahi et al. (2016) proposed the Social LSTM for predicting pedestrian trajectories in crowded spaces. The attention mechanism is increasingly being employed in trajectory prediction tasks. Messaoud et al. (2020) used a multi-head attention mechanism to emphasize the impact of adjacent vehicles, achieving competitive trajectory prediction performance.

Graph Neural Networks (GNNs) rely on message aggregation processes for prediction. Agents in the environment are represented as nodes, and relationships between agents are represented as edges. GNNs excel at aggregating spatial information and handling dynamic data, making them well-suited for complex trajectory scenarios. Shi et al. (2021) introduced SGCN for pedestrian trajectory prediction. The results demonstrate that it effectively captures adaptive interactions between pedestrians and their motion tendencies.

Generative models can be utilized for generating diverse trajectories. Notably, some of these methods have shifted their focus from predicting trajectories to predicting the distribution of trajectories. This shift better supports downstream tasks such as trajectory matching. Dendorfer et al. (2020) presented Goal-GAN, an end-to-end trajectory prediction model that generates diverse trajectories consistent with physical constraints. Xu et al. (2022) introduced SocialVAE to capture the uncertainty and multi-modality of human navigation decision-making.

3. Methodology

Our objective is to predict pedestrian trajectories based on their features and historical trajectories within built environments.

- Pedestrian trajectories represent temporal data which can be efficiently encoded by sequential models.
- Architectural spaces often contain various obstacles, such as walls and atriums. Pedestrians have the ability to perceive their surroundings, so an environment encoder can be used to gather spatial information.
- Pedestrians also have the ability to perceive and anticipate the actions of other individuals. Therefore, neighbour information should be aggregated through message aggregators in graph network layers.
- To learn the distribution of pedestrian features from real-world data, we introduce a feature generation model.

In summary, we present a hybrid trajectory prediction and generation model that consists of Temporal Convolutional Networks (TCN) for trajectory encoding, CNNs for environmental encoding, graph message aggregators for compiling neighbour information, and Conditional Variational Autoencoder (CVAE) for synthesizing pedestrian features. In the output section, we employ a Multilayer Perceptron (MLP) to merge the features obtained earlier and to predict the mean and variance of the trajectories. This model is named the Graph Temporal Convolutional Network (GTCN). We optimize the predicted values to match the ground truth by minimizing the negative log likelihood of the Gaussian likelihood function.

3.1. TRAJECTORY ENCODER

Trajectory is a type of temporal sequence data. We represent the pedestrian location at a given moment with two-dimensional coordinates. The sampling interval for the trajectory is denoted as t_interval, and the interval length is denoted as interval_length. Note that as time progresses, the interval_length gradually increases. Therefore, we capture the data up to a maximum time duration of max_length. Consequently, for n pedestrians, the shape of the trajectory data is (n, max_length, 2). It is important to recognize that the historical trajectory lengths of pedestrians do not always equate to max_length, necessitating the use of a variable-length temporal encoder.

Temporal models in deep learning include LSTM, TCN, Transformer, and others. LSTM introduces gated units that can store and forget memories, making it well-suited for processing temporal information. However, it calculates by the sequence order, which is not easily parallelizable, making it slow for long sequence data. The Transformer does not use recursive structures, allowing for parallelization to speed up computations and effectively managing long-distance dependency information. However, it has a large number of model parameters, requiring extensive training data and computational resources, and it is prone to overfitting on short sequences.

TCN is a variant of CNN that processes temporal data through convolution. It can compute in parallel, flexibly adjust its receptive field based on the size of the convolutional kernel to acquire information from different temporal scales, and with fewer model parameters, it is easier to train and less likely to overfit, making it suitable for encoding trajectory information with lengths from single to tens of digits. Therefore, we choose TCN as our trajectory encoder.

