Pedestrian volume prediction using a Diffusion Convolutional Gated Recurrent Unit Model

Yiwei Dong, Tingjin Chu, Lele Zhang, Hanfang Yang, Hadi Ghaderi

***Abstract*—Development of models to analyse and predict pedestrian flow is important to ensure the safety of both pedestrians and other road users. Such tools are also beneficial to optimise the design and geometry of infrastructure and support the economic utility of wider interacting communities. The deployment of city-wide automatic pedestrian counting systems is providing researchers with invaluable data to develop and train deep learning applications for a better understanding of traffic and crowd flows. Benefiting from real-world data provided by the City of Melbourne pedestrian counting system, this study presents a pedestrian flow prediction model, as an extension of Diffusion Convolutional Recurrent Neural Network (DCRNN) with dynamic time warping, named DCGRU-DTW, which features the spatial dependency of pedestrian flow captured by the diffusion process and the temporal dependency captured by Gated Recurrent Unit (GRU). By performing extensive numerical experiments, we demonstrate that the proposed model outperforms the classic DCGRU, the vector autoregressive model, and the historical average in terms of multiple model accuracy measures. Our observation from applying DCGRU-DTW for pedestrian flow prediction is that the model is capable of incorporating complex spatial correlations and (long-term) temporal dependencies. Finally, this study could significantly benefit both transport technologists and city planners for urban design, deployment and utilisation of pedestrian counting systems.**

***Index Terms*—Diffusion convolutional gated recurrent unit, dynamic time warping, pedestrian flow, prediction, crowd, crowd management.**

I. Introduction

Development of technologies and methods to understand the flow of pedestrians in dense urban areas has been receiving additional momentum in the intelligent transport systems (ITS) research [1]. Poorly managed pedestrian flow could result in sub-optimal use of transport systems and also lead to adverse events such as fatalities and injuries [2]. Therefore, tools for analysis of pedestrian flows are essential for the planning and geometric aspects of cities; design of infrastructure in normal times and addressing safety concerns [3]. In this context, the number of crowd and pedestrian related disasters has been on the rise around the world for a variety of reasons, including a lack of timely information about special conditions and anomalies in crowd behaviour [4]. Without such insight, identifying the appropriate crowd and pedestrian response strategy remains ineffective [5]. Therefore, predicting accurate spatiotemporal information about pedestrian activity such as volume, velocity and direction is fundamental to public safety and efficient urban planning [6].

In recent years, there have been significant improvements in the form factor, performance and acquisition expenses of sensor technologies, allowing them to be widely deployed in a variety of applications related to transport and mobility [7]. Specifically, video-based solutions [8], infrared [9], wifi [10] and Bluetooth sensors [11], supported by other information and communication technologies are becoming mainstream applications to improve crowd safety and optimised pedestrian flow via the enhanced decision making [12]. A sizeable stream of extant literature pertaining to the applications of emerging technologies for the crowd and pedestrian management focuses on evaluating their performance and effectiveness in varying environmental and operational conditions [13]. In this context, studies applying such technologies, predominantly, aim to improve public safety and minimise potential stampede for special events [14] or public transportation systems [15] based on controlled observations and experiments [16]. Such applications are characterised by confined spatial and temporal boundaries and have limitations when applied to city-wide contexts [17].

The City of Melbourne has deployed a pedestrian counting system that benefits from thermal and laser based sensors. This non-vision based technology, allows for recording multi-directional pedestrian movements by forming a counting zone, without invading privacy [18]. The technology stores data using onsite data loggers, which are later transferred to a central server using third generation mobile network (3G) every 10 to 15 minutes for visualisation and access [19]. We benefited from this data to develop a deep learning based tool for the assessment and prediction of pedestrian flows and patterns. Such a tool, with a large degree of accuracy, could provide insights into pedestrian activity patterns and potentially be used to identify anomalies and extreme conditions presented in pedestrian flow behaviour at a city-wide scale [20].

In recent years, deep learning applications have demonstrated their effectiveness in forecasting traffic and crowd flow [21]. Among these models, diffusion convolutional recurrent neural network (DCRNN) is particularly compelling at incorporating complex spatial correlations and (long-term) temporal dependencies [22]. This study proposes an extension of the DCRNN model, named DCGRU-DTW, which has

Manuscript received July 25, 2023. Corresponding author: Hadi Ghaderi

Yiwei Dong is with School of Statistics, Renmin University of China, China (email: ydong@ruc.edu.cn). Tingjin Chu and Lele Zhang are with School of Mathematics and Statistics, The University of Melbourne, Australia (email: tingjin.chu@unimelb.edu.au and lele.zhang@unimelb.edu.au). Hanfang Yang is with School of Statistics, Renmin University of China, China (email: hyang@ruc.edu.cn). Hadi Ghaderi is with School of Business, Law and Entrepreneurship, Swinburne University of Technology, Australia (email: hghaderi@swin.edu.au).

the spatial dependence of pedestrian flow captured by the diffusion process and the temporal dependency captured by gated recurrent unit (GRU), a variant of RNN.

The present work provides several contributions to the literature on pedestrian flow analysis, particularly the stream that is aligned with the applications of artificial intelligence and sensor technologies. These contributions are summarised as follows:

- To the best of our knowledge, this study is the first to apply DCRNN for the purpose of pedestrian volume forecasting.
- Unlike spatiotemporal forecasting of vehicles, there lacks a good metric to measure the proximity between different locations for pedestrian volume forecasting, because pedestrians can choose their paths with great freedom. This work is the first to leverage the time series data between sensor signals to quantify the distance between the corresponding sensor locations and combines it with DCRNN.
- This study employs the time series similarity measured using dynamic time warping (DTW) as additional information to construct the adjacency matrix of DCRNN. The established DCGRU-DTW model achieves the best results among the five methods, including the original DCRNN and the classic VAR model.

In the remainder of this paper, we first provide a summary of extant literature by focusing on three streams of statistical, machine learning and deep learning based methods. We then introduce the DCGRU model and its application for pedestrian volume prediction. Next, experiments set up procedures and results are presented, followed by concluding remarks and directions for future research.

II. Literature Review

This section briefly reviews the most relevant literature pertaining to the statistical methods, machine learning models, and relevant to the scope of this study, deep learning models that are applied to forecasting the flows of crowds, pedestrians and (road) traffic. Compared to the flows of crowds and pedestrians, traffic flow prediction has received more attention in the literature, mainly because of its historical importance for the management of congestion [23]. Not surprisingly, it is natural to see models were first developed and applied for traffic prediction and then modified or re-purposed to forecast pedestrian or crowd flow.

A. Statistical methods

Among the classical statistical methods, one of the most widely used is the autoregressive integrated moving average model (ARIMA). ARIMA-based models show satisfactory results in traffic congestion prediction for datasets characterised by non-stationary and non-normally distributed short-term traffic time series [24]. Appropriate data preprocessing and model design could improve the performance of the ARIMA-based method in most cases [25]. Over the years, a number of ARIMA variants have been studied. One is the seasonal ARIMA model, which has been shown to be effective for traffic forecasting by capturing the behaviour of various traffic streams. In the field of traffic flow, the work in [26] explains the theoretical basis of seasonal ARIMA models. In this study, the authors analyzed empirical results and examined promising applications of seasonal ARIMA in the field of ITS. The subset model is another variant of ARIMA that demonstrates superiority over the classical model in applications related to short-term traffic volume forecasting. The procedure of time series modelling using the subset ARIMA is elaborated in [27]. Shahriari et al. [28] combined Bootstrap with the conventional parametric ARIMA model to perform traffic volume prediction. The proposed E-ARIMA retains the theory adherence and meantime outperforms the comparative methods. ARIMA can be modified to conduct multivariate time series prediction by adding multiple time series as exogenous variables, such as ARIMAX [29].

The vector autoregressive model (VAR) is another statistical model for multivariate time series [30]. The literature reports on its effectiveness for traffic forecasting [31], [32]. Chandra et al. [31] found that the VAR model is more powerful than the traditional ARIMA model when predicting freeway traffic speed and volume, using the case of the downtown area of Orlando, Florida. The superior performance can partly be attributed to the fact that VAR could take advantage of upstream and downstream location information, while the traditional ARIMA model neglects this information. Variants of VAR are being explored with interest by researchers to address various traffic forecasting applications. For example, Schimbinschi et al. [33] proposed topology-regularized universal vector autoregression (TRU-VAR), which relies on the topology-designed adjacency matrix. Experiments of TRU-VAR demonstrate it can yield reliable traffic forecasts in urban areas. With a similar structure, the Bayesian vector autoregressive moving average model, whose parameters are estimated via the Bayesian inference framework, is another statistical model capable of achieving accurate and efficient short-term traffic flow forecasting [34].

B. Machine learning based models

Various machine learning models have been proposed to tackle the traffic forecasting problem, and some have yielded satisfying results in traffic volume and speed forecasting [35]. The k-nearest neighbour (KNN) is a classical machine learning algorithm. Using modifications of KNN, satisfactory forecasting outcomes can be obtained for various traffic-related applications. In [36], a fully automatic dynamic procedure KNN (DP-KNN) was proposed, showing accurate results on average for short-term traffic forecasting. Cai et al. [37] leveraged the spatiotemporal correlation and developed a KNN model capable of multi-step forecasting with enhanced accuracy. Similarly, Xia et al., [38] proposed a spatiotemporal weighted k-nearest neighbour model (STW-KNN) for short-term traffic flow forecasting. In this context, support vector regression (SVR) is a well-established supervised machine learning algorithm for regression applications [39]. The least squares support vector machine (LS-SVM) was proposed and utilized to predict the travel time index (TTI) in [40].

Experiments showed that LS-SVM can produce smaller mean absolute percentage errors and variance of absolute percentage errors in the testing set, demonstrating good generalization ability. Hu et al. [**?**] used quarterly conversion coefficients combined with the SVM model to establish a predictor of traffic volume of freeway toll stations. Unsupervised machine learning algorithms, such as the hidden Markov model [41], [42], are also applied to traffic forecasting problems.

While several statistical and machine learning based models have been used extensively for traffic forecasting, limited applications exist for the prediction of pedestrian and crowd flows. Wang et al. [18] applied three methods, namely ARIMA, SVR, and multiple linear regression (MLR), to predict pedestrian flow in the CBD of Melbourne using data from the pedestrian counting system. They found ARIMA performed best in terms of MAPE, whilst SVR is the worst performing compared to the other two methods. Tan et al. [43] conducted experiments on a similar dataset and compared the prediction effectiveness of ARIMA, VAR, and spatiotemporal autoregression (STAR). They found ARIMA was outperformed by VAR. As pointed out by Zhag et al. [6], the classic statistical and machine learning models have limitations in capturing complex spatial and temporal dependencies, preventing them from achieving higher prediction accuracy.

C. Deep learning based models

Deep learning is one of the fastest developing techniques in the field of ITS, particularly for applications related to big data analytics [44], [45], [46]. Owing to the improvements in computational power, it is possible to efficiently conduct parameter estimation in deep neural networks [47]. Deep learning based traffic forecasting methods are more efficient in modelling the highly nonlinear and stochastic characteristics of complex transportation systems compared to traditional statistical and machine learning-based models, hence, resulting in substantial prediction performance e.g., ([48], [49]). Among them, the recurrent neural network (RNN) featuring long short-term memory (LSTM) network is one of the earliest models applied to the traffic forecasting problem [48]. The joint application of LSTM and gated recurrent unit (GRU) neural networks is another example used for traffic prediction [50]. Since RNNs can mainly deal with temporal dependence, researchers have proposed new variants that can achieve spatiotemporal forecasting. For example, the deep multi-view spatial-temporal network (DMVST-Net) framework was proposed to model both spatial and temporal relations, which was used to solve a large-scale real taxi demand data effectively [46]. Zhang et al. [51] designed a spatiotemporal feature selection algorithm (STFSA), which can facilitate the convolution neural network (CNN) to improve traffic flow forecast accuracy. There are also other attempts such as the development of hybrid predictors using both RNNs and CNNs to endow models with spatiotemporal forecasting ability [52]. In addition, stacked autoencoder (SAE) models excel in feature extractions and are also able to yield satisfactory prediction results for traffic flow forecasting tasks [45].

