ArrivalNet: Predicting City-wide Bus/Tram Arrival Time with Two-dimensional Temporal Variation Modeling

Zirui Li, *Graduate Student Member, IEEE,* Patrick Wolf, Meng Wang, *Senior Member, IEEE*

Abstract—Accurate arrival time prediction (ATP) of buses and trams plays a crucial role in public transport operations. Current methods focused on modeling one-dimensional temporal information but overlooked the latent periodic information within time series. Moreover, most studies developed algorithms for ATP based on a single or a few routes of public transport, which reduces the transferability of the prediction models and their applicability in public transport management systems. To this end, this paper proposes *ArrivalNet*, a two-dimensional temporal variation-based multi-step ATP for buses and trams. It decomposes the one-dimensional temporal sequence into intra-periodic and inter-periodic variations, which can be recast into two-dimensional tensors (2D blocks). Each row of a tensor contains the time points within a period, and each column involves the time points at the same intra-periodic index across various periods. The transformed 2D blocks in different frequencies have an image-like feature representation that enables effective learning with computer vision backbones (e.g., convolutional neural network). Drawing on the concept of residual neural network, the 2D block module is designed as a basic module for flexible aggregation. Meanwhile, contextual factors like workdays, peak hours, and intersections, are also utilized in the augmented feature representation to improve the performance of prediction. 125 days of public transport ta from Dresden were collected for model training and validation. Experimental results show that the root mean square error, mean absolute error, and mean absolute percentage error of the proposed predictor decrease by at least 6.1%, 14.7%, and 34.2% compared with state-of-the-art baseline methods.

Index Terms—Tram/bus arrival time, time series forecasting, temporal variation modeling.

I. INTRODUCTION

As urban populations swell and individual car ownership rises, traffic congestion, energy consumption, and air pollution pose increasing challenges to urban transport systems. Public transport is one of the promising options to address the issues and achieve sustainable transport [1, 2]. However, existing public transport (PT) systems (buses, trams, subways, etc.) often suffer from issues of low reliability and prolonged delays in arrival times [3, 4]. [5] indicated that users were significantly concerned with the accuracy of travel and arrival time predictions (ATP), which greatly influences their travel choices

Zirui Li and Meng Wang are with the Chair of Traffic Process Automation, "Friedrich List" Faculty of Transport and Traffic Sciences, TU Dresden, 01069, Dresden, Germany. (Email: zirui.li@tu-dresden.de; meng.wang@tu-dresden.de)

Patrick Wolf is with the Traffic Management Department of the Dresden Transport Company, Dresdner Verkehrsbetriebe AG, 01129, Dresden, Germany. (Email: patrick.wolf@dvbag.de)

Corresponding author: Meng Wang.

and experiences. Fig. 1 illustrates the multi-step bus/tram ATP. In this scenario, when the bus/tram is on Link 1 (between stop 0 and stop 1) and has been delayed 55 seconds, the passengers in the future stops would like to know **when will the bus/tram arrive at stops 3-5?**

Accurate ATP not only facilitates the rescheduling and dispatching of public transport for operators but also helps passengers make informed decisions regarding their traveling plans and mode choices[6, 7]. It needs to take into account of many influencing factors: the traffic situation (e.g., traffic volume and speed), platform facility design, passenger demand, the propagation of delay, traffic signals, etc [8–11]. All elements above lead to the complexity and difficulty in the accurate ATP. To cope the issues, numerous solutions have been proposed to predict the arrival time [12, 13], which can be divided into three categories: state estimation-based and statistical learning-based and deep learning-based methods. In the sequel, we survey the main methods in each category to identify the knowledge gaps.

State estimation-based methods treat the public transport operation process as a dynamic system and the kinematic state of public transport vehicles as the system state to be estimated. To this end, popular state estimation approaches of Kalman filter (KF) and Bayesian networks from the systems engineering domain can be applied for ATP [14–16]. KF is an efficient state estimation method, as it can update the linear system state when new observations become available continuously. [17] proposed a KF-based model for vehicle arrival/departure prediction using real-time and historical data. It makes optimal estimation of the location and speed of the vehicle based on the streaming data from the automatic vehicle location (AVL) system. [16] formulated a spatial KF to detect the unknown order of spatial dependence, and then learn its linear, non-stationary spatial correlations for this detected order. However, the performance is limited by the linear formulation of the state-space model. [18] proposed a hierarchical Bayesian framework for bus dwell time prediction with minimal historical data. It made predictions using a small set of continually updated model parameter distributions, which was inherently adaptive to the time-varying duration of dwell time. The Bayesian architecture also provided the confidence in dwell time estimation to support the decision-making of scheduling under uncertainty. In [19], a Bayesian Gaussian mixture model (GMM) was developed to generate probabilistic forecasting of bus travel time. It characterizes the strong dependencies between adjacent buses (e.g., correlated speed

Fig. 1: The problem illustration of multi-step bus/tram arrival time prediction. **Bottom left**: The city-wide public transport link delay distribution of tram in Dresden, Germany. (A link is the route segment between adjacent stops.) **Top**: When the bus/tram is on Link 1 (between stop 0 and stop 1) and has delayed 55 seconds, the potential passengers in the future stops are concerned on *when the bus/tram will arrive at stop 3/5*? **Bottom right**: The analysis from the two-dimensional perspective. In a sequence of tram delay information with 39 stops, the temporal pattern of falling and fluctuation are labelled in red and green, respectively.

and smooth variation of headway). In the inference stage, an efficient Markov chain Monte Carlo (MCMC) algorithm was applied for probabilistic prediction. Furthermore, a conditional forecasting model for bus travel time and passenger occupancy was proposed with a similar strategy [20]. The advantage of state estimation-based methods is their ability to explicitly model the relationships among historical observations, noise, and uncertainties. Additionally, these methods stand out for their efficiency, requiring only a minimal dataset to operate effectively. However, the relationship between some factors and ATP is nonlinear (e.g., weather conditions), making it daunting to formulate these implicit and intertwined relationships explicitly. Meanwhile, for long-term predictions, the influence between nonadjacent samples and the propagation of delays in time series becomes the bottleneck of the state estimation-based methods [8].

Statistical learning-based approaches for public transport ATP aim at capturing complex public transport modalities and nonlinear relationships inherent in the collected data from the perspective of statistics [8, 9]. [21] integrated support vector machine (SVM) and genetic algorithm to search the best parameters in predicting the bus arrival time in various traffic conditions. [22] proposed a k-nearest neighbor (k-NN)-based framework for bus arrival time estimation, which outperformed simple artificial neural networks (ANNs). In [23], ANN, KF, k-NN, and linear regression (LR) were adopted for the bus ATP

at the same bus stop but with different routes using real-world data. Compared with the bus information of the same route, multi-source from several routes can provide more benefit in the timeliness and reliability of the information. Meanwhile, the performance of several prediction methods are assessed and a valuable insight for algorithm selection was provided, which guided the further development of travel time predictors. In [9], various statistic machine learning algorithms were compared and evaluated. It demonstrated that real-time traffic information from taxis could improve the performance of ATP based on bus GPS data under both normal and abnormal traffic conditions. [24] integrated KF and k-NN algorithm for bus travel time estimation. It applied k-NN algorithm to capture the related input features and then combined exponential smoothing technique with a recursive estimation scheme based on KF to generate the prediction.

The above methods primarily focus on one-step prediction of public transport arrival time, which only consider the ATP of the next adjacent stop. The long-term multi-step prediction is also crucial for the public transport management system and it can provide more useful reference information for decisions of transport operators and travelers. With the development of deep neural networks (DNNs), DNN-based time series prediction methods were widely used in ATP. [8] proposed combining convolutional Long short-term memory (ConvL-STM) with ensemble learning and employing the eXtreme

Gradient Boosting (XGBoost) statistical method to model heterogeneous traffic characteristics, thereby achieving prediction of bus arrival times under various traffic conditions. [25] developed a KF-LSTM (Kalman filter-LSTM) deep learning method to predict bus travel time. With the statistical analysis, it was found that the KF-LSTM approach can outperform the ensemble learning strategy. [26] proposed a traffic pattern-centric segment coalescing framework to learn heterogeneous traffic patterns and incorporated LSTM in each cluster to predict the bus travel time. Besides investigating the interaction of arrival time for public transport on a single route, [27] designed a parallel Gated Recurrent Unit (GRU) network-based method to capture the spatial correlation among bus stops on different lines under the condition of limited data. In summary, with the advancement of data collection technologies (e.g., AVL data), deep learning-based methods demonstrate significant superiority in capturing nonlinear relationships, time series prediction, and multi-sources heterogeneous information fusion.

