A learning based pedestrian flow prediction approach with diffusion behavior

Mai, Weiming; Duives, Dorine; Hoogendoorn, Serge

DOI
10.1016/j.trc.2025.105243

Publication date
2025

Document Version
Final published version

Published in
Transportation Research Part C: Emerging Technologies

Transportation Research Part C 179 (2025) 105243

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

A learning based pedestrian flow prediction approach with diffusion behavior☆

Weiming Mai *, Dorine Duives, Serge Hoogendoorn

Department of Transportation and Planning, Delft University of Technology, Stevinweg 1, Delft, 2628CN, Netherlands

ARTICLE INFO

Keywords:
Real-time pedestrian flow prediction
Pedestrian simulation
Interpretable learning-based method
Crowd diffusion
Online learning

ABSTRACT

In public spaces such as city centers, train stations, airports, shopping malls, and multi-modal hubs, accurately predicting pedestrian flow is crucial for effective crowd management e.g. congestion prevention and evacuation planning. Traditional microscopic simulation models offer fine-grained insights by simulating each pedestrian individually, but they are computationally intensive and typically used at the planning and design stage, making them unsuitable for real-time interventions in high-demand scenarios. Macroscopic models, on the other hand, reduce computational cost by aggregating pedestrian behavior and solving partial differential equations, but they typically require estimates of traffic states such as density and speed — quantities that are difficult to measure accurately in practice. Additionally, as the complexity of these physics-based models increases, their computational feasibility for real-time use becomes even more limited. Data-driven (machine learning) models provide a computationally efficient alternative, enhancing real-time prediction capabilities. However, they often require large historical datasets to generalize well, and their performance can degrade under out-of-distribution (OOD) conditions. Moreover, most black-box learning models lack interpretability and domain-specific insights, limiting their practical adoption. In this paper, we propose a novel pedestrian flow prediction model based on the theory of crowd diffusion. Our method estimates flow rates directly from sensor-observed data and infers both Origin–Destination (OD) demand and route choice probabilities to support real-time operations. To address the OOD challenge, we incorporate an online learning mechanism that continuously calibrates model parameters based on incoming observations.

1. Introduction

Predicting and estimating pedestrian movement in public spaces enables operators to anticipate where people will go and how many will be present at any given time. This information is critical for preventing overcrowding and improving the efficiency of evacuations. Existing research largely falls within the descriptive or observational domain, which focuses on assessing, exploring, or predicting evacuation behavior through experimentation or physics-based microscopic simulation models (Haghani, 2020; Rasouli, 2021; Helbing and Molnar, 1995; Geraerts, 2010). However, these models are typically not designed to be dynamically updated during operation, limiting their applicability for real-time crowd management. As a result, they are insufficient for anticipating and mitigating crowd-related risks in fast-evolving scenarios.

☆ This article belongs to the Virtual Special Issue on "Data Analytics and ML".

* Corresponding author.
E-mail address: w.m.mai@tudelf.nl (W. Mai).

https://doi.org/10.1016/j.trc.2025.105243
Received 27 November 2024; Received in revised form 23 May 2025; Accepted 13 June 2025
Available online 11 July 2025

Data-driven models can be employed for real-time predictions because they are pre-trained on historical data to learn complex pedestrian activity patterns. This allows them to make predictions based on only current information, making the process more computationally efficient and capable of responding to real-time situations. There has been a growing interest in the application of deep learning techniques for predicting the flow, for instance of inbound / outbound passengers (Liu et al., 2019) at a train station, or demand prediction in subway networks (Fang et al., 2019). To predict pedestrian flow, these models rely on external factors, such as weather conditions, flow data from nearby stations, and daily or weekly activity patterns, which typically enable prediction horizons extending beyond 15 min. However, in large infrastructures or open areas like city centers, there is a need to forecast dense crowd movements during events like evacuations or entertainment gatherings. Here, predictions are required for the immediate future — often within minutes or even seconds to effectively direct and manage crowds in real-time.

A few studies have analyzed pedestrian flow data to estimate origin–destination (OD) demand at train stations (Jia et al., 2024; Hänseler et al., 2017), but these are not designed for predicting real-time pedestrian traffic states. In Bamaqa et al. (2022), synthetic crowd datasets are generated using agent-based simulations for predicting crowd severity levels. The simulation outputs include agents' positions, speeds, and headings, which are processed to obtain crowd density, speed, and heading direction as inputs for the prediction model. Similarly, Makinoshima and Oishi (2022) employed a particle filter simulation approach for real-time crowd state prediction, using observation data to estimate latent parameters. While both studies leverage simulation tools for crowd analysis and prediction, they are limited to small indoor areas and rely on detailed data, such as agent speeds, densities, and precise positions — information that is challenging to acquire in real-world scenarios. Other approaches, including computer vision-based methods for individual pedestrian trajectory prediction (Zheng, 2015; Karamouzas et al., 2018; Korbmacher and Tordeux, 2022), focus on short-term microscopic trajectory forecasts within confined spaces, which are not applicable for macroscopic prediction in larger environments.

Currently, there is a notable lack of effective data-driven (machine learning) models for macroscopic pedestrian flow prediction in large public infrastructures. This gap stems from two key limitations in existing approaches. First, these models require extensive historical datasets that capture a wide range of conditions, including both routine and emergency scenarios. However, such data are difficult to obtain Manibardo et al. (2022), Li et al. (2022), as pedestrian flows are highly non-stationary — changing rapidly within seconds, and crowd dynamics often do not repeat. As a result, data-driven models must be frequently retrained or updated, which is computationally demanding and undermines their practicality for real-time prediction, especially during critical events (Seedat et al., 2022).

Second, most existing deep learning models function as black boxes, lacking integration of physical principles or interpretable structure. This makes their predictions difficult for practitioners to understand and trust, reducing their applicability in operational contexts. Moreover, their complex architectures are difficult to calibrate, further limiting their usability in high-stakes, real-time decision-making settings.

In this paper, we develop a data-driven pedestrian flow prediction framework that builds upon the basic crowd diffusion model proposed by Liu et al. (2015). Our model consists of two primary components: the route velocity model, which estimates pedestrian flow speed, and the route choice model, which estimates the probability of pedestrians selecting specific routes. By leveraging these components, we can predict pedestrian flow between any two accessible locations in the infrastructure and estimate the origin–destination (OD) matrix, while keeping the prediction process interpretable. Additionally, we integrate online machine learning techniques to handle non-stationary scenarios, enabling the model to adapt to new situations.

In summary, our contributions are as follows.

- We formulate the problem of pedestrian flow prediction in indoor environments/large public areas using graph theory and propose a lightweight, traffic knowledge-infused, data-driven model that utilizes diffusion behavior theory for flow prediction. To the best of our knowledge, this approach has not yet been explored for real-time crowd flow prediction.
- We designed a learning-based route velocity estimation module and a route choice module to estimate the parameters of the diffusion behavior model. Together, these modules enable our prediction model to not only forecast flow but also estimate traffic conditions between sensors, even with incomplete data.
- We present an online learning framework that is adaptable to general data-driven methods to address the out-of-distribution (OOD) issue. Experiments demonstrate the efficacy of this online learning framework.
- To evaluate our model's performance, we develop three comprehensive synthetic datasets and one real-world dataset, encompassing various scenarios that could occur in a public area. Experimental results demonstrate that our model's prediction accuracy is comparable to and sometimes outperforms traditional data-driven models. Additionally, the model can capture general traffic phenomena.

In the following of this paper, we first formulate the flow prediction problem and define the concept of scenario drift in Section 2. Next, we introduce the crowd diffusion model and our extended data-driven crowd diffusion model in Section 3. Section 4 describes the synthetic dataset and the setting of online learning. Section 5 and Section 7 analyzes the prediction performance of our model and evaluates its reliability in terms of the estimated velocity and OD matrix. Lastly, we close with a discussion of the main findings and potential enhancements of our new modeling framework.

2. Problem formulation

In this section, we first formulate the pedestrian flow prediction problem and then introduce the concept of scenario drift, which refers to the OOD scenarios mentioned earlier.

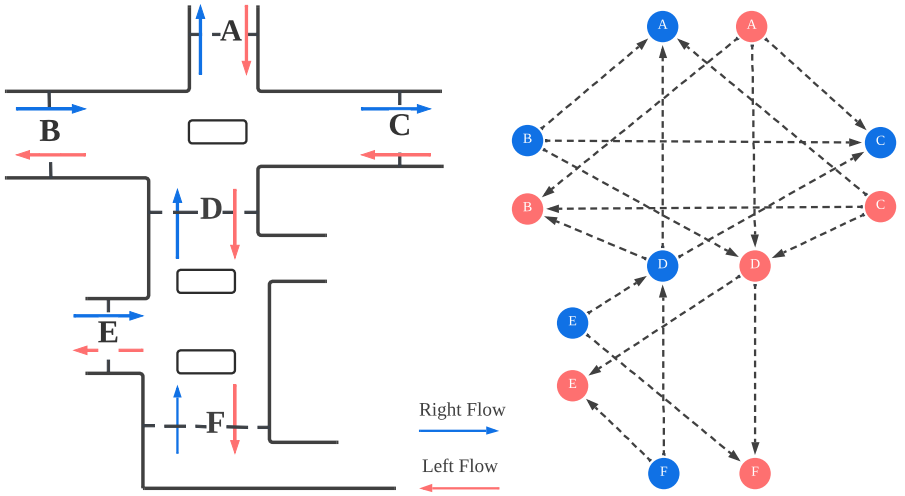


Fig. 1. Graph representation of the sensor network.

2.1. Mathematical definition of the flow prediction problem

This paper focuses on pedestrian flow prediction in large infrastructure or public spaces. Pedestrian flow represents directed crowd movement and can be more easily and accurately acquired by most surveillance systems compared to speed and density data. It is defined as the number of people crossing a line (i.e., cross-section) within a certain period. Suppose we have a public area as depicted in Fig. 1(a). Each of the sensors records crowd flow in real-time, capturing instant changes in the flow rate. They are installed in the hallway (black dashed lines in the corridor) within an infrastructure. Although pedestrians crossing a line can move in multiple directions, we define only two directions for a line sensor without loss of generality (Tordeux et al., 2018; Lilasathapornkit et al., 2022): right-directed flow (blue arrow) and left-directed flow (red arrow).

Many studies have used graphs to model road networks (Han et al., 2023) or large infrastructures such as buildings or transit terminals (Løvås, 1994; Borgers and Timmermans, 1986) to analyze traffic flow. In a similar manner, this paper represents the sensor network as a graph (Fig. 1(b)). These sensors divide the infrastructure into multiple segments, where each segment is bounded by sensors such that all entering and exiting pedestrians can be detected. The distance between sensors varies from 20 meters to 100 m.

Formally, we define the sensor network as a directed graph $\mathcal{G}\{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_1, \ldots, v_{|\mathcal{V}|}\}$ is the set of sensor nodes and $\mathcal{E} = \{e_{ij}, v_j \text{ is downstream of } v_i\}$ is the set of directed edges indicating the direction of flow from the upstream nodes to the downstream nodes. Note that each node in the graph only captures the uni-directional flow of a line sensor, and two nodes are connected if these points are accessible within the infrastructure.

The flow of a node is influenced by its upstream node. The flow starts from the root node and spreads through the network. We consider the root node as independent from the other nodes in the network, making it unpredictable based solely on flow information within the network. For example, we cannot predict how many people will arrive in the building just based on the current flow information. In contrast, nodes with ancestors are dependent and thus predictable. The flow prediction task can hence be described as follows: Given the historical flow data $[q^{t-M'+1}, \ldots, q^t] \in \mathbb{R}^{|\mathcal{V}| \times M'}$, where $q^k \in \mathbb{R}^{|\mathcal{V}|}$ is the vector of flow rates from all sensors at time step k, predict the future flow rate $[q^{t+1}, \ldots, q^{t+M}] \in \mathbb{R}^{|\mathcal{V}_{down}| \times M}$ of the downstream nodes. Here, M' is the size of the sliding window, M is the prediction horizon, and t is the current time step.

2.2. Scenario drift

Pedestrian movement patterns vary due to changing intentions and conditions over time, such as different times of day, events, or sudden changes in the environment. These changes manifest themselves as dynamically changing OD demands varying pedestrian activity routes, and dynamically changing walking speeds. Additionally, unexpected factors, such as adverse events and disruptions can also lead to significant changes. For instance, in a train station, a sudden delay may shift passenger movement patterns, causing congestion in previously underutilized areas. Traditional offline models may fail to adapt and produce inaccurate flow predictions. The scenario drift refers to the changes in the underlying patterns and distributions of pedestrian flows influenced by these factors. The mathematical definition can be analogized to the concept drift problem (Wares et al., 2019), it can be defined as: $p_{t_0}(x, y) \neq p_{t_1}(x, y)$, where p_t is the distribution of the pedestrian flow in scenario t with input variables $x = [q^{t-M'+1}, \ldots, q^t]$ and ground truth $y = [q^{t+1}, \ldots, q^{t+M}]$. The changes in the scenarios pose challenges for crowd flow prediction.

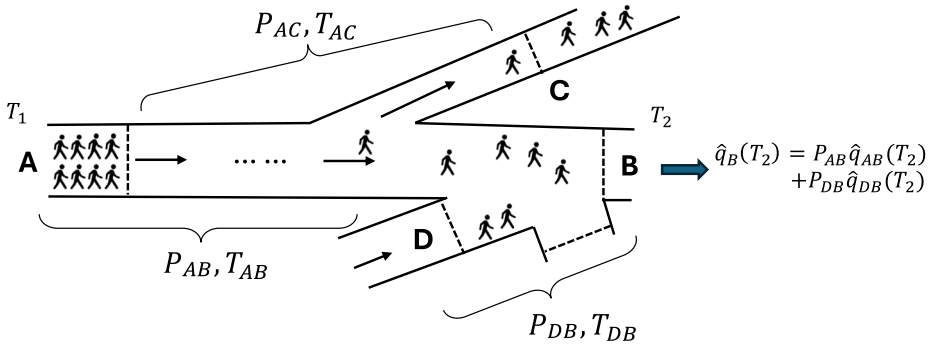


Fig. 2. Illustration of predicting the downstream flow with the upstream route flows.

