Fine-Grained Service-Level Passenger Flow Prediction for Bus Transit Systems Based on Multitask Deep Learning

Dan Luo, Dong Zhao, *Member, IEEE*, Qixue Ke, Xiaoyong You, Liang Liu, *Member, IEEE*, Desheng Zhang, *Member, IEEE*, Huadong Ma, *Senior Member, IEEE*, and Xingquan Zuo, *Senior Member, IEEE*

***Abstract*— Bus services play a crucial role in urban transit. It is significant to achieve the fine-grained service-level passenger flow prediction (SPFP), namely to predict the total number of passengers for each service of each bus line passing through each station during the next short-term interval. However, it faces great challenges due to complex factors including inter-station and inter-line spatial dependencies, intra-station and inter-service temporal dependencies, and internal/external influences. To address these challenges, we propose a multitask deep-learning (MDL) approach, called *MDL-SPFP*, to jointly predict the arriving bus service flow, line-level on-board passenger flow and line-level boarding/alighting passenger flow by leveraging well-designed deep neural networks called *ARM*. The MDL framework can mutually reinforce the prediction of each type of flow, and finally integrate the outputs to achieve the fine-grained service-level prediction. The ARM network combines three modules, Attention mechanism, Residual block and Multi-scale convolution, to well capture various complex non-linear spatio-temporal dependencies and influence factors. Extensive experiments based on a large-scale realistic bus operation dataset are conducted to confirm that our MDL-SPFP approach outperforms 10 state-of-the-art baselines, and improves 22.39% accuracy than the best baseline.**

***Index Terms*— Traffic passenger flows prediction, bus transit systems, multitask learning, deep learning.**

I. Introduction

BUS services are the most common way to move people over short and medium distances in almost every city around the world. It is reported that buses account for about 55% of public transport in Europe, 751 million passenger trips annually in the USA, and 250 million passenger trips per day in China [1], [2]. To improve the service quality of Bus Transit Systems (BTS), it is of great significance to accurately predict passenger flows for each running bus. From the bus operators' perspective, it can assist in optimizing resource scheduling, e.g., dynamically adjusting the service frequency and vehicle size in fine granularity. From the passengers' perspective, it can help to obtain a comfortable travel experience by avoiding crowded lines. From the government's perspective, it is useful for risk assessment and guaranteeing public safety.

In this paper, we focus on the fine-grained Service-level Passenger Flow Prediction (SPFP) for BTS. Specifically, many different bus services (i.e., individual round trips) belonging to the same bus line always depart from the starting station one by one at different times in one day and pass through many stations of the line. Our objective is to predict the total number of on-board passengers for each service of each bus line passing through each station during the next short-term interval (e.g., 5 minutes) in fine granularity. Please note that it is different from the station-level passenger flow prediction, which concerns the total number of on-board passengers passing through each station during a time interval, while not distinguishing passengers from different bus lines; it is also different from the line-level passenger flow prediction, which concerns the total number of on-board passengers passing through each station/region from each individual bus line during a time interval, while not distinguishing passengers from different bus services that may belong to the same bus line.

The fine-grained SPFP is the key to realize transit crowdedness predictions for individual bus services, based on which Google Maps introduced a new crowdedness prediction service in June 2019 [3]. Nevertheless, the Google's service relies heavily on crowdsourcing data, i.e., asking Map users to share information about their experience and count how many seats are available or if riders have to stand. Nowadays, with the widespread use of Automatic Fare Collection (AFC) devices that can record passengers' payments using smart cards, and GPS embedded On Board Units (OBUs) that can track the bus, it offers us a new and promising alternative way to achieve this goal. However, we still face great

Manuscript received December 1, 2019; revised March 18, 2020; accepted June 3, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2018AAA0101201, in part by the National Natural Science Foundation of China (NSFC) under Grant 61972044, Grant 61722201, and Grant 61632008, in part by the Funds for International Cooperation and Exchange of NSFC under Grant 61720106007, and in part by the 111 Project under Grant B18008. The Associate Editor for this article was J. E. Naranjo. *(Corresponding author: Dong Zhao.)*

Dan Luo, Dong Zhao, Xiaoyong You, Liang Liu, and Huadong Ma are with the Beijing Key Laboratory of Intelligent Telecommunication Software and Multimedia, School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: dzhao@bupt.edu.cn).

Qixue Ke is with the School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China.

Desheng Zhang is with the Department of Computer Science, Rutgers University, New Brunswick, NJ 08901 USA.