However, there are two inherent shortcomings for current solutions. Firstly, most methods for multi-step ATP focus on exploring the relationships between different time points alongside the one-dimensional temporal variation. Here, the one-dimensional variation refers to considering changes in features over a continuous duration. Specifically, for the public transport ATP, it involves analyzing the information at different stops on a specific route. While one-dimensional time series analysis helps capture continuity, periodicity and trends, real-world time series often have intricate temporal patterns (e.g., rising, falling, etc.). These different patterns are always mixed, overlapped with each other and their characteristics are obscured deeply in time series [28–30]. As shown in the bottom right of Fig. 1, for a sequence of public transport arrival information with 39 stops, the temporal pattern of falling and fluctuation are hidden in the time series, which are labeled in red and green, respectively. These similar patterns are difficult to be captured by the one-dimensional based algorithms. Therefore, for the multi-step ATP, the modeling of multiple temporal variations is required to model the connection between similar temporal patterns. Secondly, most of the existing studies only used the data collected from one single or few routes [31, 32], which are rather limited [33]. It reduces the predictive capability for newly added, modified, or unmodeled routes.

To address these issues, in this paper, a two-dimensional temporal variation-based bus/tram ATP model is proposed, which is termed *ArrivalNet*. It transforms one-dimensional time series into two-dimensional tensors to represent the intra-period and inter-period interactions. It can capture the implicit periodic information in the multi-step sequential prediction of arrival time. Meanwhile, a large scale city-wide dataset is constructed for validation [34]. The primary contributions of this paper are summarized as follows:

- A two-dimensional temporal variation-based bus/tram ATP model that can capture the intra-period and inter-period information in time series is proposed;
- The proposed model is validated on city-wide public transport data, which includes bus/tram operational data

from 125 days.

The remainder of this paper is organized as follows. The problem of public transport ATP is formulated in Section II. Then, details of *ArrivalNet* are presented in Section III. Section IV shows the experimental setting and the comparison results. Finally, Section V presents the conclusion and future work.

II. PROBLEM FORMULATION

This paper focuses on the multi-step bus/tram ATP. The problem formulation of ATP is divided into three parts: the description of the ATP problem, the construction of variables, and the series stationarization.

A. The formulation of bus/tram arrival time prediction

Following the formulation in [19], in this work, the prediction of arrival time T_i^{a} at stop i is converted to the prediction of delay T_i^{d} relative to the scheduled arrival time T_i^{s}, which implicitly includes the dwell time at stop $i-1$ in the travelling time between stops $i-1$ and i.

$$T_i^{\text{a}} = T_i^{\text{d}} + T_i^{\text{s}} \tag{1}$$

With (1), the multi-step ATP is transformed into the sequential prediction of delay in future N_f stops based on the information in the past N_p stops, which can be formulated as follow:

$$\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{d}} = f_\theta(\mathbf{F}_{i-N_p+1:i}^{\text{temporal}}, \mathbf{F}^{\text{static}}) \tag{2}$$

$$\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{a}} = \hat{\mathbf{T}}_{i+1:i+N_f}^{\text{d}} + \mathbf{T}_{i+1:i+N_f}^{\text{s}} \tag{3}$$

where $\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{a}}$, $\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{d}}$ and $\mathbf{T}_{i+1:i+N_f}^{\text{s}}$ are predicted arrival time, predicted delay time and scheduled arrival time for stops $i+1$ to $i+N_f$, respectively. $\mathbf{F}_{i-N_p+1:i}^{\text{temporal}}$ are temporal features in the past N_p stops and $\mathbf{F}^{\text{static}}$ is the static contextual information, which can be flexibly designed and embedded as the input. f_θ is the prediction model with trainable parameters θ.

B. The construction of variables and series stationarization

Considering that this work aims to develop a generic bus/tram ATP model, at each stop i, the index of the specific line is not considered as the input feature. At stop i, the temporal features $\mathbf{F}_{i-N_p+1:i}^{\text{temporal}} = [\mathbf{F}_{i-N_p+1}, ..., \mathbf{F}_i]^\top \in \mathbb{R}^{N_p \times C}$ are the serial combination of information $\mathbf{F}_j$ ($\forall j \in [i - N_p + 1, i]$, $j \in \mathbb{Z}$) in each past stop:

$$\mathbf{F}_j = [S_j^{\text{t}}, T_j^{\text{t}}, T_j^{\text{d}}, I_j, \overline{T}_j^{\text{t}}] \tag{4}$$

where C is the length of feature space. S_j^{t} is the traveling distance between stop j and previous stop $j - 1$, which is highly related to the arrival time and can be obtained from the route of the specific line. T_j^{t} is the scheduled traveling time between stop j and previous stop $j - 1$ in the daily updated timetable. T_j^{d} is the delay at stop j. It is calculated from the difference of timetable and real-time arrival information. I_j is is a boolean value indicating whether there is a traffic light between stop j and previous stop $j - 1$. $\overline{T}_j^{\text{t}}$ is the average of

Fig. 2: The overall framework of the proposed *ArrivalNet*. It consists of five parts: feature embedding, fast fourier transform (FFT) from time to frequency domain, reshape from 1D tensor to 2D image like tensor, vision backbone (CNN-based or Transformer-based) and adaptive aggregation. Note that two possible algorithms are shown, but only one is enough for feature extraction.

T_j^{t} at stop j in collected data. S_j^{t} and T_j^{t} are daily updated from the centralized data collection system, which take into account of stop and route changing caused by adjustment in road infrastructure. The average travel time is calculated by statistically aggregating the time on the link between stop $j-1$ and j. A significant discrepancy between the actual travel time and the average travel time may indicate the presence of exceptional circumstances affecting the delay.

C. Series stationarization

As for the data collected from the public transport system, the sequences always reflect non-stationarity, which is characterized by the continuous change of joint distribution over time. This reduces the predictability of time series. In [35], an effective normalization-and-denormalization strategy was proposed to normalize instances with learnable parameters in the transformation from raw to normalized time series. It alleviated the temporal distributional shift by learning the affine transformation of input in the normalization and restoring the corresponding output in the denormalization. From the

perspective of ATP, the shift of the temporal distribution is the variation of input features' joint distribution along the temporal dimension. For example, the corresponding relationship between average link travel time and distance may change along the same public transport line. [36] experimentally demonstrated that the algorithm also worked without learnable parameters. In normalization, it standardizes instances with varied means and variances. Then, the prediction is recovered with the same statistical properties in denormalization.

For the original input sequence $\mathbf{F}^{\text{temporal}}_{i-N_p+1:i}$, apply the normalization and de-normalization along the temporal dimension to obtain the normalized input $\mathbf{F}^{\text{temporal}}_{i-N_p+1:i}{}' = [\mathbf{F}_{i-N_p+1}{}', ..., \mathbf{F}_i{}']^\top \in \mathbb{R}^{N_p \times C}$:

$$\mu = \frac{1}{N_p} \sum_{j=i-N_p+1}^{i} \mathbf{F}_j \qquad (5)$$

$$\sigma^2 = \frac{1}{N_p} \sum_{j=i-N_p+1}^{i} (\mathbf{F}_j - \mu)^2 \qquad (6)$$

$$\mathbf{F}_j{}' = \frac{1}{\sigma} \odot (\mathbf{F}_j - \mu) \qquad \forall j \in [i-N_p+1, i], \quad j \in \mathbb{Z} \qquad (7)$$

where μ and $\sigma \in \mathbb{R}^{C \times 1}$ are mean and standard deviation of time series. $\odot$ is the element wise product. The normalized input feature $\mathbf{F}^{\text{temporal}}_{i-N_p+1:i}{}' = [\mathbf{F}_{i-N_p+1}{}', ..., \mathbf{F}_i{}']^\top \in \mathbb{R}^{N_p \times C}$ is combined with $\mathbf{F}^{\text{static}}$ as $\mathbf{F}_{\text{1D},i} \in \mathbb{R}^{N_p \times (C+N_c)}$, where N_c is the feature dimension of contextual information. To simplify the expression, $\mathbf{F}_{\text{1D},i}$ is substituted with $\mathbf{F}_{\text{1D}}$. In the following parts, $\mathbf{F}_{\text{1D}}$ is the feature representation at a specific stop. Then, $\mathbf{F}_{\text{1D}}$ is sent to the prediction module to generate the estimation $\hat{\mathbf{T}}^{\text{d}'}_{i+1:i+N_f} = [\hat{\mathbf{T}}^{\text{d}'}_{i+1}, ..., \hat{\mathbf{T}}^{\text{d}'}_{i+N_f}]^\top \in \mathbb{R}^{N_f \times C}$. Then, the de-normalization is operated for final multi-step predictions $\hat{\mathbf{T}}^{\text{d}}_{i+1:i+N_f} = [\hat{\mathbf{T}}^{\text{d}}_{i+1}, ..., \hat{\mathbf{T}}^{\text{d}}_{i+N_f}]^\top \in \mathbb{R}^{N_f \times C}$:

$$\hat{\mathbf{T}}^{\text{d}}_k = \sigma \odot \hat{\mathbf{T}}^{\text{d}'}_k + \mu \qquad \forall k \in [i+1, i+N_f], \quad k \in \mathbb{Z} \qquad (8)$$

After the de-normalization, the outputs have the same feature space as the original inputs. The two-step operations can reduce the influence of non-stationarity in time series.