Among deep neural network models, graph neural networks (GNNs) have received additional attention [53], [54], [55]. Because of their ability to handle complexities among nodes, GNNs are ideal tools to model the spatial dependency between different sites/roads/regions in traffic forecasting problems [53]. Typically in GNNs, the traffic graph is constructed as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{\mathcal{A}})$, where $\mathcal{V}$ is the node set , $\mathcal{E}$ is the edge set, and $\boldsymbol{\mathcal{A}}$ is the adjacency matrix. While the adjacency matrix in GNNs plays an important role in capturing the spatial dependencies, various ways exist to represent such a matrix in traffic forecasting problems [53]. For example, typically for a road based matrix, the road connection matrices, transportation connectivity matrices, and direction matrices are merged together. This way of constructing the adjacency matrix allows for modelling complex road networks such as arterial roads, highways, and subway systems effectively [54]. The distance-based matrix concentrates on the spatial closeness between nodes. This type of adjacency matrix is used in [56] to model the distance, direction and positional relationship. Another type of adjacency matrix is a similarity based matrix, which considers traffic patterns or functional similarity matrices [49]. Similarity based matrix is useful to reveal the intrinsic mutual dependencies, correlations and distributions of different locations. For example, in [57], dynamic time warping (DTW) between different nodes is used to understand the temporal similarity of traffic data and guide the graph partitioning method. The authors found that the prediction accuracy of this method combined with spatial-temporal graph convolutional networks in an edge-computing system (STGCN-EC) significantly improves. Lastly, dynamic matrices are also considered in traffic forecasting tasks as an alternative to pre-defined adjacency matrices [58], [59], [60], [61].

Compared to motor vehicle related modelling, there are fewer works in the literature dedicated to pedestrian or crowd flow forecasting. We identified three groups of related studies: (i) passenger flow prediction at public transportation stations such as rail transit stations or bus stations [62], [63], [64], [65], (ii) crowd flow prediction using GPS data collected from cell phones, taxis, shared bikes, etc. [66], [67], [6], [68], [69], and (iii) pedestrian volume prediction using the data collected by various types of counting systems such as camera or thermal sensors across a large region [70].

Although the models explained in previous sections, as well as other traffic forecasting approaches can be used for pedestrian or crowd flow forecasting, we acknowledge that pedestrian flow forecasting problems represent unique complexities [70]. Unlike motor vehicles, whose trajectories and velocities are restricted by road network conditions and traffic regulations, pedestrian flows and patterns often present significant uncertainties [71]. As such, in many cases, it is naturally more difficult to simply employ traffic forecasting models developed for motor vehicles for pedestrian flow prediction and various special features of pedestrian flow need to be considered when designing the model [70], [72].

On the other hand, a large portion of the crowd and pedestrian flow production studies is dedicated to understanding passengers' behaviour around transit stations. For example, Liu et al. [64] proposed a deep learning framework based on LSTM to model spatiotemporal dependencies and external factors, such as weather conditions and holidays to predict the in-/out flow of metro passengers. In [65], the proposed dynamic graph recurrent convolutional neural network (Dynamic-GRCNN), is able to capture the periodicity and trends of passenger flows with satisfactory performance to model both station-level bus and subway passenger flows. Similarly, Fang et al. [62] designed a meta-learning based multi-source spatiotemporal network with an encoder-decoder structure. They conducted experiments using data from the Beijing public traffic system to verify the effectiveness of their method.

As for crowd prediction, Zhang et al. [66] developed a deep spatiotemporal residual network (ST-ResNet) using convolution based residual networks, while Zhang et al. [67] applied multi-graph neural networks equipped with graph attention networks (GAT) and self-attention to extract spatiotemporal correlations. Yuan et al. [68] adopted double-branch residual attention networks named MV-RANet to capture spatiotemporal correlations and effects of external factors such as weather conditions. More recently, Zhang et al. [6] proposed an integrated model using the gated recurrent unit (GRU) to capture the temporal structure, the convolutional neural network (CNN) to capture the spatial structure, and k-nearest neighbours (KNN) for fusion. For evaluating the performance of the proposed models, the studies [66], [67] used the taxi GPS datasets and the bike rental dataset while, respectively. While the works in [68], [6] used the GPS datasets from cell phones and taxis. With the GPS data, works such as [6] can partition the studied area into regular grid cells, and handle the grid data by a graphic based method. The CNN method used in [6] can be generalised to graphical-based methods [73], [74], [70], which allow for higher weights to closer locations. However, the pedestrian flow data of this study is collected through sensors and cannot be transformed into grid data, therefore, we use a graphical based method for this study. Duives et al. [69] applied a recursive neural network with a gated recurrent unit (RNN-GRU) to forecast crowd movement (from one grid cell to another) using GPS trace data. This work provides a detailed methodology for data cleaning for identifying the pedestrian on the move.

In the same vein, Liu et al. [70] studied the pedestrian volume data recorded by sensors with high granularity, specifically pedestrian counts per minute. The authors examined the standard geographic conventional network (GCN) and its spatiotemporal graph convolutional network (STGCN) extension. Experiments conducted in this study compared the performance of their pedestrian forecasting flow with baselines of historical average (HA) and four other models namely, ARIMA, support vector machine (SVM), convolutional neural network (CNN) and long short-term memory (LSTM). The results of experiments showed that STGCN performs the best in terms of MRSE, MAE, and R^2. Although the GCN was marginally outperformed by STGCN, it was much more computationally efficient. The authors stated that the GCN models were more effective because they can capture the graph structure of the pedestrian network and spatial dependency. However, one limitation indicated by this study was that the adjacency matrix was solely based on the distances of detectors.

III. DCGRU Model

In this section, we present the modified DCGRU model for pedestrian volume prediction. Based on the original DCGRU [74], time series similarity between pedestrian counting sensors is added to the adjacency matrix of the sensor graph. Compared with the original DCGRU, the modified DCGRU improves the spatiotemporal forecasting ability by incorporating the connectivity and proximity between pedestrian counting sensors via DTW. In the remainder of this section, we first define the formulation of the pedestrian volume forecasting problem and then introduce the components of our model, as well as the overall architecture.

A. Pedestrian Volume Forecasting Problem Formulation

We consider a system of sensors deployed at intersections in an urban area to measure pedestrian volumes. The pedestrian volume forecasting problem in this study is aimed to predict the number of pedestrians passing through each sensor in the future. Let $X_{i,t}$ be the observed number of pedestrians detected at the i-th sensor during the timestamp t for $t = 1, 2, \ldots, T$. The size of a timestamp in this study is one hour. The collection of the data from all the sensors forms a multivariate time series $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_T\}$, where $\boldsymbol{X}_t = (X_{1,t}, \ldots, X_{N,t})^T \in \mathbb{R}^N$ and N is the number of sensors. The goal of the pedestrian volume forecasting problem is to seek a function $h(\cdot)$ that maps the data of the past T timestamps to predict the future values of T' timestamps, given some other information $\mathcal{I}$ such as the geographical information in the traffic study:

$$[\boldsymbol{X}_1, \cdots, \boldsymbol{X}_T; \mathcal{I}] \xrightarrow{h(\cdot)} [\boldsymbol{X}_{T+1}, \cdots, \boldsymbol{X}_{T+T'}] .$$

B. Gated Recurrent Unit

Gated Recurrent Unit (GRU) [75] is one type of recurrent neural networks that shows excellent ability in sequence modelling. GRU is used in our framework to model the temporal dynamics of pedestrian counts. [75] showed that GRU can effectively capture long-term dependencies of time series, and it is less vulnerable to both the vanishing gradient problem and the exploding gradient problem when conducting back-propagation. Compared with other RNN structures, GRU contains fewer parameters and has a simpler architecture, yet GRU can achieve satisfying results in most cases. In addition, GRU is easier to train compared to LSTM, which can significantly improve the training efficiency of the overall model by facilitating convergence and parameter updates. Therefore, considering the computing power and time cost, using GRU in sequence modelling problems is often preferable.

The structure of the input and output in each timestamp of the GRU is the same as that of the vanilla RNN [76]. At each timestamp, there is an input denoted as $\boldsymbol{X}_t$, and the hidden state denoted as $\boldsymbol{H}_{t-1}$, which is passed from the previous state of $t-1$. The previous hidden state $\boldsymbol{H}_{t-1}$ contains relevant information from the previous time. Using the input $\boldsymbol{X}_t$ and the previous hidden state $\boldsymbol{H}_{t-1}$, the current hidden state $\boldsymbol{H}_t$ is computed by the GRU cell and then passed forward. Additionally, an output denoted as $\boldsymbol{Y}_t$ for the current timestamp is optionally produced by the GRU. A part of this recursive process is depicted in Figure 1.

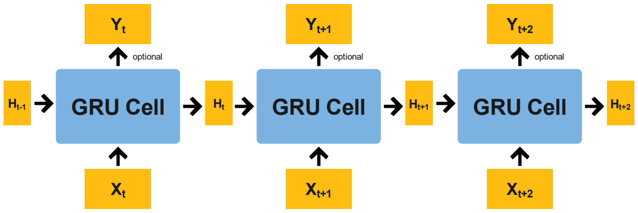


Fig. 1. The input and output of GRU cells

The inner structure of one GRU cell can be summarized as follows. There are two gates in each GRU cell: reset and update. The t-th states of the two gates are computed according to the following formulas:

$$\begin{aligned} \boldsymbol{r}_t &= \sigma\left(\boldsymbol{W}_{\boldsymbol{r}}[\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_{\boldsymbol{r}}\right) \\ \boldsymbol{z}_t &= \sigma\left(\boldsymbol{W}_{\boldsymbol{z}}[\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_{\boldsymbol{z}}\right) \end{aligned} \tag{1}$$

where $\boldsymbol{r}_t$ denotes the state of the reset gate and $\boldsymbol{z}_t$ denotes the state of the update gate. Parameters $\boldsymbol{W}$ and $\boldsymbol{b}$ with different subscripts are learnable weights and biases, respectively of the two gates. Operator $[\cdot, \cdot]$ denotes the concatenation of matrices. For example, $[\boldsymbol{X}_t, \boldsymbol{H}_{t-1}]$ means a new matrix containing both $\boldsymbol{X}_t$ and $\boldsymbol{H}_{t-1}$. The function of the two gates has been widely explored and variants of the GRU with different gating mechanisms have been designed in the literature [77], [78].

The gating mechanism is used to control the information flow. The reset gate is responsible for the short-term memory of the network, determining how much past information to forget. It is used to calculate the current candidate activation $\boldsymbol{c}_t$:

$$\boldsymbol{c}_t = \tanh\left(\boldsymbol{W}_{\boldsymbol{c}}[\boldsymbol{X}_t, (\boldsymbol{r}_t \odot \boldsymbol{H}_{t-1})] + \boldsymbol{b}_{\boldsymbol{c}}\right), \tag{2}$$

where $\odot$ is the element-wise multiplication and $\boldsymbol{W}_{\boldsymbol{c}}$ and $\boldsymbol{b}_{\boldsymbol{c}}$ are again weights and biases respectively. As we can infer from Equation (2), the closer $\boldsymbol{r}_t$ is to 0, the less information of the previous state would be included in the current candidate activation. When the reset gate is off, i.e., $\boldsymbol{r}_t$ approximates 0, then nearly all the previous information is forgotten by $\boldsymbol{c}_t$ and it seems we begin to learn the pattern of the sequence since time t.

The update gate $\boldsymbol{z}_t$ is responsible for the long-term memory of the network. It controls how much information to carry forward from the previous hidden state $\boldsymbol{H}_{t-1}$ and also controls how much new information from the current candidate activation needs to be added to the current hidden state $\boldsymbol{H}_t$. The current hidden state is then computed by:

$$\boldsymbol{H}_t = \boldsymbol{z}_t \odot \boldsymbol{H}_{t-1} + (1 - \boldsymbol{z}_t) \odot \boldsymbol{c}_t. \tag{3}$$

Accordingly, the GRU cell finishes the entire calculation process of time t. We can conclude how GRU achieves the trade-off between memory forgetting and memory retaining via the gating mechanism. Generally, considering multiple GRU cells stacked as a GRU network, the reset gates of those cells focusing short-term dependencies are more active (if $\boldsymbol{r}_t$ is 1 and $\boldsymbol{z}_t$ is 0, this GRU cell is equivalent to a standard RNN cell, which can handle short-term dependencies), and the update gates of cells aim for capturing long-term dependencies and are relatively active ($\boldsymbol{z}_t$ is close to 1). Therefore, each GRU cell can adaptably capture dependencies of different time scales when it is trained to model the time series. The illustration of the inner structure of one GRU cell is depicted in Figure 2.

