SOCIAL LSTM WITH DYNAMIC OCCUPANCY MODELING FOR REALISTIC PEDESTRIAN TRAJECTORY PREDICTION

A PREPRINT

Ahmed Alia[1,3], Mohcine Chraibi[1], Armin Seyfried[1,2]

[1] Institute for Advanced Simulation, Forschungszentrum Jülich, 52425 Jülich, Germany

[2] Faculty of Architecture and Civil Engineering, University of Wuppertal, 42285 Wuppertal, Germany

[3] Faculty of Information Technology and Artificial Intelligence, An-Najah National University, P4110257 Nablus, Palestine

November 14, 2025

ABSTRACT

In dynamic and crowded environments, realistic pedestrian trajectory prediction remains a challenging task due to the complex nature of human motion and the mutual influences among individuals. Deep learning models have recently achieved promising results by implicitly learning such patterns from 2D trajectory data. However, most approaches treat pedestrians as point entities, ignoring the physical space that each person occupies. To address these limitations, this paper proposes a novel deep learning model that enhances the Social LSTM with a new Dynamic Occupied Space loss function. This loss function guides Social LSTM in learning to avoid realistic collisions without increasing displacement error across different crowd densities, ranging from low to high, in both homogeneous and heterogeneous density settings. Such a function achieves this by combining the average displacement error with a new collision penalty that is sensitive to scene density and individual spatial occupancy. For efficient training and evaluation, five datasets were generated from real pedestrian trajectories recorded during the Festival of Lights in Lyon 2022. Four datasets represent homogeneous crowd conditions—low, medium, high, and very high density—while the fifth corresponds to a heterogeneous density distribution. The experimental findings indicate that the proposed model not only lowers collision rates, but also enhances displacement prediction accuracy in each dataset. Specifically, the model achieves up to a $31\,\%$ reduction in the collision rate and reduces the average displacement error and the final displacement error by $5\,\%$ and $6\,\%$, respectively, on average across all datasets compared to the baseline. Moreover, the proposed model consistently outperforms several state-of-the-art deep learning models across most test sets.

Keywords Deep Learning · Social LSTM · Dynamic Occupied Space Loss Function · Pedestrian Trajectory Prediction · Realistic Collision Avoidance · Intelligent Human Motion Modeling · Urban Planning

1 Introduction

The accurate and realistic prediction of pedestrian trajectories in crowded environments is crucial for a wide range of applications, including robotic navigation [1], urban planning [2], and crowd management systems [3]. Despite significant progress, this task remains challenging in dense crowds due to the complexity of the dynamic nature of pedestrian behavior [4, 5, 6]. In such environments, individuals continually adjust their paths in response to their surroundings, especially in reaction to the movements of others. Early research addressed these challenges using models based on physics or models inspired by robotics, such as the social force model [7, 8], cellular automata model [9], and the optimal reciprocal collision avoidance [10]. These methods, grounded in well-defined rules or differential equations, established a strong foundation for understanding pedestrian movement and continue to provide valuable

insights [11, 12]. With advances in deep learning and its success in various human crowd analysis tasks [13, 14], deep learning algorithms have gained increasing attention for pedestrian trajectory prediction. These methods have demonstrated reliable performance in forecasting several seconds into the future (e.g., $4\,\mathrm{s}$) [15, 16, 17]. Among the most widely used deep learning methods are Long Short-Term Memory networks (LSTM) [18] and Generative Adversarial Networks (GAN) [19]. Although these models effectively learn and predict movement sequences, they often treat each pedestrian's path independently, overlooking the influence of nearby individuals. This limits their ability to produce accurate and realistic forecasts in crowded environments.

To address this limitation, advanced models such as Social-LSTM [20] and Social-GAN [21] incorporate mechanisms that account for the influence of nearby movements using 2D trajectories of multiple pedestrians. Social-LSTM, for example, assigns an individual LSTM to each pedestrian and introduces a social pooling layer that aggregates the hidden states of neighbors within a local spatial grid. This design allows the model to adjust the predictions based on the motion of the surrounding individuals. To avoid confusion, the term "social" in these models does not denote genuine social interaction in a semantic sense, since 2D trajectory data alone cannot capture behaviors that depend on non-kinematic cues such as intentions, gaze, gestures, or verbal communication [22, 23].

Although these advances improve the modeling of pedestrian motion, they often fall short in ensuring that the predicted trajectories remain physically realistic, particularly in dense crowds. A key limitation is that pedestrians are represented as single points, which neglects their physical dimensions and spatial occupancy in the environment, often leading to unrealistic collisions [12, 24]. Recently, a metric has been proposed to evaluate a model's ability to generate collision-free paths between the bodies of pedestrians, representing each individual as a circular disk with a fixed radius of $0.2\,\mathrm{m}$ to approximate occupied space []. Building on this motivation, a new loss function for Social LSTM was introduced that not only minimizes displacement errors, but also explicitly learns collision avoidance [25]. While effective in low- and medium-density environments, this approach underperforms in highly dense and heterogeneous crowds.

To overcome these limitations, this paper proposes a novel deep learning model based on Social-LSTM, enhanced by a new Dynamic Occupied Space (DOS) loss function. The proposed model leverages this loss to explicitly learn realistic collision avoidance while also improving displacement accuracy across various crowd densities, both homogeneous and heterogeneous.

The contribution of this paper can be summarized as follows.

1. This paper presents a novel deep learning model that combines the Social LSTM architecture with a new loss function to predict pedestrian trajectories more accurately and realistically in diverse and dynamic crowd environments.
2. It introduces a collision-aware loss function based on a circular disk with a dynamic radius that adapts to scene density, enabling Social-LSTM to better capture physical realism while improving displacement accuracy.
3. The study uses five datasets that represent low, medium, high, and very high crowd densities, along with a heterogeneous dataset that includes a mix of all four. The data source for these datasets is real pedestrian trajectories recorded during Lyon's Festival of Lights in 2022.

The remainder of this paper is organized as follows. Section 2 reviews existing studies on the prediction of pedestrian trajectory. Section 3 outlines the methodology behind the proposed deep learning framework to predict realistic pedestrian trajectory in crowded environments. Section 4 presents the experimental setup, results, and comparative analysis. Finally, Section 5 summarizes the key findings, discusses limitations, and outlines potential directions for future research.

2 Related Work

Efficient prediction of pedestrian trajectory in crowded environments remains a challenging task, particularly due to the need to model nearby movements and ensure physically plausible, collision-free paths. Recent deep learning methods have significantly advanced the field by learning from large-scale trajectory datasets [26, 27, 28].

2.1 Neighbor-based Models

In this category, models leverage the movements of nearby pedestrians to account for the influence of surrounding trajectories. Early models such as Social LSTM [29] introduced social pooling to implicitly model interactions between each individual and nearby pedestrians. Based on this, SR-LSTM [30] and STGAT [31] incorporated attention mechanisms and spatio-temporal graph structures to capture richer interaction patterns. To broaden the spatial context,

Xue et al. [32] proposed a joint Location–Velocity Attention LSTM model for pedestrian trajectory prediction, where an attention mechanism learns to combine location and velocity cues to improve forecasting accuracy and generalization across scenes.

More recent architectures, such as Social-BiGAT [33] and AgentFormer [34], employed attention-based frameworks—Social-BiGAT used a graph attention network for social interactions, while AgentFormer leveraged a transformer to jointly capture temporal and social dependencies. Advancements such as SocialFormer [35] and Knowledge-Aware Graph Transformer [36] extended attention-based frameworks by incorporating semantic interaction features and adaptive mechanisms to better capture complex relational dependencies.

2.2 Generative-based Models

Generative models have emerged to address the multimodal and uncertain nature of human behavior by generating multiple plausible future trajectories. Social GAN [21] was among the first to apply a generative adversarial framework, combining LSTM-based encoders and decoders with a discriminator to produce diverse and socially compliant paths. Trajectron++ [37] proposed a graph-structured recurrent model to forecast multi-agent trajectories by incorporating agent dynamics and environmental context.

More recent approaches, such as GSGFormer [38], integrate graph attention with CVAE modules for semantically consistent generation, while TUTR [39] simplifies the process by using a transformer-based framework to directly predict trajectory modes and their probabilities.

2.3 Physical Realism and Collision Avoidance

Despite these advances, most of the existing deep learning–based trajectory prediction approaches still model pedestrians as abstract points, ignoring their physical space requirements. This limits their ability to ensure realistic and collision-free predictions in dense environments [12, 40].

To address this limitation, recent research has incorporated representations of personal body space and penalized overlapping trajectories. For example, Time-to-Collision (TTC)-Social LSTM [25] uses a TTC-based loss function and represents each pedestrian as a circular disk with a fixed radius of $0.2\,\mathrm{m}$. This design helps the model learn realistic collision avoidance behavior while maintaining displacement accuracy, thereby improving realism and predictive accuracy in crowded scenes. However, TTC-Social LSTM still faces challenges in dense and heterogeneous crowds, where avoiding collisions can reduce the accuracy of the prediction. One contributing factor is the use of a single fixed body-space size for all pedestrians, regardless of scene density.

As a result, this article proposes a novel deep learning model that integrates the Social-LSTM architecture with a dynamic loss function of the occupied space, which adapts the space size of each person according to the density of the scene. This approach is designed to reduce collision occurrences and displacement errors in various conditions, including homogeneous low-, medium-, and high-density scenes, as well as heterogeneous density distributions. The following section presents a detailed overview of the proposed model.

3 Methodology

This section begins with the pedestrian trajectory prediction problem. Then it introduces the proposed model that predicts realistic human trajectories. Finally, it presents the evaluation metrics used to assess the model.