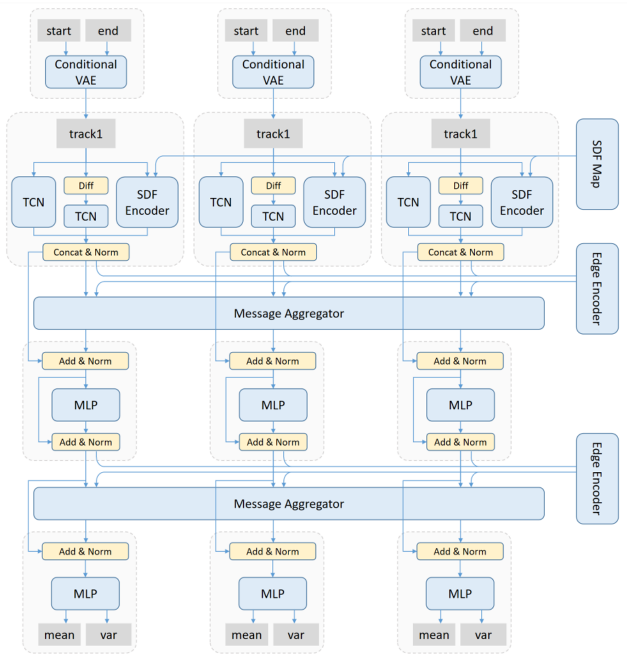

Figure 1. Model structure of GTCN.

3.2. MAP ENCODER

The architectural spaces are complex and diverse, encompassing walls, atria, ramps, furniture, and more. Pedestrians can perceive these obstacles and adjust their movement strategies accordingly. Therefore, it's crucial to encode obstacle information.

The Signed Distance Field (SDF) is commonly used in computer graphics. It represents a distance field where each point stores the distance to the nearest object. In our case, we consider all inaccessible areas within the space as obstacles and compute the Signed Distance Field of that space, denoted as SDF_space, to represent the global environmental information. For the k-th pedestrian, we extract a surrounding fixed-size SDF_k to represent the local environmental information. We then input SDF_k into a map encoder. We use a CNN as the encoder, utilizing a five-layer structure of Conv2d - ReLU - MaxPool2d for convolution and pooling. This is followed by a fully connected layer, which outputs a 64-dimensional vector to serve as the environmental encoding.

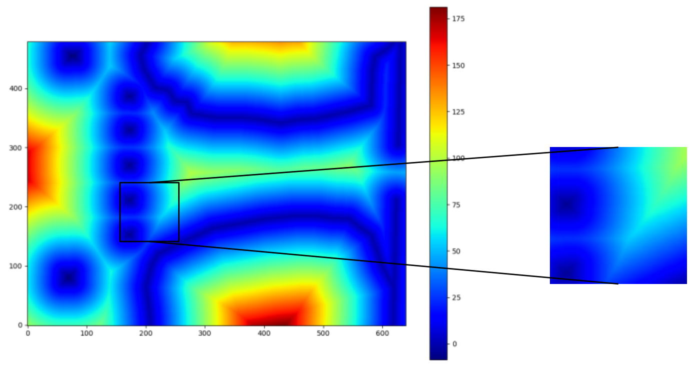

Figure 2. Left: global Signed Distance Field. Right: local Signed Distance Field.

3.3. GRAPH NEURAL NETWORK

Pedestrians as sentient beings, are capable of perceiving and predicting the movements of nearby pedestrians. To avoid collisions or catch up with companions, they adjust their movements accordingly. Therefore, we introduce Message Aggregators to gather neighbourhood information.

The perception range of a pedestrian is relatively fixed, but the number of other pedestrians within this range constantly changes. This requires a model that can handle complex non-Euclidean data. Graph Neural Networks (GNNs) are particularly suitable for this task. We represent the features of agents as nodes and compute the distance between each pair of nodes. We then assign an edge to every pair of nodes that are within a certain distance threshold. Nodes and edges are fed into a Graph Convolutional Network layer for aggregation. The number of layers in the graph is a significant hyper-parameter. Through experimentation, we found that 1 to 2 layers of GCN can effectively aggregate neighbour information. In contrast, more than 3 layers, although capable of collecting long-distance information, may lead to over-smoothing. Therefore, we set the number of layers to 2.

3.4. CONDITIONAL VARIATIONAL AUTOENCODER

When applying a trajectory prediction model to the pedestrian flow simulation within architectural spaces, a key issue is initializing pedestrian features. In common trajectory prediction tasks, the initial features of an object, namely its historical trajectory, serve as inputs to the model. However, in the task of predicting flow within a building space, we typically use path-finding algorithms to determine the starts and goals of the pedestrians. Additional features, such as historical trajectory, body gait and stride, need to be manually set or generated out of thin air. By learning the distribution of pedestrian features in dataset, we can generate reasonable feature while avoiding complex manual inputs. The starts and goals can be used as conditions to guide the feature initialization for the generative model.