Xingquan Zuo is with the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China, and also with the Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, Beijing 100876, China.

Digital Object Identifier 10.1109/TITS.2020.3002772

challenges, as the fine-grained SPFP is affected by many complex factors:

- **Spatial dependencies.** It contains not only *inter-station* dependencies similar to *inter-region* dependencies in the traditional traffic flow prediction [4], but also unique and complex *inter-line* dependencies. The inter-station dependencies exist not only between adjacent stations, but also between stations that are far away from each other (e.g., between home and work place, or between one tourist attraction and another). The inter-line dependencies are affected by the transfers between two lines, which obviously reflect passenger flows from one line to another.
- **Temporal dependencies.** The current passenger flows are always correlated with the historical flows [4], which essentially belongs to the *intra-station* temporal dependencies. Meanwhile, the *inter-service* temporal dependencies exist in different services of the same line. For example, two successive services may have similar passenger flows.
- **Internal and external influences.** Besides some *external* factors, such as weather conditions and events [4], some *internal* features of bus lines and stations, including the accessibility, connectivity, and surrounding functional zones, also have significant influences on passenger flows.

Therefore, it is necessary to design a novel fine-grained model to simultaneously capture complex spatio-temporal dependencies and internal/external influences from line-level and station-level to service-level. Most existing studies on passenger flow prediction [5]–[11] are based on traditional time series models or machine learning models, failing to capture such complex non-linear dependencies and influences. Several most recent studies [12], [13] have exploited deep learning models to capture complex non-linear dependencies and influences for BTS, but are still limited to the line-level [12] or station-level [13] instead of service-level.

To the best of our knowledge, this is the first work that investigates the fine-grained SPFP using the off-the-shelf bus operation data based on deep learning. To address the above challenges, we mainly make the following contributions:

- We identify diverse spatiotemporal dependencies, and heterogeneous flow dependencies among arriving bus service flow, line-level on-board passenger flow, line-level boarding/alighting passenger flow and inter-line transfer passenger flow (see Table I). A multitask deep learning (MDL) approach, *MDL-SPFP*, is proposed to simultaneously predict different types of flows by adequately capturing our identified dependencies. Finally, we integrate the outputs of arriving bus service flow and line-level on-board passenger flow predictions to achieve the fine-grained SPFP.
- We propose a novel deep neural network (DNN) model called *ARM* in the MDL framework, which consists of three modules: (i) *Attention mechanism* to automatically capture different weights of temporal influences, (ii) *Residual block* to efficiently capture both near and far inter-station spatial dependencies, and (iii) *Multi-scale convolution* to further enhance the learning ability of spatial dependencies. Moreover, we integrate the inter-line transfer passenger flow to improve the prediction performance of the line-level boarding/alighting passenger flow by leveraging the ARM network, which can capture the inter-line spatial dependencies very well.
- We evaluate our MDL-SPFP approach using over 16.6-million AFC transaction records and the corresponding over 9.9-million bus arrival time records collected during 3 months in Jinan City, China. The results verify that our MDL-SPFP achieves the best prediction performance compared to 10 baselines, and improves 22.39% accuracy than the best baseline. We have released the code and data for public use.[1]

II. Motivation

In this section, we explain why it is significant to capture the dependencies among three types of flows (i.e., arriving bus service flow, line-level on-board passenger flow, and line-level B/A passenger flow), which is also our main motivation to design a MDL approach.

To achieve SPFP, one direct method is to use the service-level historical passenger flow to predict the number of on-board passengers during the next time interval, as achieved in [11]. However, this method loses much information from the inter-station spatial dependencies and inter-line transfers. Another alternative method is to first predict the line-level on-board passenger flow, which may aggregate passenger flows from multiple services during a time interval, as achieved in [12]. Generally, different services belonging to the same bus line always depart from the starting station one by one at different times. If each service arrives at a certain station on time and we could achieve sufficiently short-term prediction so that there is only one service during that interval, then we could directly get the service-level passenger flow. However, it is very common to see that multiple bus services belonging to the same line arrive at the same station almost at the same time due to various uncertain factors such as traffic congestions and accidents (see Sect. III-B), not to mention the difficulty of achieving ultra-short-term prediction. Thus, it is very difficult to accurately predict how many passengers will get on different bus services belonging to the same line at a specific time. To cope with this difficulty, **it is necessary to simultaneously model the arriving bus service flow, namely to predict the number of bus services arriving at each station during each time interval.**