3.1 Problem Statement

As shown in fig. 1, the proposed model processes the observed trajectories X of all individuals n in a scene, formulated as $X = \{X_1, X_2, \ldots, X_n\}$, and predicts their future trajectories $\hat{Y}$, given by $\hat{Y} = \{\hat{Y}_1, \hat{Y}_2, \ldots, \hat{Y}_n\}$. Each observed trajectory X_i of the pedestrian i over time steps t from t_1 to t_{obs} is expressed as:

$$X_i = \{(x_i^t, y_i^t) \mid t = t_1, t_2, \ldots, t_{obs}\}, \tag{1}$$

where t_1 refers to the first observed time step, t_{obs} denotes the final observed time step, and (x_i^t, y_i^t) represents the spatial coordinates of the pedestrian i in the time step t. Similarly, the future trajectory $\hat{Y}_i$ predicted of individual i over time steps t from t_{obs+1} to t_{pred} is defined as:

$$\hat{Y}_i = \{(\hat{x}_i^t, \hat{y}_i^t) \mid t = t_{obs+1}, t_{obs+2}, \ldots, t_{pred}\}, \tag{2}$$

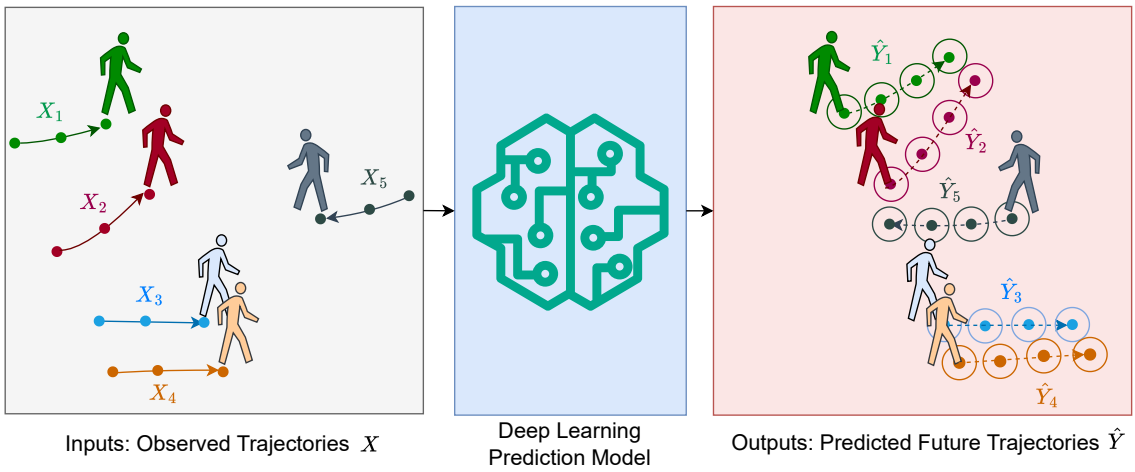

Figure 1: Illustration of the pedestrian trajectory prediction problem, where the goal is to minimize displacement error and avoid collisions between the predicted future paths. The circles with a radius of $0.2\,\mathrm{m}$ represent the body of a person and its space requirement.

where t_{pred} represents the final time step for the future trajectory predicted by the person i. Furthermore, the ground truth trajectories corresponding to $\hat{Y}$ can be defined as $Y = \{Y_1, Y_2, \ldots, Y_n\}$, where Y_i is given by:

$$Y_i = \{(x_i^t, y_i^t) \mid t = t_{obs+1}, t_{obs+2}, \ldots, t_{pred}\}. \tag{3}$$

In addition, the observed trajectory X_i and the future trajectory Y_i together form the composed trajectory $a_i(t_1)$ of the person i within a scene, starting from the time step t_1 and ending at t_{pred}. The trajectory $a_i(t_1)$ is defined as:

$$a_i(t_1) = \{(x_i^t, y_i^t) \mid t = t_1, t_2, \ldots, t_{\text{pred}}\}. \tag{4}$$

To determine whether a collision occurs between two pedestrians i and j at the time step t, each pedestrian is modeled as a circle with a fixed radius of $R = 0.2$ m [41, 29, 25]. A collision is detected if their respective circles intersect. Mathematically, this condition is evaluated using the Euclidean distance between the centers of the two pedestrians, which is expressed as:

$$\text{collision}(i, j, t) = \begin{cases} 1 & \text{if } \sqrt{(x_i^t - x_j^t)^2 + (y_i^t - y_j^t)^2} < 2R, \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

the collision indicator function returns 1 if a collision is detected and 0 otherwise.

3.2 Dynamic Occupied Space-based Social LSTM Model

3.2.1 Social LSTM

Social LSTM is a recurrent neural network architecture designed to predict the future trajectories of individuals in crowded spaces. As illustrated in fig. 2, the model extends the traditional LSTM by introducing a social pooling layer, which allows neighboring LSTMs to share their hidden states that encode time-varying motion patterns. In this architecture, each pedestrian is modeled by a separate LSTM that processes a sequence of their observed positions over time. To explicitly capture interactions among pedestrians, the social pooling layer aggregates the hidden states of pedestrians located within a predefined spatial neighborhood at each time step. For example, in fig. 2, $LSTM_1$ and $LSTM_3$ share their hidden states (h_1 and h_3) with $LSTM_2$ via its pooling layer, since individuals 1 and 3 fall within the predefined neighborhood of person 2. The aggregated features (H_2) from the pooling layer, combined with the hidden internal state of $LSTM_2$ (h_2), are then used to predict the future trajectory of individual 2. In contrast, $LSTM_2$ does not share its hidden state (h_2) with$LSTM_4$, as individual 4 lies outside the predefined spatial threshold. For more details about Social LSTM, the reader is referred to [20].

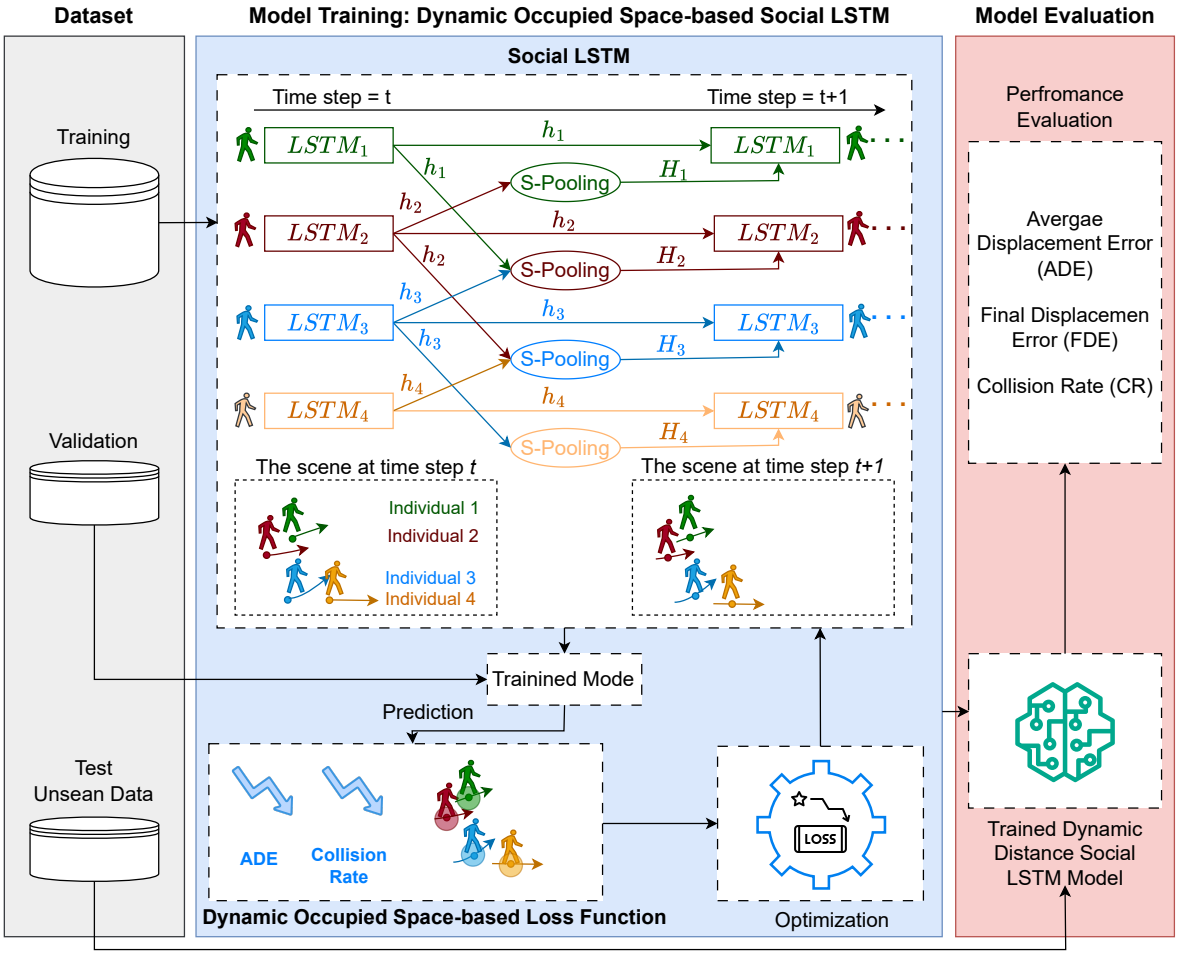

Figure 2: Overview of our model, including its training and evaluation process. In Social LSTM, S-Pooling refers to social pooling layers that share hidden states (h) among nearby LSTMs, while circles represent the occupied space of individuals.

Despite its ability to reduce displacement errors in crowded scenes, Social LSTM occasionally fails to prevent pedestrian collisions on the predicted trajectories [29]. This limitation arises because the model is trained to minimize displacement error without explicitly considering collision avoidance [33]. As a result, the notion of collision avoidance is expected to be learned implicitly. However, this implicit modeling can lead to unrealistic trajectory predictions, particularly in dense crowd scenarios. Therefore, the original Social LSTM model should be enhanced to train not only to minimize displacement error but also to explicitly avoid collisions.

3.2.2 Dynamic Occupied Space-based Loss Function

Here, we propose Dynamic Occupied Space ($\mathcal{L}_{DOS}$), a new and efficient loss function for Social LSTM that encourages collision avoidance and reduces displacement errors. This leads to more realistic crowd movement predictions across scenarios with different densities.