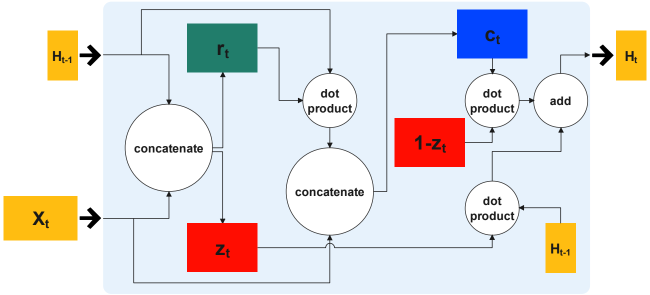


Fig. 2. The inner structure of one GRU cell

C. Diffusion Convolutional Layer

Modelling the spatial dependency of pedestrian volumes at different locations is challenging and has not been well investigated [79]. The diffusion convolution, as well as the corresponding diffusion convolutional layer, which utilizes bidirectional random walks on the graph, is an effective model to capture the spatial dependency on road networks [74]. Concretely, the sensor network is represented as a graph where the nodes represent the sensors and the edges represent their proximity. Formally, the sensor network graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{\mathcal{W}})$, where $\mathcal{V}$ is a set of nodes whose number is equal to the number of sensors $|\mathcal{V}| = N$, $\mathcal{E}$ is a set of edges and $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{N\times N}$ is a weighted adjacency matrix. Here, $\boldsymbol{\mathcal{W}}$ is calculated based on predefined road network distances and is typically not symmetric because of one-way streets and traffic control. The traffic flow data observed on this graph at time t is called the graph signal, which is denoted as $\boldsymbol{X}_t \in \mathbb{R}^{N\times P}$, where P is the number of features of each node. The diffusion convolution operation on this graph signal is then defined using a filter $f_{\boldsymbol{\theta}}$ that is parameterized by the parameter θ:

$$\boldsymbol{X}_t[:,p] \star_{\mathcal{G}} f_{\boldsymbol{\theta}} = \sum_{k=0}^{K-1}\left(\theta_{k,1}\left(\boldsymbol{D}_O^{-1}\boldsymbol{\mathcal{W}}\right)^k + \theta_{k,2}\left(\boldsymbol{D}_I^{-1}\boldsymbol{\mathcal{W}}^\top\right)^k\right)\boldsymbol{X}_t[:,p] \quad \text{for } p \in \{1, \cdots, P\}, \tag{4}$$

where $\boldsymbol{D}_O = \operatorname{diag}(\boldsymbol{\mathcal{W}}\mathbf{1})$ is the out-degree diagonal matrix and $\mathbf{1} \in \mathbb{R}^N$ denotes a vector of 1's. The matrix $\boldsymbol{D}_O^{-1}\boldsymbol{\mathcal{W}}$ in Equation (4) is a state transition matrix of the diffusion process, where $\theta_{k,1}$ is the corresponding parameter of $f_{\boldsymbol{\theta}}$ to be learned, and $\boldsymbol{D}_O^{-1}$ can be regarded as a normalizer for weighted adjacency matrix $\boldsymbol{\mathcal{W}}$. Similarly, $\boldsymbol{D}_I = \operatorname{diag}(\boldsymbol{\mathcal{W}}^T\mathbf{1})$ is the in-degree diagonal matrix and $\theta_{k,2}$ is the parameter related to reversed diffusion process. When $\boldsymbol{\mathcal{W}}$ is symmetric,

$\boldsymbol{D}_O$ and $\boldsymbol{D}_I$ become the same and the terms in Equation (4) can be optionally merged. Constant K is the maximum diffusion step. The essence of $\left(\boldsymbol{D}_O^{-1}\boldsymbol{\mathcal{W}}\right)^k \boldsymbol{X}_t[:,p]$ is, after the k-step diffusion, the information contained in related nodes is aggregated to each node according to relevance, and the relevance is quantified by the probability of transition between nodes.

The diffusion convolutional layer employs the diffusion convolution operation defined in Equation (4) to conduct further transformation to the original P-dimensional features. It maps the input $\boldsymbol{X}_t \in \mathbb{R}^{N\times P}$ to the output $\boldsymbol{O}_t \in \mathbb{R}^{N\times Q}$ by using filters $\{f_{\boldsymbol{\Theta}_{q,p,:,:}}\}$ whose parameters are $\boldsymbol{\Theta}_{q,p,:,:}$, and an activation function $\boldsymbol{a}$ such as ReLU [80] and Sigmoid [81]:

$$\boldsymbol{O}_t[:,q] = \boldsymbol{a}\left(\sum_{p=1}^{P} \boldsymbol{X}_t[:,p] \star_{\mathcal{G}} f_{\boldsymbol{\Theta}_{q,p,i,:}}\right) \qquad \text{for } q \in \{1,\cdots,Q\} \tag{5}$$

The diffusion convolutional layer has demonstrated effectiveness in feature extraction for graph data and is a useful mechanism for capturing spatial dependencies. In the following section, we use $\boldsymbol{\Theta} \in \mathbb{R}^{Q\times P\times K\times 2}$ to denote the collection of all parameters in the diffusion convolutional layer.

In this study, we adopt the diffusion convolutional layer to model the spatial dependency of pedestrian counts at various locations. Hence, the number of features is $P = 1$. For road traffic networks, distances between sensors are relatively well-defined as vehicles travel on roads and follow traffic rules and so is the adjacency matrix $\boldsymbol{\mathcal{W}}$. In contrast, pedestrians could travel freely in any direction and their flow is less controllable, which makes it difficult to define distances and calculate the adjacency matrix. The following subsection will explain how $\boldsymbol{\mathcal{W}}$ is computed for pedestrian sensor networks.

D. Dynamic Time Warping

The adjacency matrix $\boldsymbol{\mathcal{W}}$ has a great impact on the performance of graph neural networks, hence recently, it has received further attention in graph theory research [82], [83], [84]. Geographic information, such as latitude and longitude, is commonly used to construct the adjacency matrix. However, in the field of transportation (both vehicular and pedestrian), the geographic information alone may not be sufficient to capture the relationships between different sensors [70], [85]. An example from the data source utilised in this study (see Figure 3) shows the pedestrian volumes at three locations involving Sensor 13 (Spencer St-Collins St (North)), Sensor 19 (Southbank), and Sensor 20 (Queen St (West)) on a representative day. Despite Sensor 13 being geographically closer to Sensor 20, the time series of pedestrian volumes at Sensor 13 has a pattern more similar to that of Sensor 19. This study proposes to incorporate the distance of two time series, besides the distance based on geographical information. That is, we define the weighted adjacency matrix to be

$$\boldsymbol{\mathcal{W}} = \boldsymbol{\mathcal{W}}_{geo} + \beta \boldsymbol{\mathcal{W}}_{ts}, \tag{6}$$

where $\boldsymbol{\mathcal{W}}_{geo}$ is calculated based on geographical information of sensors, $\boldsymbol{\mathcal{W}}_{ts}$ is calculated based on time series using DTW, and β is a hyper-parameter that controls how much $\boldsymbol{\mathcal{W}}_{ts}$ contributes to the final weighted adjacency matrix.

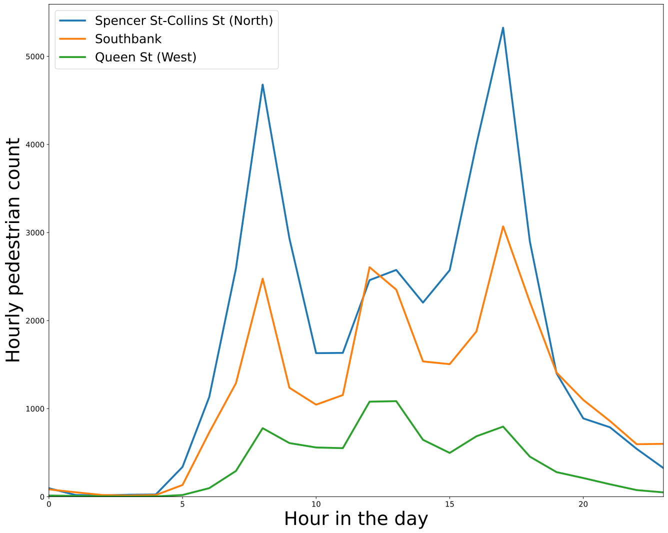


Fig. 3. Pedestrian counts collected by three sensors in one day

DTW is a well-established approach to computing the distance between time series, and it can effectively deal with scaling and translation on the time axis. Intuitively speaking, for a point in a time series, DTW can match the points around the corresponding position of the point in another time series. In this way, the time axis can be viewed as stretchable, rather than one-to-one like the traditional Euclidean distance. A formal definition of the DTW distance is given as follows.

Let $X = \{x_i\}_{i=1}^N$ be a time series of length N and $Y = \{y_j\}_{j=1}^M$ be another time series of length of M, where N is not necessarily equal to M. We denote that the local distance between the i-th element x_i in X and the j-th element y_j in Y is $c(x_i, y_j)$. Here, c is a metric defined in the space $\mathcal{F}$ where x_i and y_j locate (assumed to be the same), $c : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$. Define the cost matrix $C \in \mathbb{R}^{N\times M}$ as $C(i,j) = c(x_i, y_j)$. Additionally, the warping path can be expressed as a sequence of pairs $(p_1, \ldots, p_L)$, where each $p_l = (i_l, j_l)$ belongs to the Cartesian product of the intervals $[1 : N]$ and $[1 : M]$, and L represents the number of steps, which is not predetermined. The sequence p must fulfil three conditions: (1) the boundary condition requires $p_1 = (1,1)$ and $p_L = (N, M)$, (2) the monotonicity condition demands that $i_1 \leq i_2 \leq \ldots \leq i_L$, while $j_1 \leq j_2 \leq \ldots \leq j_L$, and (3) the step size condition requires that $p_{l+1} - p_l$, i.e. the difference between two consecutive pairs of elements in p can only be one of the following three tuples: $(1,0)$, $(0,1)$, or $(1,1)$, for l in the range $[1 : L-1]$. Then, the total cost of a warping path p between time series X and Y is given by the sum of the costs of each point in the path, calculated as $c_p(X,Y) = \sum_{l=1}^{L} c(x_{i_l}, y_{j_l})$. Finally, the DTW distance between X and Y is defined as the minimum of the total costs of all possible warping paths, which can be expressed as:

$$\mathrm{DTW}(X,Y) = \min\{c_p(X,Y) \mid p \text{ is a warping path}\}. \tag{7}$$

To illustrate the calculation of DTW, we consider two pedestrian count time series, both with a length of 7, involving two sensors at Melbourne Central and Princes Bridge. We choose $c(x_i, y_j) = |x_i - y_j|$. We draw the corresponding cost matrix and optimal warping path in Figure 4.

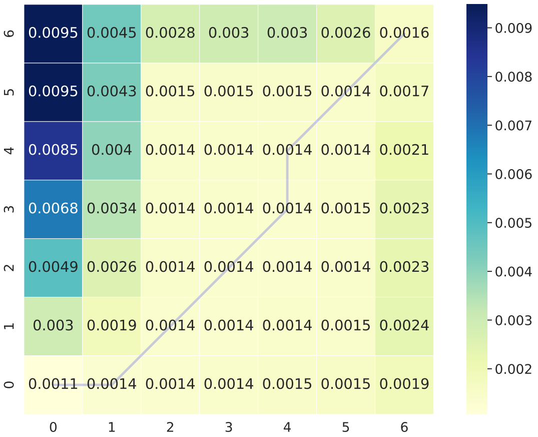


Fig. 4. Cost matrix and warping path

Figure 4 shows that the optimal warping path is: $[(0,0),(1,0),(2,1),(3,2),(4,3),(4,4),(5,5),(6,6)]$. The corresponding match between the points of the two time series is shown by the dashed line in Figure 5. If we choose the distance induced by the L_p norm to measure the dissimilarity between the two time series, points with the same time index are strictly paired together, that is, all the dashed lines are supposed to be vertical.

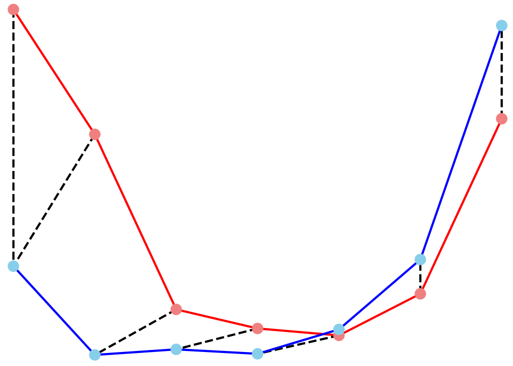


Fig. 5. The match of points between the two time series

E. Sequence to Sequence Framework

Sequence to sequence (Seq2Seq) framework is an end-to-end deep learning framework [86], and it can effectively tackle the multi-step to multi-step sequence learning problems. Seq2Seq consists of an encoder and a decoder. In most cases, both the encoder and decoder are RNNs. As illustrated in Figure 6, the encoder encodes the input sequence into a latent vector, which is referred to as *context*, and then the context is decoded by the decoder to yield outputs. During the decoding process, the decoder recurrently leverages the true label (during training) or output of the last timestamp (during inference) as the input of the current timestamp, and performs the decoding operation to yield the new hidden state and output until the stop symbol is the output.