3. Methodology

The gas/fluid dynamic models have long been used to describe the crowd flow by non-linear partial differential equations. These models (Henderson and Lyons, 1972; Henderson, 1974) rely on classical fluid dynamics theory, which assumes the conservation of momentum and energy. Another type of macroscopic model, continuum models describes the flow dynamics in more aggregate terms, using quantities such as flow, densities, and speeds. It assumes that the mass conservation equation must be satisfied at any point of the pedestrian flow (Yuan et al., 2020). However, these physics-based models typically require strong assumptions and depend on additional traffic state inputs (e.g., density and velocity), which are often difficult to obtain in practice.

In this section, we introduce the rudimentary crowd diffusion model that offers several advantages. First, it is data-efficient, requiring only observed flow and average walking speed. Second, the model is formulated in a discrete form, making it well-suited for time-series prediction tasks. Third, the simplicity of the model enables fast computation, a crucial property for real-time applications. In the following, we introduce the extension of this model to the network level by introducing a route velocity module and a route choice module to capture spatial–temporal dynamics across infrastructure locations.

3.1. Diffusion behavior with route choice probability

The crowd diffusion model is a macroscopic simulation model developed by Liu et al. (2015), which aims to study the uni-directional pedestrian movement. The relationship between the flow of a downstream and upstream sensor can be described with this model, since it discretizes the travel time and walking speed between two sensors in geometric distribution to mimic the diffusion behavior. The flow of the downstream sensor is thus can be estimated with the historical flow data from the upstream node using a diffusion process. This diffusion process is in essence a uni-directional relationship of the flow rate between one upstream sensor and one downstream sensor. To estimate the flow of the downstream sensor $\hat{q}_B(j)$ using the historical upstream sensor flow q_A, it is described by the following formula:

$$\begin{aligned}\hat{q}_B(j) = {} & F \cdot q_A(j - T_{AB}) + F(1-F) \cdot q_A(j - 1 - T_{AB}) + \cdots \\ & + F(1-F)^n q_A(j - n - T_{AB}) + \cdots \\ & + F(1-F)^{j - T_{AB} - 1} q_A(1).\end{aligned} \tag{1}$$

In Eq. (1), j is the current timestep, F is the diffusion coefficient, and T_{AB} is the minimum travel time between sensors A and B. To discretize the travel time, we divide it by the time unit which is 10 s in our cases.

We now consider modeling the route choice of the pedestrian. The transition probability from A to B is defined as $P_{AB} \triangleq P(D = A \mid U = B)$, where U denotes the upstream nodes and D the downstream nodes. This probability is then incorporated into the model to make it applicable in a network:

$$\begin{aligned}\hat{q}_{AB}(j) = {} & P_{AB}[F_{AB} \cdot q_A(j - T_{AB}) + F_{AB}(1 - F_{AB}) \cdot q_A(j - 1 - T_{AB}) + \cdots \\ & + F_{AB}(1 - F_{AB})^n \cdot q_A(j - n - T_{AB}) + \cdots \\ & + F_{AB}(1 - F_{AB})^{j - T_{AB} - 1} q_A(1)].\end{aligned} \tag{2}$$

Finally, the total flow $\hat{q}_B(j)$ of sensor B at timestep j is the weighted sum between the transition probability and the upstream flows: $\hat{q}_B(j) = \sum_{V \in \mathcal{N}_B} \hat{q}_{VB}(j)$, where $\mathcal{N}_B$ is the set of the upstream sensors of the predicted downstream sensor B. Fig. 2 demonstrates the process of predicting the downstream flow with the upstream route flows. In Section 3.3, we will introduce how the route choice probability and route travel time are estimated.

3.2. Multi-steps prediction

Eq. (1) describes the calculation for the single-step prediction. Theoretically, we can predict the future $j - T_{AB}$ steps pedestrian flow by recursively computing the flow in B ($\hat{q}_B$) by means of Eq. (3):

$$\hat{q}_B(j+n) = F \cdot q_A(j+n-T_{AB}) + (1-F) \cdot \hat{q}_B(j+n-1), \quad 0 \le n \le j - T_{AB}. \tag{3}$$

Accordingly, at the network level, we can compute the multi-step prediction on a link with the transition probability P_{AB}:

$$\hat{q}_{AB}(j+n) = P_{AB}[F_{AB} \cdot q_{AB}(j+n-T_{AB}) + (1-F_{AB}) \cdot \hat{q}_{AB}(j+n-1)], \quad 0 \le n \le j - T_{AB}. \tag{4}$$

Therefore the $n'+1$ steps ahead prediction for a downstream sensor is the aggregation of all the predicted flows from its upstream:

$$\hat{q}_B(j+n') = \sum_{v \in \mathcal{N}_B} \hat{q}_{vB}(j+n_v), \quad 0 \le n_v \le j - T_{vB}, 0 \le n' \le \min_v(j - T_{vB}). \tag{5}$$

In practice, the travel time T_{vB} between each of the upstream sensors and the downstream sensors varies, if we directly aggregate the upstream flow to predict the downstream flow, the desired downstream prediction horizon n' is limited to the minimum upper bound of the upstream prediction horizon (i.e. $\min_v(j - T_{vB})$). To address this issue, we impute the upstream sensor flow data with their averaged historical flow. For instance, if $n' = 6$ but the upstream sensor only provides $n_v = 4$ steps, we fill in the remaining steps $n_v + 1$ and $n_v + 2$ with the average historical flow from node v. This is a simple and computationally efficient method for aligning prediction horizons. Empirically, the required imputation length is typically short (1 to 3 steps) if the discrepancy of the travel times from the upstream to the downstream sensor is not significantly large, the historical averaging can still be accurate within a short time range.

3.3. Parameterizing velocity and route choice probability

The walking speed of the pedestrians and the route condition are changing dynamically, thus the diffusion coefficient F_{AB} and transition probability P_{AB} need to be parameterized as functions. In the uni-directional model, the diffusion coefficient is defined as a function of the minimum travel time, and it can be represented as:

$$F_{AB} = \frac{1}{1 + \gamma_1 T_{AB}}, \tag{6}$$

where $T_{AB} = \gamma_2 \frac{L_{AB}}{V_{AB}}$, V_{AB} is the maximum walking speed of the pedestrian, L_{AB} is the distance between the sensors, and γ_1 and γ_2 is two calibrated parameter that controls the diffusion coefficient and the average travel time, in our model they are combined and represented by a learnable parameter α. In the pedestrian network, links accommodate bidirectional flow, so a pedestrian's walking speed depends on both the same-direction and counter-direction flows. The distinct α on each network link is used to represent the unique route condition on the link. Finally, the calculation of the diffusion coefficient of route A to B is $F_{AB} = \frac{1}{1+\alpha_{AB}\frac{L_{AB}}{V_{AB}}}$. In practice, the average speed is not fixed and it changes as the pedestrian type or the overall scenario changes. Based on the fundamental diagram, we parameterize the route speed as a function of the upstream[1] flow (q_u) and downstream flow (q_d):

$$V_{ud} = V_{\boldsymbol{\Theta}}(q_u, q_d). \tag{7}$$

Here, $\boldsymbol{\Theta}$ represents the learnable parameters, and this function can be modeled using various data-driven approaches, such as neural networks or decision regression trees. In this study, we choose neural networks because they are more adaptable to different input data formats and can be effectively integrated into our online learning framework.

As for the transition probability, we know that it should be a function between the flow of upstream nodes and the flow of downstream nodes. The travel time between the sensors could also affect the choice of the pedestrian. Hence, the transition probability can be parameterized with a data-driven model and computed by the softmax function, with q_u, q_d, and T_{ud} as input :

$$P(D = d \mid U = u) = \frac{\exp(f_{\mathbf{w}}(q_u, q_d, T_{ud}))}{\sum_{k \in \mathcal{N}_u} \exp(f_{\mathbf{w}}(q_u, q_k, T_{uk}))}. \tag{8}$$

Here, $f_{\mathbf{w}}(,)$ is a score function that estimates the dependency between upstream and downstream. $\mathcal{N}_u$ is the set of downstream sensors of the sensor u. T_{ud} is the travel time between sensor u and d.

3.4. Route flow prediction paradiagm

Our framework employs two neural networks to parameterize Eqs. (7) and (8). This section provides an overview of the complete prediction framework. Fig. 3 illustrates the workflow of our data-driven diffusion behavior model, which consists of three separate parts: a route velocity prediction, a route choice probability estimation, and a route flow prediction layer. In the route velocity prediction module, the objective is to estimate the velocity on the edges (a route is represented by an edge). The flow data from the upstream and downstream sensors are encoded by two multi-layer perceptrons (MLP). The resulting embeddings (z_u and z_d)

[1] To simplify the notation, here q_u represents the historical flow $q_u^{t-M'+1:t}$ of node u.

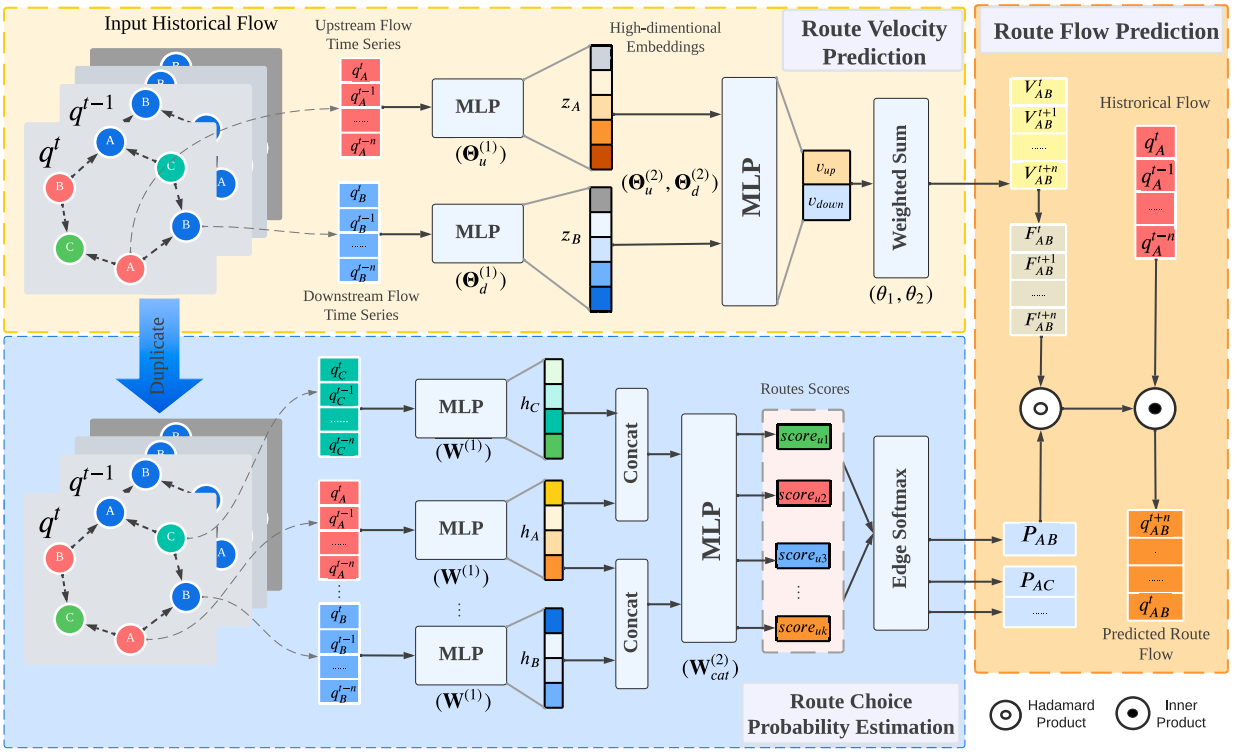


Fig. 3. Route flow prediction paradigm. q^t denotes the pedestrian flow at time step t. Route scores are computed using $score_{uk} = \text{LeakyReLU}(\mathbf{W}_{\text{cat}}[h_u \| h_k \| T_{uk}])$. These scores are then used to estimate the route choice probabilities as described in Eq. (14).

are then fed into a linear layer, reduced to a scalar (v_{up} and v_{down}) using the sigmoid function σ. Although we experimented with tuning the output dimensionality of the second last layer, we found that a two-dimensional embedding performs well empirically. Furthermore, each dimension can be interpreted as the estimated upstream (v_{up}) and downstream velocities (v_{down}), enhancing the interpretability of the final output. The corresponding computations are shown below:

$$z_u = \text{ReLU}(\Theta_u^{(1)} q_u + b_u^{(1)}), \tag{9}$$

$$z_d = \text{ReLU}(\Theta_d^{(1)} q_d + b_d^{(1)}), \tag{10}$$

$$v_{up} = \sigma(\Theta_u^{(2)} z_u + b_u^{(2)}), \tag{11}$$

$$v_{down} = \sigma(\Theta_d^{(2)} z_d + b_d^{(2)}). \tag{12}$$

In the above equations, b refers to the learnable bias in the linear layers. The final predicted route velocity is the weighted sum of v_{down} and v_{up} using two learnable parameters θ_1 and θ_2. To ensure the output is positive, we use the softplus function to transform the prediction:

$$V_{ud} = \log(1 + \exp(\theta_1 v_{down} + \theta_2 v_{up})). \tag{13}$$

Inspired by the work in Velickovic et al. (2017), the route choice module calculates the route choice probability of a node by normalizing the importance scores which is the output of $f_{\mathbf{w}}$ for each of its downstream nodes. First, the flow data from all the sensors are encoded by a MLP with trainable parameters $\mathbf{W}$. Then, the embeddings of the downstream and upstream sensors are concatenated and fed into a fully connected layer to calculate the scores. The LeakyReLU function is applied to introduce nonlinearity. It is an extended version of the ReLU function that allows a small, non-zero gradient when the input is negative, preventing the vanishing gradient issue. As such, the transition probability in Eq. (8) is estimated by the following formula:

$$P(D = d \mid U = u) = \frac{\exp(\text{LeakyReLU}(\mathbf{W}_{\text{cat}} \left[h_u \| h_d \| T_{ud}\right]))}{\sum_{k \in \mathcal{N}_u} \exp(\text{LeakyReLU}(\mathbf{W}_{\text{cat}} \left[h_u \| h_k \| T_{uk}\right]))}. \tag{14}$$

Here, h_u and h_d denote the embeddings of the upstream sensor and the downstream sensor. $\mathbf{W}_{\text{cat}}$ is the learnable parameter in the final linear layer to learn the correlation of upstream, downstream embedding and travel time. $(\cdot \| \cdot)$ is the concatenation operator, we use the concatenation of the embeddings and travel time between upstream and downstream sensors as the input for the softmax function.