The proposed loss function combines the Average Displacement Error (ADE) with a new Collision Penalty ($\mathcal{CP}$) metric, and is defined as:

$$\mathcal{L}_{\text{DOS}} = ADE + \lambda \times \mathcal{CP}, \tag{6}$$

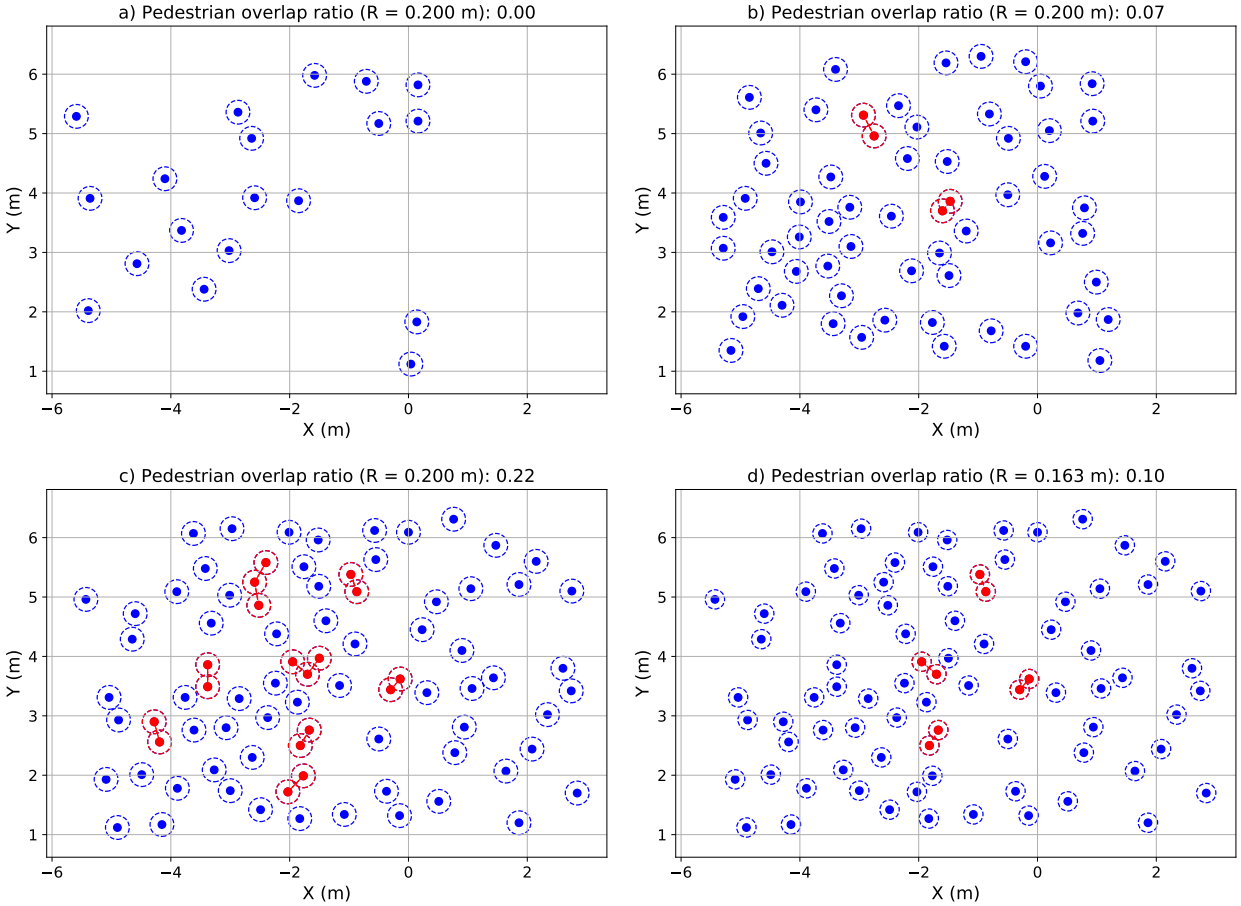

Figure 3: Visualized examples of pedestrians within a scene at a specific frame, illustrating varying overlap ratios. Each circle with radius R represents the physical space occupied by an individual. Red circles indicate pedestrians involved in collisions, with red lines showing the distance between the centers of the colliding individuals. Blue circles represent pedestrians without any overlaps.

where λ is a predefined weight that controls the influence of the $\mathcal{CP}$ on the overall loss function. ADE is used to calculate the average distance between the predicted trajectories $\hat{Y}$ and the corresponding ground truth trajectories Y. ADE plays a key role in improving the performance of distance-based metrics (more details are provided in section 3.3). In contrast, the primary objective of the $\mathcal{CP}$ is to reduce collisions by modeling the dynamic physical space occupied by each pedestrian, rather than relying solely on point-based representations. The $\mathcal{CP}$ consists of two main modules: Dynamic Radius Estimation for Occupied Space and Collision Penalty Calculation, which are described in the following sections.

Dynamic Radius Estimation

The Dynamic Radius Estimation for Occupied Space module aims to provide a more accurate representation of pedestrians by adapting to changes in crowd density, ranging from sparse to highly congested conditions. In several pedestrian trajectory prediction approaches [29, 25], individuals are commonly represented as circles with a fixed R of $0.2\,\mathrm{m}$. This simplification is intended to enhance the realism of pedestrian representation, particularly in crowded scenarios, where physical space and interactions play a crucial role. Figure 3a presents a visual example of empirical data on pedestrian positions, in which each circle represents the space requirement of a person. However, this fixed-radius representation does not accurately model the space requirements in high-density environments. As illustrated in fig. 3b and c, the radius R tends to decrease as crowd density increases, often becoming smaller than $0.2\,\mathrm{m}$. When a constant radius of $0.2\,\mathrm{m}$ is applied uniformly in dense crowds, it can lead to false collisions between pedestrians during model training. Figure 3d shows that reducing the radius helps mitigate these false collisions by better aligning with the actual spacing observed on the ground. Therefore, estimating the radius R of occupied area to closely match the actual physical space requirement could enhance the proposed model's ability to predict realistic future pedestrian trajectories

in various dense homogeneous and heterogeneous crowds. As a result, this module is responsible for estimating the average radius $\bar{R}$ of the occupied area of individuals within each prediction window, spanning from t_{obs+1} to t_{pred} time steps.

First, the set of colliding pedestrian pairs $\mathcal{O}^t$ at time step t is identified by computing the Euclidean distance d_{ij}^t between their 2D positions (x_i^t, y_i^t) and (x_j^t, y_j^t), for all $i, j \in \mathcal{N}^t$ with $i < j$.

$$\mathcal{O}^t = \left\{ (i,j) \in \mathcal{N}^t \times \mathcal{N}^t \;\middle|\; i < j,\ d_{ij}^t < 0.4\,\mathrm{m} \right\}. \tag{7}$$

For illustration, Figure 3c shows such overlapping pairs, highlighted by red circles and connected by red lines.

Afterward, the average radius of the occupied space R^t in time step t is computed as the mean of half the distances from center to center of all colliding pairs $\mathcal{O}^t$. The factor $1/2$ ensures that the measure is expressed as a radius per pedestrian rather than as a pairwise diameter:

$$R^t = \frac{1}{2|\mathcal{O}^t|} \sum_{(i,j)\in\mathcal{O}^t} d_{ij}^t, \tag{8}$$

where $|\mathcal{O}^t|$ denotes the number of pairs colliding at time t.

Finally, the overall average radius $\bar{R}$ over the prediction horizon is obtained by averaging R^t across all time steps from $t_{\text{obs}} + 1$ to t_{pred}:

$$\bar{R} = \frac{1}{t_{\text{pred}} - t_{\text{obs}}} \sum_{t=t_{\text{obs}}+1}^{t_{\text{pred}}} R^t. \tag{9}$$

In summary, averaging the radius representing the physically occupied area of pedestrians over the prediction window helps to strike a balance between collision avoidance and displacement-based error, regardless of the level of crowd density. Specifically, a smaller radius can increase the likelihood of collisions, while a larger radius can lead to higher displacement errors. Figure 3d illustrates an example where using a radius value of $0.163\,\mathrm{m}$ helps mitigate the false collision ratio compared to using a fixed radius of $0.2\,\mathrm{m}$ (see fig. 3c). Moreover, using a single radius value for each prediction window reduces computation time during the training process compared to assigning a separate value to each pedestrian at each time step t, without negatively impacting model performance.

Collision Penalty Calculation

The Collision Penalty module aims to penalize predicted positions of the primary pedestrian when collisions with neighboring pedestrians occur during the prediction window. The estimated radius $\bar{R}$ plays a crucial role in the effectiveness of the $\mathcal{CP}$ module. The module operates in four steps:

Step 1: Calculate the pairwise distances $\mathcal{D}_i^t$ between the future position predicted $(\hat{x}_i^t, \hat{y}_i^t)$ of the pedestrian i and $(\hat{x}_j^t, \hat{y}_j^t)$ of each pedestrian j in time step t, where $j \in \mathcal{N}_i^t$, and $\mathcal{N}_i^t$ denotes the set of individuals present in the scene along with the pedestrian i in time step t. For clarity, the Euclidean distance d_{ij}^t between $(\hat{x}_i^t, \hat{y}_i^t)$ and each $(\hat{x}_j^t, \hat{y}_j^t)$ is calculated at the time step t to construct $\mathcal{D}_i^t$. Specifically, $\mathcal{D}_i^t$ is defined as:

$$\mathcal{D}_i^t = \left\{ d_{ij}^t \;\middle|\; j \in \mathcal{N}_i^t,\ i \neq j \right\}, \tag{10}$$

Step 2: Detect potential collisions $\mathcal{C}_i^t$ between the primary pedestrian i and pedestrians $j \in \mathcal{N}_i^t$. A collision is defined based on a distance threshold τ, which is set as twice the estimated radius $\bar{R}$:

$$\tau = 2\bar{R}. \tag{11}$$

In this loss function, a collision between a pedestrian i and a pedestrian j is identified if the pairwise distance $d_{ij}^t \in \mathcal{D}_i^t$ is smaller than the threshold τ. This threshold balances collision avoidance and displacement accuracy. For each detected collision, the corresponding distance d_{ij}^t is stored as a collision weight. That is, this step selects and stores only the distances that indicate collisions (i.e., values below the threshold).