Common generative models include Variational Autoencoders (VAE), Generative Adversarial Networks (GAN), and Diffusion Models. GAN consists of a generator (G) and a discriminator (D), which iteratively optimize during the training process to generate highly convincing data. However, GANs are difficult to train and can fall into producing overly limited samples. Diffusion Models have recently become a popular generative model, especially in the field of image generation. They can produce high-quality samples but require a considerable number of time steps in the diffusion process, leading to high computational time and resource requirements.

VAE compresses data into latent vectors through an encoder and then uses a decoder to generate data from the latent vector. It optimizes the reconstruction accuracy by comparing the original data with the reconstructed data to minimize the Mean Squared Error (MSE) loss and generate data from noise by pushing the latent vector towards a standard normal distribution to minimize the Kullback–Leibler divergence (KL) loss. VAE has a low computational cost and are suitable for learning continuous and smooth features, thus is able to generate structurally simple pedestrian features. Further, by concatenating generation conditions to the latent vector, we can achieve a conditional generation model.

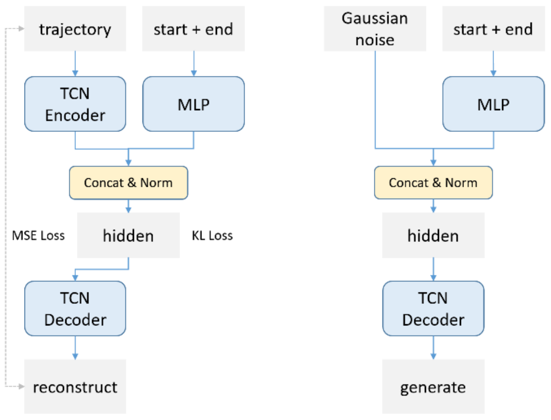

Figure 3. Left: training of Conditional Variational Autoencoder. MSE loss will be calculated between label trajectory and reconstructed trajectory. KL loss will be calculated in hidden states. Right: generate trajectory data from start location, end location and Gaussian noise.

4. Experiment

4.1. SHORT-TERM PREDICTION ON VIDEOS

To visually assess the efficacy of our model, we conducted an experiment using videos. We captured a video of a pedestrian intersection at Shanghai Hongqiao Railway Station, spanning a duration of 90 minutes with a frame rate of 30 and containing a total of 162,000 frames. We employed YOLOX (Ge et al., 2021) as the object detector to detect pedestrians and ByteTrack (Zhang et al., 2022) as the object tracker to obtain continuous pedestrian trajectories.

For each pedestrian at each moment, the object detector provided a four-dimensional dataset containing the top-left corner coordinates, as well as the width and height of the detection box. In this case, we not only predicted the coordinates but also the width and height of the detection box. We considered a maximum historical length of 8 frames, which served as input to predict the trajectory for the next 12 frames. We divided all 162,000 frame data into 8100 groups with 20 frames per group, further dividing them into 80% training set, 10% validation set, and 10% test set. We employed Average Displacement Error (ADE) to calculate the prediction error (Yue et al., 2022). ADE was calculated as the L2 error between a predicted trajectory and the ground truth. We used the mean of the predictions as the trajectory prediction value and visualized the predicted variance as the thickness of the detection box. We used a batch size of 128, a learning rate of 0.005, and an Adam optimizer. After about 50 epochs, ADE was reduced to about 4.1 pixels. As a comparison, when using Kalman filtering as prediction model, the ADE is approximately 7 pixels, which shows that the AI model performs much better than the traditional model.