On the other hand, the on-board passenger flow can only be derived from the boarding/alighting (B/A) pairs of passengers (i.e., AFC transaction records). We can easily extract the line-level B/A passenger flow. However, it is impossible to infer the number of on-board passengers during a specific time interval directly from the historical line-level B/A passenger flow, as there are different numbers of bus services arriving at various stations during different time intervals. Let us take Fig. 1 as an example: at time t_3, the line-level B/A passenger

[1] https://github.com/DanLuo-work/keras-MDL-SPFP

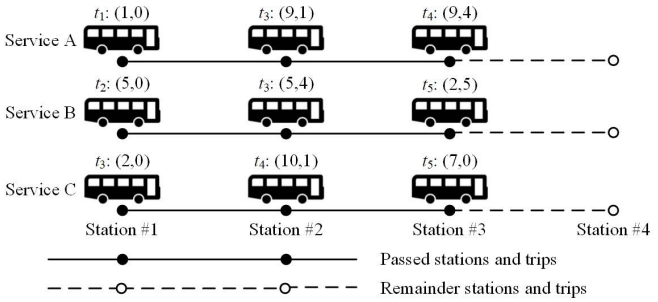


Fig. 1. Illustration of three bus services belonging to the same bus line, which depart from the starting station in the order (A, B, C). The information above each bus, $t : (n_1, n_2)$, means that the bus arrives at a station during the time interval t, and the numbers of boarding and alighting passengers are n_1 and n_2 respectively.

TABLE I
VARIOUS DEPENDENCIES REQUIRED TO CAPTURE

Perspectives	Categories of dependencies
Spatial	Inter-station dependencies (Fig. 2), Inter-line dependencies (Fig. 3)
Temporal	Intra-station dependencies (Fig. 4), Inter-service dependencies (Figs. 5, 6)
Flows	Dependencies between on-board passenger flow and bus service flow (Fig. 7) Dependencies between on-board passenger flow and B/A passenger flow (Fig. 8) Dependencies between B/A passenger flow and transfer passenger flow (Fig. 9)

flow at station 2 is $(9+5, 1+4) = (14, 5)$; at time t_5, the line-level B/A passenger flow at station 3 is $(2 + 7, 5 + 0) = (9, 5)$; while at time t_5, as service C catches up with B, the line-level on-board passenger flow at station 3 includes passengers from both services B and C, and is calculated as $(5+5-4+2-5)+(2+10-1+7) = 21$, which is different from the sum of line-level B/A passenger flows. Thus, it implies that **the line-level on-board passenger flow and the line-level B/A passenger flow are complementary to each other, both of which are needed to be modeled explicitly.**

Different types of flows contain complex dependencies. Intuitively, the on-board passenger flow should coincide with the bus service flow, since more bus services naturally bring in more passengers. Second, the B/A passenger flows have direct influences on the on-board passenger flow, since the on-board passenger flow is obtained according to a set of B/A pairs. Third, the transfer passenger flow has correlations with the B/A passenger flow, since, in essence, the former is a part of the latter. **In summary, the key to achieve find-grained SPFP is to well capture various complex spatio-temporal dependencies as listed in Table I.**

III. DATA DESCRIPTION AND PRELIMINARY ANALYSIS

A. Dataset

This work is based on a large-scale bus operation dataset collected from approximately 3 million smart card holders and 327141 bus services in Jinan City, China from November 2018 to January 2019. This dataset cosists of two parts:

TABLE II
SOME IMPORTANT STATISTICS FOR OUR USED DATA

Jinan bus operation dataset	
AFC transactional records	16595274 records
Bus arrival time records	9929015 records
Time span	11/1/2018 - 1/31/2019
Road network	11 lines and 267 stations (the longest line has 81 stations)
Other data for extracting internal/external features	
# of holidays	27 days (e.g., weekends, festivals)
Weather	4 types (sunny, cloudy, rainy, snowy)
Temperature /°C	[-10, 21]
Points of Interest	143095 records of 8 types

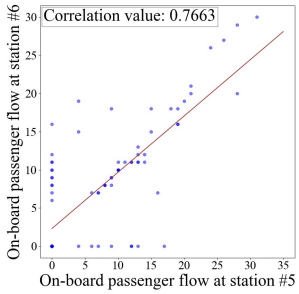


Fig. 2. Inter-station spatial dependencies between on-board passenger flows at adjacent stations from one bus line during $12:00-12:05$ for 3 months.

over 16.6-million AFC transaction records, and the corresponding over 9.9-million bus arrival time records. AFC transaction records include items of card ID, line ID, boarding station, boarding time, orientation (i.e., two opposite directions of the same bus line) and trip ID (representing different bus services). Bus arrival time records include items of trip ID, arrival station with its longitude and latitude, and arrival time. In addition, several other open data sources are used in this study for extracting internal/external features: holiday events, weather, temperature, road networks and points of interest (PoI). Some important statistics for our used data are shown in Table II.