III. METHODOLOGY

The proposed *ArrivalNet* can be divided into five parts: feature embedding, fast fourier transform (FFT) from time to frequency domain, reshape from 1D tensor to 2D image like tensor, vision backbone for feature extraction and adaptive aggregation, which are shown in Fig. 2. For the first part, the normalized input feature is expended from the original feture space to the model space, which increases the learning ability of model. The last four parts form the 2D block. It is the basic module of *ArrivalNet*. In the FFT, the sequential input in the time domain is converted into frequency domain with different frequencies. Then, the 1D tensor is reshaped to various 2D tensors according to the frequencies and periods from FFT. The useful information in two-dimensional image like features is captured by generic vision backbone (CNN and Swin Tranformer). the parameters from the periodic decomposition are designed to adaptively aggregate features of different periods.

A. Feature embedding

The normalized feature $\mathbf{F}^{\text{temporal}}_{i-N_p+1:i}{}'$ is combined with $\mathbf{F}^{\text{static}}$ as the final input sequence $\mathbf{F}_{\text{1D}} \in \mathbb{R}^{N_p \times (C+N_c)}$. Time series data is inherently sequential, with the order of N_p data points carrying significant information about temporal dynamics. Unlike recurrent neural network (RNN)-based models that intrinsically understand sequence order, the two dimensional tensor of 2D block doesn't have a built-in mechanism to recognize the order of inputs. The positional encoder (PE) provide a way to incorporate this crucial information by adding a unique positional signal to each data point in the sequence, enabling the model to recognize and utilize the order of observations [37]. Meanwhile, to better understand and capture complex patterns in multivariate time series, including non-linear relationships and interactions between different features, a value encoder (VE) is applied to expand features to a high-dimensional model space d_{model} and increase model's learning ability [38]. Wtih $\mathbf{F}^{\text{static}}$ from the series stationarization, the formulation of PE and VE are shown as follow:

$$\mathbf{F}^{(pos,2j)}_{\text{PE}} = \sin\left(pos/(2N_p)^{2j/d_{\text{model}}}\right) \qquad (9)$$

$$\mathbf{F}^{(pos,2j+1)}_{\text{PE}} = \cos\left(pos/(2N_p)^{2j/d_{\text{model}}}\right) \qquad (10)$$

$$\mathbf{F}_{\text{VE}} = \operatorname*{conv1D}_{(C+N_c) \to d_{\text{model}}} (\mathbf{F}_{\text{1D}}) \qquad (11)$$

$$\mathbf{X}_{\text{1D}} = \operatorname*{Linear}_{N_p \to (N_p+N_f)} (\mathbf{F}_{\text{PE}} + \mathbf{F}_{\text{VE}}) \qquad (12)$$

where $j \in \{1, ..., \lfloor d_{\text{model}}/2 \rfloor\}$ and $pos \in \{1, ..., \lfloor N_p \rfloor\}$. $\mathbf{F}_{\text{PE}} \in \mathbb{R}^{N_p \times d_{\text{model}}}$ is the unique positional values. $\mathbf{F}_{\text{VE}} \in \mathbb{R}^{N_p \times d_{\text{model}}}$ is the encoded model features. conv1D is the one dimension convolutional along the temporal dimension of $\mathbf{F}_{\text{1D}}$. It expend the feature space from lower to higher dimension. Linear$(\cdot)$ is the linear neural network to align the past sequence length N_p to full sequence length $N_p + N_f$. In (12), the combination of $\mathbf{F}_{\text{PE}}$ and $\mathbf{F}_{\text{VE}}$ is the element-wise sum. To the end, $\mathbf{F}_{\text{1D}}$ is encoded as $\mathbf{X}_{\text{1D}} \in \mathbb{R}^{(N_p+N_f) \times d_{\text{model}}}$.

B. Fast fourier transform from time to frequency domain

It is well recognized that time series can be analyzed from two perspectives: the time domain and the frequency domain [39, 40]. The RNN-based and attention-based methods focus on the modeling of temporal relationships in the time domain, which is termed one-dimensional temporal variation [37, 41]. The periodic trends (e.g., daily, weekly, monthly) can be reflected by the multi-periodic positional encoder [38]. However, some hidden periodic information may be neglected (e.g. rising, falling, fluctuation). It involves establishing connections between time points with the same time index across different periods. Fourier analysis serves as a common tool for transforming serial input from the time domain to the frequency domain [42]. [29] designed a frequency-enhanced block to capture these obscured variations by Fast Fourier Transform (FFT). In [30], a similar strategy was used for the transformation from one-dimensional to two-dimensional tensors. In this work, FFT is selected as the tool to convert

$\mathcal{X}_{1D} \in \mathbb{R}^{N_f \times d_{\text{model}}}$ $\mathcal{X}_{1D} \in \mathbb{R}^{N_f \times d_{\text{model}}}$ $\mathcal{X}_{2D} \in \mathbb{R}^{f_1 \times p_1 \times d_{\text{model}}}$

N_f stops (timesteps)

N_f stops (timesteps)

Padding

$\mathcal{X}_{1D} \in \mathbb{R}^{(f_i * p_i - N_f) \times d_{\text{model}}}$

p_i period length

f_i periods

d_{model}

padding elements

Embedded time series   **time series with padding**   **2D tensor reshaping**

Fig. 3: The illustration 2D tensor padding.

sequences from time to frequency domain. With the embedded feature $\mathbf{X}_{1D}$, the process of FFT is expressed as:

$$\mathbf{A} = \text{FFT}(\mathbf{X}_{1D}) \qquad (13)$$

$$\{f_1, \cdots, f_{\frac{T}{2}}\} = \text{Avg}(\text{Amp}(\mathbf{A})) \qquad (14)$$

where $\text{FFT}(\cdot)$ is the Fast Fourier Transform[1] from the time domain to the frequency domain and $\mathbf{A}$ comprises a series of complex numbers, each representing the magnitude and phase of frequency components within the sequence. $\text{Amp}(\cdot)$ is the amplifying function of complex number and $\text{Avg}(\cdot)$ is the mean of elements in the feature dimension. $\{f_1, \cdots, f_{\frac{T}{2}}\}$ is the obtained frequencies. Due to the conjugacy in the transformed frequency domain, f_* is only selected within $\{1, \cdots, \lfloor \frac{T}{2} \rfloor\}$. Simply considering the effects of all frequencies will lead to a decrease in prediction performance because high-frequency changes in the sequence may be caused by noise. To avoiding the noise from meaningless high frequencies, only the most prominent k frequencies are chosen.

$$\{f_1, \cdots, f_k\} = \underset{f_* \in \{1, \cdots, \lfloor \frac{T}{2} \rfloor\}}{\arg \text{Topk}}(\mathbf{A}) \qquad (15)$$

$$p_i = \left\lfloor \frac{T}{f_i} \right\rfloor \quad i \in \{1, \cdots, k\} \qquad (16)$$

where $\{f_1, \cdots, f_k\}$ is the frequency of kth period with length p_i and the unnormalized calculated amplitude is as follows:

$$\mathbf{A}_{\text{Top K}} = \{\mathbf{A}_{f_1}, ..., \mathbf{A}_{f_k}\} \qquad (17)$$

C. Reshape from 1D to 2D image like tensor

Based on (16) and (17), for each frequency $f_j, j \in [1, k]$ $j \in \mathbb{Z}$, the one-dimensional time serie $\mathbf{X}_{1D} \in \mathbb{R}^{(N_p + N_f) \times d_{\text{model}}}$ can be transformed and spliced into a tensor $\mathbf{X}_{2D}^j \in \mathbb{R}^{f_j \times p_j \times d_{\text{model}}}$. 2D means that the inter-periodic and intra-periodic information is represented in $\mathbf{X}_{2D}^j$. It is an

[1]FFT is implemented by Pytorch library: pytorch.org.

image-like tensor with the channel length d_{model}. Each row of the tensor contains the time points within a period, and each column of it involves the time points at the same intra-periodic index across various periods. Ideally, the relationship of $f_j, p_j, j \in [1, k]$ $j \in \mathbb{Z}$ is $f_j \times p_j = (N_p + N_f)$. However, in some cases, (16) cannot be divisible integrally. A padding operation is needed for $\mathbf{X}_{1D}$.

$$\mathbf{X}_{1D}^{\text{padding}} = \underset{p_j, f_j}{\text{Padding}}(\mathbf{X}_{1D}), \quad j \in \{1, \dots, k\} \qquad (18)$$

$$\mathbf{X}_{2D}^j = \text{Reshape}_{p_j, f_j}\left(\mathbf{X}_{1D}^{\text{padding}}\right), \quad j \in \{1, \dots, k\} \qquad (19)$$

where $\text{Padding}(\cdot)$ is applied to impute the feature with zeros for successful reshape. Fig. 3 illustrates the process of padding and reshape.