Algorithm 1 Flow Prediction with Test-then-Train

Require: Sensor network $\mathcal{G}\{\mathcal{V},\mathcal{E}\}$, offline sensor flow data $D=\left[q^1,\cdots,q^N\right]$, retraining iterations T.
Ensure: Velocity model $V_{\mathbf{\Theta}}$, route choice model $f_{\mathbf{w}}$, α for diffusion coefficient, prediction error e.

Initialization: Initialize $\mathbf{\Theta}$, $\mathbf{W}$ and α. Initialize replay buffer $\mathcal{R}$.
$\theta,\mathbf{w},\alpha \leftarrow$ Offline Batch Training(D),
while previous n time steps observation $x=\left[q^{t-n},\cdots q^{t-1},q^t\right]_v,\forall v\in\mathcal{V}$ is available **do**
  Store observation into buffered dataset $\mathcal{R}$.
  Estimate route velocity of the network with Eq. (7).
  Compute route choice probability with Eq. (8).
  Predict flow for all the downstream sensors with Eq. (5).
  Compute prediction error e_t += e_{t-1} when $t+n$ steps ground truth flow is observed.
  Compute sampling probability $\mathbf{p}$ for the data in $\mathcal{R}$ based on the arriving order $r(x_i)$: $p_i=\frac{r(x_i)}{\sum_j r(x_j)}$.
  for $i\in\{1,\cdots,T\}$ **do**
    Sample mini-batch data $\mathcal{X}_{batch}$ from $\mathcal{R}$ with probability $\mathbf{p}$.
    Update route choice model $f_{\mathbf{w}}$ based on $\mathcal{X}_{batch}$.
    if i *mod* 2 **then**
      Update velocity model $V_{\mathbf{\Theta}}$ and α based on $\mathcal{X}_{batch}$.
    end if
    Terminate retraining when validation error meets the early-stop requirement.
  end for
end while
return $V_{\mathbf{\Theta}}$, $f_{\mathbf{w}}$, α and e

After obtaining the route choice probability and route velocity, the diffusion coefficient can be calculated with Eq. (6). In the final layer of the paradiagm, Eq. (4) is applied to perform the multi-step flow prediction.

3.5. Online learning crowd flow prediction framework

In traditional machine learning, models are trained offline using a fixed historical dataset under the assumption that future data follow the same distribution. However, in real-world crowd dynamics, the OD demand and walking behaviors vary over time due to factors like passenger composition, time of day, and unexpected events. These dynamic and often non-repeating patterns make it difficult for static models to generalize, particularly during rare or extreme scenarios.

To address this, we adopt an online learning framework that continuously adapts the model to streaming input. Online learning is a typical machine learning paradigm that aims to make streaming decisions that evolve continuously over time (Wares et al., 2019). Research has focused on developing fast-adapting algorithms to handle the data distribution drift problem. The online gradient descent (OGD) algorithm (Hoi et al., 2021) is a simple yet effective approach to solve the online convex optimization problem. More advanced approaches include tree-based models with adaptive sliding windows (Domingos and Hulten, 2000) and the online sequential extreme learning machine (OS-ELM) (Liang et al., 2006), a neural network-based method for time-series prediction. However, these methods are tied to specific model architectures, such as Hoeffding trees or single-layer feedforward neural networks, and are therefore not compatible with our traffic knowledge-oriented pedestrian flow prediction model.

We choose OGD to train our model. First, it is model-agnostic, allowing integration with our custom-designed, traffic knowledge-infused prediction model without requiring structural changes. Second, OGD is well-suited for non-stationary environments: using a buffered dataset (Hoi et al., 2021) that retains both recent and newly arrived data, the model can incrementally update its parameters via stochastic gradient descent on the buffer without re-training from scratch on the full offline dataset. This enables rapid adaptation to evolving pedestrian flow patterns in real-time.

To evaluate the performance of the model, we use the test-then-train Wares et al. (2019) evaluation framework. In this approach, the model is evaluated on the current chunk of data before being trained on the chunk. After the testing, the chunk of data becomes available for training. The evaluation process is described in Algorithm 1. Before starting the test-then-train procedure, the model needs to be trained on offline data collected from a typical scenario under normal conditions to establish a baseline for online learning. The model can then be retrained continuously with this framework to adapt to changing conditions.

4. Experimental design

Through a set of simulation experiments, we aim to test our new prediction model from three perspectives: its ability to predict accurately under normal conditions, its effectiveness for downstream operations in adverse conditions such as emergency evacuations, and its reliability in terms of the route velocity model and route choice model. Specifically, we evaluate the prediction

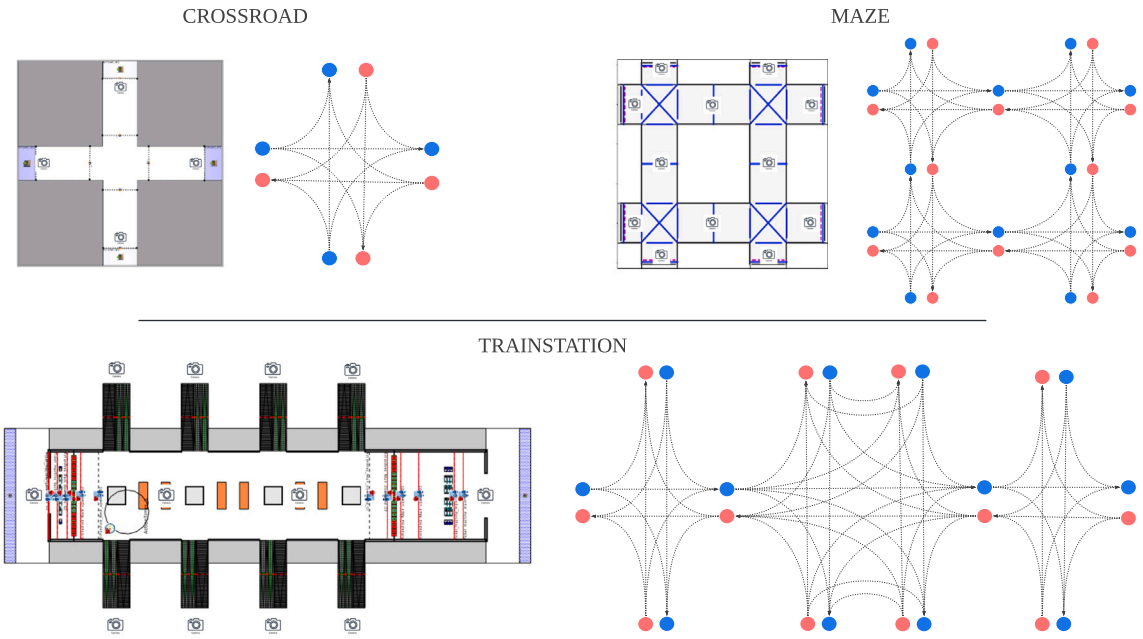


Fig. 4. The architectural plans of the three infrastructures, with sensor locations marked by camera icons and the sensor network topological structures of three infrastructures.

Table 1
The parameters of the three simulation environments. The average flow rate is calculated based on the mean flow rate at each sensor. The number of origins represents the number of locations from which people can arrive.

Parameters	Crossroad	Maze	Trainstation	Eifpd
Num. of Nodes	8	24	24	12
Num. of Edges	12	48	54	30
Num. of Origins	4	8	10	6
Num. of Scenarios	15	15	14	11
Total Recording Time (mins)	300	765	770	2,075
Average Flow Rate (peds/10s)	14	3	4	0~1
Min/Max Walking Speed (m/s)	0.06~1.75	0.2~1.6	0.06~1.75	–
Scale of the Infrastructure (m^2)	60 × 60	100 × 110	50 × 117	12 × 16

accuracy by comparing the model with baseline methods and validate whether the velocity model can learn the fundamental relationship between velocity and flow, and whether the probability model can accurately estimate the OD matrix. Underneath we describe the experimental design of the simulation experiments in more detail. First, the scenario development is briefly touched upon. Accordingly, the characteristics of the synthetic datasets and a real-world dataset are introduced.

4.1. Dataset development and scenarios

Currently, there is no publicly available crowd-flow dataset that combines both high demand and a large sensor network (10 or more sensors). To thoroughly evaluate our model, we develop a comprehensive set of test cases using simulated and real-world data. We construct three infrastructures using the simulation tools *Pedestrian Dynamics* (Dynamics, 2024) and *Nomad* (Sparnaaij et al., 2023), both of which are microscopic agent-based simulation platforms. Fig. 4 shows the architectural plans of the three designed infrastructures, the following subsections describe how we set up the simulation to develop different scenarios for each infrastructure. These three datasets are designed for different purposes and feature different spatial scales. The *Maze* and *CrossRoad* datasets primarily change the OD demand and activity patterns, while the *TrainStation* dataset aims to simulate the more realistic and very complex changes in demand at a train station during peak demand when trains arrive and emergency evacuations. Additionally, we include a real-world dataset collected in the main building of the School of Informatics at the University of Edinburgh. This dataset contains pedestrian trajectory data and is used to assess the model's predictive performance in actual environments. The parameters of the four datasets are summarized in Table 1.

4.1.1. Cross road

The $CossRoad$ is the most basic infrastructure. We develop this simulation using the simulator $PedestrianDynamic$. There are four entrances and exits in the $CrossRoad$ infrastructure, forming a simple intersection where pedestrian flow can be observed from

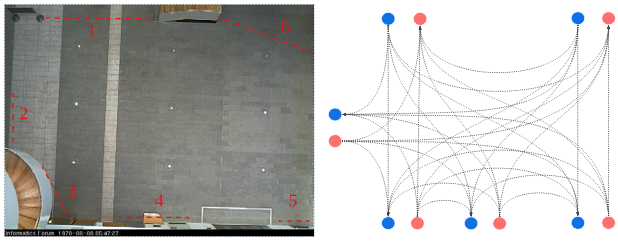


Fig. 5. The top view of Edinburgh Informatics Forum (l.h.s.) and its sensor network (r.h.s.).

multiple directions. Pedestrians enter the building from one of the four entrances and choose an exit with varied probabilities. The walking speed of the pedestrians follows a triangular distribution $Triangular(1.35, 0.8, 1.75)$, indicating that the lowest walking speed is 0.8 m/s, the highest is 1.75 m/s, and the mode is 1.35 m/s. The flow rate of each entrance ranges from 1 to 8 (peds/10s).

4.1.2. Maze

The $Maze$ infrastructure is a more complex version of $CrossRoad$ infrastructure, consisting of multiple interconnected hallways and intersections. This design allows for more intricate routes and interactions. The simulation is built using the simulator $Nomad$.

The flow rate at each entrance ranges from 0 to 3 (peds/10s). To simulate changes in demand, this flow rate varies over time. Pedestrian walking speed follows a normal distribution with a mean of 1.5 m/s and a standard deviation of 0.6 m/s. To prevent extreme velocities, the speed is bounded with a minimum of 0.2 m/s and a maximum of 1.6 m/s.

4.1.3. Train station

The *TrainStation* infrastructure simulates a real-world transit hub, featuring multiple platforms, ticketing areas, and entrances/exits. The simulation is developed based on *Pedestrian Dynamic*.

In each scenario, the agent generator generates a group of people at each time interval. The number of people in this group follows a uniform distribution $U(2, 5)$, meaning that 2 to 5 people will be in this group. The time interval is a random variable following an exponential distribution, such that the arrival of the passenger groups follows a Poisson distribution. We set the mean interval $s = 2$ for the rush hour condition, which means that on average every 2 s a group of passengers would arrive at the station. For off-peak hours, we set $s = 4$. The demand for each train line is described by the probability that passengers will take that line. We categorize demand levels as high, normal, and low for a specific train line. For example, the demand distributions for line 1 are: high: $[52\%, 16\%, 16\%, 16\%]$, normal: $[25\%, 25\%, 25\%, 25\%]$, and low: $[10\%, 30\%, 30\%, 30\%]$.

The maximum walking speed follows the triangular distribution $Triangular(2, 1.5, 2.5)$ under high-throughput conditions, indicating that the lowest maximum walking speed is 1.5 m/s, the highest is 2.5 m/s, and the mean is 2 m/s. Under normal conditions, the maximum walking speed is $Triangular(1.35, 0.8, 1.75)$. Further details about the simulation setup can be found in Mai et al. (2025).

4.1.4. Edinburgh informatics forum

We adopted one real-world dataset for our evaluation. The dataset (Majecka, 2009) comprises detected targets of pedestrians walking through the Informatics Forum, the main building of the School of Informatics at the University of Edinburgh. Collected over several months, the data contain approximately 1,000 observed trajectories per working day. We extracted and processed 12 days of trajectory data into flow data. The sensor locations are visualized on the left side of Fig. 5. Multiple entrance/exit points surround the forum, each monitored by a sensor.

The average walking speed is approximately 1.2 m/s with a standard deviation of 1.72; however, precise maximum speeds cannot be determined due to noisy trajectories with very short distances. Additionally, this real-world dataset is relatively sparse compared to synthetic datasets, with an average flow rate of 0.13 pedestrians per 10 s.

4.2. Setting of the experiment

This section introduces the format of the spatial–temporal flow data, the benchmark models, and the evaluation metrics we used for the experiments. The datasets we created and the code for our and the benchmark models can be found in this github repository.[2]

[2] https://github.com/WaimenMak/Crowd-Flow-Inference

Table 2
Performance of online crowd flow prediction on synthetic datasets, measured in pedestrians per 10 s (*peds/10 s*). The value in red indicates the lowest prediction error, while the value in blue marks the second-lowest.

Dataset	Metric	MA	GCN	GAT	LSTM	GCN-LSTM	Diffusion
Crossroad	RMSE	4.50	5.44	8.31	8.63	10.90	5.86
	MAE	2.72	3.11	4.07	4.67	6.00	3.45
Maze	RMSE	1.98	2.25	2.06	2.14	2.26	2.03
	MAE	1.39	1.64	1.50	1.51	1.57	1.41
Trainstation	RMSE	4.80	4.23	4.21	3.54	3.72	3.68
	MAE	2.96	2.84	2.81	2.35	2.46	2.31
Eifpd	RMSE	0.46	0.46	0.47	0.48	0.49	0.46
	MAE	0.23	0.24	0.25	0.22	0.18	0.19

4.2.1. Time-series flow data

The input data is a three-dimensional matrix $X \in \mathbb{R}^{N \times |\mathcal{V}| \times M'}$, which represents a sequence of pedestrian flow with N samples, $|\mathcal{V}|$ sensors and M' time lags. The length of the time window M' should be carefully chosen: a window that is too long may include uncorrelated information, while a window shorter than the minimum travel time $T_{\min}$ between sensors may fail to capture the relevant upstream influence because the downstream inflow at time t corresponds to the upstream outflow at time $t - T_{\min}$ under theoretical conditions (free flow and identical walking speed). Therefore, using a time window shorter than $T_{\min}$ may result in missing critical causal relationships in the data.