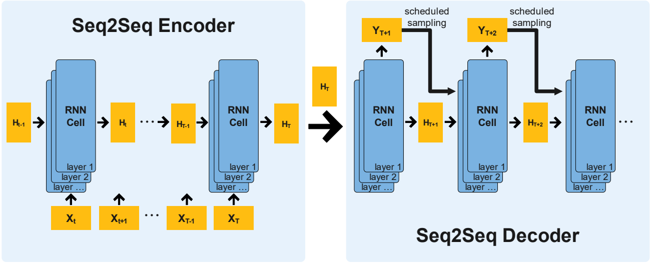


Fig. 6. The Seq2Seq framework

Compared with simple RNNs, including those with a gated mechanism, the Seq2Seq framework has the following advantages. First, due to more efficient architecture, Seq2Seq performs better in practical applications, especially in long-range time series forecasting and machine translation [87]. This can partly be attributed to the way Seq2Seq produces predictions. When decoding, i.e., making predictions, Seq2Seq takes into account the dependency between outputs, which cannot be done by the conventional RNNs. The RNNs solve multi-step prediction problems by outputting vectors at once, which means dependency between outputs is not considered. Second, the Seq2Seq framework is very flexible and can deal with variable-length input and output. Moreover, a series of off-the-shelf effective methods and techniques in the field of deep learning can be integrated into the Seq2Seq framework to further improve the performance of the model, such as the attention mechanism [88] and scheduled sampling [89].

F. Diffusion Convolutional Gated Recurrent Unit Model

Leveraging on the aforementioned model components and techniques, we now explain the overall architecture of the diffusion convolutional gated recurrent unit (DCGRU) model for the pedestrian volume forecasting problem. See Figure 7. The DCGRU also benefits from the Seq2Seq framework, and each layer in both the encoder and decoder is a DCGRU cell, whose inner structure is described as:

$$\boldsymbol{r}_t = \sigma\left(\boldsymbol{\Theta}_r \star_{\mathcal{G}} [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_r\right) \tag{8}$$

$$\boldsymbol{z}_t = \sigma\left(\boldsymbol{\Theta}_z \star_{\mathcal{G}} [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_u\right) \tag{9}$$

$$\boldsymbol{c}_t = \tanh\left(\boldsymbol{\Theta}_c \star_{\mathcal{G}} [\boldsymbol{X}_t, (\boldsymbol{r}_t \odot \boldsymbol{H}_{t-1})] + \boldsymbol{b}_c\right) \tag{10}$$

$$\boldsymbol{H}_t = \boldsymbol{z}_t \odot \boldsymbol{H}_{t-1} + (1 - \boldsymbol{z}_t) \odot \boldsymbol{c}_t \tag{11}$$

where $\boldsymbol{X}_t$ are the input and $\boldsymbol{H}_{t-1}$ is the hidden state of timestamp t. $\Theta_r \star_{\mathcal{G}}$, $\Theta_z \star_{\mathcal{G}}$ and $\Theta_c \star_{\mathcal{G}}$ are diffusion convolution operations for the corresponding filters. By Equation (8) - (11), the DCGRU cell substitutes the linear transformations in the original GRU with diffusion convolutions, and thus achieves the spatiotemporal feature extraction. Moreover, several DCGRU cells can be stacked in the encoder and the decoder of Seq2Seq to obtain better feature extraction results.

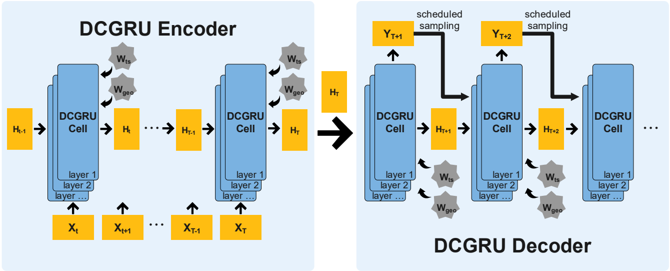


Fig. 7. The overall framework of DCGRU

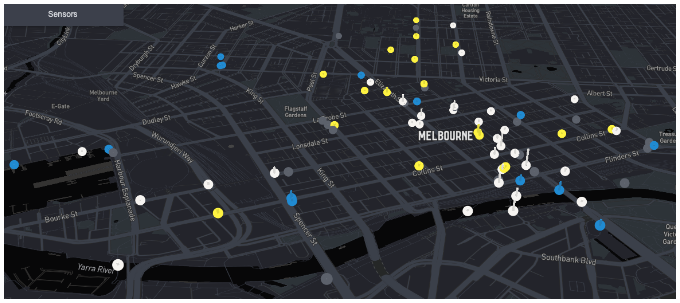


Fig. 8. The sensor map of the pedestrian counting system in Melbourne

As suggested by the original DCRNN, scheduled sampling [89] is used to mitigate the performance degradation issues during model testing, which is caused by the discrepancy between training and inference. Specifically, during the inference phase of standard Seq2Seq models, the output of each timestamp has an association with the outputs of previous timestamps. If a wrong output is generated, the input state of all the following timestamps will be affected, and the error will continue to accumulate. Scheduled sampling alleviates this problem by no longer completely using true labels as inputs at each timestamp during training. Instead, the true label is fed into the decoder with a probability ϵ_i at i-th iteration, and the output of the model itself is chosen to be fed into the decoder with probability $1-\epsilon_i$. ϵ_i is typically large at the beginning because the model is not sufficiently trained. As the training process goes on, the model is better trained, and then ϵ_i should gradually decrease to 0. In this way, the model should be more effective in tackling the cumulative error problem and performing more consistently, as it is exposed to a similar scenario during both the training and inference phases.

IV. Numerical Experiments

A. Dataset and Data Preprocessing

We obtained the pedestrian data from the open data website of the City of Melbourne [1]. To determine variations in pedestrian activity throughout the day, the City of Melbourne has built a pedestrian counting system, which is composed of pedestrian counting sensors installed across the central city area of Melbourne. The dataset records the hourly counts of pedestrians passing by various sensors in the City of Melbourne. Locations of sensors can be obtained via the Sensor Locations website [2]. See Figure 8. In order to avoid the impact of the COVID-19 pandemic and the consequential lockdowns, we selected the data from April 1 to December 31 of 2019. There are missing values in the dataset due to various reasons, such as sensor malfunction. We selected 30 sensors with the least number of missing values for further analysis. For the missing values in the data of these 30 sensors, we chose the average of the non-missing values of the same sensor at the same time to fill in. This resulted in a dataset containing $6,600$ rows and 30 columns.

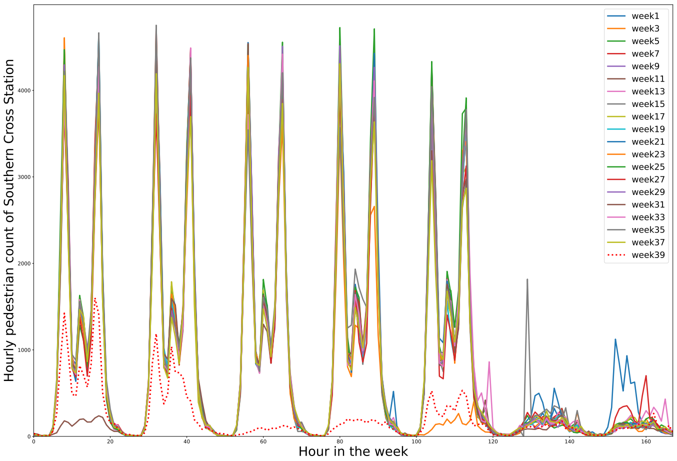


Fig. 9. Hourly pedestrian counts at Southern Cross Station in different weeks. The data for week 39 (from December 23 to December 29) marked as a dotted line, shows a different pattern compared with the other weeks.

Besides missing data, our exploratory analysis also identified abnormal data. We found that the data of certain weeks present very different characteristics from others; an example is shown in Figure 9. This discrepancy may be due to factors such as the Christmas holidays or extreme weather. To unveil the general spatiotemporal patterns of pedestrian volumes, we applied an anomaly detection algorithm based on k-medoids clustering to eliminate the influence of data in abnormal weeks. Compared with other clustering algorithms such as k-means, the k-medoids method is less sensitive to outliers due to the fact that k-medoids only allow existing data points to be cluster centres, which is beneficial for anomaly detection [90]. For example, when an outlier is assigned to a cluster, it would affect the mean value of its cluster in the k-means algorithm, resulting in a large deviation between the mean value and most of the data in the cluster. This may cause the true abnormal data to be mixed with normal data. In contrast, the k-medoids algorithm can reduce this adverse effect. The procedure of the k-medoids algorithm adopted for the multivariate time series data is summarized in Algorithm 1.

Note that each data point $\mathcal{D}_j = [\boldsymbol{X}_{1,j}, \ldots, \boldsymbol{X}_{N,j}]$ in Algorithm 1 is a 168×30 matrix, where $\boldsymbol{X}_{i,j}$ denotes the time series of the i-th sensor for the j-th week. Before calculating the distance, we first re-scaled the dataset for each sensor, that

[1] http://www.pedestrian.melbourne.vic.gov.au/

[2] https://data.melbourne.vic.gov.au/explore/dataset/pedestrian-counting-system-sensor-locations/information/

Algorithm 1 k-medoids for multivariate time series data

```
1:  Input The data $\mathcal{D}_j$ of week $j, j = 1, 2, \ldots, N$, the number of clusters $k$, the distance function $a$, and the maximum iteration number $P$
2:      Compute the distance matrix $A = (a_{jj'})_{N \times N}$ between the data of each week
3:      Randomly select $k$ points as medoids
4:      Set iteration counter $p = 1$
5:      Calculate the distance from each data point to the medoids
6:      Assign each data point to its nearest medoids and then $k$ clusters are formed
7:      for each cluster in $k$ clusters
8:          Choose one data point as the target and compute the sum of distances of all data points to the target until each data point in this cluster has been selected as the target.
9:          Select the data point that minimizes the above sum of distances as the new medoids
10:     end for
11:     while new medoids are not completely the same as the former medoids and $p \leq P$
12:         Repeat steps 5 - 10
13:     end while
14: Return New medoids and nearest medoid of $\mathcal{D}_j$, $j = 1, 2, \ldots, N$
```

is, for t in the j-th week,

$$X_{i,t} = \frac{X_{i,t} - \min \boldsymbol{X}_{i,j}}{\max \boldsymbol{X}_{i,j} - \min \boldsymbol{X}_{i,j}}. \tag{12}$$

The distance between $\mathcal{D}_j$ and $\mathcal{D}_{j'}$ is

$$a_{jj'} = \sum_{i=1}^{N} \mathrm{DTW}(\boldsymbol{X}_{i,j}, \boldsymbol{X}_{i,j'}), \tag{13}$$

which is the sum of DTW distances of each sensor between the j-th week and the j'-th week.

We determine the number of clusters, k, based on the Silhouette score [91]. The Silhouette score is a measure of how well each data point fits within its assigned cluster and can be used to guide the number of clusters in a clustering algorithm. It measures both the compactness of a cluster and the separation between different clusters. A higher Silhouette score indicates that the data points are well-clustered and have a clear separation between the clusters.

The Silhouette score for a single data point i, which belongs to cluster I, is calculated as follows:

1) Calculate the average distance between data point i and all other data points in the same cluster. Denote this value as a_i:

$$a_i = \frac{1}{|C_I| - 1} \sum_{j \in C_I, j \neq i} a_{ij}. \tag{14}$$

where $|C_I|$ is the number of data points in cluster I, the same cluster as i, and a_{ij} is the distance between data points i and j. If the single data point i itself is a cluster, i.e. $|C_I| = 1$, then the Silhouette score of i is 0.

2) Calculate the average distance between i and all other data points in the nearest neighbouring cluster. Denote this value as b_i:

$$b_i = \min_{K \neq I} \frac{1}{|C_K|} \sum_{j \in C_K} a_{ij}. \tag{15}$$

3) Calculate the Silhouette score for data point i as:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}. \tag{16}$$

The Silhouette score for the clustering algorithm is then calculated as the average of the Silhouette scores for all N data points:

$$\text{Silhouette score} = \frac{1}{N} \sum_{i=1}^{N} s_i. \tag{17}$$

The Silhouette score ranges from -1 to 1. A score close to 1 indicates that the data points are well-clustered, while a score close to -1 indicates that they are poorly clustered and may have been assigned to the wrong clusters. A score close to 0 indicates that the data points may belong to multiple clusters. Therefore, supported with this information, k with the largest Silhouette score is regarded as the optimal number of clusters. In practice, we run the k-medoids algorithm for $1,000$ times, using different random seeds for the initialization of medoids. The optimal number of clusters we found is $k = 2$.