B. Preliminary Data Analyses

Before formally introduce our problem and approach, it is significant to first conduct preliminary data analyses for directly verifying various spatio-temporal dependencies and heterogeneous flow dependencies. We use Pearson correlation analysis to evaluate various dependencies. The time interval length is 5 minutes by default.

1) Spatio-Temporal Dependencies:

a) Inter-station spatial dependencies: Intuitively, the passenger flows from adjacent bus stations should have strong spatial correlation. We analyze on-board passenger flows in a pair of adjacent stations from one bus line in a certain time interval for three months, which presents a strong spatial correlation, as shown in Fig. 2.

b) Inter-line spatial dependencies: Different bus lines may meet at the same station in BTS, which provides chances for passenger transfer. In general, the more lines a certain bus station contains, the more urban settlements exist around this

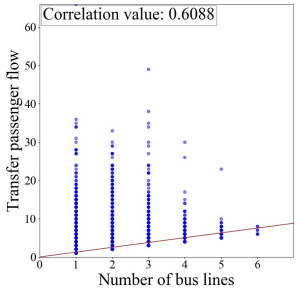


Fig. 3. Inter-line spatial dependencies between the transfer passenger flow and the number of bus lines at all individual stations during all of time intervals for 3 months.

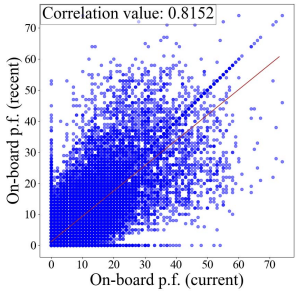


Fig. 4. Intra-station recent temporal dependencies for on-board passenger flows at one node (p.f. is short for "passenger flow").

station, and the more passengers will be attracted to transfer at this station. After data preprocessing, we observe that there are over 600-thousand transfer records in the dataset. The average number of daily travel records is approximately 96 thousands, and the number of transfer records is nearly 7 thousands, i.e., over 14.7% of all travel records. We plot the relationship of the transfer passenger flow and the number of bus lines at all individual stations during all of time intervals for 3 months in Fig. 3, which presents an obvious correlation.

c) Intra-station temporal dependencies: We analyze intra-station temporal dependencies with different types (*recent*, *daily-periodic* and *weekly-periodic*) of time fragments, respectively. Take *recent* type as an example. The horizontal axe of Fig. 4 represents the on-board passenger flow at a certain node (i.e., a certain station of a certain bus line) during each time interval for 3 months, and the vertical axe represents that during each of the last 3 time intervals, which shows a strong correlation value, 0.8193. Meanwhile, both the daily-periodic and weekly-periodic types show a strong correlation, although with diminishing correlation values, 0.7647 and 0.7409, respectively.

d) Inter-service temporal dependencies: BTS are usually affected by a variety of factors (e.g., weather conditions, traffic accidents), resulting in random bus arrival times. Figure 5 shows the standard deviations of arrival times for one bus line at a certain station during different time intervals in all of days for 3 months. It is observed that the arrival times vary greatly, and the maximum standard deviation can reach 137 seconds. Table III shows the occurrence number and percentage for different numbers of bus services belonging to the same line arriving at the same station during the same time interval for

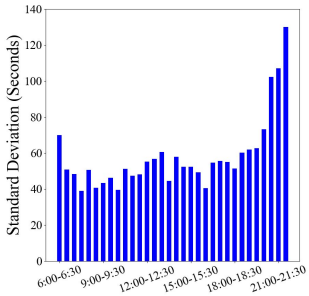


Fig. 5. The standard deviation of bus services' arrival times in all of days for 3 months.