D. Vision backbone

Two popular deep-learning architectures, CNN and Transformer, are chosen for extracting features from the 2D tensors. We remark that the proposed prediction model is generic and allows the use of other architectures.

1) CNN-based feature extraction: For each element $\mathbf{X}_{2D}^j$ $j \in \{1, \dots, k\}$ in $\mathbf{X}_{2D}$, the first and second dimensions represent the intra-periodic and inter-periodic information, which is similar to the feature space of image in computer vision. The frequency f, period p and d_{model} dimensions are corresponding to the height, width and channel dimensions in an image. Therefore, the tensor $\mathbf{X}_{2D}$ can be easily processed by the parameter-efficient inception block and finally transformed back to one-dimensional feature space [43]:

$$\hat{\mathbf{X}}_{2D}^j = \text{Inception}_{\text{Conv2d}}(\mathbf{X}_{2D}^j), \quad j \in \{1, \dots, k\} \qquad (20)$$

$$\hat{\mathbf{X}}_{1D}^j = \text{Reshape}(\hat{\mathbf{X}}_{2D}^j), \quad j \in \{1, \dots, k\} \qquad (21)$$

where $\hat{\mathbf{X}}_{2D}^j \in \mathbb{R}^{f_j \times p_j \times d_{\text{model}}}$ is the tensor processed by $\text{Inception}(\cdot)$ and $\hat{\mathbf{X}}_{1D}^j \in \mathbb{R}^{(N_p + N_f) \times d_{\text{model}}}$ is the output feature of the module. As shown in Fig. 4, in the parameter-efficient inception block, kernels with different sizes operate the two

Fig. 4: CNN-based feature extraction.

Fig. 5: Swin Transformer-based feature extraction.

dimensional convolution along dimensions $f * p$. All kernels work parallelly and are summed together in the first layer. Then, with a nonlinear actvation function, it is sent to the second layer with the similar operation. By carefully designing the relationship of kernel size and padding length in each layer, the input and output of Inception$(\cdot)$ are tensors with same size, which doesn't influence the transformation from two to one dimensional tensor.

From the analysis in Section III-B and III-C, the 1-dimensional tensor obtained by feature embedding is transformed into 2D tensor after FFT and padding processes. For the designed information extractor within the 2D block, besides employing a CNN-based inception model to extract useful features from the image-like tensor, other popular vision backbones can also be utilized. In this work, an attention-based approach, named Swin Transformer, is applied to the 2D tensor.

Fig. 6: The computationally efficient operation after the window shifting.

2) Attention-based feature extraction: In deep learning, the development of Transformer has significantly enhanced model performance in sequential analysis and natural language processing (NLP) based on the attention mechanism. However, for vision tasks, transformer-based models struggle to process image information efficiently. The difficulty arises from the high resolution of the image, while calculating attention across all pixels is computationally prohibitive. To address this issue, [44] proposed dividing the image into several non-overlapping patches and extracting useful information through a hierarchical structure. However, compared to CNN-based methods, this approach struggles to establish connections between adjacent pixels across different patches, and the computation time increases quadratically with image size. Inspired by the sliding operation in CNNs, [45] introduced a local transformer method based on sliding windows, termed Swin Transformer. It models adjacent pixels through the movement of local windows, while its computational complexity increases linearly relative to the image size. Therefore, in this work, the Swin Transformer is employed to extract useful information from image-like 2D tensors.

As illustrated in Fig. 5, we assume that the frequency and period length of the 2D tensor are 4 and 6, respectively. Compared to high-resolution images, the number of pixels in a 2D tensor is relatively low. Hence, the patch size is set to 1 and each 2x2 local window contains only 4 pixels, and the 2D tensor comprises 6 windows. After applying Window-based Multi-head Self Attention (W-MSA) within local windows, features from different local windows are recombined. The

self-attention mechanism is formulated as follow:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_{\text{model}}}}\right)\mathbf{V} \qquad (22)$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{M^2 \times d_{\text{model}}}$ are *query*, *key* and *value*, respectively. M^2 is the number of pixels (patches) in a local window. In the self-attention, $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ are same features. To introduce cross-window connections while maintaining efficient computation of non-overlapping windows, a shifted window partitioning strategy is implemented. This involves moving the windows in the 2D tensor starting from the top left by the half size of a local window. In this case, it will expand the number of local windows from 6 to 12. To reduce the computational complexity, [45] proposed an efficient computation approach based on the mask operation in the attention mechanism, which is detailed in Fig. 6. It shifts some partited windows to the opposite positions and make a re-partition of 2D tensor. To aovid unreasonable connection of pixels in the original 2D tensor, the mask matrices are generated to block the relationship between non-connected pixels. This efficient operation ensures that the computational complexity remains consistent with that of the initial window partition. The masked attention mechanism is formulated as follow:

$$\text{Mask Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}, \mathbf{M}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_{\text{model}}}} + \mathbf{M}\right)\mathbf{V}$$
$$(23)$$

where $\mathbf{M} \in \mathbb{R}^{M^2 \times M^2}$ represents the mask matrix that is added to the scores resulting from $\mathbf{Q}\mathbf{K}^\top$. The masked element is set to a large negative value. The shifting process and mask attention mechanism are collectively referred to as Shifting Window multi-head Self Attention (SW-MSA). Based on W-MSA, SW-MSA and the 2D tensor input $\mathbf{X}_{\text{2D}}^j \quad j \in \{1, \ldots, k\}$, the entire Swin Transformer consists of two layers and is computed as:

$$\hat{\mathbf{X}}_{\text{2D}}^{j,1} = \text{W-MSA}(\text{LN}(\mathbf{X}_{\text{2D}}^j)) + \mathbf{X}_{\text{2D}}^j \qquad (24)$$

$$\mathbf{X}_{\text{2D}}^{j,1} = \text{MLP}(\text{LN}(\hat{\mathbf{X}}_{\text{2D}}^{j,1})) + \hat{\mathbf{X}}_{\text{2D}}^{j,1} \qquad (25)$$

$$\hat{\mathbf{X}}_{\text{2D}}^{j,2} = \text{SW-MSA}(\text{LN}(\mathbf{X}_{\text{2D}}^{j,1})) + \mathbf{X}_{\text{2D}}^{j,1} \qquad (26)$$

$$\mathbf{X}_{\text{2D}}^{j,2} = \text{MLP}(\text{LN}(\hat{\mathbf{X}}_{\text{2D}}^{j,2})) + \hat{\mathbf{X}}_{\text{2D}}^{j,2} \qquad (27)$$

where $\text{LN}(\cdot)$ and $\text{MLP}(\cdot)$ are layernorm and multi-layer peception, respectively. $\hat{\mathbf{X}}_{\text{2D}}^{j,1}$ and $\hat{\mathbf{X}}_{\text{2D}}^{j,2}$ are the output of local window attention. Similar to (21), the output of second layer $\mathbf{X}_{\text{2D}}^{j,2}$ can be transformed back to one-dimensional tensor.