We chose the clustering result with the highest frequency as the final clustering setting, based on which data from abnormal weeks are identified and deleted. We identified the outliers using Tukey's method [92]:

1) Calculate distances between the data of each week and its nearest centre;
2) Obtain the lower quartile Q_1 and upper quartile Q_3 of the calculated distances for each center;
3) Calculate the upper critical value $Q_3 + q(Q_3 - Q_1)$ and lower critical value $Q_1 - q(Q_3 - Q_1)$, where q represents the hyperparameter of the method.

In Tukey's method, q is typically chosen to be 1.5 or 3, and a data point is regarded as an outlier if its distance to the nearest centre is either greater than the upper critical value or less than the lower critical value. Here, we chose $q = 1.5$ and deleted the data point whose distance to the nearest centre is greater than the upper critical value. After the pre-processing stage, the dataset of Melbourne's pedestrian volume with $5,712$ hours of records was obtained.

B. Numerical Experiment Setup

In this section, we describe the implementation details of the DCGRU model and experiment settings. We use the first 70% of the data in chronological order as the training set, the next 10% as the validation set, and the last 20% of the data as the test set. Similar to most time series supervised learning tasks, the sliding window method is used to construct the input data of the DCGRU model.

As shown in Figure 10, let L_{input} denote the input length of our window and L_{output} denote the output length. Then, for the current timestamp t, the input multivariate time series data

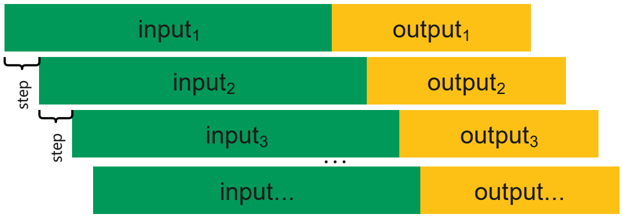


Fig. 10. Input data and forecasting target of DCGRU

of the DCGRU model is $\left[\boldsymbol{X}_{t-L_{input}+1}, \cdots, \boldsymbol{X}_t\right]$ and the expected output is $\left[\boldsymbol{X}_{t+1}, \cdots, \boldsymbol{X}_{t+L_{output}}\right]$. In the experiment, we keep $L_{output} = 5$ and the step of sliding window be 1. The input length L_{input} is set to 5, 24 (one day), and 168 (one week), and the performance of the three input lengths is evaluated.

For the adjacency matrix, $\mathcal{W}_{geo}$ is constructed by using sensors' longitude and latitude coordinates. More specifically, the geographical adjacency matrix is calculated via a thresholded Gaussian kernel:

$$\mathcal{W}_{geo}(i, i') = \begin{cases} \exp\left(-\frac{d(\boldsymbol{s}_i, \boldsymbol{s}_{i'})^2}{\sigma_g^2}\right), & \text{if } d\left(\boldsymbol{s}_i, \boldsymbol{s}_{i'}\right) \leq \kappa, \\ 0, & \text{otherwise}, \end{cases} \tag{18}$$

where κ is a threshold, σ_g is the standard deviation of all the geographical distances, $\boldsymbol{s}_i$ and $\boldsymbol{s}_{i'}$ denote the coordinates of the sensors i and i', respectively. The thresholded Gaussian kernel can make the adjacency matrix sparse, which benefits the computation of the diffusion convolution operation. The thresholded Gaussian kernel is also used to compute $\mathcal{W}_{ts}(i, i')$. It is worth mentioning that, we select a representative period to calculate DTW distances between sensors to reduce the computation complexity. That is,

$$\mathcal{W}_{ts}(i, i') = \begin{cases} \exp\left(-\frac{\text{DTW}(\boldsymbol{c}_i, \boldsymbol{c}_{i'})^2}{\sigma_t^2}\right), & \text{if DTW}(\boldsymbol{c}_i, \boldsymbol{c}_{i'}) \leq \kappa, \\ 0, & \text{otherwise}, \end{cases} \tag{19}$$

where $\boldsymbol{c}_i$ is the i-th column of the cluster centre derived in Section IV-A, which represents the pattern of typical weekly pedestrian counts of the i-th sensor, σ_t is the standard deviation of all the time series distances. In our experiment, the sample standard deviation is used for σ_g and σ_t, and the threshold κ is set to be 0.1.

Three evaluation metrics are used to quantify the performance of different models:

- Mean Absolute Error (MAE)

$$\text{MAE}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \frac{1}{NL} \sum_{t=t_0}^{t_0+L} \sum_{i=1}^{N} \left| \hat{X}_{i,t} - X_{i,t} \right|$$

- Mean Absolute Percentage Error (MAPE)

$$\text{MAPE}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \frac{1}{NL} \sum_{t=t_0}^{t_0+L} \sum_{i=1}^{N} \left| \frac{\hat{X}_{i,t} - X_{i,t}}{X_{i,t}} \right| \times 100\%$$

- Root Mean Square Error (RMSE)

$$\text{RMSE}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \sqrt{\frac{1}{NL} \sum_{t=t_0}^{t_0+L} \sum_{i=1}^{N} \left(\hat{X}_{i,t} - X_{i,t} \right)^2}$$

where N is the number of sensors, L is the length of the data to be tested, t_0 is the start timestamp of the test data, and $X_{i,t}$ and $\hat{X}_{i,t}$ denote the true and the predicted pedestrian volumes observed by sensor i at time t, respectively.

For choosing the hyper-parameters, we used the grid search method and repeated the experiment 3 times for each set of parameters. The range of weight β is $[0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5]$; The maximum diffusion step K is [1, 2, 3]. The range of the learning rate in the Adam optimizer is $[0.001, 0.005, 0.01]$; The batch size is either 32 or 64; The number of layers of the DCGRU cell is 1 or 2. For each set of parameter combinations, the best-performing model on the validation set is further used to make predictions on the test data. In addition, the model iterates for 50 epochs in each experiment, and the model parameters corresponding to the epoch with the smallest loss on the validation set are reloaded when conducting spatiotemporal forecasting on future data.

C. Results and Analysis

We conducted a comparative experiment on the Melbourne pedestrian count dataset using five methods.

- *Historical average*: The historical average method sets the predicted value in the test set to be the average of the timestamps of the same day in the week from the training set.
- *VAR*: the classic statistical vector autoregressive model.
- *GRU*: the original GRU model, which does not incorporate the spatial correlation.
- *DCGRU*: DCGRU represents the classic DCGRU model that only relies on geographic information when building the sensor graph.
- *DCGRU-DTW*: The proposed DCGRU-DTW model incorporates both geographic information and time series.

All three neural networks, GRU, DCGRU, and DCGRU-DTW, are trained under the sequence-to-sequence framework introduced in Section III-E. The averaged experiment results are summarized in Tables I–III with input lengths of 5, 24, and 168 respectively. Different columns (1h, 2h, 3h, 4h, 5h) represents the number of hours in advance for making predictions.

Table I compares the performance of the five methods with the input length $L_{input} = 5$, that is, using the past 5 hours' information for prediction. First, we consider the three deep neural network methods of GRU, DCGRU and DCGRU-DTW. Compared with the classic DCGRU model, the average MAPE of DCGRU-DTW is around 1.3% lower than that of DCGRU when predicting one hour in advance. Moreover, We notice that the advantage of DCGRU-DTW increases as the forecasting range extends. For example, when DCGRU-DTW makes 3 hours to 5 hours ahead prediction, the average MAPE of DCGRU-DTW is 3% to 4% lower than that of DCGRU. The RMSE and MAE achieved by DCGRU-DTW are also smaller

TABLE I
PERFORMANCE COMPARISON OF DIFFERENT MODELS FOR THE VARIOUS PREDICTION INTERVALS (1 TO 5 HOURS) WITH $L_{input} = 5$

Model	**Metric**	1 h	2 h	3 h	4 h	5 h
Historical Average	MAE	122.215	122.215	122.215	122.215	122.215
	MAPE	31.30%	31.30%	31.30%	31.30%	31.30%
	RMSE	279.823	279.823	279.823	279.823	279.823
VAR	MAE	82.877	104.246	113.448	118.342	121.832
	MAPE	31.58%	43.25%	52.68%	63.02%	70.63%
	RMSE	207.392	244.629	255.542	261.815	267.395
GRU	MAE	112.304	173.680	225.612	264.483	294.915
	MAPE	39.47%	71.49%	106.64%	160.94%	224.02%
	RMSE	237.811	334.622	429.453	507.356	554.252
DCGRU	MAE	80.534	109.435	131.126	147.224	157.427
	MAPE	27.39%	35.23%	41.42%	45.50%	46.91%
	RMSE	191.606	237.132	277.204	310.549	324.449
DCGRU-DTW	MAE	78.486	104.766	125.336	141.557	151.363
	MAPE	26.06%	32.77%	37.02%	41.15%	43.33%
	RMSE	190.672	234.447	271.101	306.298	319.305

than DCGRU when making 3 hours to 5 hours ahead forecasting: the MAE of DCGRU-DTW is around 6 units smaller, and the RMSE is around 4 units smaller, compared to DCGRU. In addition, it can be concluded that the classic DCGRU shows significantly better predictive capability than GRU. When making predictions 1 hour to 2 hours ahead, DCGRU reduces nearly half of the MAPE produced by GRU, and DCGRU's MAE is about 32 units smaller than GRU. DCGRU's RMSEs are also significantly lower. When predicting 3 hours to 5 hours in advance, the advantages of DCGRU and DCGRU-DTW over GRU increase. This could be associated with the fact that DCGRU's diffusion convolution operation can effectively capture the spatial dependency between different sensors, and utilizing this correlation would yield more accurate prediction results.

The performance of VAR is not competitive in this experiment, indicating that traditional statistical models have difficulty in capturing complex spatiotemporal correlations of data compared to deep neural networks, and thus, they are unable to yield accurate predictions. On the other hand, surprisingly, the simple historical average achieves better results than the other methods for predicting 2 hours or longer. This seems to be a one-sided comparison as the historical average makes use of all previous information in the training set. Nevertheless, the result suggests that if we take advantage of the daily and weekly periodicity of the data, the prediction accuracy may be improved. We verify this in the following experiment.

We increase L_{input} to 24, which means incorporating daily patterns into three deep neural network methods, and we present the results by the three deep neural network methods in Table II. We note that the result produced by the historical average does not depend on L_{input}. For the VAR method, the best order is chosen by cross-validation, and the results are presented in Table I. Hence, the results of the historical average and VAR are not included in Table II. Compared with $L_{input} = 5$, the performance of GRU, DCGRU and DCGRU-DTW in each forecast range has witnessed significant improvement, especially when forecasting longer hours ahead. In particular, DCGRU-DTW shows a drop by approximately 8% in MAPE when forecasting 4 or 5 hours ahead. It appears that to forecast farther ranges, methods with $L_{input} = 24$ outperform the ones with $L_{input} = 5$. A possible reason for this is that pedestrian flow in the short term is more determined by the situation in the previous few hours. Thus, given the historical data of the last 5 hours, $L_{input} = 5$, models can effectively predict the pedestrian volume in the next 1 to 2 hours. But when the forecast range increases, insufficient useful information can be extracted from the observations in the last 5 hours. So leveraging the periodicity of the pedestrian volume, while increasing the input length can help to further improve the model performance. Similar to results in Table I, DCGRU-DTW also achieves the best results among all three neural network methods for all forecasting ranges. For example, when making the prediction 5 hours ahead, the MAPE by DCGRU-DTW is about 5% smaller than DCGRU, the MAE is about 13 units lower, and the RMSE is about 14 lower.