With above, we set the window size to 6, meaning that we use the previous 1 min of flow data as input to predict the future 1 min flow rate of each downstream sensor. The distance between two line sensors is estimated by dividing the average travel time by the average walking speed of the pedestrians.

We generate different scenarios by varying parameters such as walking speed, OD demand, and activity patterns. These scenarios are then concatenated to create four comprehensive datasets for online testing. This setup also enables us to simulate scenario drifts within a continuous data stream.

4.2.2. Benchmark models

As there are no models specifically designed for crowd flow prediction in public spaces, we selected several off-the-shelf data-driven models for comparison. Since the models will be trained and tested online, a relatively small model with fewer trainable parameters would be preferred. We choose the conventional time-series prediction models such as LSTM (Long-Short Term Memory), non-parametric method, and moving average (MA). It is a simple yet effective method for estimating crowd flow and is still commonly used in some train station crowd flow prediction systems. We also included graph models like GAT (Graph Attention Network) (Velickovic et al., 2017), GCN (Graph Convolutional Network) (Kipf and Welling, 2017), and the spatial–temporal model GCN-LSTM. Note that all baseline models are trained online as described in Algorithm 1.

4.2.3. Model evaluation method

We calculate the prediction error using the mean absolute error (MAE) and the root mean square error (RMSE), since the model output prediction in an online manner, we compute the MAE and RMSE over all chunks of data stream for comparison. The following equations compute the MAE and RMSE between the prediction $\hat{y}_i$ and ground truth y_i.

$$MAE = \frac{\sum_{i=1}^{i=N} |\hat{y}_i - y_i|}{N}. \tag{15}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{i=N} (\hat{y}_{it} - y_{it})^2}{N}}. \tag{16}$$

We evaluate the prediction accuracy of the models by comparing the RMSE and MAE. The goal is not to achieve state-of-the-art prediction accuracy but rather to attain performance comparable to other well-known data-driven models. The prediction accuracy of existing methods is already sufficient for most scenarios.

The data stream is collected periodically, with flow data recorded every 10 s. We set the model update frequency to 5 min, meaning the model is re-trained after every 30 data points. The replay buffer size should not be too large, as it occupies significant storage in practice. However, it should not be too small either, as adequate space is needed for the model to be re-trained on previous scenarios. We set it to 1000 empirically, allowing it to store data from the three most recent scenarios, each containing around 360 data points.

5. Online prediction performance

Among these four datasets, the *CrossRoad* and *Maze* datasets primarily assess the models' robustness to changes in pedestrian activity patterns, while the *TrainStation* dataset evaluates their robustness to sudden changes in flow rate. The dataset *EIFPD* tests the model's performance in real-world scenarios. This section first compares the online prediction error of each model and then analyses the result by visualizing the prediction and the ground truth.

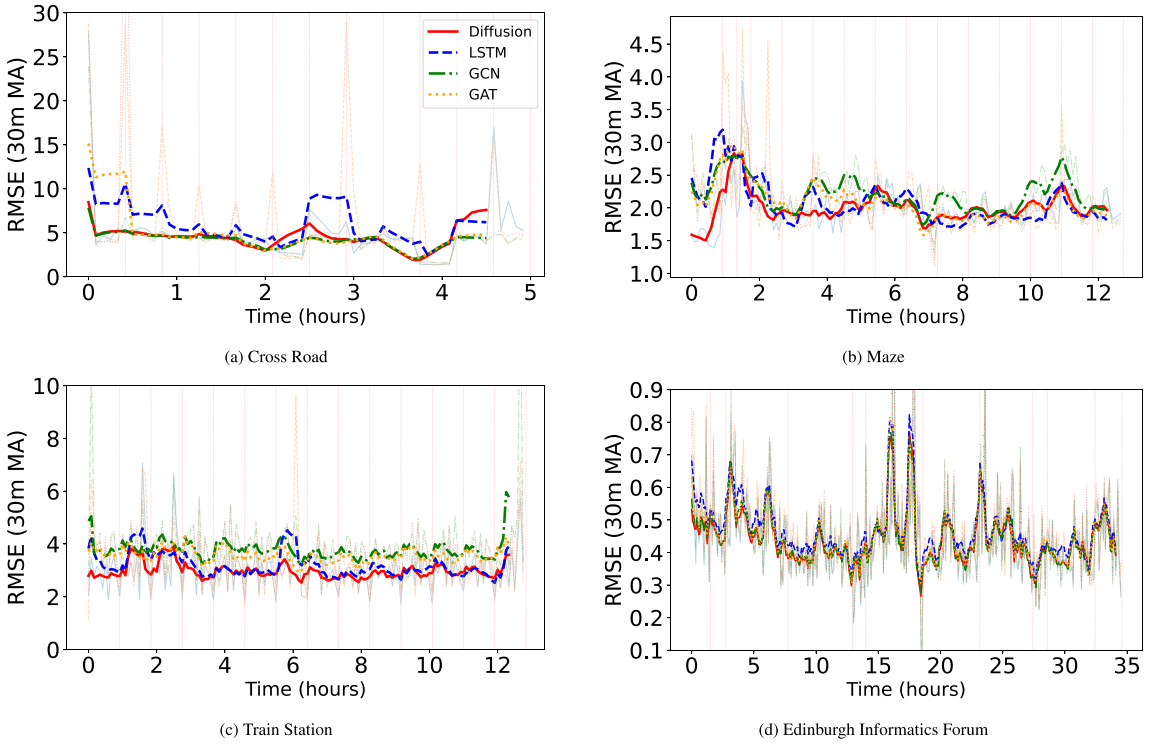


Fig. 6. Prediction error (RMSE) curves on four datasets. The vertical red dotted line indicates the shift in scenarios. Baseline model errors often increase after scenario changes, while the diffusion model demonstrates more stability.

5.1. Prediction error analysis

We compare the qualities of the 1 min ahead multi-step predictions in the first three datasets and the predictions of 10 s ahead in the real-world dataset. The results are shown in Table 2. MA performs the best in both *CrossRoad* and *Maze*. The following reasons may explain this result: (1) The demands in both datasets change gradually, thus the flow at each sensor remains smooth without sudden drops or increases. In such cases, a simple MA is sufficient to capture the trend. (2) MA does not have any trainable parameters and does not need to adapt to different scenarios. (3) During network disruptions, MA can still provide stable predictions by averaging historical flow, whereas data-driven models may struggle due to reliance on broader, potentially misleading inputs.

In the *TrainStation* dataset, the performance of MA model deteriorates due to the highly fluctuating flow caused by train arrivals. In contrast, other data-driven models can react to peaks more quickly because they have encountered similar situations in the previous training data. Another interesting finding is that the prediction accuracies of the graph-based models are comparable to that of the LSTM. We believe this is because the graph sizes of the datasets are not large enough for the graph models to have a significant advantage.

The diffusion behavior model performs the second best in *Maze* and *TrainStation* regarding RMSE. Because it tends to overestimate flow during congestion or network disruptions, resulting in a higher RMSE than the other models in these scenarios. However, the main goal is not to develop a model with the best prediction accuracy among all scenarios but to design a model that can obtain acceptable accuracy and quickly adapt to new scenarios. We recorded the online prediction RMSE for each chunk of data and plotted it in Fig. 6. The prediction error curves are smoothed using a 30 min sliding window for clarity. As shown in the figures, the diffusion behavior model maintains consistent prediction errors across different scenarios, demonstrating its robustness to scenario drifts. In Fig. 6(d), all models perform comparably, with the prediction error remaining within a small range due to the sparsity of pedestrian flow.

In summary, our model leverages a diffusion process for pedestrian flow prediction. The results show that its predictive accuracy is comparable to widely used data-driven models. Although these approaches achieve similar levels of accuracy across different datasets, the diffusion-based model may be preferred for its transparent and interpretable prediction process.

5.2. Prediction visualization

We visualize the prediction of the model and the ground truth for 5000 s of a sensor in Fig. 7. This figure shows the 40-s prediction of a sensor in the hallway in *TrainStation*. In Fig. 7(a), the diffusion behavior model responds more effectively than MA,

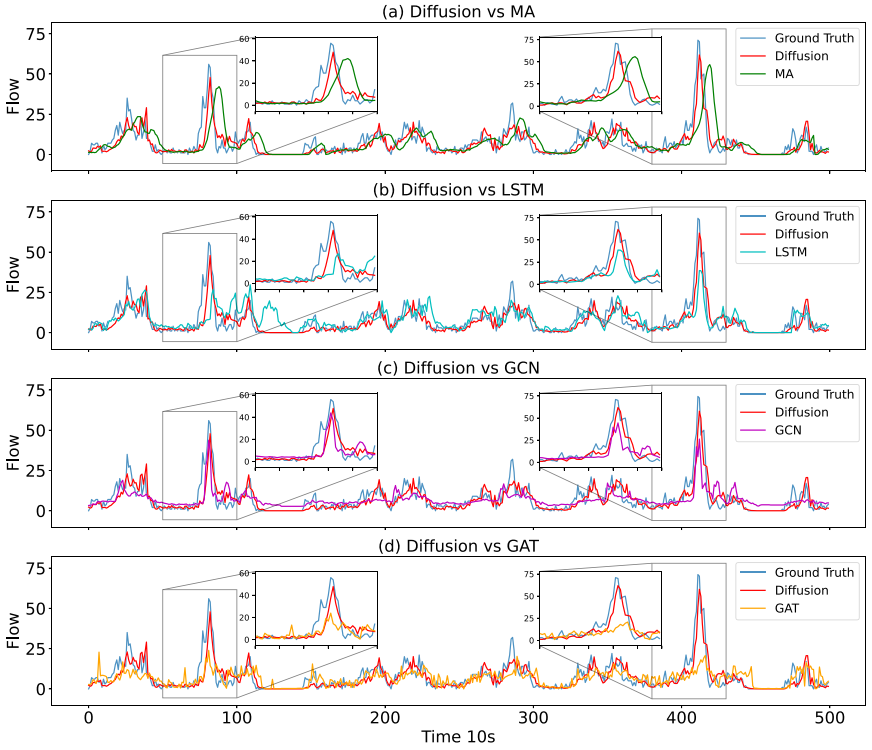


Fig. 7. Models' 40-s ahead predictions on one sensor in *TrainStation*.

which exhibits noticeable delays. Fig. 7(b) compares our model with LSTM, demonstrating that our model captures peak flow more accurately, while LSTM sometimes underestimates flow and completely misses demand peaks. In Fig. 7(c) and (d), we compare our model with graph-based models. GCN effectively captures peak flows, but tends to underestimate them. On the other hand, GAT accurately captures the periodic rise in flow patterns but shows limited sensitivity to sudden changes. These visualizations show that the diffusion model can capture peak demands more effectively than the other models. More visualizations of the prediction in other locations are presented in Appendix A.

5.3. Prediction model as a congestion detector

Under normal conditions (free flow), a good prediction model should aim for high accuracy. However, during congestion or disruptions, it is beneficial if the model can also act as an early warning system. Our proposed crowd flow prediction model exhibits this potential. To illustrate this, we calculate the total difference between inbound and outbound flows at the train station over time (inbound–outbound), as shown in Fig. 8(a). The inbound flow includes both passengers alighting from trains and those entering to board. The two peaks in this difference indicate moments when outbound flow significantly lags behind inbound flow, pointing to severe congestion between the gate and the exits.

The model's 40-s predictions at the main and back exit are shown in Fig. 8(b) and Fig. 8(c), our model predicts extremely high flow at these two moments, followed by a rapid decline. However, the actual downstream flow increases more gradually. This discrepancy between the model's prediction and the actual flow indicates that while many people are approaching from upstream, congestion slows their movement, delaying their arrival at the downstream sensor. The overestimation of the model becomes a useful signal for detecting congestion and could enhance real-time alert systems and emergency response strategies.