E. Adaptive output aggregation

Based on the useful feature extraction of vision backbone, the formulation of multi-step ATP is converted into the feature extraction from the image-like tensor. Inspired by the well-developed theory of residual connection between different layers [46], the developed 2D block, are stacked and connected by the residual learning framework. If the *ArrivalNet* consists of L layers, for the lth $l \in \{1, \ldots, L\}$ layer, the residual connection is formulated as:

$$\mathbf{X}_{\text{1D}}^l = f_{\text{2D}}\left(\mathbf{X}_{\text{1D}}^{l-1}\right) + \mathbf{X}_{\text{1D}}^{l-1} \qquad (28)$$

Fig. 7: The residual connection of 2D blocks.

where $f_{\text{2D}}(\cdot)$ is the 2D block. The residual connection is illustrated in Fig. 7. The processed outputs $\mathbf{X}_{\text{2D}}^j \quad j \in \{1, \ldots, k\}$ need to be aggregated adaptively based on the normalized weight $\mathbf{A}_{\text{Top K}} = \{\mathbf{A}_{f_1}, \ldots, \mathbf{A}_{f_k}\}$ in periodic decomposition.

$$\hat{\mathbf{A}}_{\text{Top K}} = \text{Softmax}(\mathbf{A}_{\text{Top K}}) \qquad (29)$$

$$\mathbf{X}_{\text{1D}} = \sum_{j=1}^{k} \mathbf{A}_{f_j} \times \hat{\mathbf{X}}_{\text{1D}} \qquad (30)$$

where $\text{Softmax}(\cdot)$ is the normalization of weight. $\mathbf{X}_{\text{1D}} \in \mathbb{R}^{(N_p + N_f) \times d_{\text{model}}}$ is the output of 2D Block. And the estimation $\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{delay}}$ of the proposed *ArrivalNet* is obtained by aligning d_{model} back to one and cut the last N_f elements in the temporal dimension, which means only predicting delays.

$$\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{delay}} = \text{Trun}(\underset{d_{\text{model}} \to 1}{\text{Linear}}(\mathbf{X}_{\text{1D}})) \qquad (31)$$

where $\text{Linear}(\cdot)$ is the linear neural network to embed the feature space from d_{model} to 1. $\text{Trun}(\cdot)$ is the truncation for estimated $\hat{\mathbf{T}}_{i+1:i+N_f}^{\text{delay}}$.

IV. EXPERIMENTS

To validate the performance of *ArrivalNet* for bus/tram multi-step ATP, 125 days of public transport operational data in Dresden, Germany is collected. In this section, we describe the dataset, experimental settings (including evaluation metrics and comparison baselines), and results.

A. Dataset

1) Basic description of DVB data: The urban public transport system in Dresden, Germany, is operated by DVB (Dresdner Verkehrsbetriebe AG) and consists of two modes: tram and

(a) The joint distribution of tram. The average values of link travel distance and link travel time are 0.4702 km and 92.19 s, respectively.

(b) The joint distribution of tram. The average values of link travel distance and link travel time are 0.46 km and 80.79 s, respectively.

Fig. 8: The relationship between link travel distance and link travel time.

(a) The vehicle does not pass the trigger point between the current and next stop and opens the door.

(b) The vehicle passes the trigger point between the current and next stop and opens the door.

(c) The vehicle passes the stop without opening doors.

Fig. 9: Three conditions when the vehicle arrives at the stop.

Fig. 10: The selecting process of related traffic signal infrastructure.

bus. All operating public transport vehicles send their real-time status to the central data collection system at approximately 15 second intervals, which includes current time, current location, and distance from the previous stop within the route segment. In this work, only the status for the time of arrival is used. Additionally, the public transport system collects daily updated information on operating routes and stop locations, which may vary due to factors like road construction. In the constructed dataset, operational tram/bus data in 125 days from all bus and tram routes is collected. 4.97M valid sequences were extracted, where each sequence is the sequential status of a bus/tram at the stop from departure platform to the destination platform. The massive and useless information for the bus/tram on the link between platforms is removed. The average values of link travel distance and link travel time for tram are 0.4702 km and 92.19 s, respectively. For bus, these average characteristics are 0.46 km and 80.79 s. In Fig. 8, the relationship of link travel distance and link travel time for tram and bus is presented. It indicates a positive correlation between travel distance and travel time, where an increase in travel distance tends to lead to an increase in travel time. This, in turn, impacts the ATP. Furthermore, it demonstrates the rationality of using travel distance and time as one of the selected features in Section II.

2) The calculation of delays: In this work, the ATP is formulated as predicting the delay time at each future stop in (3) for a specific vehicle. In the data pre-processing, it is necessary to accurately capture the actual arrival time to calculate the real delay in seconds. For the collected data, the enabling information of the door opening operation can serve as an indicator of vehicle arrival time, which is event-triggered and not affected by the approximate 15-second transmission frequency. Specifically, vehicle arrivals at the stop can be categorized into three conditions: the vehicle does not pass the trigger point between the current and next stop & opens the door, the vehicle passes the trigger point between the current and next stop & opens the door, and the vehicle passes the stop without opening doors. The details and differences are illustrated in Fig. 9. With these three conditions, the arrival time is precisely captured, which is applied for the calculation of delay in (3).

3) The selection of contextual information: In this work, contextual information is incorporated into the *ArrivalNet*, which can improve the performance of the model. Whether a link connecting two stops passes through a traffic signal infrastructure is the dynamic contextual feature, which may change temporally in a sequence. The matching process between infrastructure and daily updated public transport routes in Dresden is illustrated in Fig. 10. The criterion for judgment is whether the distance between the traffic signal infrastructure's position and any waypoint in the link is less than a threshold. $\mathbf{F}^{\text{static}}$ consists of two static and contextual binary features: peak/off peak hour and weekday/weekend. The morning rush hour is set from 7:00 AM to 9:00 AM, and the evening rush hour is set from 4:00 PM to 7:00 PM on weekdays. The selection of related traffic signal infrastructure will be detailed in the experiments section.

TABLE I: The setting of parameters in *ArrivalNet*

Description	Notation	Value
learning rate	-	0.001
embedding feature length	d_{model}	16
number of 2D block	-	2
selected frequency	k	3
number of kernels in CNN	-	6
input series length	N_p	10
output series length	N_f	5, 10
local window size	-	2
threshold in related signal selection (m)	-	20

B. Experimental settings, metrics and baselines

In the training process, 90% samples ($\sim$4.47M sequences) in the dataset are randomly selected as the training set, and the left samples ($\sim$0.5M sequences) are the testing set. The mean square error (MSE) is chosen as the loss function of *ArrivalNet*. The details of parameter settings are presented in Table I. For *ArrivalNet* and all baseline algorithms, the learning rate is set as 0.001, which is fit for the convergence of deep neural network. Following the hyperparameters in [30], the length of embedded feature, the number of 2D block and the number of kernels in CNN are set as 16, 2 and 6, respectively. Considering that most of valid sequence length

is between 15 and 20, the input series length N_p is set as 10 and the output lengths N_f are 5 and 10. It can reflect the influence of sequence length to the model performance. The minimum length of the whole sequence is 15, which can be decomposed into 4 frequencies. Therefore, the 3 top frequencies are selected. The minimum width of the 2D tensor is 2, which is equal to the local window size. It can guarantee the effectivity of basic operation in the Swin Transformer. As for the threshold in related signal selection, it is the maximum distance between the location of signal infrastructure and the nearest waypoint on the link. If the distance to the nearest waypoint is less than the threshold, it is judged as the related signal. Experimentally, nearly all signal infrastructure can be captured and paired with the corresponding platform when the threshold is set to 20 m.

Three metrics are chosen for model performance evaluation, root mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE). Compared to MAE, RMSE is more sensitive to large errors in prediction by squaring the errors before averaging. MAPE is useful in expressing errors as a proportion of actual values. At stop i_k, the formulations of three metrics are detailed as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N_f} \sum_{t=i_k+1}^{i_k+N_f} (\mathbf{T}_{i_k+t}^{\text{arrival}} - \hat{\mathbf{T}}_{i_k+t}^{\text{arrival}})^2} \quad (32)$$

$$\text{MAE} = \frac{1}{N_f} \sum_{t=i_k+1}^{i_k+N_f} \left| \mathbf{T}_{i_k+t}^{\text{arrival}} - \hat{\mathbf{T}}_{i_k+t}^{\text{arrival}} \right| \quad (33)$$

$$\text{MAPE} = \frac{100}{N_f} \sum_{t=i_k+1}^{i_k+N_f} \left| \frac{\mathbf{T}_{i_k+t}^{\text{arrival}} - \hat{\mathbf{T}}_{i_k+t}^{\text{arrival}}}{\mathbf{T}_{i_k+t}^{\text{arrival}}} \right| \quad (34)$$

where N_f is number of future steps. i_k is the stop index of kth sequence. $\hat{\mathbf{T}}_{i_k+t}^{\text{arrival}}$ and $\mathbf{T}_{i_k+t}^{\text{arrival}}$ are estimation and groundtruth values of kth testing sequences at $(i_k+t)^{th}$ stop, respectively.