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT MODELS FOR THE NEXT 1 TO 5 HOURS WITH $L_{input} = 24$

Model	**Metric**	1 h	2 h	3 h	4 h	5 h
GRU	MAE	84.945	112.379	129.692	140.134	147.731
	MAPE	28.50%	37.94%	47.27%	54.87%	57.59%
	RMSE	196.913	241.335	273.455	294.747	306.290
DCGRU	MAE	85.013	107.315	119.998	130.157	136.940
	MAPE	27.18%	32.53%	35.13%	36.98%	40.80%
	RMSE	199.657	240.018	263.474	282.574	293.403
DCGRU-DTW	MAE	74.738	94.146	107.837	117.755	123.628
	MAPE	26.03%	29.24%	31.27%	33.51%	35.43%
	RMSE	182.632	220.303	249.720	270.128	279.355

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT MODELS FOR THE NEXT 1 TO 5 HOURS WITH $L_{input} = 168$

Model	**Metric**	1 h	2 h	3 h	4 h	5 h
GRU	MAE	79.288	103.376	118.899	129.827	136.224
	MAPE	27.91%	38.18%	47.97%	53.83%	56.22%
	RMSE	187.635	226.591	253.308	273.713	282.853
DCGRU	MAE	72.812	89.781	100.697	107.623	111.761
	MAPE	26.66%	29.77%	32.33%	33.70%	34.80%
	RMSE	184.072	216.248	240.139	253.995	259.775
DCGRU-DTW	MAE	72.232	87.635	97.162	103.256	106.853
	MAPE	25.81%	27.62%	28.72%	29.60%	31.30%
	RMSE	186.871	217.423	235.401	244.235	249.744

In Table III, we consider the longest input length, $L_{input} = 168$, which implies that the weekly pattern is included. Compared to the result for $L_{input} = 24$, the performance of all neural network methods is further improved. The performance of DCGRU-DTW is the best among the three methods. When forecasting 5 hours in advance and $L_{input} = 168$, the MAE of DCGRU-DTW is approximately 5 units smaller than DCGRU and 30 units smaller than GRU. Furthermore, the MAPE of DCGRU-DTW is 3.5% smaller than DCGRU and 25% smaller than GRU, while the RMSE of DCGRU-DTW is 10 units smaller than DCGRU and 33 smaller than GRU.

Next, we compare the proposed DCGRU-DTW method with the historical average method. For the datasets with the daily or weekly pattern, the historical average method is quite robust for longer-range forecasting [74], [93], [94]. For the sequence neural network models to work in real-world applications, it is important to include these patterns in the model building. Here, this is achieved by setting $L_{input} = 24$ for the daily pattern and $L_{input} = 168$ for the daily and weekly patterns. It can be seen that the proposed DCGRU-DTW method with $L_{input} = 24$ is best for forecasting up to 4 hours, while the proposed DCGRU-DTW method with $L_{input} = 168$ is best for forecasting up to 5 hours.

We also conduct a specific analysis of the prediction result of each sensor when DCGRU-DTW forecasts different hours ahead. For illustration, we randomly select 10 sensors and one-week data obtained from these sensors. Figure 11 compares the true values of hourly pedestrian counts (in blue) and the predicted values by DCGRU-DTW (in red). It is observed that DCGRU-DTW can generally provide satisfactory predictions of pedestrian volume. DCGRU-DTW can capture the complex nonlinear trends in pedestrian volumes effectively, and in most cases, it can produce accurate predictions when the pedestrian volume presents drastic and oscillating changes within a short period of time, such as the data of sensor 4 in week 5 and the data of sensor 13 in week 3 (see Figure 11).

On careful inspection, we find that inaccurate predictions mostly occur in two situations. The first instance is when the pedestrian volume observed at the sensor level demonstrates a clear, regular daily and/or weekly pattern and yet presents extremely abnormal behaviour (e.g. a steep rise or drop in volume) on a certain day. In that case, DCGRU-DTW is unable to effectively capture the abnormal variations in the number of pedestrians passing through that sensor, resulting in large prediction errors on that day. The mismatch between the predicted value and the ground truth of sensor 16 in week 3 is such a case, highlighted with green circles (Figure 11). Another situation is when pedestrian volume at the sensor does not have a regular pattern and instead varies significantly from week to week. Figure 12 illustrates an example for Sensor 29 located at Bourke St Bridge. It can be seen that the time series of some weeks in the test set presents distinct patterns compared with others, and this 'abnormal pattern' seldom occurs in the training data. As a result, DCGRU-DTW can not accurately capture the pedestrian volume trend. Table IV provides the MAPE results for all 30 sensors, in which the MAPE of 1-hour prediction for sensor 29 is 56.87%, the worst among all the sensors. This result shows a strong heterogeneity in the prediction results for different sensors. For sensor 11 and sensor 13, regular daily and weekly patterns are observed in the pedestrian flow and their MAPEs are low.

TABLE IV
MAPEs OF EACH SENSOR OF DIFFERENT HOURS AHEAD PREDICTION WITH $L_{input} = 168$

Id	**Sensor location**	1 h	2 h	3 h	4 h	5 h
1	Melbourne Central (MC)	10.61%	12.35%	13.45%	14.53%	15.56%
2	Town Hall (W)	11.39%	13.86%	15.19%	17.06%	18.41%
3	Princes Bridge	19.18%	24.04%	25.77%	27.44%	28.68%
4	Flinders St Station Underpass	11.37%	12.82%	14.22%	16.03%	16.22%
5	Southern Cross Station	45.77%	55.43%	51.43%	47.64%	50.72%
6	New Quay	49.24%	49.14%	50.54%	49.80%	50.40%
7	Collins Place (N)	32.20%	27.38%	31.42%	26.03%	28.44%
8	Chinatown-Swanston St (N)	24.87%	25.00%	26.74%	27.64%	26.60%
9	Chinatown-Lt Bourke St (S)	22.30%	23.26%	23.95%	25.31%	26.45%
10	Bourke St-Russell St (W)	13.41%	14.86%	15.80%	16.92%	18.20%
11	Flinders St-Elizabeth St (E)	9.45%	11.20%	12.27%	13.70%	15.31%
12	Spencer St-Collins St (S)	27.78%	27.99%	24.29%	25.72%	26.86%
13	Spencer St-Collins St (N)	13.75%	15.89%	17.02%	17.09%	18.39%
14	QV Market-Elizabeth St (W)	12.82%	14.57%	15.80%	16.50%	17.17%
15	The Arts Centre	23.44%	25.49%	26.26%	27.04%	26.87%
16	Lonsdale St (S)	41.98%	55.05%	65.44%	72.58%	77.00%
17	Lygon St (W)	25.10%	26.82%	27.46%	30.97%	32.97%
18	Flinders St-Spark La	44.01%	43.96%	43.11%	43.01%	43.45%
19	Southbank	18.89%	20.94%	22.57%	24.45%	27.07%
20	Queen St (W)	32.71%	32.97%	33.23%	37.26%	45.67%
21	Lygon St (E)	30.25%	31.67%	33.32%	32.26%	33.49%
22	Lonsdale St-Spring St (W)	30.15%	26.53%	29.09%	29.50%	32.75%
23	Grattan St-Swanston St (W)	39.11%	45.96%	47.79%	45.65%	46.15%
24	MC-Elizabeth St (E)	12.09%	13.62%	14.06%	14.37%	15.09%
25	QVM-Queen St (E)	24.38%	22.86%	23.36%	25.11%	27.00%
26	Faraday St-Lygon St (W)	30.95%	29.19%	33.06%	36.66%	41.74%
27	QVM-Franklin St (North)	25.56%	27.64%	26.09%	24.75%	26.02%
28	Elizabeth St-Lonsdale St (S)	21.01%	21.19%	23.58%	22.64%	23.09%
29	Bourke St Bridge	56.87%	61.26%	59.19%	63.23%	65.92%
30	Bourke St-Spencer St (N)	13.51%	15.73%	16.08%	17.17%	17.35%

V. CONCLUSION

This study presents a diffusion convolution gated recurrent unit model (called DCGRU-DTW) for predicting pedestrian volumes at a city-wide scale. We used the data from the City of Melbourne pedestrian counting sensors to set up the experiments and evaluate the performance of the DCGRU-DTW model. Specifically, our model considers the static geographic relationship among sensors, as well as, the dynamic similarity between pedestrian counts over time. Experiments conducted in this study show that the proposed model outperforms the classic DCGRU model, the VAR model and the historical average in terms of MAE, MAPE and RMSE. Furthermore, it is found that the prediction accuracy could be further improved by increasing the input length so that daily and weekly trends are captured. Our observation of applying the DCGRU model for the purpose of pedestrian flow prediction indicates that despite being a deep learning neural network, its overall architecture is not too complicated to be used for transport and mobility based problems. More specifically, DCGRU is comparatively lightweight and easy to train.

The proposed model comes with some limitations, which open new avenues for future research directions. First, the prediction accuracy could be impacted when unexpected extreme deviations from regular behaviour occur. From the anomaly detection perspective, nevertheless, such observation is considered as one. We also found that for real-time crowd or pedestrian management, further data granularity (e.g., counts per five minutes) is required. In the future, we suggest combining techniques such as continual learning [95] and distribution shift detection [96] with DCGRU-DTW to further improve the model that can adjust the prediction results in time when abnormal situations occur, thereby reducing the adverse and potentially long-term impact of unexpected extreme deviations on the prediction accuracy.

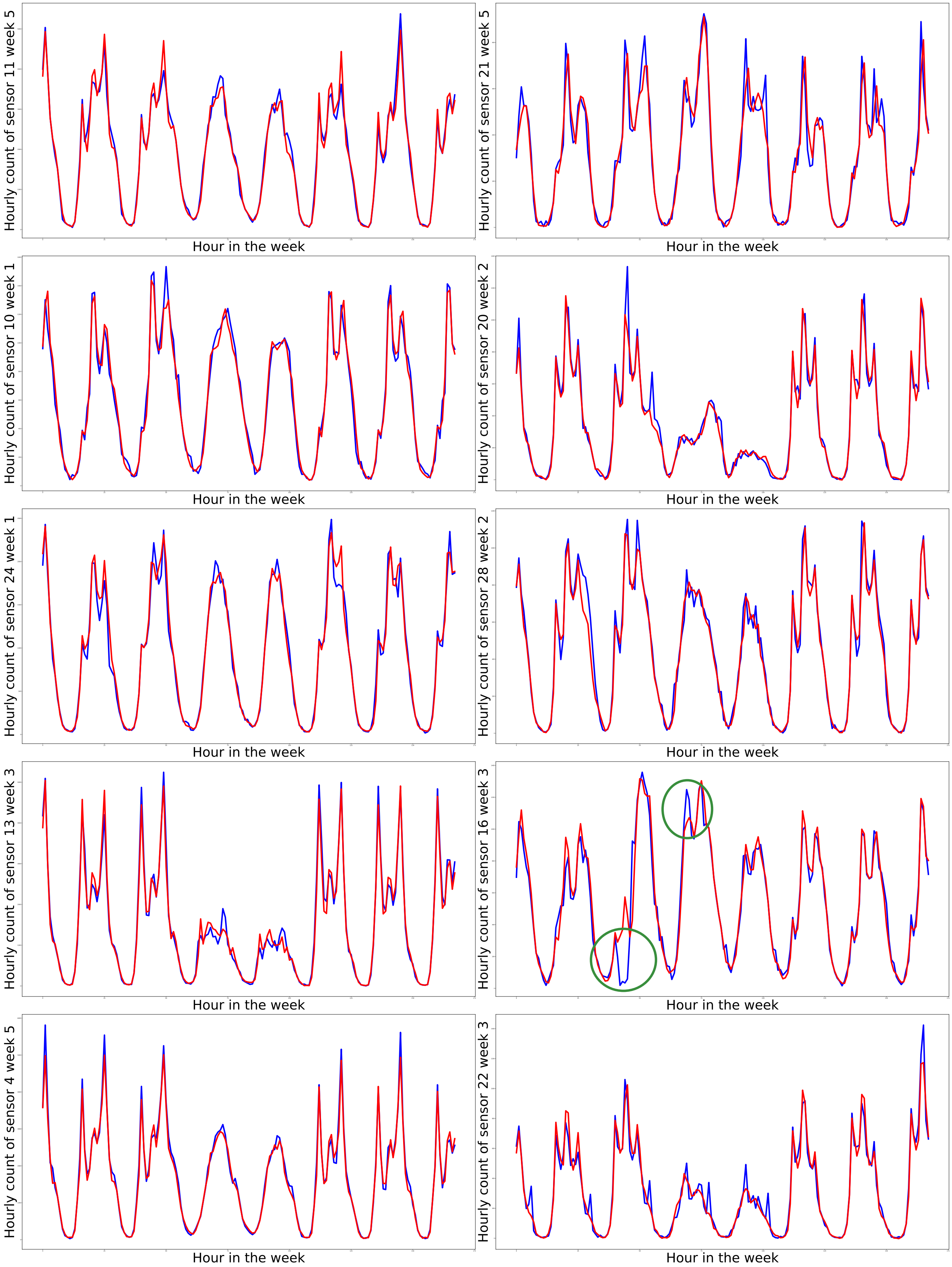


Fig. 11. Ground truths (in blue) and 1-hour-ahead prediction results (in red) of 10 selected sensors in different weeks