5.4. Online uncertainty quantification with negative binomial likelihood

Incorporating uncertainty in pedestrian flow prediction can significantly improve real-time decision making. To evaluate the capability of our model in uncertainty quantification, we adopt a parametric probabilistic approach by modeling pedestrian counts using the negative binomial distribution - a well-established choice for positive count data (Chapados, 2014). This enables us to formulate the training process as a likelihood maximization problem. The corresponding negative binomial likelihood is defined as follows:

$$l(y|\mu,\beta) = \frac{\Gamma(y+\frac{1}{\beta})}{\Gamma(y+1)\Gamma(\frac{1}{\beta})}\left(\frac{1}{1+\beta\mu}\right)^{\frac{1}{\beta}}\left(\frac{\beta\mu}{1+\beta\mu}\right)^{y}. \tag{17}$$

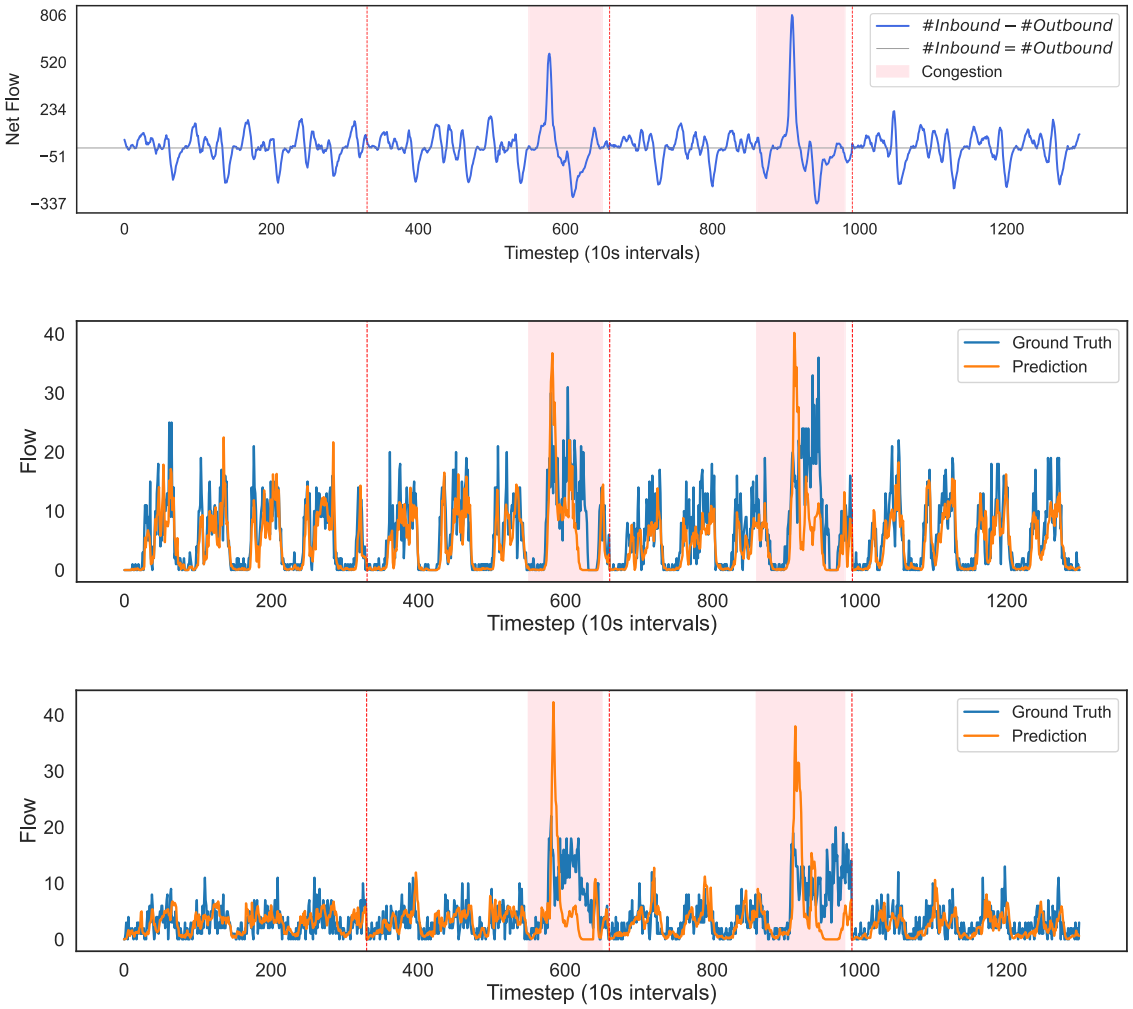


Fig. 8. Visualization of the prediction of the diffusion model when there is high congestion. The model overshoots prediction when there is high congestion. The vertical red dotted line indicates the shift in scenarios.

In this formulation, y denotes the one-step prediction of pedestrian flow, while μ and β are the mean and shape parameters of the negative binomial distribution. In the setup in Salinas et al. (2020), parameters μ and β are estimated using two separate neural networks. Specifically in our approach, the process of predicting the mean ($\hat{\mu}$) of the distribution is the same as the non-probabilistic prediction, however, we use an extra MLP to predict the shape parameter $\hat{\beta}$ of a sensor using its upstream flows as inputs. Furthermore, solely optimizing the negative log-likelihood (NLL) loss function could lead to a poor accuracy for the single-value prediction. Therefore, to train the model, we combine the NLL loss for the first step with the mean squared error (MSE) loss for subsequent steps, forming the following joint objective:

$$\mathcal{L} = \sum_{i=1}^{i=N} \left[\sum_{t=2}^{T} (\hat{y}_i^t - y_i^t)^2 - \log l(y_i^1 | \hat{\mu}, \hat{\beta}) \right]. \tag{18}$$

We assess the quality of the distribution prediction using the coverage probability calibration curve, comparing our model with benchmark models (e.g., LSTM and GCN-LSTM) trained under the same loss function (Eq. (18)) with the dataset *TrainStation*. In addition, we included DeepAR (Salinas et al., 2020), a sophisticated probabilistic forecasting model, for further comparison. Coverage probability measures the proportion of observed data points that fall within the predicted intervals. For a well-calibrated model, the calibration curve should align with the nominal prediction interval (the reference line). If the curve falls below the reference line, the model underestimates the uncertainty, producing overly narrow prediction intervals; if it lies above, the model overestimates the uncertainty, resulting in excessively wide intervals.

As can be observed in Fig. 9, when the crowding level is below 40 (peds/10 s), the diffusion model aligns closely with the reference line, indicating a better calibration compared to the other methods. As crowding level increases, alternative models begin to underestimate the prediction intervals, reflecting overconfidence in their forecasts. To illustrate this behavior, we present

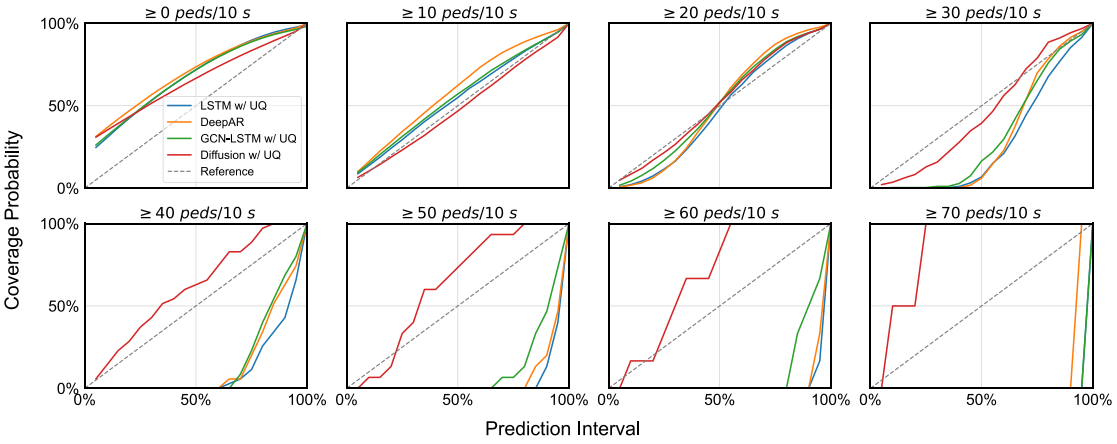


Fig. 9. Coverage probability calibration curve in different crowding levels.

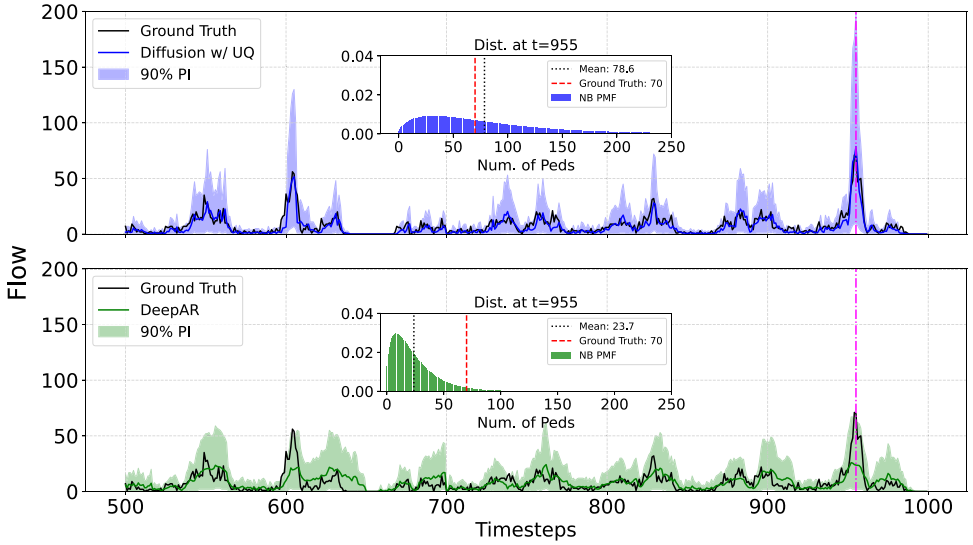


Fig. 10. The estimated 90% prediction interval and predicted probability distribution during evacuation. The pink dashed line indicates the moment when there is high crowding.

a snapshot of the probabilistic predictions in Fig. 10. In this example, an evacuation occurs at timestep 955. The corresponding predicted probability mass function (PMF) is shown in the center of the figure. The diffusion model captures the spike and predicts high uncertainty at this moment, whereas DeepAR produces a much narrower prediction interval, which almost fails to encompass the true pedestrian flow.

To conclude this section, we demonstrate that our model can effectively capture predictive uncertainty by incorporating a negative binomial likelihood into training loss. With this simple modification, the model is able to achieve better-calibrated prediction intervals compared to baseline models, particularly under extreme or rapidly changing crowd conditions. This capability enhances the model's applicability for real-time crowd monitoring and management, where reliable uncertainty estimation is crucial.

6. Analysis of online learning

In this section, we first validate the efficacy of online learning in the prediction framework, then the OGD's sensitivity to the update frequency and its real-time computational cost will be analyzed.

6.1. Efficacy of retraining

Fig. 11 presents the model's average prediction RMSE on each data chunk, comparing their performance with and without online retraining. The results clearly indicate a large increase in prediction error when online learning is not employed (online retraining

Table 3
Percentage of error decrease of each model with online learning.

Model	Crossroad	Maze	Trainstation
LSTM	48.9% (↓)	36.2% (↓)	18.1% (↓)
GCN	66.1% (↓)	33.6% (↓)	2.8% (↓)
GAT	49.9% (↓)	22.8% (↓)	7.5% (↓)
Diffusion	60.8% (↓)	4.3% (↓)	1.9% (↓)

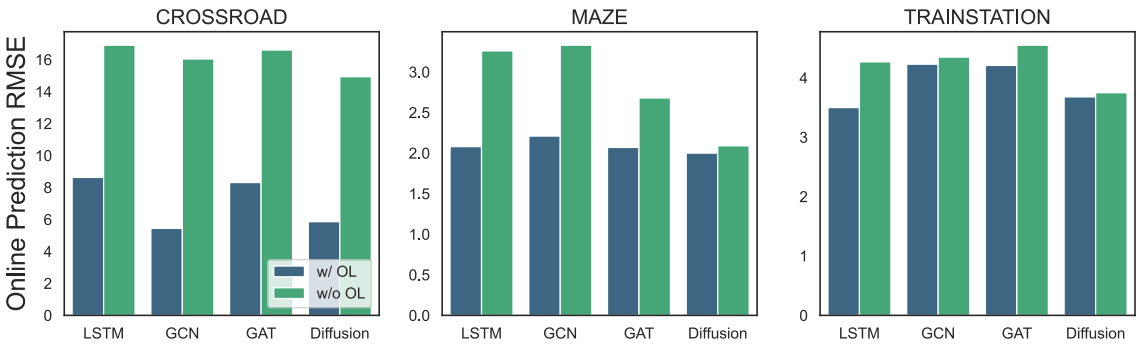


Fig. 11. Online prediction performance with and without online learning.

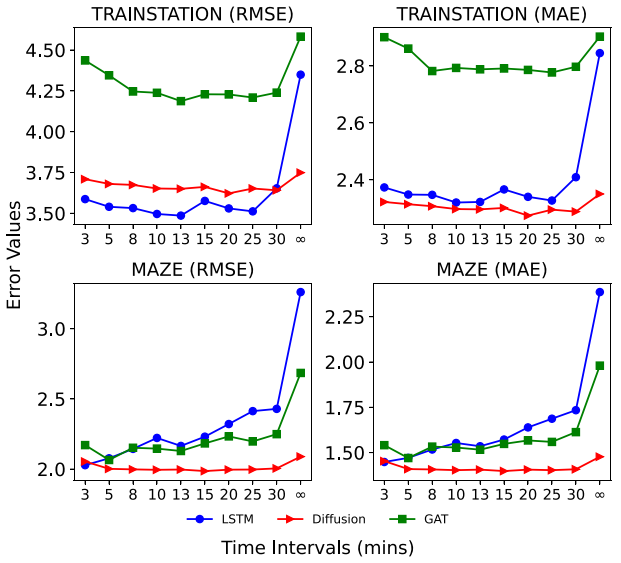


Fig. 12. Variation in prediction error with respect to different update frequencies. (∞ means no update at all).

step set to 0). Specifically, the non-graph-based model LSTM benefits the most from online learning in total, as shown in Table 3 for the percentage decrease in prediction error. Since LSTM predicts without using the topological information of the graph, changes in the OD demand can significantly affect its prediction accuracy. In contrast, graph-based models exhibit more robust behavior.

6.2. Sensitivity to update frequency

We further examine the effect of the update frequency on the accuracy of the prediction. The chunk size of the data stream determines how frequently the model is re-trained. As described in Algorithm 1, a portion of the chunk data is sampled as an evaluation set, and retraining will be stopped early if the prediction error on the evaluation set exceeds a specified threshold to prevent overfitting. Therefore, choosing an appropriate chunk size is critical: a small chunk size may lead to an unreliable evaluation set, while a large chunk size reduces the update frequency, potentially impacting the model's ability to adapt.

We tested three models with different update frequencies in Fig. 12. In general, we can observe that the models do not necessarily achieve optimal performance when the update frequency is set to the highest every 3 min (with a chunk size of 15), as the evaluation set is too small to provide reliable feedback. As the update frequency decreases, the prediction accuracy gradually

Table 4
Number of trainable parameters of different models across three datasets.

Model	# of trainable params			
	Crossroad	Maze	Trainstation	Eifpd
GCN-LSTM	48,272	116,976	116,976	63,468
LSTM	55,344	65,680	65,680	54,028
GAT	7602	7602	7602	7523
GCN	1670	1670	1670	1153
Diffusion	1428	1464	1470	1574

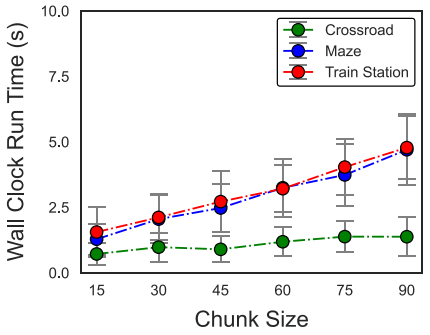


Fig. 13. The wall clock run time with different chunk sizes of each retraining of our model.

improves. However, when the frequency becomes too low, the performance begins to deteriorate. In the dataset $Maze$, the route choice behavior of the pedestrians is more variant than in the dataset $TrainStation$, we can see that the performance of LSTM drops as the update frequency decreases and is more sensitive compared to those graph-based models. The diffusion model shows the least sensitivity to the update frequency, suggesting that the structure of the model can be generalized well in different scenarios.