TABLE III
DISTRIBUTION OF BUS SERVICES

Num. of bus services during a time interval	0	1	2	3	≥ 4
Occurrence number	9430788	5512627	1299042	241132	49915
Percentage	57.04%	33.34%	7.86%	1.46%	0.30%

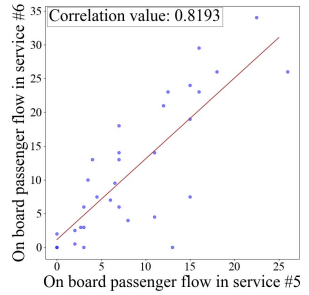


Fig. 6. Inter-service temporal dependencies in successive services #5 and #6 in one day.

3 months. It is observed that there are nearly 10% cases where more than two bus services arrive at the same station during the same interval. In fact, at most 9 bus services arrive at the same station during the same time interval in Jinan dataset. It implies a great challenge but significance for accurately predicting the bus service flow during a short-term interval. We further analyze the on-board passenger flow for two successive bus services of one bus line arriving at the same station in one day, which presents a strong correlation, as shown in Fig. 6.

2) Heterogenous Flow Dependencies:

a) Dependencies between on-board passenger flow and bus service flow: It is well known that, individual passenger always moves with much randomness, and thus the passenger flows during the same periods of different days may vary very dynamically. Nevertheless, the on-board passenger flow heavily depends on the bus service flow. Figure 7 shows the relationship between the on-board passenger flow and the bus service flow at all individual stations during all of time intervals for 3 months, which presents a strong correlation.

b) Dependencies between on-board passenger flow and B/A passenger flows: We randomly select a node (i.e., a certain station of a certain bus line) and analyze the relationships between the on-board passenger flow and B/A passenger flows at this node during all of time intervals for 3 months.

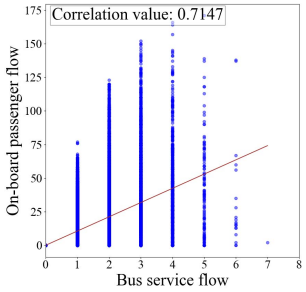


Fig. 7. Relationship between the on-board passenger flow and the bus service flow at all individual stations during all of time intervals for 3 months.

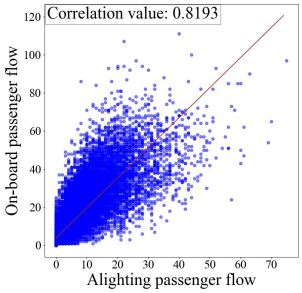


Fig. 8. Relationship between the on-board passenger flow and the alighting passenger flow at one node during all of time intervals for 3 months.

They present strong correlations, as shown in Fig. 8 (only alighting flow is presented due to similar results for boarding flow).

c) Dependencies between B/A passenger flows and transfer passenger flow: We randomly select a node and analyze the relationships between the B/A passenger flows and transfer flow at this node during all of time intervals for 3 months. As an example, Fig. 9 plots the alighting passenger flow and the number of transfer passengers alighting from this node and boarding to other lines, presenting strong correlation.

C. Trajectory Estimation

The AFC transaction data in the Jinan bus operation dataset only provides boarding records (corresponding to the starting station of a passenger's trajectory) but without alighting records. In fact, the lack of alighting records is very common in urban BTS with flat rate fare. To obtain passengers' complete trajectories, we utilize the boarding data to estimate the alighting station for different passengers using the probabilistic estimation method [11]. Considering that the alighting station from regular historical trajectories always produce a completely opposite orientation with boarding station, we perform an additional operation before using the probabilistic model, namely using passengers' next start stations to estimate their alighting stations.

IV. Problem Formulation

In this section, we formally define several important concepts and various types of flows, and formulate the SPFP problem.

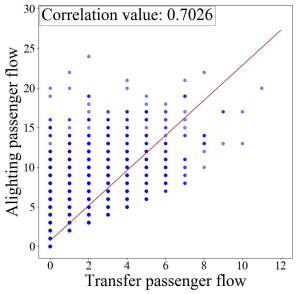
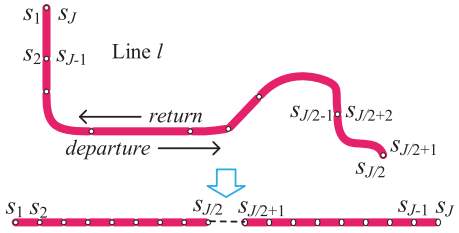
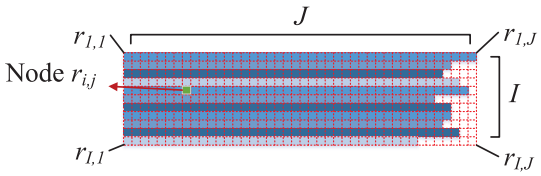


Fig. 9. Relationship between the alighting passenger flow and the transfer passenger flow at one node during all of time intervals for 3 months.