To validate the performance of *ArrivalNet*, a few baseline methods are implemented for comparison, including the traditional time series smoothing method, the RNN-based method, the attention-based method, and the CNN-based method. Meanwhile, three variants of *ArrivalNet* are designed for comparative study. An overview of each method is as follows:

- **Traditional time series smoothing method**. Autoregressive integrated moving average (ARIMA) is selected as the traditional smoothing method, which is a widely used statistical approach for time series forecasting [47]. It is a combination of the differenced autoregressive (AR) model with the moving average (MA) model. The AR part of ARIMA shows that the time series is regressed on its own past data. The integrated part is used to make the time series stationary by differencing the observations a certain number of times. The MA part indicates that the forecast error is a linear combination of past respective errors.
- **RNN-based method**. LSTM [41], as a typical recurrent network, is chosen for comparison. LSTM enhances the representation capability for one-dimensional temporal

TABLE II: The Comparative Results of Tram arrival Time Prediction. The average of each metric is the mean of 10→5 and 10→10.

Metric	Length	ARIMA	LSTM	Transformer	TCN	*ArrivalNet-1* (wo context)	*ArrivalNet-2* (Swin)	*ArrivalNet-3* (CNN)
RMSE (second)	10→ 5	64.3	60.4	49.8	49.2	47.9	<u>46.4</u>	**46.2**
	10→ 10	79.3	72.4	69.1	68.5	61.4	**56.8**	<u>57.4</u>
	Average	71.8	66.4	59.45	58.85	54.65	**51.6**	<u>51.8</u>
MAE (second)	10→ 5	55.2	49.9	38.8	42.5	34.2	<u>32.7</u>	**31.5**
	10→ 10	61.2	57.7	50.7	48.6	41.1	<u>38.6</u>	**37.4**
	Average	58.2	53.8	44.75	45.55	37.65	<u>35.65</u>	**34.45**
MAPE (%)	10→ 5	4.94	4.86	3.83	3.59	2.73	<u>2.54</u>	**2.39**
	10→ 10	6.71	5.12	4.32	3.93	3.15	<u>2.66</u>	**2.55**
	Average	5.825	4.99	4.075	3.76	2.94	<u>2.6</u>	**2.47**

The $1^{st}/2^{nd}$ best results are indicated in **bold**/<u>underline</u>.

TABLE III: The Comparative Results of Bus arrival Time Prediction. The average of each metric is the mean of 10→5 and 10→10.

Metric	Length	ARIMA	LSTM	Transformer	TCN	*ArrivalNet-1* (wo context)	*ArrivalNet-2* (Swin)	*ArrivalNet-3* (CNN)
RMSE (second)	10→5	66.1	63.7	53.1	52.8	49.3	<u>48.7</u>	**48.3**
	10→10	83.3	74.2	65.1	67.9	61.9	**57.3**	<u>58.3</u>
	Average	74.7	68.95	59.1	60.35	55.6	**53**	<u>53.3</u>
MAE (second)	10→5	53.8	49.5	39.2	41.4	36.1	<u>34.2</u>	**33.4**
	10→10	61.4	59.3	51.0	49.8	42.7	<u>38.5</u>	**38.3**
	Average	57.6	54.4	45.1	45.6	39.4	<u>36.35</u>	**35.85**
MAPE (%)	10→5	5.84	5.32	3.94	3.91	3.67	<u>3.42</u>	**3.35**
	10→10	8.73	6.30	4.27	4.39	3.87	**3.65**	<u>3.74</u>
	Average	7.285	5.81	4.105	4.15	3.77	**3.535**	<u>3.545</u>

The $1^{st}/2^{nd}$ best results are indicated in **bold**/<u>underline</u>.

(a) *ArrivalNet*.   (b) Transformer.

Fig. 11: Radar charts for tram and bus with different metrics and experimental settings.

processes by modeling both long and short-term memories. It has been widely applied in the prediction of public transport arrival times [8, 34].

- **Attention-based method**. Transformer is a well-developed, attention-based method and has been widely applied in natural language processing and trajectory prediction [37]. Due to the attention mechanism, correlations between non-adjacent time points in long temporal sequences can be captured.

- **CNN-based method**. Temporal convolutional network (TCN) is a typical time series method developed based on CNN [48]. It captures temporal correlations through convolutional kernels along the time dimension, which has also been widely used in sequential human action recognition and trajectory prediction [49].

- *ArrivalNet-1* is the *ArrivalNet* without considering the contextual features, which is equipped with CNN vision backbone.

(a) Tram case 1.

(b) Tram case 2.

Fig. 12: Two cases of tram ATP.

- **ArrivalNet-2** is the *ArrivalNet* with Swin Transformer vision backbone.
- **ArrivalNet-3** is the *ArrivalNet* with CNN vision backbone.

C. Results

1) Quantitative results: The results regarding the ATP of tram and bus are presented in Tables II and Table III. In each table, three metrics (RMSE, MSE, and MAPE) are shown for two different output lengths (10→5 and 10→10).

In Table II, three variants of proposed methods (*ArrivalNet-1*, *ArrivalNet-2* and *ArrivalNet-3*) outperform other baselines across all three metrics. Among three *ArrivalNet*, the absence of contextual features leads to a decrease in performance, while *ArrivalNet-2* or *ArrivalNet-3* achieve the best results in all six comparisons. In the tram ATP, compared to the four baseline methods (average of two lengths), the best-performing *ArrivalNet* model reduces RMSE by 28.26% (51.6 vs 71.8), 22.29% (51.6 vs 66.4), 13.20% (51.6 vs 59.45), 12.32% (51.6 vs 58.85), MAE by 40.81% (34.45 vs. 58.2), 35.97% (34.45

(a) Bus case 1.

(b) Bus case 2.

Fig. 13: Two cases of bus ATP.

vs. 53.8), 23.02% (34.45 vs. 44.75), 24.37% (34.45 vs. 45.55), and MAPE by 57.59% (2.47 vs. 5.825), 50.50% (2.47 vs. 4.99), 39.39% (2.47 vs. 4.075), 34.31% (2.47 vs. 3.76). In four baseline methods, for each metric, the performance of three deep learning methods (LSTM, Transformer, TCN) surpasses that of the autoregressive method (ARIMA), indicating that deep learning-based approaches have advantages in sequential prediction.

Similar to Table II, Table III presents the corresponding results for bus ATP. In the bus ATP, for the average of each

metric in two lengths, the best-performing *ArrivalNet* model reduces RMSE by 29.05% (53 vs. 74.7), 23.13% (53 vs. 68.95), 10.32% (53 vs. 59.1), 12.18% (53 vs. 60/35), MAE by 37.76% (35.85 vs. 57.6), 34.10% (35.85 vs. 54.4), 20.51% (35.85 vs. 45.1), 21.38% (35.85 vs. 45.6), MAPE by 51.48% (3.535 vs. 7.285), 39.16% (3.535 vs. 5.81), 13.89% (3.535 vs. 4.105), 14.82% (3.535 vs. 4.15). The findings from Table III are similar to those of Table II. Overall, Table II and Table III indicate that *ArrivalNet* achieves the best performance in both tram and bus ATP. To more clearly illustrate the performance

Fig. 14: The boxplot of MAE for each predicted stop. (vehicle type: tram, prediction length: 10→10)

comparison among all algorithms, Fig. 11 presents two radar charts for tram and bus across two length settings and three metrics. The radar charts also demonstrate that the proposed *ArrivalNet* consistently achieves the best performance across different metrics and experimental settings.

(a) *ArrivalNet*.

(b) Transformer.

Fig. 15: The relationship of between MAE and ground-truth delays (vehicle type: tram, prediction length: 10→10).

2) Case study: Fig. 12 and Fig. 13 present several cases of tram and bus ATP. Each case is a sequential arrival information of tram or bus with 20 stops. The labels of the x axis only represents the platform index of the corresponding sequence, it indicates that the same index in different cases doesn't mean the same physical public transport stop. In all cases, the length

of both input and output sequences are set to 10, corresponding to platforms 1-10 and 11-20, respectively. In all cases, the vehicle departs from the 10th platform, which is highlighted with a golden star. In each figure, the left one is the comparison of ground truth, predicted and scheduled arrival time, while the right one is the comparsion of ground truth, predicted delay. Considering that Transformer is the method with best performance in all one-dimensional algorithms overall, it is selected for the comparative study.

For all cases of trams and buses, the scheduled arrival time do not match the ground truth arrival time, indicating that discrepancies between actual arrival times and timetables are common. It emphasizes the significance of public transport ATP. Comparing the ground truth delay, predicted delay (*ArrivalNet*) and predicted delay (Transformer) across all cases, *ArrivalNet* is good at capturing the small variations in the sequence, such as rising, falling, fluctuation. For example, the platforms 13-15 in tram case 1, the platforms 14-18 in tram case 2, the platforms 18-20 in bus case 1 and the platforms 11-15 in bus case 2. On the contrary, Transformer cannot respond to these changes promptly, leading to only minor increases or decreases in the prediction of delay. It tends to generate an average of multi-step prediction.