6.3. Computational cost

Online learning can address the drift in data distribution through frequent retraining. We investigate the computational cost of each round of retraining during online learning. The time complexity of the forward pass of our model is twofold. For the route velocity module, the complexity is $O(|\mathcal{E}|Td_v)$, where T represents the number of input time steps, and d_v is the dimension of the hidden layer. Here, Td_v accounts for the complexity of generating the embeddings for upstream and downstream flow. The time complexity for the route choice estimation module is $O(|\mathcal{V}|Td_c + |\mathcal{E}|d_c)$, where d_c is the embedding size. This complexity accounts for the generation of embeddings for each node, as well as the calculation of importance scores in Eq. (14) for each edge.

The variable chunk size is correlated to the retraining computational time since the model first trains on the chunk of the data and then trains on randomly sampled recently stored data from the replay buffer. Other variables, such as an increase in the size of the sensor graph and the number of trainable parameters in the model, can also increase the spatial and temporal complexity of the algorithm. Table 4 summarizes the number of trainable parameters in different datasets. As we can see, the number of sensors does not affect the parameters of the graph-based model, whereas in the LSTM model, the parameters largely increase with the number of sensors. For the diffusion model, the number of parameters increases as the number of edges in the sensor network grows, as there is one trainable parameter α for each edge. However, this increment is slight and can almost be overlooked. In general, the diffusion model has the fewest trainable parameters.

In practice, the model makes predictions every 10 s and completes the retraining within that time frame once the update is required. Fig. 13 shows how the chunk size of the data stream affects the retraining computational time. The experiments were conducted on a MacBook Pro with an Apple M1 Pro CPU, 16 GB RAM, and a 500 GB SSD. In the dataset $CrossRoad$, the sensor graph is relatively small; therefore, the computational time is much lower than $Maze$ and $TrainStation$. For the larger networks, even with a chunk size of 90 (equivalent to an update frequency of 15 min), the average retraining time remains around 5 s, well below the 10-s threshold. Therefore, the computational time for each retraining round is manageable. Although the largest network, consisting of 24 nodes and 54 edges, is not as extensive as an urban-scale network, it is sufficient for large infrastructure.

7. Qualitative evaluation of route velocity and route choice models

In this section, we evaluate the prediction quality of the velocity model and route choice model. To evaluate the route choice model, we assess its prediction of the OD matrix in different scenarios and compare it to the ground-truth ODs, since the OD can reflect the route choice probability of the pedestrians. For the velocity model, we investigate its understanding of the relationship between speed and flow derived from the data.

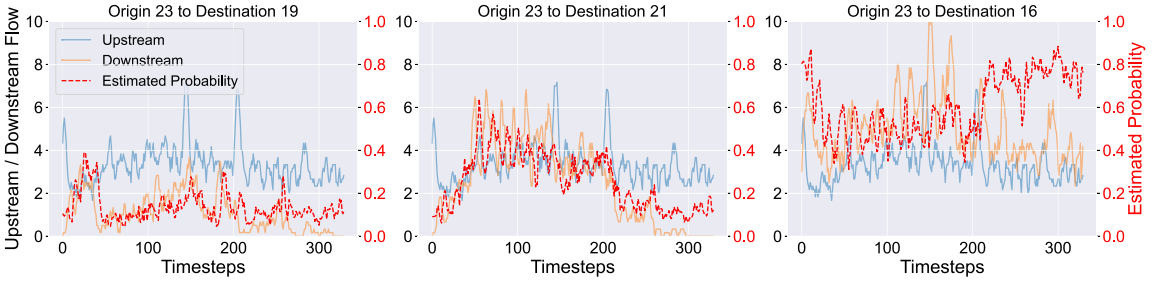
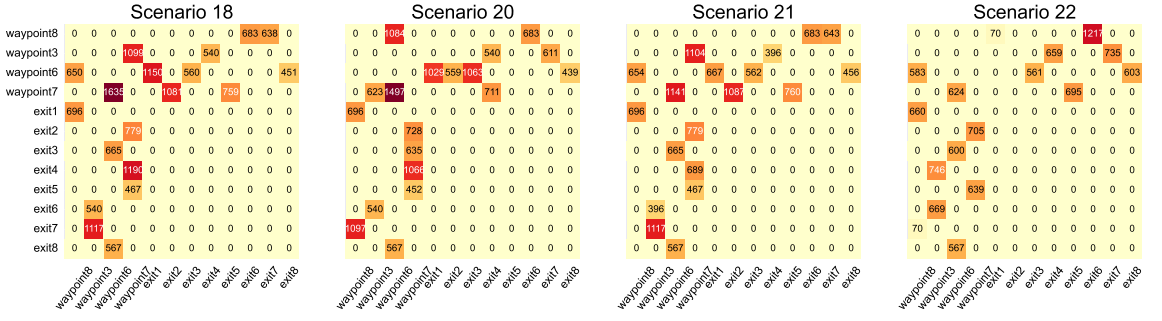
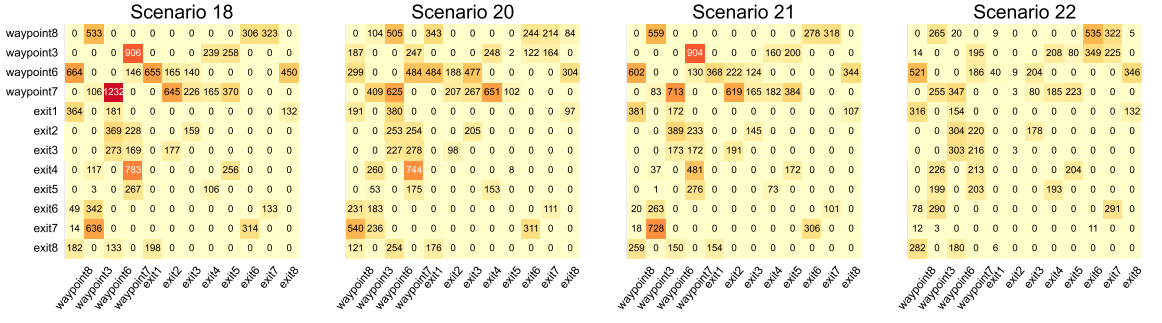


Fig. 14. The change in the estimated route probability as the downstream flow changes.



(a) Ground truth OD matrix of *Maze*



(b) Estimated OD matrix of *Maze*

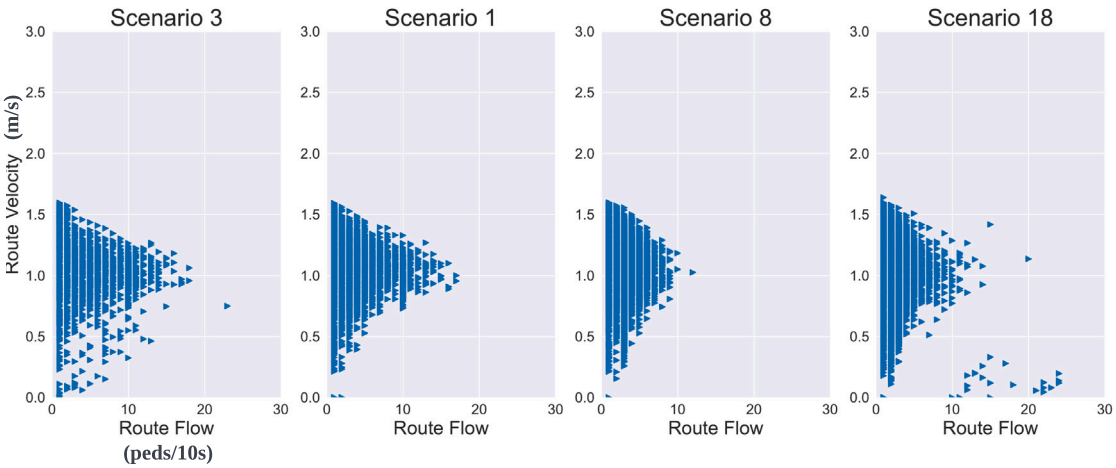
Fig. 15. Comparison between the ground truth OD and the estimated OD, where the colors indicate the number of people crossing — darker colors represent higher numbers.

7.1. OD estimation based on route choice model

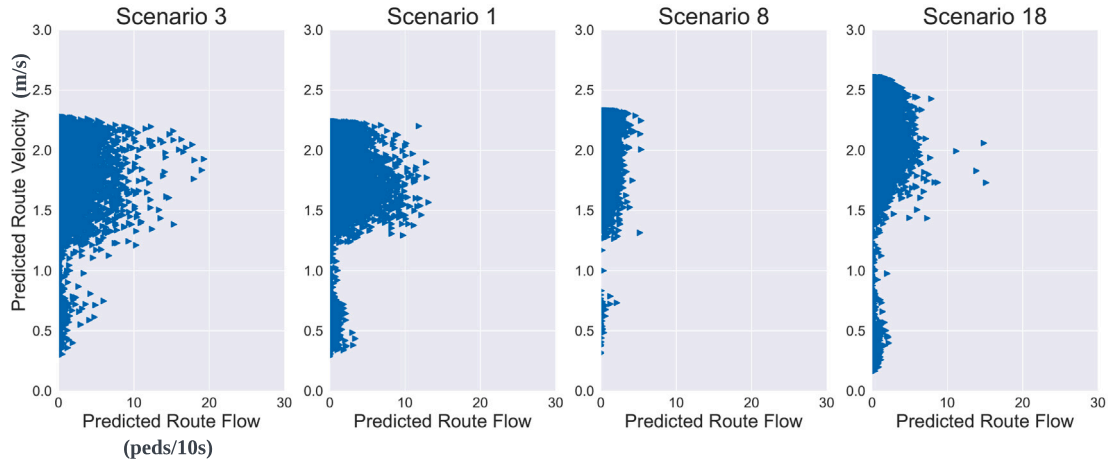
The trajectory data are extracted from the simulator to generate the OD matrix for the *Maze* infrastructure. Using these trajectory data, we calculated the number of people moving from point A to point B during a specific period, enabling us to extract the OD matrix.

In Fig. 14, we focus on a selected upstream sensor and visualize the estimated route choice probabilities for its downstream paths. The results show that the estimated probabilities generally increase with the corresponding downstream flows, following similar trends. In the third plot, which illustrates the estimated probability from origin 23 to destination 16, we notice that even though the downstream flow declines after the 200th timestep, the estimated probability continues to rise. This occurs because the model detects a steeper decline in flow on the other two downstream routes, making the current route relatively more favorable. This response is consistent with intuition: in public spaces, a higher number of pedestrians entering a location often indicates greater attractiveness or utility of that route, and thus more likely to be chosen by others.

We present the ground truth OD matrix for four different scenarios with different pedestrian activity patterns in Fig. 15(a), and the model's estimation in Fig. 15(b). The x-axis represents the origins and the y-axis represents the destinations. We observe that the



(a) Ground truth FD of *Maze*



(b) Estimated FD of *Maze*

Fig. 16. Comparison between the ground truth FD and the estimated FD.

model can identify OD pairs with high demand, given only the node sensor information. However, when pedestrian activity patterns are highly diverse, the accuracy of the route choice model's estimates may decrease. For instance, in Scenario 18, the ground truth OD shows no movement from *waypoint8* to *waypoint3*. Nevertheless, people are coming from *exit7* to *waypoint3* at the same time, so the downstream sensor at *waypoint3* still detects people passing through. Therefore, the model might assume that there is a chance for people to travel from *waypoint8* to *waypoint3*. As a result, the model might infer a possible route from *waypoint8* to *waypoint3*. Since the model relies solely on flow data from node sensors, it is reasonable to expect some discrepancies between the exact ground truth and the estimated OD flows.

7.2. Flow and velocity fundamental diagram

To validate whether the route velocity model can capture the characteristics of the velocity-speed fundamental diagram (FD). We determine the walking speed of agents on a route by computing their travel distance divided by travel time at each time step, then averaging these values to obtain the route velocity. The number of people currently on the route represents the flow. Using this information, we construct the fundamental diagram (FD). In Fig. 16(a), we visualize the ground truth velocity-flow FD, while Fig. 16(b) shows the relationship between the predicted route flow and the predicted route velocity. Please note that the estimated FD and ground-truth FD data points are not a one-to-one correspondence, since we plot the estimated FD using the predicted route velocity and flow. The main goal of this comparison is to see whether our model can learn a general shape of the pedestrian fundamental diagram and pick up the characteristics of different scenarios.

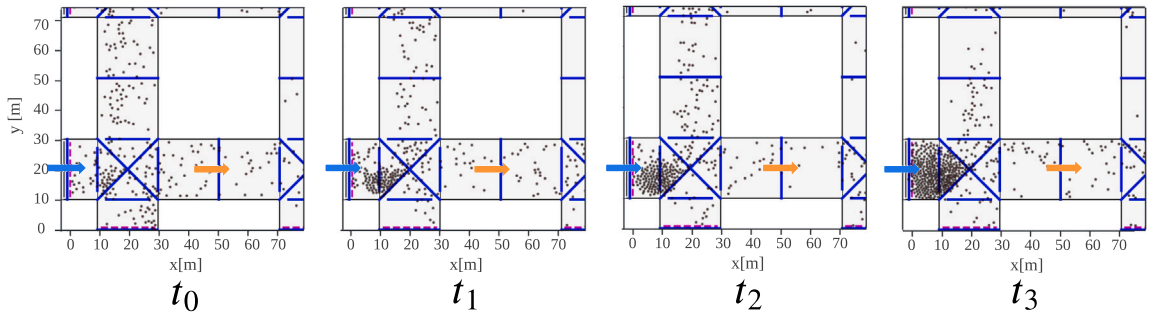
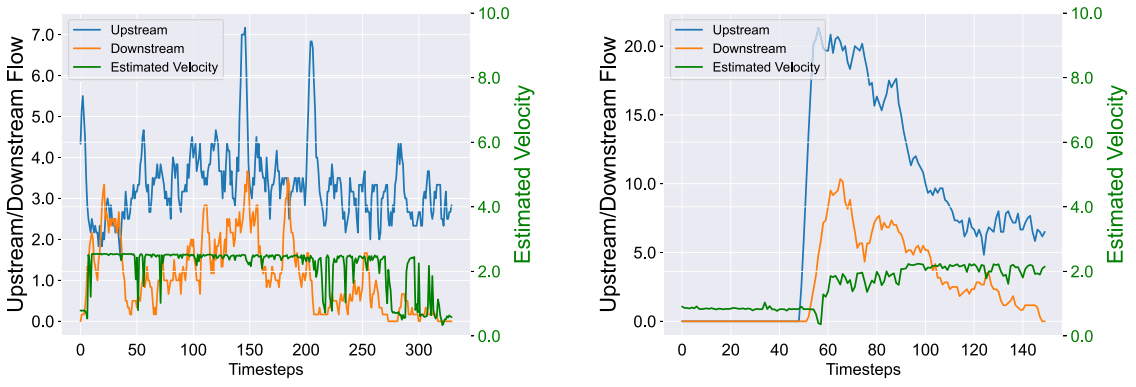


Fig. 17. The evolution of the spillback. Blue arrow represents the upstream flow, orange represents the downstream flow.