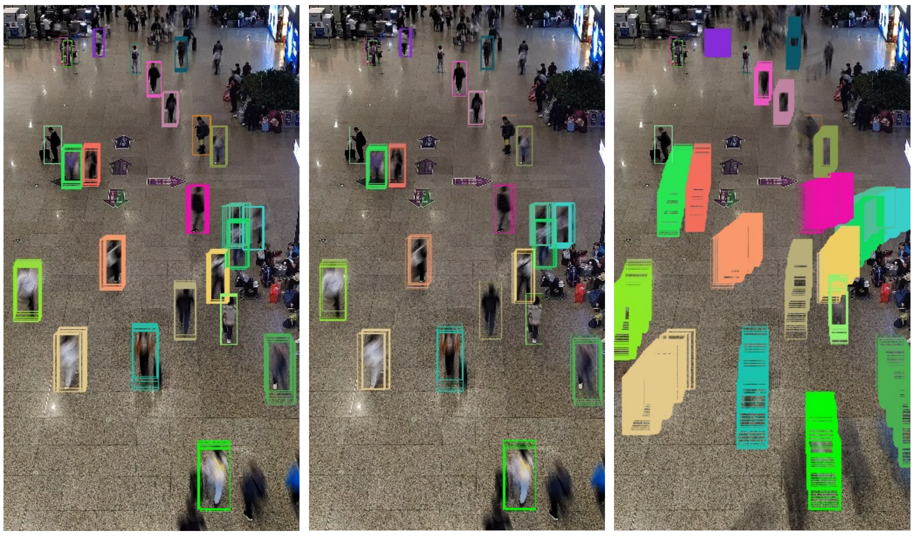

Figure 4. Left: the ground truth of detection box trajectory, with a length of 12 frames, corresponding to 0.4 seconds. Middle: the predicted detection box trajectory with a length of 12 frames. The predicted values are very close to the ground truth. Right: the long-term predicted trajectory obtained by iterating the prediction results, with a length of 60 frames, corresponding to 2 seconds.

4.2. COMPARISON ON OPEN SOURCE DATASETS

To ensure the versatility of our trajectory prediction model, we pretrained it on the Hongqiao Railway Station Dataset and fine-tuned it on the ETH Dataset (Pellegrini et al., 2009) and the Stanford Drone Dataset (Robicquet et al., 2016). This approach enabled the model to adapt to different scenarios.

We compared our model with S-GAN (Gupta et al., 2018) , Sophie (Sadeghian et al., 2019) , Trajectron++ (Salzmann et al., 2020) , SGCN (Shi et al., 2021) on the ETH and SDD datasets. Note that the Social Force Model is a traditional physics-based model that requires manual parameter settings and destination settings, which are not included in the dataset. Therefore, it cannot be directly compared with other trajectory models. The test metrics were ADE and FDE, and the test results are summarized in the following table.

Table1. Results on ETH/UCY and SDD. Our model outperforms baseline methods in both ADE and FDE.

Methods	Metrics	ETH	Hotel	ZARA1	ZARA2	SDD
S-GAN	ADE	0.82	0.70	0.33	0.42	27.23
	FDE	1.52	1.63	0.69	0.81	42.60
Sophie	ADE	0.70	0.76	0.31	0.38	17.21
	FDE	1.40	1.68	0.63	0.80	29.34
Trajectron++	ADE	0.71	0.22	0.30	0.23	17.90
	FDE	1.68	0.46	0.78	0.60	30.83
SGCN	ADE	0.87	0.68	0.33	0.42	30.31
	FDE	1.62	1.38	0.69	0.83	41.76
GTCN (Ours)	ADE	0.68	0.23	0.29	0.21	22.81
	FDE	1.49	0.44	0.63	0.69	34.43

4.3. MODEL APPLICATION IN ARCHITECTURAL SPACES

The simulation of pedestrian movement in architectural spaces typically involves two main components: global path-finding for each agent to reach their destination, and local motion modeling to simulate agent movement. Path-finding typically utilizes heuristic algorithms such as A*, while the motion modeling aspect uses a motion model at the local level (Mashhadawi, 2016). The building traffic system encompasses both horizontal and vertical traffic elements. Horizontal traffic includes floors, carpets, and other walking areas, while vertical traffic includes escalators, elevators, and evacuation stairs.

To model the reachable building space for pedestrian simulations, we represent it as a graph. Each floor that is reachable is divided into grids, and each grid serves as a node in the graph. Edges are established between adjacent nodes. Then, we randomly distribute or manually input the initial positions and destinations of pedestrians, use path-finding algorithms to determine the path for each pedestrian. We utilize the trajectory prediction model as a motion model to simulate the local movement of the crowd. For each newly generated agent, we consider their starting point and destination, generate initial features using the CVAE model, and input them into the trajectory prediction model to predict their movement. The trajectory prediction model predicts the movement of each pedestrian for 12 frames, approximately 0.4 seconds, and then updates their position and trajectory in a loop until they reach the exit. After calculation, approximately 780 updates and 312 seconds later, all pedestrians have exited the building. We accumulate the positions of all pedestrians over time to create a spatial distribution heatmap. This heatmap allows us to quantitatively identify crowded areas and optimize the design accordingly.