The collision set $\mathcal{C}_i^t$ at time step t is formally defined as:

$$\mathcal{C}_i^t = \left\{ d_{ij}^t \;\middle|\; d_{ij}^t < \tau,\ d_{ij}^t \in \mathcal{D}_i^t \right\}. \tag{12}$$

Step 3: Once the collision distances $\mathcal{C}_i^t$ are identified, this step aims to assign penalties to each collision distance. First, the distances are normalized to a standard scale of $[0, 1]$ to facilitate a consistent comparison of the severity of

the collision in different scenarios. Specifically, for each collision detected between the primary pedestrian i and a pedestrian j at the time step t, the pairwise distance d_{ij}^t is normalized by dividing it by the collision threshold τ. $\widetilde{\mathcal{C}}_i^t$ is defined as:

$$\widetilde{\mathcal{C}}_i^t = \left\{ \frac{d_{ij}^t}{\tau} \,\middle|\, d_{ij}^t \in \mathcal{C}_i^t \right\}. \tag{13}$$

This results in a set of normalized distances $\widetilde{\mathcal{C}}_i^t$ that reflect the relative severity of the collisions: values close to 0 indicate strong (i.e. very close) collisions, while values closer to 1 indicate less severe interactions near the threshold. Next, the normalized distances ($\widetilde{\mathcal{C}}_i^t$) are converted into collision penalty values, denoted by $\mathcal{P}_i^t$, by applying an inverse mapping:

$$\mathcal{P}_i^t = 1 - \widetilde{\mathcal{C}}_i^t. \tag{14}$$

Alternative formulation: This formulation ensures that the penalty increases with the size of the overlapping area, while distances near the threshold produce smaller penalties.

Step 4: The normalized collision penalties $\mathcal{P}_i^t$ for all predicted future trajectories $\hat{Y}$ are summed to compute the total collision penalty $\mathcal{CP}$. For each predicted trajectory $\hat{Y}_i$, penalties are added in all future time steps. Then, the total penalties of all trajectories are summed to produce the final $\mathcal{CP}$, given by:

$$\mathcal{CP} = \sum_i \sum_{t=t_{\text{obs+1}}}^{t_{\text{pred}}} \mathcal{P}_i^t, \tag{15}$$

3.3 Evaluation Metrics

To evaluate the performance of the proposed model and the comparison models (baseline models), two distance-based metrics [42, 43], ADE and the final displacement error (FDE), are utilized. ADE and FDE serve as standard measures for quantifying the displacement error of the predicted trajectories. However, these metrics do not account for potential overlaps or collisions within a scene. Therefore, the Collision Rate (CR) metric [29] is also used to measure the collision rate among predicted trajectories. These metrics are defined below.

- **ADE**: Measures the average Euclidean distance between predicted and ground-truth trajectories over all time steps. It is defined as:

$$\text{ADE} = \frac{1}{M(t_{\text{pred}} - t_{\text{obs}})} \sum_{i=1}^{M} \sum_{t=t_{\text{obs}}+1}^{t_{\text{pred}}} \sqrt{(x_i^t - \hat{x}_i^t)^2 + (y_i^t - \hat{y}_i^t)^2}, \tag{16}$$

  where M is the total number of predicted future trajectories across all scenes in the test set.

- **FDE**: Calculates the Euclidean distance between the predicted final position and the ground-truth final position at end of the prediction window t_{pred}.

$$\text{FDE} = \frac{1}{M} \sum_{i=1}^{M} \sqrt{(x_i^{t_{\text{pred}}} - \hat{x}_i^{t_{\text{pred}}})^2 + (y_i^{t_{\text{pred}}} - \hat{y}_i^{t_{\text{pred}}})^2}. \tag{17}$$

- **CR**: Measures the percentage of predicted trajectories that result in at least one collision. A predicted trajectory $\hat{Y}_i$ for person i is considered a collision trajectory if, at any future time step t, there is at least one collision with a neighboring person j in the predicted scene. Equation (5) determines whether a collision occurs between pedestrians i and j at a given time step t. For example, in Figure 4, the predicted trajectories $\hat{Y}_1$ and $\hat{Y}_2$ are considered collision trajectories because there is an overlap between person 1 and person 2 at time step $t_{\text{obs+5}}$. In contrast, $\hat{Y}_3$ does not collide with any neighbor at any time step. This metric reflects the model's ability to learn collision avoidance behavior. The CR is computed as:

$$\text{CR} = \frac{1}{|\mathcal{S}|} \sum_{\hat{Y} \in \mathcal{S}} \text{COL}(\hat{Y}), \tag{18}$$

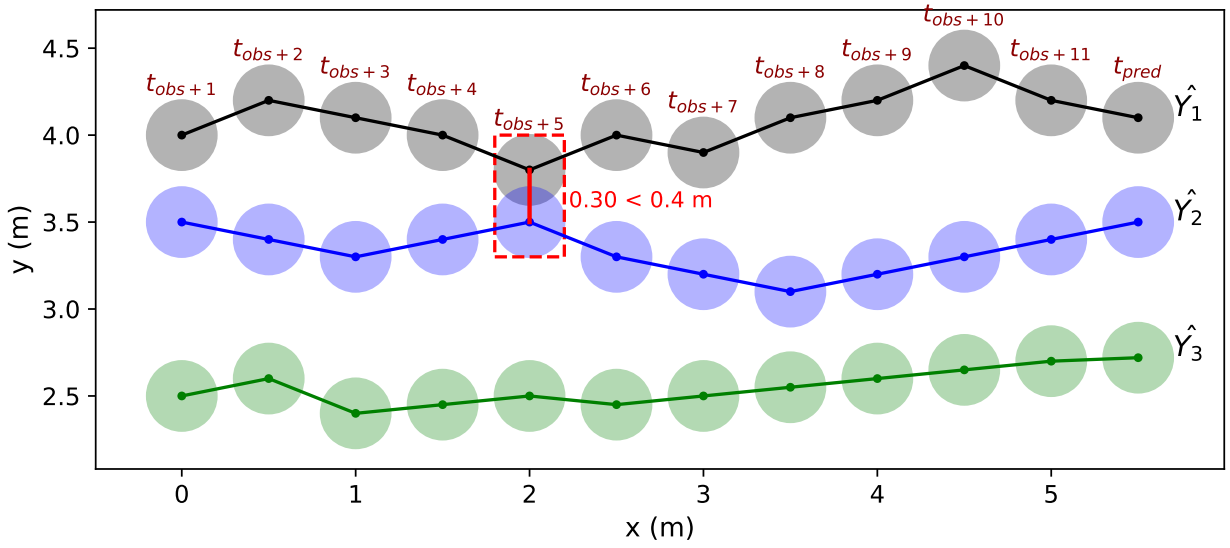

Figure 4: Illustration of collision rate (CR) computation. Each circle represents the space occupied by a person (with radius $0.2\,\mathrm{m}$). A collision occurs when two circles overlap. In this example, $\hat{Y}_1$ and $\hat{Y}_2$ collide at $t_{\text{obs}+5}$, while $\hat{Y}_3$ remains collision-free. Thus, the CR for this scene is $2/3$.

where $\mathcal{S}$ is the set of all scenes in the test set, and $\hat{Y} = \{\hat{Y}_1, \hat{Y}_2, \ldots, \hat{Y}_n\}$ represents the predicted future trajectories of all n individuals in the corresponding scene over the prediction window from $t_{\text{obs}+1}$ to t_{pred}. The function $\text{COL}(\hat{Y})$ is defined as:

$$\text{COL}(\hat{Y}) = \frac{1}{n}\sum_{i=1}^{n} \min\left(1, \sum_{\substack{j=1 \\ j\neq i}}^{n} \sum_{t=t_{\text{obs}}+1}^{t_{\text{pred}}} \left[\sqrt{(\hat{x}_i(t)-\hat{x}_j(t))^2 + (\hat{y}_i(t)-\hat{y}_j(t))^2} < 0.4\right]\right), \tag{19}$$

if we refer to P as $\sqrt{(\hat{x}_i(t)-\hat{x}_j(t))^2 + (\hat{y}_i(t)-\hat{y}_j(t))^2} < 0.4$, then the Iverson bracket is defined as:

$$[P] = \begin{cases} 1, & \text{if } P \text{ is true,} \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

As illustrated in fig. 4, the $\text{COL}(\hat{Y})$ in the predicted scene from $t_{\text{obs}+1}$ to t_{pred} is $\text{COL}(\{\hat{Y}_1, \hat{Y}_2, \hat{Y}_3\}) = \frac{2}{3}$, since two trajectories ($\{\hat{Y}_1, \hat{Y}_2\}$) out of three result in collisions. Because this example contains only one scene (prediction window), the overall CR is also $\frac{2}{3}$.

4 Experiments and Results

4.1 Datasets

This section aims to create the data sets required for training and evaluating the proposed model, the baseline methods, and the current approaches. Covers the data source and the necessary preparation steps to introduce the required datasets, including homogeneous datasets categorized into low density (lowD), medium density (mediumD), high density (highD) and very high density (veryHD) and a heterogeneous dataset (allD).

4.1.1 Data Source

The data of the pedestrian trajectory extracted and utilized in the most relevant approach (TTC-Social LSTM) [25] serves as the primary data source for the creation of the data sets. Initially, videos were recorded with a top view camera during Lyon's Festival of Lights 2022 [44], specifically within a region $9\,\mathrm{m} \times 6.5\,\mathrm{m}$ on Place des Terreaux, highlighted by the red rectangle in fig. 5. Subsequently, the PeTrack software [45] was used to accurately extract

Figure 5: Top view of Lyon's Festival of Lights 2022, with the trajectory tracking region highlighted in red. Figure from [44].

pedestrian trajectories from recorded footage of the tracking region. The extracted pedestrian coordinates (x, y) over times are measured in meters.