(a) Normal Situation

(b) High Congestion

Fig. 18. The model estimates the velocity based on the difference between up and downstream flow.

In the ground truth fundamental diagram, the relationship between speed and flow aligns well with classical traffic flow theory, exhibiting both the free-flow regime (higher speed, lower density) on the descending branch of the speed–flow curve, and the congested regime (lower speed, higher density) on the ascending branch of the speed–flow curve. As the route flow approaches its maximum, the data points cluster around the critical speed. A similar pattern is observed in the estimated FD. However, the predicted speeds are slightly higher than those in the ground truth. This discrepancy can be attributed to two reasons: (1) The model estimates the maximum travel speed between two sensors, and (2) there is a parameter α to control the scale of the predicted speed. As a result, the model tends to predict higher values. Moreover, although the model captures that low flow conditions can correspond to both low and high speeds, it does not fully reproduce the congestion dynamics of the flow-speed relationship. For instance, in Scenario 18, the ground truth FD displays a cluster of data points with high flow and very low speed, which indicates congestion. This pattern is not captured in the estimated FD, suggesting that the model may not fully learn the complex dynamics of congested pedestrian traffic.

However, our model is able to reflect the general changes in speed as flow changes. We investigate the relationship between estimated speed and observed flow in Fig. 18. Fig. 18(a) shows the upstream and downstream flow of one link in the sensor network and the corresponding estimated velocity in a normal situation. The model learns that when there is a significant difference between the upstream and downstream flows, the upstream flow being much higher than the downstream flow, the route velocity should be low (as seen in the drop in velocity in Fig. 18(a)). This property aligns with a general physical phenomenon: if we analogize the corridor to a pipe and people to fluid, a higher influx than outflux indicates low fluid velocity, suggesting potential congestion at the connection.

Fig. 18(b) shows the estimated speed in a highly congested corridor, corresponding to the scenario visualized in Fig. 17. Initially, there is high demand at the entrance point, but people get stuck in the corridor, causing the downstream flow to decrease. This congestion back-propagates through the upstream sensor, leading to a decrease in upstream flow. In this case, the predicted speed first drops due to the high upstream inflow but low downstream outflow. The model fails to recognize the congestion afterward because the difference between the upstream and downstream flows is not large enough to trigger the model to predict a low velocity. It also cannot detect spillback effects, as the model does not explicitly incorporate the principle of mass conservation in pedestrian flow. Since the sensors continue to record pedestrian movement, the model may interpret that pedestrians are still passing through the hallway. However, it lacks the information needed to identify congestion and when it will propagate upstream and affect the previous node. This limitation explains why the model does not fully capture the congestion patterns in the fundamental diagram. Given the absence of direct measurements for link-level flow and density, this represents a trade-off between relying solely on observable data and achieving a more complete representation of traffic dynamics.

8. Limitations

This section outlines the limitations related to both the data used in this study and the design of the proposed model. The synthetic datasets were developed to assess the performance of the models in high-demand scenarios featuring diverse pedestrian activity patterns. Although synthetic simulation enables the generation of controlled and repeatable scenarios by adjusting simulation parameters, such data may not fully capture the complexities of real-world pedestrian behavior.

Real-world data, on the other hand, can provide authentic representations of pedestrian dynamics. However, high-quality datasets that meet the spatial and temporal granularity required for this study, particularly in large-scale public environments, are still limited. Although we include a real-world dataset (EIFPD) in our experiments, its limited scale and sparsity reduce its effectiveness for comprehensive model evaluation. This limitation emphasizes the value of using synthetic data and highlights the need to develop richer, large-scale pedestrian datasets in future work.

Regarding the prediction model, it is designed to be interpretable and informed by traffic flow theory. It infers intermediate variables such as walking speed and route choice, rather than directly fitting observed flow using black-box techniques. While this approach limits the model's flexibility compared to purely data-driven methods, it enhances interpretability and provides additional traffic information that is valuable for operational decision-making.

An additional limitation is the model's inability to predict during congestion, as it relies solely on sensor flow data and does not incorporate density measurements between sensor locations. This design choice reflects a trade-off between the limited availability of real-time density data, which are often difficult to obtain, and the practical need for timely predictions based on accessible inputs (e.g. counts/flow).

9. Conclusion and future work

In this paper, we present a learning-based pedestrian flow prediction model utilizing diffusion behavior theory. The model features a decoupled architecture for enhanced transparency. In addition, we incorporate the model into an online learning framework to address the scenario drift problem. Our model can estimate the traffic conditions between sensors using only the sensor flow data, serving as a short-term crowd state estimator to support real-time intervention. Overall, this paper demonstrates the potential of combining crowd diffusion models with neural networks for macroscopic pedestrian flow prediction in highly dynamic public environments, with better robustness than the other off-the-shelf models.

In the future, we plan to explore more advanced online learning techniques beyond the simple OGD, which requires less retraining time. We may incorporate uncertainty estimation to determine the optimal moments for model updates. Furthermore, as discussed in the previous section, the velocity model may fail to make accurate predictions during congestion when relying solely on sensor data. In the future, we may explore designing a module in the neural network to implicitly infer the density between sensors, e.g. using the information of the cumulative number of pedestrians that pass through upstream and downstream sensors. This could enhance the accuracy of both the velocity and diffusion models, particularly under congested conditions. Lastly, a promising direction for future work is to integrate our model into real-time crowd management or control algorithms, using its short-term predictions as input. This can provide approximate traffic state estimates that support more effective operational decisions.

CRediT authorship contribution statement

Weiming Mai: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Dorine Duives:** Writing – review & editing, Supervision, Software, Methodology, Funding acquisition, Conceptualization. **Serge Hoogendoorn:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

AI Tools Disclosure

During the preparation of this work, the authors used ChatGPT (OpenAI) to assist with writing revision and language polishing. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We acknowledge the support of the NWO, VENI project CrowdIT space and TU Delft AI Labs programme.

Appendix A. Prediction on different locations

This section presents the prediction visualizations on different locations in the infrastructure (see Figs. A.19–A.23).

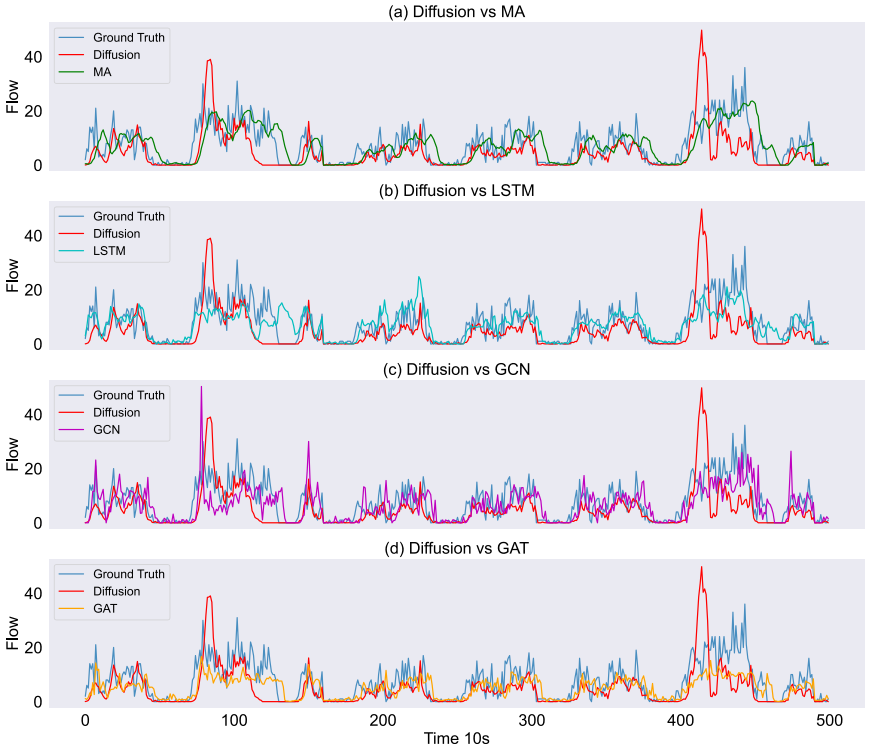


Fig. A.19. Models' 40-s ahead predictions at main exit in *TrainStation* (Evacuation).

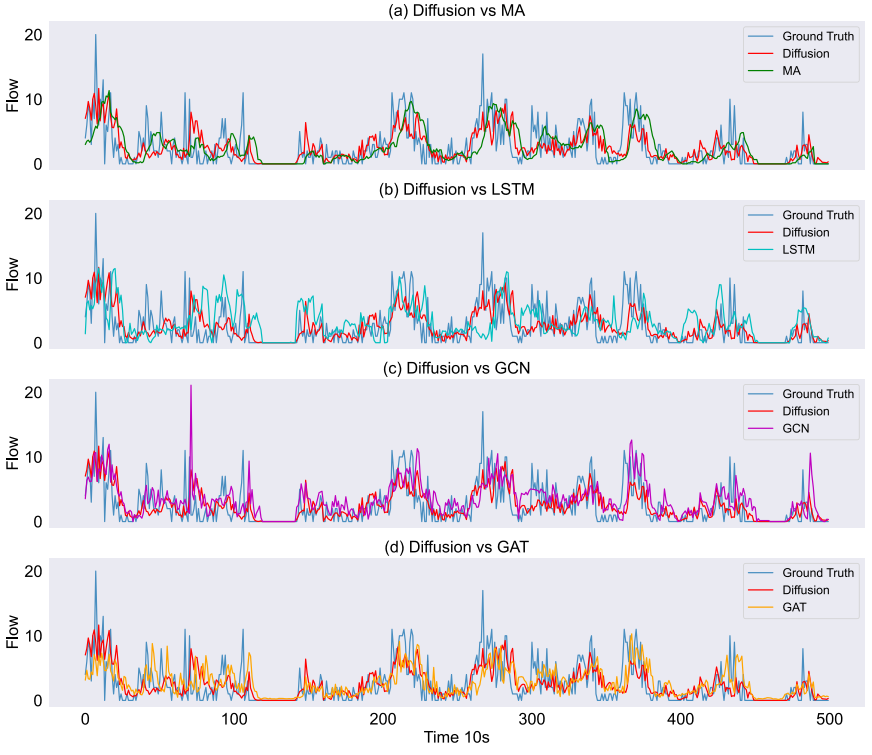


Fig. A.20. Models' 40-s ahead predictions at platform entrance in *TrainStation*.

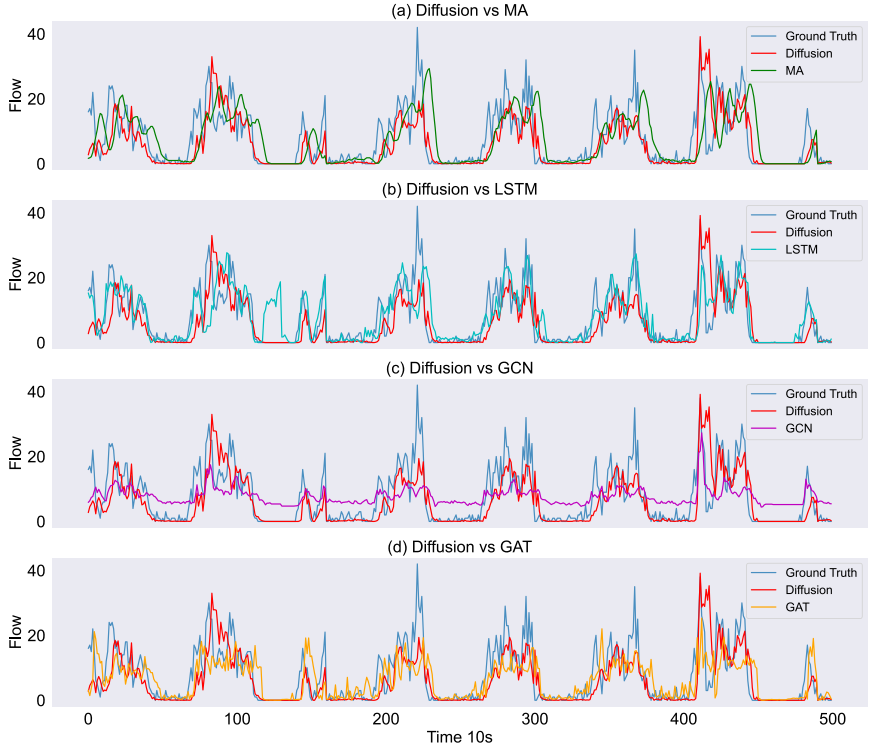


Fig. A.21. Models' 40-s ahead predictions at the main hall in *TrainStation* (Evacuation).