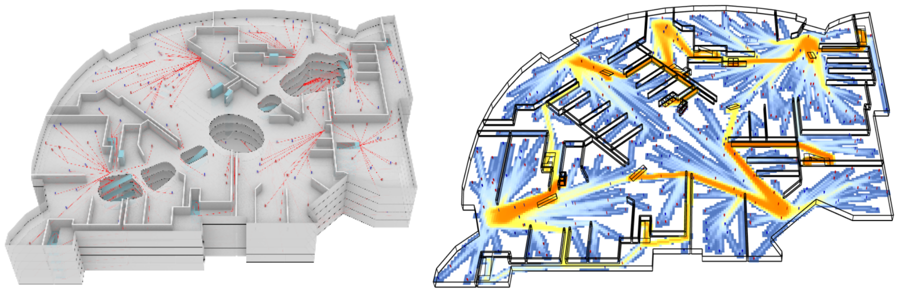
Figure 5. Left: Path-finding algorithms. The red lines represents the route. Right: Representation of the simulation results using a heatmap. The red areas indicate potential congestion zones.

5. Conclusion

Pedestrians are intelligent individuals, with the ability to perceive crowds and environments and predict the behavior of others. Pedestrians can move regularly, but may also change their intentions to make unpredictable movements. Therefore, physics-based models, such as Social Force Model, are difficult to simulate diverse pedestrian behaviors. We have built a novel AI model, called GTCN, to predict pedestrian trajectories in architectural spaces. The model comprehensively considers the perception and prediction abilities of pedestrians, as well as the diversity of pedestrian characteristics.

Video experiments have shown that GTCN can simulate trajectories with higher accuracy than traditional Kalman filters. Experiments on open-source dataset have shown that GTCN outperforms other AI models. We applied the model in architectural spaces to predict pedestrian distribution and movement. Through quantitative analysis, we obtained indicators such as spatial distribution heatmaps. Based on visual results, architects can identify design vulnerabilities such as congestion or accident-prone areas, to optimize building efficiency or improve safety.

References

Bendali-Braham, M., Weber, J., Forestier, G., Idoumghar, L., & Muller, P. A. (2021). Recent trends in crowd analysis: A review. Machine Learning with Applications, 4, 100023.
Gweon, H., & Saxe, R. (2013). Developmental cognitive neuroscience of theory of mind. Neural circuit development and function in the brain, 3, 367-377.
Helbing, D., & Molnar, P. (1995). Social force model for pedestrian dynamics. Physical review E, 51(5), 4282.

Narayanan, S., Moslemi, R., Pittaluga, F., Liu, B., & Chandraker, M. (2021). Divide-and-conquer for lane-aware diverse trajectory prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 15799-15808).

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).

Messaoud, K., Yahiaoui, I., Verroust-Blondet, A., & Nashashibi, F. (2020). Attention based vehicle trajectory prediction. IEEE Transactions on Intelligent Vehicles, 6(1), 175-185.

Shi, L., Wang, L., Long, C., Zhou, S., Zhou, M., Niu, Z., & Hua, G. (2021). SGCN: Sparse graph convolution network for pedestrian trajectory prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8994-9003).

Dendorfer, P., Osep, A., & Leal-Taixé, L. (2020). Goal-gan: Multimodal trajectory prediction based on goal position estimation. In Proceedings of the Asian Conference on Computer Vision.

Xu, P., Hayet, J. B., & Karamouzas, I. (2022, October). Socialvae: Human trajectory prediction using timewise latents. In European Conference on Computer Vision (pp. 511-528). Cham: Springer Nature Switzerland.

Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). Yolox: Exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430.

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., ... & Wang, X. (2022, October). Bytetrack: Multi-object tracking by associating every detection box. In European Conference on Computer Vision (pp. 1-21). Cham: Springer Nature Switzerland.

Yue, J., Manocha, D., & Wang, H. (2022, October). Human trajectory prediction via neural social physics. In European Conference on Computer Vision (pp. 376-394). Cham: Springer Nature Switzerland.

Mashhadawi, Mohammad. "MassMotion evacuation model validation." LUTVDG/TVBB (2016).

Pellegrini, S., Ess, A., Schindler, K., & Van Gool, L. (2009, September). 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.

Robicquet, A., Sadeghian, A., Alahi, A., & Savarese, S. (2016). Learning social etiquette: Human trajectory understanding in crowded scenes. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14 (pp. 549-565). Springer International Publishing.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., & Alahi, A. (2018). Social gan: Socially acceptable trajectories with generative adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2255-2264).

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/CVF conference on computer vision and pattern recognition (pp. 1349-1358).

Salzmann, T., Ivanovic, B., Chakravarty, P., & Pavone, M. (2020). Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16 (pp. 683-700). Springer International Publishing.