(a) A bus line is stretched, and stations from two directions are arranged in order.



(b) A set of nodes

Fig. 10. Illustration of converting a bus route network to a set of nodes.

A. Concepts and Models

Definition 1 (Node V): Given a bus route network with I bus lines and at most J stations for each line, all lines are stretched and merged to form a set of nodes, denoted by $V = \{r_{ij}\}, 1 \le i \le I, 1 \le j \le J$, each of which is arranged in order and represents the j-th station of the i-th line. Note that all the stations in one direction are always placed after all the stations in the opposite directions of the same line, in a reverse order, as shown in Fig. 10. For the line with less than J stations in reality, we add virtual nodes to keep consistent with the longest line.

Let (τ, l, s, φ) be a spatio-temporal point, of which τ denotes a timestamp, (l, s) denotes the station s of the bus line l, and φ denotes a specific bus service belonging to line l. Let $ar = (\tau_{ar}, l_{ar}, s_{ar}, \varphi_{ar})$ denote a bus arrival time record, i.e., a bus, which belongs to the service φ_{ar} of line l_{ar}, arrives at station s_{ar} at time τ_{ar}. Let $\mathbb{A}$ be all bus arrival time records, and Φ be all bus services appearing in $\mathbb{A}$. For convenience, we use $(l, s) \in r_{ij}$ to represent that the station s of the bus line l lies within the node r_{ij}, and $\tau \in t$ to represent that the timestamp τ is in the time interval t.

Definition 2 (Arriving Bus Service Sets and Flows $\boldsymbol{X}$): Given a set of bus arrival time records $\mathbb{A}$ and a sequence of time intervals $\mathcal{T}$, the set of bus services arriving at node r_{ij}

during the interval t is defined as

$$\Phi_{t,i,j} = \{\varphi_{ar} \in \Phi : (l_{ar}, s_{ar}) \in r_{ij} \wedge \tau_{ar} \in t\}, \tag{1}$$

and the corresponding number of bus services is defined as

$$\boldsymbol{X}_t(i, j) = |\Phi_{t,i,j}|, \tag{2}$$

where $\boldsymbol{X}_t(:, :)$ means arriving bus service flow matrix.

The movement of a passenger can be recorded as a time-ordered spatial trajectory, among which the *boarding* point and *alighting* point (i.e., B/A pair), denoted by $p_b = (\tau_{p_b}, l_p, s_{p_b}, \varphi_p)$ and $p_a = (\tau_{p_a}, l_p, s_{p_a}, \varphi_p)$, respectively. Let $\mathbb{P}$ be all B/A (i.e., (p_b, p_a)) pairs. From these B/A pairs together with bus arrival time records, we can model the passenger flows from different perspectives.

Definition 3 (B/A Passenger Flows $\boldsymbol{Y}$): Given a set of B/A pairs $\mathbb{P}$ and a sequence of time intervals $\mathcal{T}$, the service-level boarding and alighting (B/A) passenger flows belonging to service φ_k at node r_{ij} during the interval t are defined respectively as

$$\begin{aligned}\boldsymbol{Y}^S_{t,k}(1, i, j) = |\big\{&(p_b, p_a) \in \mathbb{P} : (l_p, s_{p_b}) \in r_{ij} \\ &\wedge \varphi_p = \varphi_k \wedge \tau_{p_b} \in t\big\}|,\end{aligned} \tag{3}$$

$$\begin{aligned}\boldsymbol{Y}^S_{t,k}(0, i, j) = |\big\{&(p_b, p_a) \in \mathbb{P} : (l_p, s_{p_a}) \in r_{ij} \\ &\wedge \varphi_p = \varphi_k \wedge \tau_{p_a} \in t\big\}|,\end{aligned} \tag{4}$$

where $\forall \varphi_k \in \Phi_{t,i,j}$, and $\boldsymbol{Y}^S_{t,k}(1, :, :)$ and $\boldsymbol{Y}^S_{t,k}(0, :, :)$ mean service-level B/A passenger flow matrices, respectively. Furthermore, the line-level B/A passenger flows at node r_{ij} during the interval t are defined respectively as

$$\boldsymbol{Y}^L_t(1, i, j) = \sum_{\forall \varphi_k \in \Phi_{t,i,j}} \boldsymbol{Y}^S_{t,k}(1, i, j), \tag{5}$$

$$\boldsymbol{Y}^L_t(0, i, j) = \sum_{\forall \varphi_k \in \Phi_{t,i,j}} \boldsymbol{Y}^S_{t,k}(0, i, j), \tag{6}$$

where $\boldsymbol{Y}^L_t(1, :, :)$ and $\boldsymbol{Y}^L_t(0, :, :)$ mean line-level B/A passenger flow matrices, respectively.