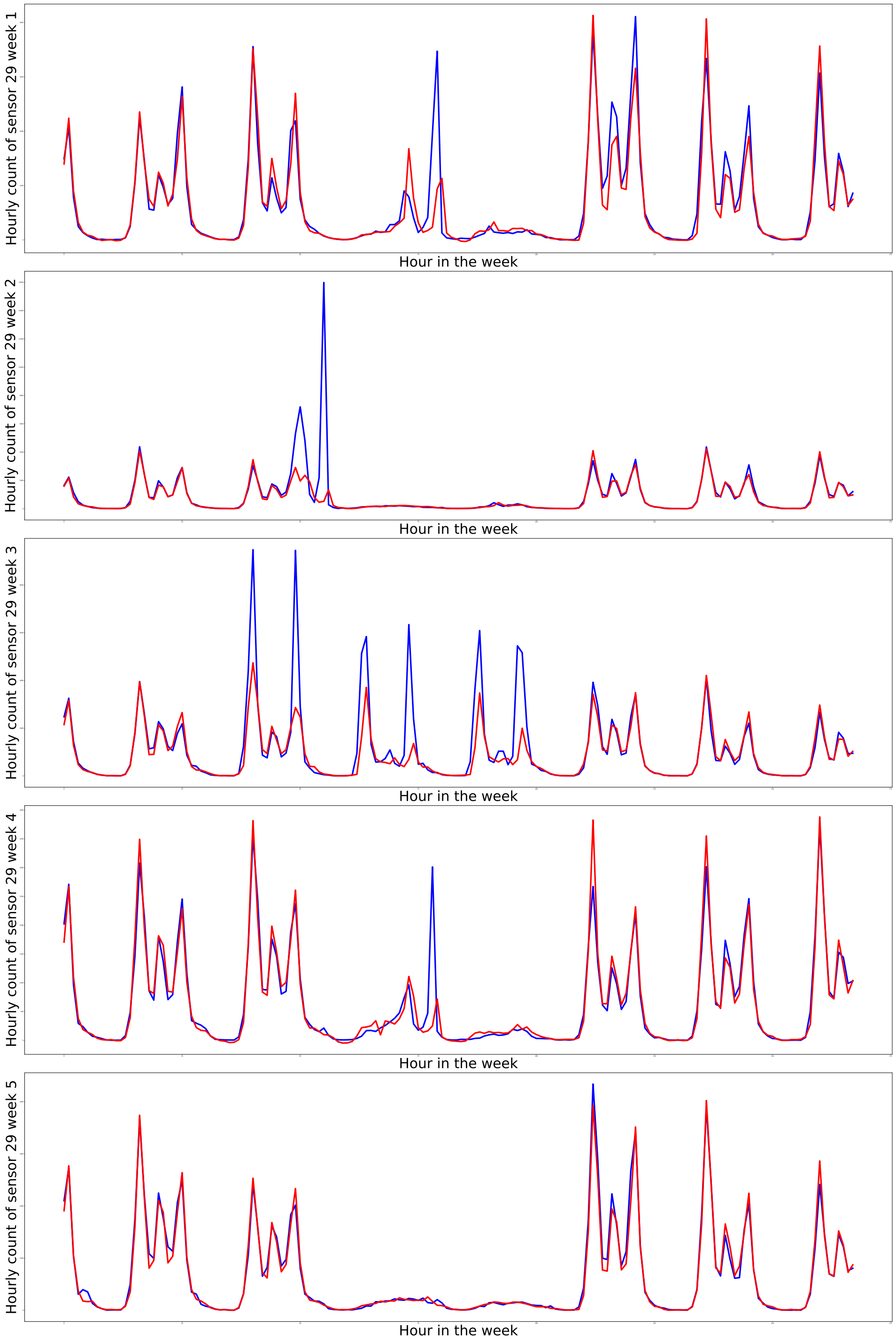


Fig. 12. Ground truths (in blue) and 1-hour-ahead prediction results (in red) of sensor 29 in 5 different weeks

Our study also provides important managerial implications. First, the DCGRU-DTW model purposed in this study can be used to detect anomalies in pedestrian flow at a city-wide scale. Early detection of such events could provide warning signals to prevent crowd-related tragedies. Subject to improved data granularity, our tool could be used by city planners to identify major pedestrian flow bottlenecks, and subsequently, improve the geometric design of infrastructure for both normal and emergency conditions. For transport technologists, we emphasise the potential value of automated pedestrian counting systems to improve the safety and seamless flow of pedestrians in dense urban areas. Such systems, when augmented by advanced analytical and predictive tools, could facilitate evidence-based and timely crowd-management decisions making. Therefore, when designing and implementing pedestrian counting systems, technology managers must carefully consider the implications associated with the geo-spatial scale, integration with other systems and expected data granularity from their deployment in a city-wide setting.

Acknowledgments

This research was partially funded by the Australian Government through the Australian Research Council Industrial Transformation Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Project ID IC200100009.

References

[1] R. Zhao, D. Dong, Y. Wang, C. Li, Y. Ma, and V. Enríquez, "Image-based crowd stability analysis using improved multi-column convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5480–5489, 2022.

[2] L. Feng and E. Miller-Hooks, "A network optimization-based approach for crowd management in large public gatherings," *Transportation Research Part C: Emerging Technologies*, vol. 42, pp. 182–199, 2014.

[3] M. Haghani, "Optimising crowd evacuations: Mathematical, architectural and behavioural approaches," *Safety Science*, vol. 128, p. 104745, 2020.

[4] C. Feliciani and K. Nishinari, "Measurement of congestion and intrinsic risk in pedestrian crowds," *Transportation Research Part C: Emerging Technologies*, vol. 91, pp. 124–155, 2018.

[5] C. Martella, J. Li, C. Conrado, and A. Vermeeren, "On current crowd management practices and the need for increased situation awareness, prediction, and intervention," *Safety Sciences*, vol. 91, pp. 381–393, 2017.

[6] X. Zhang, Y. Sun, F. Guan, K. Chen, F. Witlox, and H. Huang, "Forecasting the crowd: An effective and efficient neural network for citywide crowd information prediction at a fine spatio-temporal scale," *Transportation Research Part C: Emerging Technologies*, vol. 143, p. 103854, 2022.

[7] K. Abualsaud, M. Elfouly, Tarek, T. Khattab, E. Yaacoub, L. S. Ismail, M. H. Ahmed, and M. Guizani, "A survey on mobile crowd-sensing and its applications in the iot era," *IEEE Access*, vol. 7, pp. 3855–3881, 2019.

[8] C. Sipetas, A. Keklikoglou, and E. J. Gonzales, "Estimation of left behind subway passengers through archived data and video image processing," *Transportation Research Part C: Emerging Technologies*, vol. 118, p. 102727, 2011.

[9] H. Yang, K. Ozbay, and B. Bartin, "Enhancing the quality of infrared-based automatic pedestrian sensor data by nonparametric statistical method," *Transportation Research Record*, vol. 2264, pp. 11–17, 2011.

[10] A. Lesani and L. Miranda-Moreno, "Development and testing of a real-time wifi-bluetooth system for pedestrian network monitoring, classification, and data extrapolation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1484–1496, 2019.

[11] N. Abedi, A. Bhaskar, E. Chung, and M. Miska, "Assessment of antenna characteristic effects on pedestrian and cyclists travel-time estimation based on bluetooth and wifi mac addresses," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 124–141, 2015.

[12] N. Wijermans, C. Conrado, M. van Steen, C. Martella, and J. Li, "A landscape of crowd-management support: An integrative approach," *Safety Science*, vol. 86, pp. 142–164, 2016.

[13] C. McCarthy, I. Moser, P. P. Jayaraman, A. M. Ghaderi, H.and Tan, A. Yavari, U. Mehmood, M. Simmons, Y. Weizman, D. Georgakopoulos, F. K. Fuss, and D. Hussein, "A field study of internet of things-based solutions for automatic passenger counting," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 2, pp. 384–401, 2021.

[14] A. Felemban, Emad, F. U. Rehman, S. A. A. Biabani, A. Ahmad, A. Naseer, A. R. M. Abdul Majid, O. K. Hussain, A. M. Qamar, R. Felemban, and F. Zanjir, "Digital revolution for hajj crowd management: A technology survey," *IEEE Access*, vol. 8, pp. 208 583–208 609, 2020.

[15] H. Wang, L. Li, P. Pan, Y. Wang, and Y. Jin, "Early warning of burst passenger flow in public transportation system," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 580–598, 2019.

[16] C.-J. Jin, R. Jiang, S. Wong, S. Xie, D. Li, N. Guo, and W. Wang, "Observational characteristics of pedestrian flows under high-density conditions based on controlled experiments," *Transportation Research Part C: Emerging Technologies*, vol. 109, pp. 137–154, 2019.

[17] K. Chen and J.-K. J. K. Kämäräinen, "Pedestrian density analysis in public scenes with spatiotemporal tensor features," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 1968–1977, 2016.

[18] X. Wang, J. Liono, W. McIntosh, and F. D. Salim, "Predicting the city foot traffic with pedestrian sensor data," *ACM International Conference Proceeding Series*, pp. 1–10, 2017.

[19] City of Melbourne, "Pedestrian counting system," http://www.pedestrian.melbourne.vic.gov.au/#date=20-04-2023&time=12, 2023.

[20] M. H. Zaki and T. Sayed, "Automated analysis of pedestrian group behavior in urban settings," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 1880–1889, 2018.

[21] P. Kothari, S. Kreiss, and A. Alahi, "Human trajectory forecasting in crowds: A deep learning perspective," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23(7), pp. 7386–7400, 2022.

[22] K. Tamil Selvi, R. Thamilselvan, and S. Mohana Saranya, "Diffusion convolution recurrent neural network – a comprehensive survey," in *IOP Conf. Series: Materials Science and Engineering*, vol. 1055, 2021.

[23] Q. Q. Zhou, J. Zhang, L. Che, H. Shan, and J. Wang, "Crowd counting with limited labeling through submodular frame selection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1728–1738, 2019.

[24] T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf, and S. Shah, "Forecasting traffic congestion using ARIMA modeling," in *2019 15th international wireless communications & mobile computing conference (IWCMC)*. IEEE, 2019, pp. 1227–1232.

[25] B. Alsolami, R. Mehmood, and A. Albeshri, "Hybrid statistical and machine learning methods for road traffic prediction: A review and tutorial," *Smart Infrastructure and Applications: Foundations for Smarter Cities and Societies*, pp. 115–133, 2020.

[26] B. M. Williams and L. A. Hoel, "Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[27] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transportation Research Record*, vol. 1678, no. 1, pp. 179–188, 1999.

[28] S. Shahriari, M. Ghasri, S. Sisson, and T. Rashidi, "Ensemble of ARIMA: combining parametric and bootstrapping technique for traffic flow prediction," *Transportmetrica A: Transport Science*, vol. 16, no. 3, pp. 1552–1573, 2020.

[29] B. Dissanayake, O. Hemachandra, N. Lakshitha, D. Haputhanthri, and A. Wijayasiri, "A comparison of ARIMAX, VAR and LSTM on multivariate short-term traffic volume forecasting," in *Conference of Open Innovations Association, FRUCT*, no. 28. FRUCT Oy, 2021, pp. 564–570.

[30] E. Zivot and J. Wang, *Vector Autoregressive Models for Multivariate Time Series*. New York, NY: Springer New York, 2006, pp. 385–429.

[31] S. R. Chandra and H. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *Journal of Intelligent Transportation Systems*, vol. 13, no. 2, pp. 53–72, 2009.

[32] H.-A. T. Nguyen, H.-D. Nguyen, and T.-H. Do, "An application of vector autoregressive model for analyzing the impact of weather and nearby traffic flow on the traffic volume," in *2022 RIVF International Conference on Computing and Communication Technologies*, 2022, pp. 328–333.

[33] F. Schimbinschi, L. Moreira-Matias, V. X. Nguyen, and J. Bailey, "Topology-regularized universal vector autoregression for traffic forecasting in large urban areas," *Expert Systems with Applications*, vol. 82, pp. 301–316, 2017.

[34] T. Mai, B. Ghosh, and S. Wilson, "Multivariate short-term traffic flow forecasting using bayesian vector autoregressive moving average model," Tech. Rep., 2012.

[35] W. Xia, J. Zhang, and U. Kruger, "Semisupervised pedestrian counting with temporal and spatial consistencies," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 1705–1715, 2015.

[36] B. Sun, W. Cheng, P. Goswami, and G. Bai, "Short-term traffic forecasting using self-adjusting k-nearest neighbours," *IET Intelligent Transport Systems*, vol. 12, no. 1, pp. 41–48, 2018.

[37] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, and J. Sun, "A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 62, pp. 21–34, 2016.

[38] D. Xia, B. Wang, H. Li, Y. Li, and Z. Zhang, "A distributed spatial–temporal weighted model on mapreduce for short-term traffic flow forecasting," *Neurocomputing*, vol. 179, pp. 246–263, 2016.