3) Statistical analysis: Fig. 14 shows the MAE boxplot of all test samples (∼0.5M sequences) for different predicted platforms by *ArrivalNet-2* and Transformer. The input and output lengths set to 10. For both methods, MAE increases as the prediction length extends, which aligns with the general trend observed in time series prediction. *ArrivalNet* achieves a smaller MAE across all predicted platforms, including the mean and variance. Moreover, this difference is more pronounced for nearer platforms (predicted platforms 1, 2, 3).

To better understand the advantage of *ArrivalNet* in ATP, Fig. 15a illustrates the joint distribution between MAE and different delays of all test samples. In this figure, the horizontal axis represents the actual delay at each platform, while the vertical axis corresponds to MAE. The color of each grid represents the percentage of samples in the test set. Comparing the results of *ArrivalNet* and Transformer, MAE of *ArrivalNet* is predominantly concentrated in intervals with smaller values. As the value of delay increases, unlike *ArrivalNet*, MAE of Transformer is not concentrated in a smaller interval but tend to disperse over a broader range. This comparison suggests that the proposed *ArrivalNet* can maintain predicted errors within a smaller range.

4) The city-wide analysis: In this work, city-wide operational data is utilized. The delay in extracted sequences is the cumulative delay, which is influenced by the delay propagation along the route. To analyze the distribution of actual and predicted delays in city-wide public transport, the cumulative delay at different platforms along the same route is converted into the link delay, which represents the delay caused by tram/bus between two adjacent platforms. The transformation from cumulative to link delays is to calculate the difference of cumulative delay between adjacent platforms. As shown in Fig. 16, the comparison of actual and predicted delays shows that the predictions of *ArrivalNet* for trams are generally consistent with ground truth. For example, in the three pairs

-53.4   **delay time (s)**   **108**

Fig. 16: The city-wide tram link delay distributions (top: ground-truth link delay distribution, bottom: predicted link delay distribution from *ArrivalNet-3*).

of circle, the proposed *ArrivalNet* successfully capture the high (black), low (orange) and negative (blue) delays. It indicates that for tram/bus ATP (delay prediction), *ArrivalNet* can accurately capture the real city-wide delay distribution and pattern.

TABLE IV: The influence of negative delay (vehicle type: tram, prediction length: 10)

metric	w/ negative delay	w/o negative delay	error increase
RMSE	57.4	59.78	4.0%
MAE	37.4	39.1	4.55%
MAPE	2.55	2.84	11.37%

w/ negative delay: Same with previous experiments in Table. II and Table. III.
w/o negative delay: All negative delay in samples are set as 0.

Similarly, Fig. 17 presents the city-wide distribution of the ground truth and the average predicted link delays. Comparing the ground truth link delay between the tram and the bus at the top of Fig. 16 and Fig. 17, the overall delay of the bus is higher than that of the tram. In the comparison within Fig. 17, it indicates that *ArrivalNet* can well predict the city-wide link delay distribution with low positive values. However, it also

-60   **delay time (s)**   **100**

Fig. 17: The city-wide bus link delay distributions (top: ground-truth link delay distribution, bottom: predicted link delay distribution from *ArrivalNet-3*).

reveals that *ArrivalNet* cannot perfectly capture the actual delay modes in downtown areas, which is also highlighted by the black circle. This could be due to the presence of more traffic congestion in downtown areas. Another possible reason is that most of Dresden's tram links use dedicated rail lines that do not overlap with roads used by buses and private cars. Therefore, the arrival time prediction of trams is less affected by the uncertainty of artery traffic conditions. This makes it easier to predict than buses.

5) The influence of negative delay: In this study, the Dresden city-wide data has instances with negative delay, indicating that the vehicle arrives at the stop earlier than the scheduled

(a) The 2D heatmap.

(b) The 2D scatter distribution.

Fig. 18: The relationship of negative delay at the current stop and the delay at the next stop.

time. The causes of negative delay are varied, such as the absence of passengers boarding or alighting at specific stops, or bunching issues between consecutive vehicles. Typically, public transport drivers mitigate negative delays by increasing the dwell time at stops to prevent the propagation of the delay to subsequent stops.

Fig. 18 illustrates the relationship between delays at stops exhibiting negative delay and the delays at the subsequent stops. Specifically, Fig. 18a presents a 2D distribution of all samples in the dataset with negative delay, while Fig. 18 shows a joint plot combining scatter and 1-dimensional distributions. From these figures, it is evident that overall, vehicles tend to reduce or eliminate negative delay at the next stop. Approximately 30.31% of samples transition to a positive delay

at the subsequent stop. This indicates that public transport vehicles actively work to reduce negative delays to maintain alignment with the schedule. Ideally, negative delays would not be propagated to the next stop, thereby preventing any impact on arrival time predictions.

To further analyze the impact of negative delay on the performance of ATP, an additional experiment was conducted. For the tram ATP, all negative delays were set to zero during model training. The results of this experiment are presented in Table IV, where the length of the prediction step is 10. It shows that including negative delay helps the model learn a more accurate delay propagation process. Conversely, when negative delay is excluded, the model's performance slightly declines, though it does not result in a complete failure of the model. Future research may involve incorporating external knowledge to more precisely characterize the propagation of negative delay, thereby enhancing the training process of the deep learning model.

In addition to analyzing the city-wide link delay distribution of all collected data, this work also presents the variation in city-wide link delay at each hour of the day to investigate the performance of *ArrivalNet*'s estimation at the hour level. The data collected on Aug 4th, 2022 (Thursday) is selected to show the city-wide link delay in one day. Fig. 19 and Fig.20 present link delay distributions at different times for tram and bus, respectively. To simplify the description, three hours in one day are shown, which include 7 AM to 8 AM (morning rush hour), 3 PM to 4 PM (normal afternoon), and 6 PM to 7 PM (evening rush hour). In both figures, the first row is the city-wide ground truth link delay, and the second is for prediction by *ArrivalNet-3*. The average link delay below each map is the mean of all public transport links in Dresden. The comparison between different times in one day indicates that compared with non-peak hours, there is a higher link delay during peak hours. This finding applies to both trams and buses. It also shows that it is reasonable to use whether in the rush hour as a contextual feature in the problem formulation. Comparing the average link delay of ground truth and prediction, *ArrivalNet* can predict the trend of delay variation in one day. From the perspective of application, *ArrivalNet* can be used as a city-wide public transport delay monitor.

V. CONCLUSIONS

In this paper, we propose a two-dimensional temporal variation-based bus/tram ATP model, which is termed as *ArrivalNet*. With the FFT transformation, it decomposes one-dimensional time series into two-dimensional temporal variation, that represents both intra-period and inter-period changes. The useful temporal feature of the two-dimensional tensor can be effectively extracted by vision backbones. Moreover, drawing on the concept of ResNet [46], this two-dimensional temporal variation module can be defined as a basic module, allowing for flexible use and adjustment. Validated by a city-wide dataset from the public transport system in Dresden, *ArrivalNet* demonstrates superiority in multi-step bus/tram ATP compared to baseline methods. It can be used for traveler information systems and public transport management systems.

Fig. 19: The city-wide hour-level tram link delay distributions (top row: ground-truth link delay distribution, bottom row: predicted link delay distribution from *ArrivalNet-3*).

Fig. 20: The city-wide hour-level bus link delay distributions (top row: ground-truth link delay distribution, bottom row: predicted link delay distribution from *ArrivalNet-3*).

This study focuses on uncovering periodic regularities in temporal information but does not consider the inherent instability of the time series data itself. Future work will focus on mitigating the impact of non-stationary samples on sequential prediction capabilities.

References

[1] D. A. Hensher, "Sustainable public transport systems: Moving towards a value for money and network-based approach and away from blind commitment," *Transport Policy*, vol. 14, no. 1, pp. 98–102, 2007.

[2] R. Buehler and J. Pucher, "Making public transport financially sustainable," *Transport Policy*, vol. 18, no. 1, pp. 126–138, 2011.

[3] J. Soza-Parra, S. Raveau, J. C. Muñoz, and O. Cats, "The underlying effect of public transport reliability on users' satisfaction," *Transportation Research Part A: Policy and Practice*, vol. 126, pp. 83–93, 2019.