Definition 4 (On-board Passenger Flows $\boldsymbol{Z}$): Given a set of boarding-alighting pairs $\mathbb{P}$ and a sequence of time intervals $\mathcal{T}$, the service-level on-board passenger flow at node r_{ij} during the interval t are defined as

$$\boldsymbol{Z}^S_{t,k}(i, j) = \sum_{t'=t_1}^{t} (\boldsymbol{Y}^S_{t',k}(1, i, j) - \boldsymbol{Y}^S_{t',k}(0, i, j)), \tag{7}$$

where $\forall \varphi_k \in \Phi_{t,i,j}$, and $\boldsymbol{Z}^S_{t,k}(:, :)$ mean service-level on-board passenger flow matrix. Furthermore, the line-level on-board passenger flow at node r_{ij} during the interval t are defined as

$$\boldsymbol{Z}^L_t(i, j) = \sum_{\forall \varphi_k \in \Phi_{t,i,j}} \boldsymbol{Z}^S_{t,k}(i, j), \tag{8}$$

where $\boldsymbol{Z}^L_t(:, :)$ mean line-level on-board passenger flow matrix.

Definition 5 (Inter-Line Transfer Passenger Flows $\boldsymbol{M}$): Given a set of B/A pairs $\mathbb{P}$ and a sequence of time intervals $\mathcal{T}$, let (p_b, p_a, p'_b, p'_a) denote two continuous B/A pairs belonging to the same individual passenger, and $\{p, p'\}$ denote the set of such continuous B/A pairs, where $p = (p_b, p_a) \in \mathbb{P}$ and $p' = (p'_b, p'_a) \in \mathbb{P}$, then the inter-line transfer passenger flow from line l_i to $l_{i'}$ at node r_{ij} are defined as

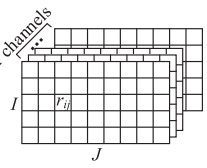


Fig. 11. Inter-line transfer passenger flow with I line channels.

$$\begin{aligned}\boldsymbol{M}_t(i, j, i') = |\{&(p_b, p_a, p'_b, p'_a) \in \{p, p'\} : (l_p, s_{p_a}) \in r_{ij} \\ &\wedge (l_{p'}, s_{p'_b}) \in r_{ij} \wedge \tau_p \in t \wedge \tau_{p'} \in t\}|,\end{aligned} \tag{9}$$

where $\boldsymbol{M}_t(:, :, i')$ mean inter-line transfer passenger flow matrix at any node r_{ij} that is transferred to a specific line $l_{i'}$, as shown in Fig. 11.

Considering all four types of flows (i.e., $\boldsymbol{X}_t, \boldsymbol{Y}^L_t, \boldsymbol{Z}^L_t, \boldsymbol{M}_t$), a time-varying bus route network is conventionally represented as a time-ordered sequence of matrixes/tensors, with each matrix/tensor corresponding to a snapshot of the bus route network during a certain time interval. These flow matrixes/tensors have complex influence and interaction with each other. In addition, we identify and integrate both internal and external features that have influences on passenger flows. External features include holidays information, weather conditions and events. Internal features mainly concern geographical attributes of bus lines and stations, which will be elaborated in Sect. V-D.

B. Service-Level Flow Prediction Problem

Generally, passenger flow prediction is a time series prediction problem, which aims to predict the passenger flow for each bus line at time interval $T + 1$ with observing historical flow data until time T. However, it is difficult for traditional approaches to capture complex spatio-temporal dependencies and influences. To achieve the fine-grained SPFP, in this paper we integrate four types of flows from various perspectives, i.e., arriving bus service flow, line-level B/A passenger flows, line-level on-board passenger flow, and inter-line transfer passenger flow, as defined above. Our goal is to first predict all these flows simultaneously, and then integrate the results to finally achieve the SPFP. In addition, we integrate both internal and external features defined above for improving prediction performance.