A total of 5,195 individual raw trajectories were collected, with an average duration of $12.38\,\mathrm{s}$, a mean velocity of $0.62\,\mathrm{m/s}$ and an average density of $0.95\,\mathrm{pedestrians/m^2}$, with three time steps per second. The trajectory data captures unidirectional and bidirectional pedestrian flows and covers a wide range of densities, from $0.2\,\mathrm{pedestrians/m^2}$ to $2.2\,\mathrm{pedestrians/m^2}$. More details of the data can be found in Ref. [25].

4.1.2 Datasets Preparation

Generating the required datasets from raw trajectory data involves two main steps: trajectory segmentation and density-based classification. These steps rely on key parameter values, including segmented trajectory length, observed trajectory length, future trajectory length, stride value, and density levels, as defined in the main baseline approach.

The trajectory segmentation step applies a slicing strategy with a stride $\Delta = 12$ time steps to generate the segmented trajectories a_i from each raw trajectory $\mathcal{A}_i = \{(x_i^t, y_i^t) \mid t = t_1, t_2, \ldots, t_T\}$ of pedestrian i, where T is the total number of time steps in the raw trajectory. Each generated segment $\mathrm{a}_i(t_1)$ of person i represents $L = 21$ consecutive time steps ($7\,\mathrm{s}$) started from time step t_1. The slicing strategy creates overlapping between consecutive trajectory segments, making the most effective use of the raw trajectory data. The generated segments a_i from the raw trajectory $\mathcal{A}_i$ are described as:

$$\text{Slicing}(\mathcal{A}_i(t_1)) = \{\mathrm{a}_i(t_{\text{init}}) \mid \text{init} = t_{1+(k-1)\times\Delta}, k = 1, 2, \ldots K\}, \tag{21}$$

where K is the total number of trajectories generated, and K is determined as:

$$K = \left\lfloor \frac{T - L}{\Delta} \right\rfloor + 1, \tag{22}$$

and $a_i(t_1)$ mathematically defined as:

$$a_i(t_1) = \left\{ \left(x_i^{t_{1+l}}, y_i^{t_{1+l}} \right) \mid l = 0, 1, \ldots, 20 \right\} \tag{23}$$

The first 9 time steps ($3\,\mathrm{s}$) of each segment $a_i(t_1)$, spanning from t_1 to t_{1+8}, for the observed trajectory X_i. The remaining 12 time steps ($4\,\mathrm{s}$), spanning from t_{1+9} to t_{1+20}, for the future trajectory Y_i. Finally, in this step, the heterogeneous density dataset (allD) is prepared as follows: 1) the raw trajectories $\mathcal{A}_i$ are divided based on the scenes into 70% for the training set and 15% each for the validation and test sets, and 2) the slicing strategy is applied independently to each set to generate the trajectories a_i, which represent the main samples in the sets. As shown in table 1, this dataset consists of a total of $8,341$ trajectories (a_i), divided into $5,773$ for the training set, $1,317$ for the

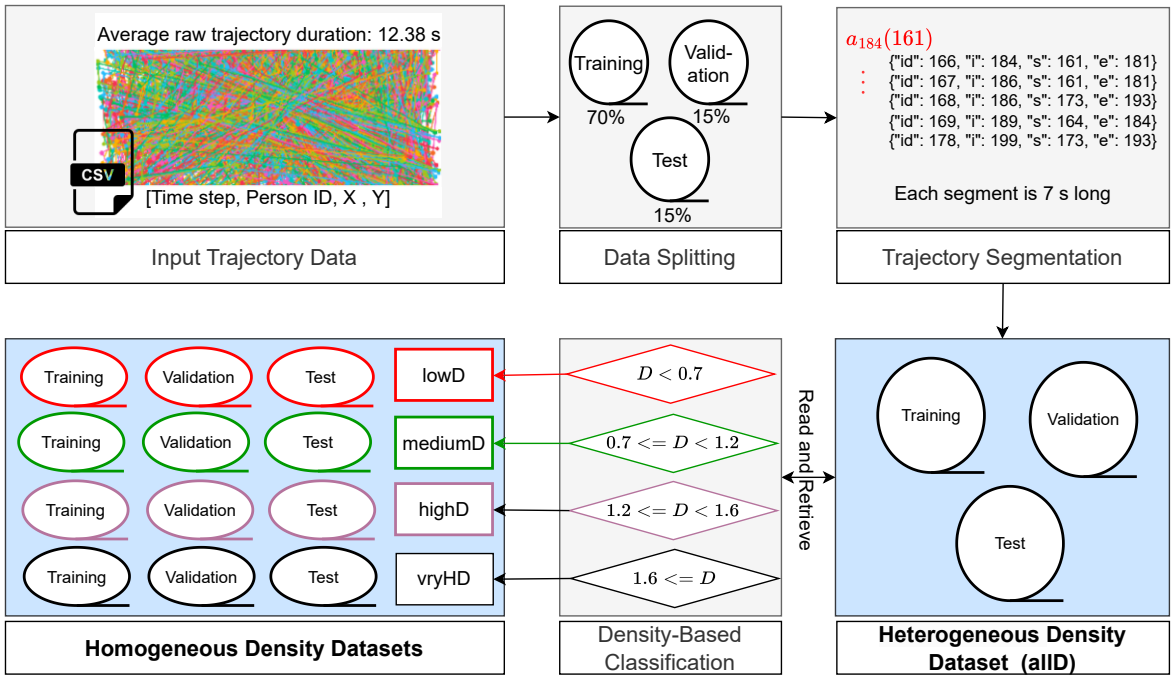

Figure 6: Dataset preparation flowchart. In the trajectory segmentation step, $a_{184}(161)$ denotes the trajectory segment of person ID 184 starting from frame 161. Here, i refers to the person ID, s to the initial frame (start time step), and e to the final time step. Time steps are represented by frame orders. In the density classification step, D denotes the density.

validation set, and $1,251$ for the test set. It is important to note that the splitting is performed first to ensure that there is no overlap of trajectories between the sets.

Table 1: Dataset distribution across training, validation, and test sets (trajectory).

Dataset	**Training**	**Validation**	**Test**	**Total**
allD	5773	1317	1251	8341
lowD	440	101	96	637
MediumD	2021	469	493	2983
highD	2288	298	352	2938
veryHD	1024	449	310	1783

The second step, density-based classification, aims at preparing homogeneous density datasets from allD. Classifies trajectories a_i into four density levels based on the density of the scenes that they traverse. Then it generates lowD, mediumD, highD, and veryHD data sets. To achieve this purpose: 1) calculate the area of the scene at each time step t in $a_i(t_1)$ using the coordinates of the trajectories of the individuals present in that scene. 2) compute the scene density Density(a_i^t) at each time step t within the segmented trajectory $a_i(t_1)$ by dividing the number of individuals in the scene by its area at time step t. 3) compute the average density of a trajectory $a_i(t_1)$, which is calculated as:

$$\text{AverageDensity}(a_i(t_1)) = \frac{1}{21}\sum_{t=t_1}^{t_{1+20}} \text{Density}(a_i^t). \tag{24}$$

4) classify the trajectory $a_i(t_1)$ into one of the four datsets based on its average density:

$$\text{Class}(a_i(t_1)) = \begin{cases} \text{lowD}, & \text{if AverageDensity}(a_i(t_1)) < 0.7\,\text{ped/m}^2, \\ \text{mediumD}, & \text{if } 0.7 \leq \text{AverageDensity}(a_i(t_1)) < 1.2\,\text{ped/m}^2, \\ \text{highD}, & \text{if } 1.2 \leq \text{AverageDensity}(a_i(t_1)) < 1.6\,\text{ped/m}^2, \\ \text{veryHD}, & \text{if AverageDensity}(a_i(t_1)) \geq 1.6\,\text{ped/m}^2. \end{cases} \tag{25}$$

Table 1 presents the number of trajectories $a_i(t_1)$ in the training, validation, and test sets in the four homogeneous density data sets.

4.2 Implementation Details

All experiments in this study were performed on a supercomputer[1] at the Jülich Supercomputing Centre using the TrajNet++ benchmark [29] implemented in PyTorch. To ensure consistency and fair comparison, all models, including the proposed model and those used for comparison, were trained and evaluated within the same framework using identical datasets (section 4.1), evaluation metrics (section 3.3), and hyperparameters.

The default values were used for most hyperparameters as defined in the original implementation of the models within the benchmark framework, consistent with the baseline approach. Specifically, we used a learning rate of 0.001 with the Adam optimizer. The observation duration was set to 9 time steps and the prediction length to 12 time steps. The collision check radius was set to $0.2\,\text{m}$. The batch size was fixed at 8. Training was carried out for up to 15 epochs or until interrupted by an early stopping mechanism with a patience of 5 epochs.

4.3 Results and Discussion

In this section, several experiments were conducted to evaluate the performance of the introduced model. These experiments are categorized as follows: (1) evaluation across various homogeneous density levels, (2) evaluation on a heterogeneous density distribution, (3) analysis of the impact of dynamic occupied space, and (4) comparison with state-of-the-art models.

4.3.1 Evaluation on Homogeneous Density Scenes

To evaluate the effectiveness of the proposed model across varying levels of homogeneous scene density, its performance was compared to that of two other models: TTC-Social LSTM and ADE-Social LSTM. The latter uses only the ADE as its loss function, without any collision penalty. It serves as the main baseline for evaluating the performance of the proposed model and TTC-Social LSTM in terms of ADE, FDE, and CR. The evaluation was carried out using four datasets, each representing a distinct and uniformly distributed density level: lowD, mediumD, highD, and veryHD.

As shown in table 2, the new model consistently outperformed the ADE-Social LSTM across all density levels in terms of ADE, FDE, and CR. For instance, in the lowD dataset, it achieved a significant CR reduction of $17.72\,\%$, along with improvements of $3.6\,\text{cm}$ in ADE and $7.3\,\text{cm}$ in FDE. Consistently, improvements in ADE, FDE, and CR were observed in the mediumD, highD, and veryHD scenarios.