[4] B. Amberg, B. Amberg, and N. Kliewer, "Robust efficiency in urban public transportation: Minimizing delay propagation in cost-efficient bus and driver schedules," *Transportation Science*, vol. 53, no. 1, pp. 89–112, 2019.

[5] T. C. Lam and K. A. Small, "The value of time and reliability: measurement from a value pricing experiment," *Transportation Research Part E: Logistics and Transportation Review*, vol. 37, no. 2-3, pp. 231–251, 2001.

[6] Q. Cheng, Y. Lin, X. S. Zhou, and Z. Liu, "Analytical formulation for explaining the variations in traffic states: A fundamental diagram modeling perspective with stochastic parameters," *European Journal of Operational Research*, vol. 312, no. 1, pp. 182–197, 2024.

[7] X. S. Zhou, Q. Cheng, X. Wu, P. Li, B. Belezamo, J. Lu, and M. Abbasi, "A meso-to-macro cross-resolution performance approach for connecting polynomial arrival queue model to volume-delay function with inflow demand-to-capacity ratio," *Multimodal Transportation*, vol. 1, no. 2, p. 100017, 2022.

[8] S. Ratneswaran and U. Thayasivam, "An improved bus travel time prediction using multi-model ensemble approach for heterogeneous traffic conditions," in *2023 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2023.

[9] J. Ma, J. Chan, G. Ristanoski, S. Rajasegarar, and C. Leckie, "Bus travel time prediction with real-time traffic information," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 536–549, 2019.

[10] T. Kim, J. Lu, R. M. Pendyala, and X. S. Zhou, "Computational graph-based mathematical programming reformulation for integrated demand and supply models," *Transportation Research Part C: Emerging Technologies*, vol. 164, p. 104671, 2024.

[11] P. Guarda, M. Battifarano, and S. Qian, "Estimating network flow and travel behavior using day-to-day system-level data: A computational graph approach," *Transportation Research Part C: Emerging Technologies*, vol. 158, p. 104409, 2024.

[12] B. Büchel and F. Corman, "Review on statistical modeling of travel time variability for road-based public transport," *Frontiers in Built Environment*, vol. 6, p. 70, 2020.

[13] N. Singh and K. Kumar, "A review of bus arrival time prediction using artificial intelligence," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 12, no. 4, p. e1457, 2022.

[14] G. Welch, G. Bishop *et al.*, *An introduction to the Kalman filter*. Chapel Hill, NC, USA, 1995.

[15] D. Heckerman, "A tutorial on learning with bayesian networks," *Innovations in Bayesian networks: Theory and applications*, pp. 33–82, 2008.

[16] A. Achar, D. Bharathi, B. A. Kumar, and L. Vanajakshi, "Bus arrival time prediction: A spatial kalman filter approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1298–1307, 2020.

[17] F. Cathey and D. J. Dailey, "A prescription for transit arrival/departure prediction using automatic vehicle location data," *Transportation Research Part C: Emerging Technologies*, vol. 11, no. 3-4, pp. 241–264, 2003.

[18] I. K. Isukapati, C. Igoe, E. Bronstein, V. Parimi, and S. F. Smith, "Hierarchical bayesian framework for bus dwell time prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 3068–3077, 2020.

[19] X. Chen, Z. Cheng, J. G. Jin, M. Trépanier, and L. Sun, "Probabilistic forecasting of bus travel time with a bayesian gaussian mixture model," *Transportation Science*, vol. 57, no. 6, pp. 1516–1535, 2023.

[20] X. Chen, Z. Cheng, A. M. Schmidt, and L. Sun, "Conditional forecasting of bus travel time and passenger occupancy with bayesian markov regime-switching vector autoregression," *arXiv preprint arXiv:2401.17387*, 2024.

[21] M. Yang, C. Chen, L. Wang, X. Yan, and L. Zhou, "Bus arrival time prediction using support vector machine with genetic algorithm," *Neural Network World*, vol. 26, no. 3, p. 205, 2016.

[22] T. Liu, J. Ma, W. Guan, Y. Song, and H. Niu, "Bus arrival time prediction based on the k-nearest neighbor method," in *2012 Fifth International Joint Conference on Computational Sciences and Optimization*. IEEE, 2012, pp. 480–483.

[23] B. Yu, W. H. Lam, and M. L. Tam, "Bus arrival time prediction at bus stop with multiple routes," *Transportation Research Part C: Emerging Technologies*, vol. 19, no. 6, pp. 1157–1170, 2011.

[24] B. A. Kumar, L. Vanajakshi, and S. C. Subramanian, "A hybrid model based method for bus travel time estimation," *Journal of Intelligent Transportation Systems*, vol. 22, no. 5, pp. 390–406, 2018.

[25] Y. Liu, H. Zhang, J. Jia, B. Shi, and W. Wang, "Understanding urban bus travel time: Statistical analysis and a deep learning prediction," *International Journal of Modern Physics B*, vol. 37, no. 04, p. 2350034, 2023.

[26] P. He, G. Jiang, S.-K. Lam, and Y. Sun, "Learning heterogeneous traffic patterns for travel time prediction of bus journeys," *Information Sciences*, vol. 512, pp. 1394–1406, 2020.

[27] C. Li, S. Ling, H. Zhang, H. Zhao, L. Liu, and N. Jia, "A sequence and network embedding method for bus arrival time prediction using gps trajectory data only," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[28] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in Neural Information Processing Systems*, vol. 34, pp. 22 419–22 430, 2021.

[29] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International Conference on Machine Learning*. PMLR, 2022, pp. 27 268–27 286.

[30] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," in *The eleventh international conference on learning representations*, 2022.

[31] X. Li, A. Cottam, and Y.-J. Wu, "Transit arrival time prediction using interaction networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3833–3844, 2023.

[32] B. Büchel and F. Corman, "What do we know when? modeling predictability of transit operations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 15 684–15 695, 2022.

[33] J. Ma, J. Chan, S. Rajasegarar, and C. Leckie, "Multi-attention graph neural networks for city-wide bus travel time estimation using limited data," *Expert Systems with Applications*, vol. 202, p. 117057, 2022.

[34] Y. Rong, Z. Xu, J. Liu, H. Liu, J. Ding, X. Liu, W. Luo, C. Zhang, and J. Gao, "Du-bus: A realtime bus waiting time estimation system based on multi-source data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 524–24 539, 2022.

[35] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *International Conference on Learning Representations*, 2021.

[36] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting,"

Advances in Neural Information Processing Systems, vol. 35, pp. 9881–9893, 2022.

[37] 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.

[38] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.

[39] L. H. Koopmans, *The spectral analysis of time series*. Elsevier, 1995.

[40] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] H. J. Nussbaumer and H. J. Nussbaumer, *The fast Fourier transform*. Springer, 1982.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[44] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.

[45] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[47] G. E. Box and D. A. Pierce, "Distribution of residual auto-correlations in autoregressive-integrated moving average time series models," *Journal of the American statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

[48] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[49] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 14 424–14 432.

Patrick Wolf received the diploma from the Dresden University of Technology (TU Dresden). Dresden, Germany, in 2018, where he is currently pursuing a Ph.D. degree in traffic engineering. From January 2019 to March 2024, he was a researcher in the Chair of Traffic Process Automation at the Faculty of Transportation and Traffic Sciences "Friedrich List" of the Dresden University of Technology (TU Dresden). From April 2024, he is with the traffic management department of the local transportation company (Dresdner Verkehrsbetriebe AG). His research focuses on operational management in public transport, traffic light control and motion forecast of public transport vehicles.

Meng Wang received the PhD degree from TU Delft in 2014. He worked as a PostDoc Researcher (2014-2015) at the Faculty of Mechanical Engineering, TU Delft, and as an Assistant Professor (2015-2021) at the Department of Transport and Planning, TU Delft. Since 2021, he has been a Full Professor and Head of the Chair of Traffic Process Automation, "Friedrich List" Faculty of Transport and Traffic Sciences, TU Dresden. His main research interests are control design and impact assessment of Cooperative Intelligent Transportation Systems. He is an Associate Editor of IEEE Transactions on Intelligent Transportation Systems and Transportmetrica B.

Zirui Li received the B.S. degree from the Beijing Institute of Technology (BIT), Beijing, China, in 2019, where he is currently pursuing the Ph.D. degree in mechanical engineering. From June, 2021 to July, 2022, he was a visiting researcher in Delft University of Technology (TU Delft). From Aug, 2022. He was the visiting researcher in the Chair of Traffic Process Automation at the Faculty of Transportation and Traffic Sciences "Friedrich List" of the TU Dresden. His research focuses on interactive behavior modeling, risk assessment and motion planning of automated vehicles.