[39] M. Awad, R. Khanna, M. Awad, and R. Khanna, "Support vector regression," *Efficient learning machines: Theories, concepts, and applications for engineers and system designers*, pp. 67–80, 2015.

[40] Y. Zhang and Y. Liu, "Traffic forecasting using least squares support vector machines," *Transportmetrica*, vol. 5, no. 3, pp. 193–213, 2009.

[41] C. Raskar and S. Nema, "Metaheuristic enabled modified hidden markov model for traffic flow prediction," *Computer Networks*, vol. 206, p. 108780, 2022.

[42] J. Jiang, T. Guo, W. Pan, and Y. Lu, "Freeway traffic flow prediction based on hidden markov model," in *International Conference on Intelligent Traffic Systems and Smart City (ITSSC 2021)*, vol. 12165. SPIE, 2022, pp. 427–434.

[43] Y. Tan, L. Zhang, T. Chu, and L. Wu, "Exploring pedestrian counts in a large central city area," in *Proceedings of the 24th International Conference of Hong Kong Society for Transportation Studies, HKSTS 2019: Transport and Smart Cities*, 2019.

[44] M. Veres and M. Moussa, "Deep learning for intelligent transportation systems: A survey of emerging trends," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 8, pp. 3152–3168, 2020.

[45] Y. Jin, W. Xu, P. Wang, and J. Yan, "SAE Network: A Deep Learning Method for Traffic Flow Prediction," in *2018 5th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS)*, 2018, pp. 241–246.

[46] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[47] C. C. Aggarwal *et al.*, "Neural networks and deep learning," *Springer*, vol. 10, no. 978, p. 3, 2018.

[48] Y. Tian and L. Pan, "Predicting short-term traffic flow by long short-term memory recurrent neural network," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, 2015, pp. 153–158.

[49] R. Jiang, D. Yin, Z. Wang, Y. Wang, J. Deng, H. Liu, Z. Cai, J. Deng, X. Song, and R. Shibasaki, "DL-traff: Survey and benchmark of deep learning models for urban traffic prediction," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, p. 4515–4525.

[50] D. Kang, Y. Lv, and Y.-y. Chen, "Short-term traffic flow prediction with lstm recurrent neural network," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–6.

[51] W. Zhang, Y. Yu, Y. Qi, F. Shu, and Y. Wang, "Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning," *Transportmetrica A: Transport Science*, vol. 15, no. 2, pp. 1688–1711, 2019.

[52] G. Yu and J. Liu, "A hybrid prediction approach for road tunnel traffic based on spatial-temporary data fusion," *Applied Intelligence*, vol. 49, pp. 1421–1436, 2019.

[53] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," *Expert Systems with Applications*, p. 117921, 2022.

[54] J. Ye, J. Zhao, K. Ye, and C. Xu, "How to build a graph-based deep learning architecture in traffic domain: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, pp. 3904–3924, 2020.

[55] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, and J. Yu, "Traffic flow prediction via spatial temporal graph neural network," in *Proceeding of WWW '20: The Web Conference 2020*, April 2020, p. 1082–1092.

[56] K. Lee and W. Rhee, "DDP-GCN: Multi-graph convolutional network for spatiotemporal traffic forecasting," *Transportation Research Part C: Emerging Technologies*, vol. 134, p. 103466, 2022.

[57] X. Xu, H. Zheng, X. Feng, and Y. Chen, "Traffic flow forecasting with spatial-temporal graph convolutional networks in edge-computing systems," in *2020 International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2020, pp. 251–256.

[58] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," *Advances in neural information processing systems*, vol. 33, pp. 17 804–17 815, 2020.

[59] Y. Xu, X. Cai, E. Wang, W. Liu, Y. Yang, and F. Yang, "Dynamic traffic correlations based spatio-temporal graph convolutional network for urban traffic prediction," *Information Sciences*, vol. 621, pp. 580–595, 2023.

[60] J. Gu, Z. Jia, T. Cai, X. Song, and A. Mahmood, "Dynamic correlation adjacency-matrix-based graph neural networks for traffic flow prediction," *Sensors*, vol. 23, no. 6, 2023.

[61] L. Han, B. Du, L. Sun, Y. Fu, Y. Lv, and H. Xiong, "Dynamic and multi-faceted spatio-temporal deep learning for traffic speed forecasting," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, p. 547–555.

[62] S. Fang, X. Pan, S. Xiang, and C. Pan, "Meta-msnet: Meta-learning based multi-source data fusion for traffic flow prediction," *IEEE Signal Processing Letters*, vol. 28, pp. 6–10, 2021.

[63] Y. Li, X. Wang, S. Sun, X. Ma, and G. Lu, "Forecasting short-term subway passenger flow under special events scenarios using multiscale radial basis function networks," *Transportation Research Part C: Emerging Technologies*, vol. 77, pp. 306–328, 2017.

[64] Y. Liu, Z. Liu, and R. Jia, "DeepPF: A deep learning based architecture for metro passenger flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 101, pp. 18–34, 2019.

[65] H. Peng, H. Wang, B. Du, M. Z. A. Bhuiyan, H. Ma, J. Liu, L. Wang, Z. Yang, L. Du, S. Wang *et al.*, "Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting," *Information Sciences*, vol. 521, pp. 277–290, 2020.

[66] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, and T. Li, "Predicting citywide crowd flows using deep spatio-temporal residual networks," *Artificial Intelligence*, vol. 259, pp. 147–166, 2018.

[67] X. Zhang, R. Cao, Z. Zhang, and Y. Xia, "Crowd Flow Forecasting with Multi-Graph Neural Networks," in *Proceedings of the International Joint Conference on Neural Networks*, 2020.

[68] H. Yuan, X. Zhu, Z. Hu, and C. Zhang, "Deep multi-view residual attention network for crowd flows prediction," *Neurocomputing*, vol. 404, pp. 198–212, 2020.

[69] D. C. Duives, G. Wang, and J. Kim, "Forecasting pedestrian movements using recurrent neural networks: An application of crowd monitoring data," *Sensors (Switzerland)*, vol. 19, no. 2, 2019.

[70] M. Liu, L. Li, Q. Li, Y. Bai, and C. Hu, "Pedestrian flow prediction in open public places using graph convolutional network," *ISPRS International Journal of Geo-Information*, vol. 10, no. 7, p. 455, 2021.

[71] Y. Xu, Z. Piao, and S. Gao, "Encoding crowd interaction with deep neural network for pedestrian trajectory prediction," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5275–5284.

[72] A. Sevtsuk and R. Kalvo, "Predicting pedestrian flow along city streets: A comparison of route choice estimation approaches in downtown san francisco," *International Journal of Sustainable Transportation*, vol. 16, no. 3, pp. 222–236, 2022.

[73] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations*, 2017.

[74] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018.

[75] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[76] L. R. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, pp. 64–67, 2001.

[77] R. Dey and F. M. Salem, “Gate-variants of gated recurrent unit (GRU) neural networks,” in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2017, pp. 1597–1600.

[78] G. Yiğit and M. F. Amasyali, “Simple but effective GRU variants,” in *2021 international conference on INnovations in intelligent SysTems and applications (INISTA)*. IEEE, 2021, pp. 1–6.

[79] L. M. Pfiester, R. G. Thompson, and L. Zhang, “Spatiotemporal exploration of Melbourne pedestrian demand,” *Journal of Transport Geography*, vol. 95, p. 103151, 2021.

[80] A. F. Agarap, “Deep learning using rectified linear units (relu),” *arXiv preprint arXiv:1803.08375*, 2018.

[81] J. Han and C. Moraga, “The influence of the sigmoid function parameters on the speed of backpropagation learning,” in *From Natural to Artificial Neural Computation: International Workshop on Artificial Neural Networks Malaga-Torremolinos, Spain, June 7–9, 1995 Proceedings 3*. Springer, 1995, pp. 195–201.

[82] C.-L. Yang, A. Setyoko, H. Tampubolon, and K.-L. Hua, “Pairwise Adjacency Matrix on Spatial Temporal Graph Convolution Network for Skeleton-Based Two-Person Interaction Recognition,” in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 2166–2170.

[83] Y. Jiang, M. Li, Y. Fan, and Z. Di, “Characterizing dissimilarity of weighted networks,” *Scientific Reports*, vol. 11, no. 1, p. 5768, 2021.

[84] X. Zhang, C. Huang, Y. Xu, L. Xia, P. Dai, L. Bo, J. Zhang, and Y. Zheng, “Traffic flow forecasting with spatial-temporal graph diffusion network,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, 2021, pp. 15 008–15 015.

[85] M. Xu and H. Liu, “Road Travel Time Prediction Based on Improved Graph Convolutional Network,” *Mobile Information Systems*, vol. 2021, p. 7161293, 2021.

[86] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” *Advances in neural information processing systems*, vol. 27, 2014.

[87] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, “A survey on long short-term memory networks for time series prediction,” *Procedia CIRP*, vol. 99, pp. 650–655, 2021.

[88] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[89] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, “Scheduled sampling for sequence prediction with recurrent neural networks,” in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS’15. Cambridge, MA, USA: MIT Press, 2015, p. 1171–1179.

[90] N. K. Kaur, U. Kaur, and D. Singh, “K-medoid clustering algorithm-a review,” *Int. J. Comput. Appl. Technol*, vol. 1, no. 1, pp. 42–45, 2014.

[91] P. J. Rousseeuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

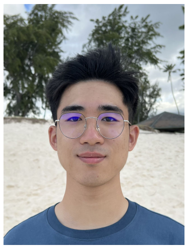
[92] C. M. Salgado, C. Azevedo, H. Proença, and S. M. Vieira, *Noise Versus Outliers*. Cham: Springer International Publishing, 2016, pp. 163–183.

[93] X. Xu, L. Zhang, Q. Kong, C. Gui, and X. Zhang, “Enhanced-historical average for long-term prediction,” in *2022 2nd International Conference on Computer, Control and Robotics (ICCCR)*. IEEE, 2022, pp. 115–119.

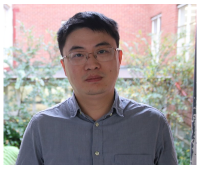
[94] Y. Li and C. Shahabi, “A brief overview of machine learning methods for short-term traffic forecasting and future directions,” *Sigspatial Special*, vol. 10, no. 1, pp. 3–9, 2018.

[95] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, “Embracing change: Continual learning in deep neural networks,” *Trends in cognitive sciences*, vol. 24, no. 12, pp. 1028–1040, 2020.

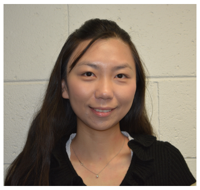
[96] M. Dragoi, E. Burceanu, E. Haller, A. Manolache, and F. Brad, “Anoshift: A distribution shift benchmark for unsupervised anomaly detection,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 32 854–32 867, 2022.



Yiwei Dong is a MSc in School of Statistics at Renmin University of China. He is interested in time series analysis, spatiotemporal data mining, and learning theory.



Tingjin Chu is a Senior Lecturer in School of Mathematics and Statistics at the University of Melbourne (UoM). He got a Bachelors Degree in Mathematics from Tsinghua University, China, and a PhD Degree in Statistics from Colorado State University, USA. (Southeast University, China), and a PhD in Operations Research (UoM). His research interest is to model the spatial and spatio-temporal data using both statistical and machine learning methods.



Lele Zhang is a Senior Lecturer in School of Mathematics and Statistics at the University of Melbourne (UoM). She has a Bachelors Degree in Engineering (Southeast University, China), and a PhD in Operations Research (UoM). Her areas of expertise are broadly transport modelling and optimisation, traffic flow theory, Monte Carlo simulation, scheduling theory, Operations Research, City Logistics, and time series analysis. She is a Chief Investigator of ARC Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), and is involved in Physical Internet Lab at UoM. She also has research collaborations with a number of industries in logistics, transport and healthcare.

Hanfang Yang is a professor of the School of Statistics, Renmin University of China (RUC). He is the deputy director of the Metaverse Research Center, and deputy director of the Blockchain Research Institute of RUC. Dr. Yang has won the second prize of the Statistical Science and Technology Progress Award of China. His research interest is machine learning and artificial intelligence in social science.

Hadi Ghaderi (IEEE Member) is an Associate Professor in Digital and Sustainable Supply Chain at Swinburne University of Technology. Hadi is also an Associate Editor for IEEE Engineering Management Review. He has led a number of industry-based research projects in the area of Intelligent Transport Systems and Supply Chain Digitalisation. His research interest is focused around digital transformation, city logistics, optimisation, operations management and intelligent transport systems.