The comparison between TTC-Social LSTM and our model demonstrated the superiority of the proposed model, particularly in reducing CR, while enhancing displacement accuracy across all density levels. In contrast, TTC-Social LSTM struggled to reduce CR without increasing ADE and FDE, especially in the highD and veryHD datasets. Furthermore, in the lowD setting, the introduced model achieved a substantial reduction $17.72\,\%$ in CR, compared to $3.12\,\%$ with TTC-Social LSTM, while also achieving better ADE and FDE values. In the mediumD dataset, although TTC-Social LSTM slightly outperformed the proposed model in displacement errors, the developed model obtained a much greater reduction in CR, $16.91\,\%$ versus $8.61\,\%$.

The collision weight λ used in the loss functions of our model and the TTC-Social LSTM was experimentally tuned to minimize CR without increasing either ADE or FDE (see eq. (6)). Table 2 presents the optimal values λ for each data set for both the presented model and TTC-Social LSTM. A higher λ typically leads to a lower CR but often at the cost of increased displacement errors. Therefore, it is crucial to adapt λ to maintain a balance between reducing CR and preserving the performance of ADE and FDE. Figure 7 illustrates the impact of different values λ on ADE, FDE, and CR for both models. In particular, the case where $\lambda = 0$ corresponds to ADE-Social LSTM, indicating the absence of a collision penalty in the loss function.

[1] JUWELS Booster is a multi-petaflop modular supercomputer operated by the Jülich Supercomputing Center at Forschungszentrum Jülich. For further details, refer to: https://www.fz-juelich.de/en/ias/jsc/systems/supercomputers/juwels

Table 2: Performance evaluation across homogeneous density levels (lowD, mediumD, highD, and veryHD). ↓ indicates improvement in the corresponding metric compared to ADE-Social LSTM, while ↑ indicates a decline in performance.

Model	lowD		mediumD		highD		veryHD	
	Value	Difference	Value	Difference	Value	Difference	Value	Difference
ADE-Social LSTM								
ADE (m)	0.499	–	0.345	–	0.241	–	0.259	–
FDE (m)	0.949	–	0.671	–	0.418	–	0.456	–
CR (%)	40.62	–	29.21	–	33.8	–	51.29	–
TTC-Social LSTM								
Optimal λ	0.001	–	0.002	–	0.001	–	0.002	–
ADE (m)	0.469	-0.03 ↓	0.307	-0.038 ↓	0.251	0.01 ↑	0.319	0.06 ↑
FDE (m)	0.904	-0.045 ↓	0.549	-0.122 ↓	0.435	0.017 ↑	0.577	0.121 ↑
CR (%)	37.5	-3.12 ↓	20.6	-8.61 ↓	19.3	-14.5 ↓	38.7	-12.59 ↓
Our model								
Optimal λ	0.01	–	0.01	–	0.001	–	0.002	–
ADE (m)	0.463	-0.036 ↓	0.323	-0.022 ↓	0.239	-0.002 ↓	0.238	-0.021 ↓
FDE (m)	0.876	-0.073 ↓	0.621	-0.05 ↓	0.413	-0.005 ↓	0.42	-0.036 ↓
CR (%)	22.9	-17.72 ↓	12.3	-16.91 ↓	25.5	-8.3 ↓	47.4	-3.89 ↓

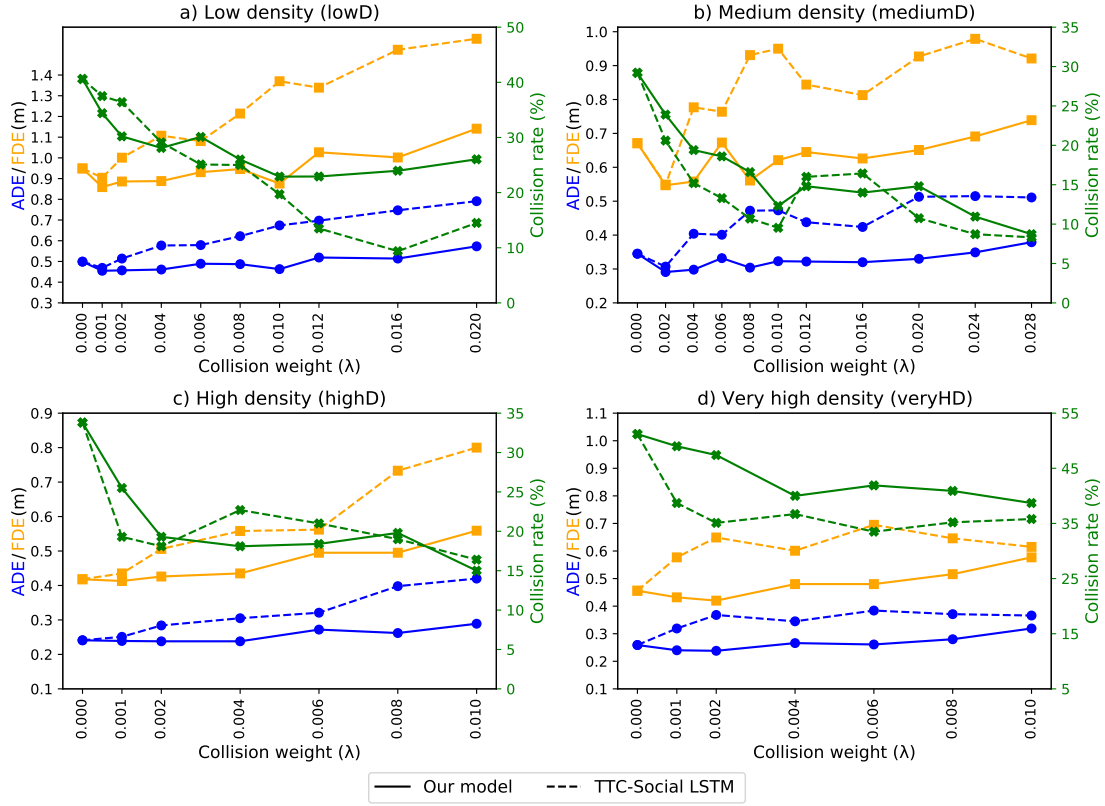

Figure 7: Impact of collision weight (λ) on models' performance in terms of ADE, FDE, and CR across four scene density levels: a) lowD, b) mediumD, c) highD, and d) veryHD. The case $\lambda = 0$ corresponds to ADE-Social LSTM, where the collision penalty is not applied.

In summary, the new model successfully reduces collisions and improves displacement accuracy across all homogeneous density levels.

4.3.2 Evaluation on Heterogeneous Density Scenes

To assess the performance of the proposed model under diverse pedestrian density conditions, it was trained and evaluated alongside ADE-Social LSTM and TTC-Social LSTM using the heterogeneous dataset (allD).

Table 3: Performance evaluation across heterogeneous density levels (allD). ↓ indicates improvement in a metric, while ↑ indicates a decline. "diff." shows the difference between ADE-Social LSTM and the other two models: the new model and TTC-Social LSTM.

Model	**allD**		**lowD**		**mediumD**		**highD**		**veryHD**	
	Value	**Diff.***	**Value**	**Diff.**	**Value**	**Diff.**	**Value**	**Diff.**	**Value**	**Diff.**
ADE-Social LSTM										
ADE (m)	0.257	–	0.402	–	0.281	–	0.224	–	0.212	–
FDE (m)	0.473	–	0.769	–	0.538	–	0.396	–	0.367	–
CR (%)	39.3	–	41.6	–	29	–	36.9	–	57.4	–
TTC-Social LSTM (Optimal λ: 0.001)										
ADE (m)	0.288	$+0.031$ ↑	0.445	$+0.043$ ↑	0.321	$+0.04$ ↑	0.258	$+0.034$ ↑	0.221	$+0.009$ ↑
FDE (m)	0.532	$+0.06$ ↑	0.85	$+0.081$ ↑	0.612	$+0.074$ ↑	0.464	$+0.068$ ↑	0.386	$+0.019$ ↑
CR (%)	26.1	-13.2 ↓	31.3	-10.3 ↓	16.6	-12.4 ↓	21.6	-15.3 ↓	44.8	-12.6 ↓
Our model (Optimal λ: 0.003)										
ADE (m)	0.248	-0.01 ↓	0.368	-0.034 ↓	0.274	-0.007 ↓	0.219	-0.005 ↓	0.203	-0.009 ↓
FDE (m)	0.445	-0.028 ↓	0.678	-0.091 ↓	0.507	-0.031 ↓	0.376	-0.02 ↓	0.355	-0.012 ↓
CR (%)	29.9	-9.4 ↓	35.4	-6.2 ↓	21.7	-7.3 ↓	26.1	-10.8 ↓	45.8	-11.6 ↓

As shown in table 3, the proposed model consistently improved CR, ADE and FDE in mixed density settings. In contrast, while TTC-Social LSTM reduced the CR, it failed to maintain the displacement accuracy, as both ADE and FDE increased. In particular, the proposed model decreased the CR by $9.4\,\%$ while simultaneously reducing displacement errors. On the other hand, TTC-Social LSTM improved CR by $13.2\,\%$, but this came at the cost of increasing displacement errors. Moreover, the introduced model trained on the allD dataset demonstrated strong generalization when evaluated on the individual homogeneous density test sets: lowD, mediumD, highD, and veryHD. It consistently reduced CR and displacement errors across all test scenarios. In contrast, the TTC-Social LSTM model failed to achieve reductions in displacement errors for any of the test sets. The optimal collision weights were experimentally determined to be 0.01 for our model and 0.001 for TTC-Social LSTM (see fig. 8).

These results highlight the effectiveness of the model in simultaneously reducing the CR and improving displacement accuracy under heterogeneous density conditions. This performance is primarily attributed to the new loss function. As a result, the model emerges as a strong candidate for predicting real-world pedestrian trajectory in complex and dynamic crowd environments.

4.3.3 Impact of Dynamic Occupied Space on Social LSTM Performance

A key contribution of the proposed loss function is its ability to dynamically estimate the radius of the space occupied by pedestrians based on the scene's density. This section aims to investigate how the incorporation of this dynamically occupied space affects the performance of the proposed model.