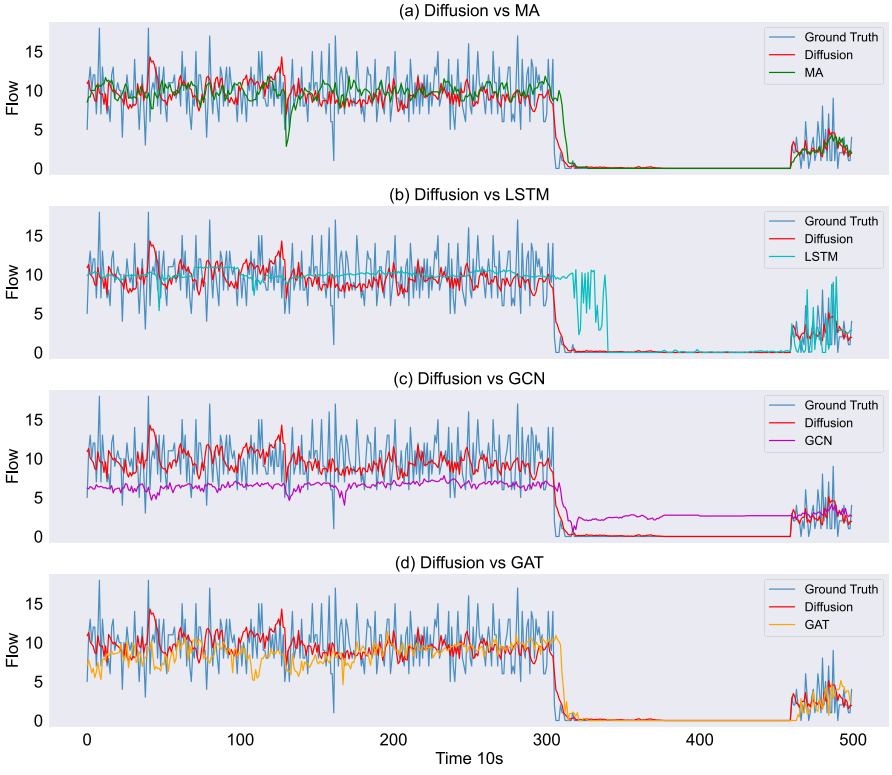


Fig. A.22. Models' 40-s ahead predictions at the main hall in *Maze*.

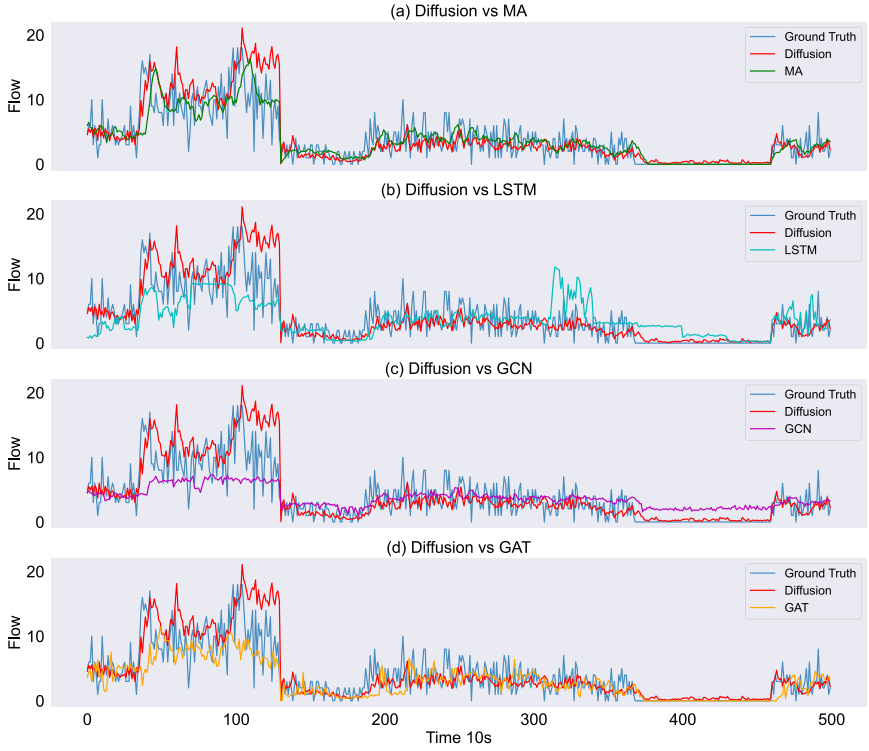


Fig. A.23. Models' 40-s ahead predictions at the main hall in *Maze*.

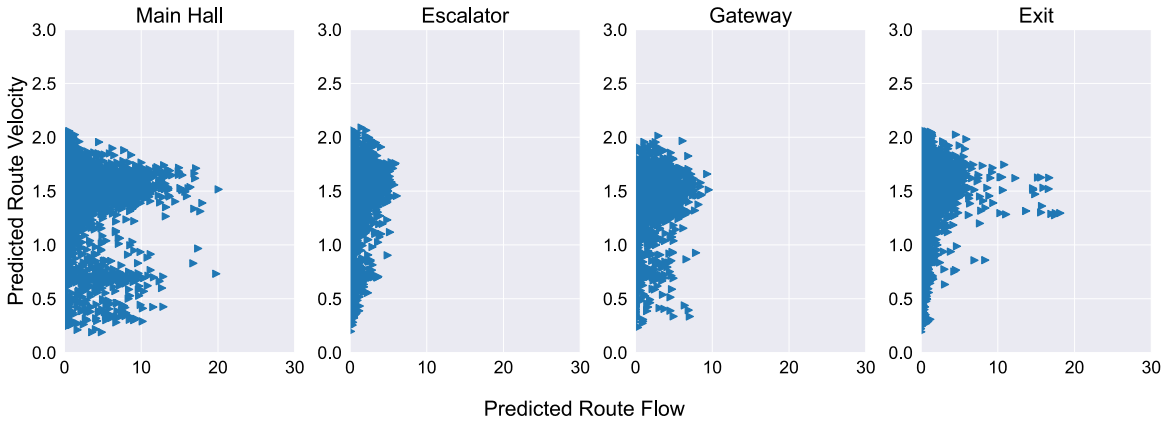


Fig. B.24. Speed-Flow FD of different locations at *TrainStation*.

Appendix B. Estimated speed-flow FD in train station

This section presents the estimated fundamental diagram on different location in the train station (see Fig. B.24).

Appendix C. Setting of the simulation

The simulation consists of two generators to generate the agent and transport element (train). The generator generates passenger and train according to the predefined parameters. During the simulation the agents would perform the activities we designed. Once

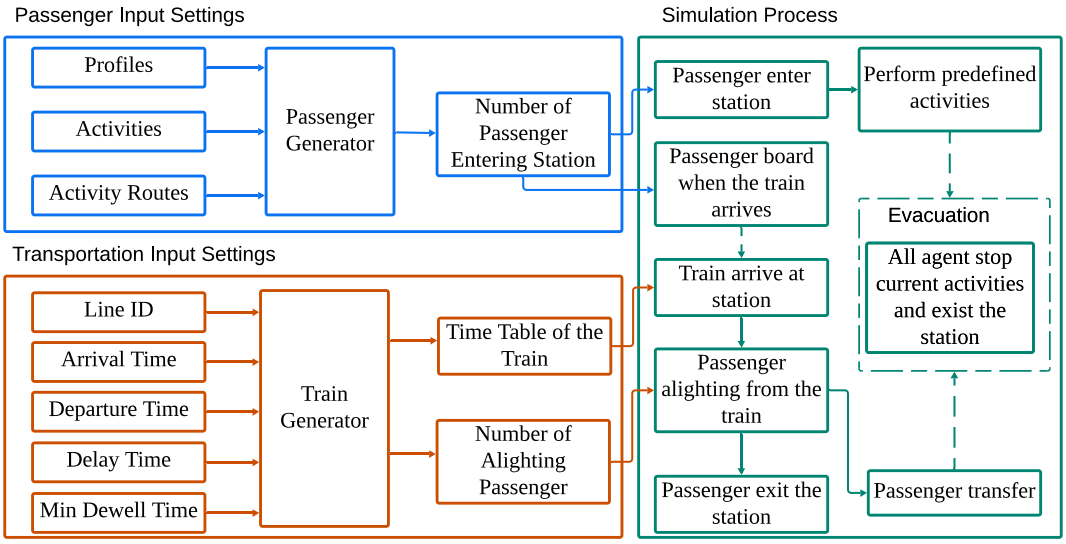


Fig. C.25. Simulation process of the synthetic dataset *TrainStation*.

the evacuation is activated, all agents need to stop their current activities and need to exit the station. For a more detailed setting, readers can refer to this paper (Mai et al., 2025) (see Fig. C.25).

References

Bamaqa, A., Sedky, M., Bosakowski, T., Bastaki, B.B., Alshammari, N.O., 2022. SIMCD: Simulated crowd data for anomaly detection and prediction. Expert Syst. Appl. 203, 117475.

Borgers, A., Timmermans, H., 1986. A model of pedestrian route choice and demand for retail facilities within inner-city shopping areas. Geogr. Anal. 18 (2), 115–128.

Chapados, N., 2014. Effective Bayesian modeling of groups of related count time series. In: Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014. In: JMLR Workshop and Conference Proceedings, 32, JMLR.org, pp. 1395–1403.

Domingos, P., Hulten, G., 2000. Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 71–80.

Dynamics, I.E., 2024. Pedestrian dynamics. Version 4.5. URL https://www.incontrolsim.com/software-platform/pedestrian-dynamics/.

Fang, S., Zhang, Q., Meng, G., Xiang, S., Pan, C., 2019. GSTNet: Global spatial-temporal network for traffic flow prediction. In: IJCAI. pp. 2286–2293.

Geraerts, R., 2010. Planning short paths with clearance using explicit corridors. In: 2010 IEEE International Conference on Robotics and Automation. IEEE, pp. 1997–2004.

Haghani, M., 2020. Optimising crowd evacuations: Mathematical, architectural and behavioural approaches. Saf. Sci. 128, http://dx.doi.org/10.1016/j.ssci.2020.104745.

Han, X., Zhao, X., Zhang, L., Wang, W., 2023. Mitigating action hysteresis in traffic signal control with traffic predictive reinforcement learning. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 673–684.

Hänseler, F.S., Molyneaux, N.A., Bierlaire, M., 2017. Estimation of pedestrian origin-destination demand in train stations. Transp. Sci. 51 (3), 981–997.

Helbing, D., Molnar, P., 1995. Social force model for pedestrian dynamics. Phys. Rev. E 51 (5), 4282.

Henderson, L.F., 1974. On the fluid mechanics of human crowd motion. Transp. Res. 8 (6), 509–515.

Henderson, L., Lyons, D., 1972. Sexual differences in human crowd motion. Nature 240 (5380), 353–355.

Hoi, S.C.H., Sahoo, D., Lu, J., Zhao, P., 2021. Online learning: A comprehensive survey. Neurocomputing 459, 249–289.

Jia, X., Feliciani, C., Tanida, S., Yanagisawa, D., Nishinari, K., 2024. Evaluating pedestrian congestion based on missing sensing data. J. Disaster Res. 19 (2), 336–346.

Karamouzas, I., Sohre, N., Hu, R., Guy, S.J., 2018. Crowd space: a predictive crowd analysis technique. ACM Trans. Graph. 37 (6), 1–14.

Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks. In: ICLR (Poster). OpenReview.net.

Korbmacher, R., Tordeux, A., 2022. Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches. IEEE Trans. Intell. Transp. Syst. 23 (12), 24126–24144.

Li, M., Tang, Y., Ma, W., 2022. Few-shot traffic prediction with graph networks using locale as relational inductive biases. arXiv preprint arXiv:2203.03965.

Liang, N.-Y., Huang, G.-B., Saratchandran, P., Sundararajan, N., 2006. A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Netw. 17 (6), 1411–1423.

Lilasathapornkit, T., Rey, D., Liu, W., Saberi, M., 2022. Traffic assignment problem for footpath networks with bidirectional links. Transp. Res. Part C: Emerg. Technol. 144, 103905.

Liu, Y., Liu, Z., Jia, R., 2019. DeepPF: A deep learning based architecture for metro passenger flow prediction. Transp. Res. Part C: Emerg. Technol. 101, 18–34.

Liu, Y., Sun, C., Bie, Y., et al., 2015. Modeling unidirectional pedestrian movement: An investigation of diffusion behavior in the built environment. Math. Probl. Eng. 2015.

Løvås, G.G., 1994. Modeling and simulation of pedestrian traffic flow. Transp. Res. Part B: Methodol. 28 (6), 429–443.

Mai, W., Duives, D., Krishnakumari, P., Hoogendoorn, S., 2025. Evaluating crowd flow forecasting algorithms for indoor pedestrian spaces: A benchmark using a synthetic dataset. IEEE Trans. Intell. Transp. Syst. 1–16. http://dx.doi.org/10.1109/TITS.2025.3559353.

Majecka, B., 2009. Statistical Models of Pedestrian Behaviour in the Forum (Ph.D. thesis). Citeseer.

Makinoshima, F., Oishi, Y., 2022. Crowd flow forecasting via agent-based simulations with sequential latent parameter estimation from aggregate observation. Sci. Rep. 12 (1), 11168.

Manibardo, E.L., Lana, I., Ser, J.D., 2022. Deep learning for road traffic forecasting: Does it make a difference? IEEE Trans. Intell. Transp. Syst. 23, 6164–6188. http://dx.doi.org/10.1109/TITS.2021.3083957.

Rasouli, A., 2021. Pedestrian simulation: A review. arXiv preprint arXiv:2102.03289.

Salinas, D., Flunkert, V., Gasthaus, J., Januschowski, T., 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. Int. J. Forecast. 36 (3), 1181–1191.

Seedat, N., Imrie, F., van der Schaar, M., 2022. Dc-check: A data-centric ai checklist to guide the development of reliable machine learning systems. arXiv preprint arXiv:2211.05764.

Sparnaaij, M., Campanella, M.C., Duives, D.C., Daamen, W., Hoogendoorn, S.P., 2023. NOMAD: A microscopic pedestrian simulation model. URL https://gitlab.tudelft.nl/active-mode/nomad. Version 1.0.

Tordeux, A., Lämmel, G., Hänseler, F.S., Steffen, B., 2018. A mesoscopic model for large-scale simulation of pedestrian dynamics. Transp. Res. Part C: Emerg. Technol. 93, 128–147.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., et al., 2017. Graph attention networks. Stat 1050 (20), 10–48550.

Wares, S., Isaacs, J., Elyan, E., 2019. Data stream mining: methods and challenges for handling concept drift. SN Appl. Sci. 1, 1–19.

Yuan, Y., Goñi-Ros, B., Bui, H.H., Daamen, W., Vu, H.L., Hoogendoorn, S.P., 2020. Macroscopic pedestrian flow simulation using smoothed particle hydrodynamics (SPH). Transp. Res. Part C: Emerg. Technol. 111, 334–351.

Zheng, Y., 2015. Trajectory data mining: an overview. ACM Trans. Intell. Syst. Technol. (TIST) 6 (3), 1–41.