To achieve this purpose, we introduced a new baseline model called Static Occupied Space Social LSTM (SOS-Social LSTM), which differs from the proposed model only in its use of a fixed occupied space with a radius of $0.2\,\mathrm{m}$, regardless of scene density. This static representation of occupied space has been used in several models in the literature, such as those in Refs. [29, 25]. The baseline model was trained and evaluated using the same datasets, parameters, and experimental setup as the proposed model. Figure 9 compares the performance of ADE-Social LSTM, SOS-Social LSTM, and our model across heterogeneous (allD) and homogeneous (lowD, mediumD, highD, veryHD) datasets. The results clearly demonstrated the effectiveness of dynamic spatial modeling in improving CR, ADE and FDE across

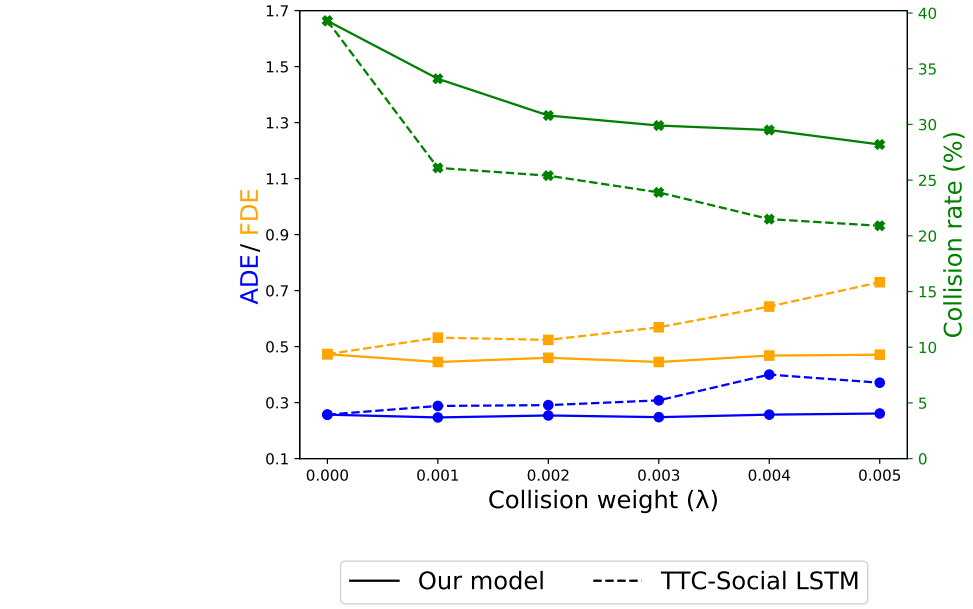

Figure 8: Impact of varying the collision weight (λ) on ADE (m), FDE (m), and CR (%) using the heterogeneous dataset (allD). When (λ) = 0, the setup corresponds to ADE-Social LSTM.

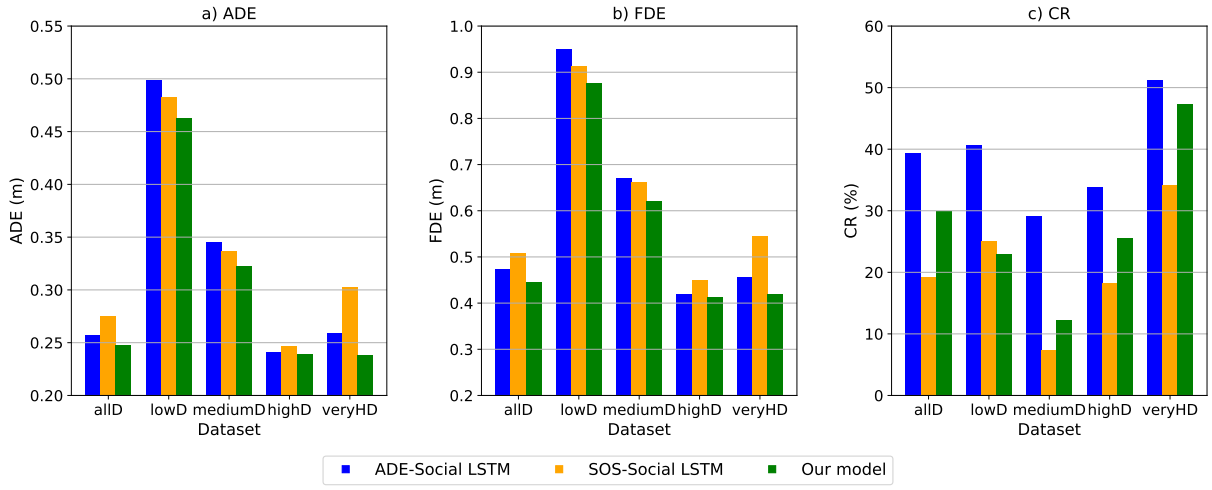

Figure 9: Impact of dynamic occupied space mechanism on the performance of the proposed model across heterogeneous (allD) and homogeneous (lowD, mediumD, highD, veryHD) crowd density scenarios.

all density conditions. In contrast, the SOS-Social LSTM failed to achieve this improvement in the allD, highD, and veryHD datasets. For example, SOS-Social LSTM reduces the CR in these three datasets by $23.04\,\%$, $15.63\,\%$, and $14.1\,\%$, respectively (see Figure 9c). However, this reduction is accompanied by an increase in displacement errors, as shown in Figures 9a,b.

Additionally, under the lowD and mediumD datasets, SOS-Social LSTM reduced the CR while also improving displacement accuracy. This suggests that a fixed occupied space radius of $0.2\,\mathrm{m}$ could be suitable for low- and medium-density environments. Despite these results, the introduced outperformed SOS-Social LSTM in the lowD setting by achieving a $2.8\,\%$ lower collision rate, a $2\,\mathrm{cm}$ lower ADE, and approximately $4\,\mathrm{cm}$ lower FDE. In the mediumD dataset, the new model achieved better ADE and FDE, while SOS-Social LSTM outperformed the proposed model in terms of collision rate. The main reason for this result is that, in high-density and heterogeneous scenes, a fixed radius becomes inefficient because the actual space occupied by pedestrians is often smaller than $0.2\,\mathrm{m}$ in crowded

conditions. Therefore, adapting the radius to reflect the realistic spacing of the crowd helps balance collision avoidance with displacement accuracy.

In summary, incorporating a dynamic occupied space into the loss function enables the proposed model to adapt to varying crowd densities. This adaptive approach successfully reduces collisions and improves displacement accuracy across both homogeneous settings—from low to high density—and heterogeneous environments. In contrast, the static occupied space fails to minimize collisions and displacement errors simultaneously, especially in dense and heterogeneous scenes.

4.4 Comparison with State-of-the-Art Approaches

For further evaluation, the proposed model is compared with two state-of-the-art methods on the homogeneous and heterogeneous datasets. These approaches are Vanilla LSTM [18] and Social-GAN [21].

The results illustrated in table 4 demonstrated that the new model significantly reduced the CR across all datasets compared to Vanilla LSTM and Social-GAN. On the lowD dataset, it lowered CR by $22.9\,\%$ and $15.6\,\%$ relative to Vanilla LSTM and Social GAN, respectively. Even in the challenging veryHD setting, where managing dense crowds is difficult, the proposed model achieved a CR of only $47.4\,\%$, clearly outperforming Vanilla LSTM ($61.6\,\%$) and Social GAN ($53.9\,\%$). In addition to its advantage in CR, our model performed as well as—or better than—the other models in terms of displacement accuracy (ADE and FDE) on most homogeneous datasets. It achieved lower displacement errors than both approaches in the lowD and veryHD datasets, and performed similarly to Vanilla LSTM in veryHD while still outperforming Social GAN. On the mediumD dataset, the introduced showed slightly higher displacement errors compared to Vanilla LSTM and Social GAN. However, it still achieved a major improvement in CR, reducing it by more than $20\,\%$ compared to both approaches. This highlighted the model's strength in balancing collision reduction with prediction accuracy. Moreover, on the allD (heterogeneous) dataset, the proposed model achieved a strong balance between displacement accuracy and collision rate. It reduced CR by $14.7\,\%$ and $9.6\,\%$, while also improving displacement accuracy compared to Vanilla LSTM and Social GAN, respectively.

To sum up, our model significantly outperformed Vanilla LSTM and Social GAN in reducing collision instances across all homogeneous and heterogeneous datasets, while also achieving better displacement accuracy in most settings. This improvement is mainly attributed to the model's ability to explicitly learn the influence of nearby pedestrians and realistic collision avoidance.

Table 4: Comparison with state-of-the-art approaches. *Diff.* refers to the difference between the introduced model and the corresponding method. A ↓ indicates that the compared method performs better in the given metric, while a ↑ indicates that our model LSTM performs better.

Approach	**Metric**	**lowD** Value (Diff.*)	**mediumD** Value (Diff.*)	**highD** Value (Diff.*)	**VeryHD** Value (Diff.*)	**allD** Value (Diff.*)
Our model	ADE (m)	0.46	0.32	0.24	0.24	0.25
	FDE (m)	0.88	0.62	0.41	0.42	0.45
	CR (%)	22.9	12.3	25.5	47.4	29.9
Vanilla LSTM	ADE (m)	0.47 (0.01 ↑)	0.30 (-0.02 ↓)	0.25 (0.01 ↑)	0.24 (0.00 =)	0.28 (0.03 ↑)
	FDE (m)	0.92 (0.04 ↑)	0.56 (-0.06 ↓)	0.45 (0.04 ↑)	0.42 (0.00 =)	0.53 (0.08 ↑)
	CR (%)	45.8 (22.9 ↑)	33.2 (20.9 ↑)	42.0 (16.5 ↑)	61.6 (14.2 ↑)	44.6 (14.7 ↑)
Social GAN	ADE (m)	0.46 (0.00 =)	0.29 (-0.03 ↓)	0.26 (0.02 ↑)	0.28 (0.04 ↑)	0.28 (0.03 ↑)
	FDE (m)	0.90 (0.02 ↑)	0.55 (-0.07 ↓)	0.47 (0.06 ↑)	0.49 (0.07 ↑)	0.51 (0.07 ↑)
	CR (%)	38.5 (15.6 ↑)	32.8 (20.5 ↑)	34.4 (8.9 ↑)	53.9 (6.5 ↑)	39.5 (9.6 ↑)

5 Conclusion and Future Works

This paper introduced a novel deep learning model designed to predict accurate, realistic, and collision-free pedestrian trajectories in dynamic and crowded environments. By integrating a new DOS-based loss function into the Social LSTM architecture, the proposed model explicitly learns to avoid pedestrian collisions while improving high displacement accuracy. The proposed loss function adaptively adjusts the occupied space radius by pedestrians based on the scene density, allowing the model to generalize across a broad range of crowd conditions, from sparse to dense crowds and from homogeneous to heterogeneous density distributions. The proposed model was trained and evaluated on

five datasets derived from real pedestrian trajectories, including four homogeneous density levels (low, medium, high, and very high density) and one heterogeneous scenario. The experimental findings showed that the model successfully reduced the number of collision instances while improving the accuracy of the displacement across all datasets. Furthermore, it consistently outperformed five baselines and state-of-the-art models in terms of both collision rate and displacement errors in most experiments.

While the proposed model effectively captures human–human interactions, it does not explicitly account for the influence of environmental constraints on pedestrian movement. To enhance its generalizability across varied environments, future work will focus on extending the model to learn human-environment interactions. This will enable the model to reflect real-world behavior in complex and structured spaces with more accuracy.

Data and Code Availability

The raw trajectory data used in this study is available at the following website: https://madras-data-app.streamlit.app/ (accessed on 1 April 2024). In addition, the TrajNet++ benchmark, which was used in the implementation of this model and for dataset construction, is described at: https://thedebugger811.github.io/posts/2020/03/intro_trajnetpp/. The code implementing the DOS-based loss function is available upon request.

Acknowledgments
The authors sincerely thank Antoine Tordeux and Raphael Korbmacher for their insightful discussions and for providing us with the trajectory datasets and the TTC-Loss function code.

Conflicts of Interest
The authors declare no conflict of interest.

References

[1] Haifeng Sang, Wangxing Chen, Jinyu Wang, and Zishan Zhao. Rdgcn: Reasonably dense graph convolution network for pedestrian trajectory prediction. *Measurement*, 213:112675, 2023.

[2] Rei Tamaru, Pei Li, and Bin Ran. Enhancing pedestrian trajectory prediction with crowd trip information. *arXiv preprint arXiv:2409.15224*, 2024.

[3] Mengya Zong, Yuchen Chang, Yutian Dang, and Kaiping Wang. Pedestrian trajectory prediction in crowded environments using social attention graph neural networks. *Applied Sciences*, 14(20):9349, 2024.

[4] Honghui Wang, Weiming Zhi, Gustavo Batista, and Rohitash Chandra. Pedestrian trajectory prediction using goal-driven and dynamics-based deep learning framework. *Expert Systems with Applications*, 271:126557, 2025.

[5] Jiajia Xie, Sheng Zhang, Beihao Xia, Zhu Xiao, Hongbo Jiang, Siwang Zhou, Zheng Qin, and Hongyang Chen. Pedestrian trajectory prediction based on social interactions learning with random weights. *IEEE Transactions on Multimedia*, 26:7503–7515, 2024.

[6] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14424–14432, 2020.

[7] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[8] Oscar Gil and Alberto Sanfeliu. Human motion trajectory prediction using the social force model for real-time and low computational cost applications. In *Iberian Robotics conference*, pages 235–247. Springer, 2023.

[9] Carsten Burstedde, Kai Klauck, Andreas Schadschneider, and Johannes Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3-4):507–525, 2001.

[10] Jur Van Den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics Research: The 14th International Symposium ISRR*, pages 3–19. Springer, 2011.

[11] Raphael Korbmacher and Antoine Tordeux. Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):24126–24144, 2022.

[12] Thomas Chatagnon, Antoine Tordeux, and Mohcine Chraibi. Exploring dense crowd dynamics: State of the art and emerging paradigms. *arXiv preprint arXiv:2505.05826*, 2025.

[13] Mohamad Abubaker, Zubayda Alsadder, Hamed Abdelhaq, Maik Boltes, and Ahmed Alia. Rpee-heads benchmark: A dataset and empirical comparison of deep learning algorithms for pedestrian head detection in crowds. *IEEE Access*, 2025.

[14] Ahmed Alia, Mohammed Maree, Mohcine Chraibi, and Armin Seyfried. A novel voronoi-based convolutional neural network framework for pushing person detection in crowd videos. *Complex & Intelligent Systems*, 10(4):5005–5031, 2024.

[15] Bogdan Ilie Sighencea, Rareș Ion Stanciu, and Cătălin Daniel Căleanu. A review of deep learning-based methods for pedestrian trajectory prediction. *Sensors*, 21(22):7543, 2021.

[16] Jing Yang, Yuehai Chen, Shaoyi Du, Badong Chen, and Jose C Principe. Ia-lstm: Interaction-aware lstm for pedestrian trajectory prediction. *IEEE transactions on cybernetics*, 54(7):3904–3917, 2024.

[17] Jiaming Jiang, Kai Yan, Xindong Xia, and Biao Yang. A survey of deep learning-based pedestrian trajectory prediction: Challenges and solutions. *Sensors (Basel, Switzerland)*, 25(3):957, 2025.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[19] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[20] 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.

[21] 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.

[22] Irtiza Hasan, Francesco Setti, Theodore Tsesmelis, Vasileios Belagiannis, Sikandar Amin, Alessio Del Bue, Marco Cristani, and Fabio Galasso. Forecasting people trajectories and head poses by jointly reasoning on tracklets and vislets. *IEEE transactions on pattern analysis and machine intelligence*, 43(4):1267–1278, 2019.

[23] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE international conference on computer vision*, pages 300–311, 2017.

[24] Mehwish Awan, Jitae Shin, and Taeg Keun Whangbo. Trajectory prediction of heterogeneous traffic agents with collision vigilance and avoidance. *IEEE Transactions on Intelligent Vehicles*, 9(1):93–102, 2023.

[25] Raphael Korbmacher and Antoine Tordeux. Toward better pedestrian trajectory predictions: the role of density and time-to-collision in hybrid deep-learning algorithms. *Sensors*, 24(7):2356, 2024.

[26] Honghui Wang, Weiming Zhi, Gustavo Batista, and Rohitash Chandra. Pedestrian trajectory prediction using dynamics-based deep learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15068–15075. IEEE, 2024.

[27] Shiji Yang and Xuezhong Xiao. Pedestrian trajectory prediction model based on self-supervised spatiotemporal graph network. *Intelligent Systems with Applications*, page 200533, 2025.

[28] Xin Chen, Chengrui Huang, Chenhao Wang, and Lisi Chen. Trajectory generation: a survey on methods and techniques. *GeoInformatica*, pages 1–26, 2025.

[29] Parth Kothari, Sven Kreiss, and Alexandre Alahi. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7386–7400, 2021.

[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/CVF conference on computer vision and pattern recognition*, pages 12085–12094, 2019.

[31] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6272–6281, 2019.

[32] Hao Xue, Du Huynh, and Mark Reynolds. Location-velocity attention for pedestrian trajectory prediction. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2038–2047. IEEE, 2019.

[33] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *Advances in neural information processing systems*, 32, 2019.

[34] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9813–9823, 2021.

[35] Meiyun Li, Yuhang Li, Zhihao Gao, Jiapei Wang, and Yangyang Li. Iaha: Intention-aware hybrid attention transformer for trajectory prediction. In *CICTP 2024*, pages 203–213. 2024.

[36] Junyou Zhu, Chao Gao, Ze Yin, Xianghua Li, and Jürgen Kurths. Propagation structure-aware graph transformer for robust and interpretable fake news detection. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4652–4663, 2024.

[37] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. *Conference on Robot Learning (CoRL)*, 155:620–645, 2020.

[38] Xingyu Luo, Junchi Zhu, Bo Dai, and Chao Wang. Gsgformer: A general scene graph transformer for trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20024–20034, 2023.

[39] Hao Song, Yifan Zhu, Yuwei Liu, Lihui Fan, and Chunhua Shen. Tutr: Trajectory unified transformer for pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2281–2290, 2023.

[40] Jingni Song, Feng Chen, Yadi Zhu, Na Zhang, Weiyu Liu, and Kai Du. Experiment calibrated simulation modeling of crowding forces in high density crowd. *Ieee Access*, 7:100162–100173, 2019.

[41] Dapeng Yan, Gangyi Ding, Kexiang Huang, and Tianyu Huang. Generating natural pedestrian crowds by learning real crowd trajectories through a transformer-based gan. *The Visual Computer*, 41(2):1079–1096, 2025.

[42] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4601–4607. IEEE, 2018.

[43] Chen Zhou, Ghassan AlRegib, Armin Parchami, and Kunjan Singh. Trajpred: Trajectory prediction with region-based relation learning. *IEEE Transactions on Intelligent Transportation Systems*, 25(8):9787–9796, 2024.

[44] Oscar Dufour, Huu-Tu Dang, Jakob Cordes, Raphael Korbmacher, Mohcine Chraibi, Benoit Gaudou, Alexandre Nicolas, and Antoine Tordeux. Dense Crowd Dynamics and Pedestrian Trajectories: A Multiscale Field Dataset from the Festival of Lights in Lyon. *Scientific Data*, 12(1):1–10, April 2025.

[45] Maik Boltes, Deniz Kilic, Tobias Schrödter, Tobias Arens, Luke Dreßen, Juliane Adrian, Ann Katrin Boomers, Alica Kandler, Mira Küpper, Arne Graf, Daniel Salden, Ricardo Martin Brualla, Paul Häger, Daniel Hillebrand, Paul Lieberenz, and Janine Klein. Petrack, April 2025.