Definition 6 (Service-Level Passenger Flow Prediction Problem): Given the historical flow observations $\{\boldsymbol{X}_t, \boldsymbol{Y}^L_t, \boldsymbol{Z}^L_t, \boldsymbol{M}_t | t = t_1, \ldots, t_T\}$ and internal/external features $\boldsymbol{F}_t$, the objective of this paper is to collectively predict $\boldsymbol{X}_{t_{T+1}}$, $\boldsymbol{Y}^L_{t_{T+1}}$ and $\boldsymbol{Z}^L_{t_{T+1}}$, and integrate the results to finally predict $\boldsymbol{Z}^S_{t_{T+1}}$ in the future.

Unless explicitly stated, in the rest of paper we respectively use "bus service flow", "B/A passenger flow", "on-board passenger flow", and "transfer passenger flow" to represent "arriving bus service flow", "line-level B/A passenger flow", "line-level on-board passenger flow" and "inter-line transfer passenger flow" by default for convenience. Table IV lists frequently used notations in this paper.

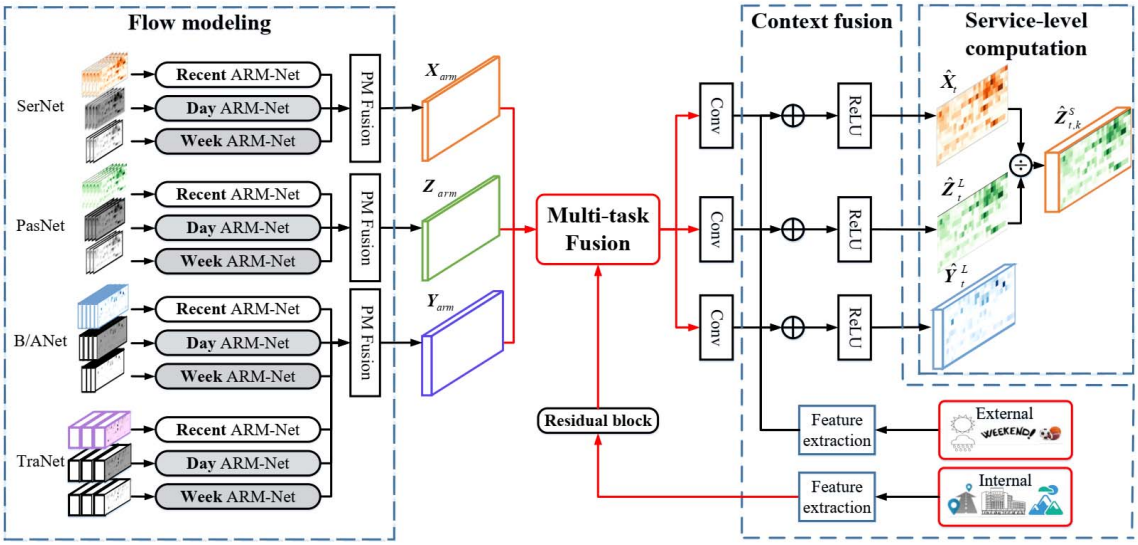


Fig. 12. The overall framework of MDL-SPFP.

TABLE IV
FREQUENTLY USED NOTATIONS

Symbol	Description
$V = \{r_{ij}\}$	set of nodes in a bus route network
$\mathcal{T} = \{t\}$	set of time intervals
$\boldsymbol{X}_t \in \mathbb{R}^{I \times J}$	arriving bus service flow matrix during the interval t
$\boldsymbol{Y}_t^L \in \mathbb{R}^{I \times J \times 2}$	line-level B/A passenger flow tensor during the interval t
$\boldsymbol{Z}_t^L \in \mathbb{R}^{I \times J}$	line-level on-board passenger flow matrix during the interval t
$\boldsymbol{Z}_t^S \in \mathbb{R}^{I \times J}$	service-level on-board passenger flow matrix during the interval t
$\boldsymbol{M}_t \in \mathbb{R}^{I \times J \times I}$	transfer passenger flow tensor during the interval t
$\varepsilon_{internal}$	internal features
$\varepsilon_{external}$	external features

V. MDL-BASED SERVICE-LEVEL PASSENGER FLOW PREDICTION

As shown in Fig. 12, our MDL-SPFP framework consists of four main components: *flow modeling*, *multi-task fusion*, *context fusion* and *service-level computation*. The *flow modeling* component uses four types of streaming data as input: (i) bus service flow, which is a time-ordered sequence of two dimensional matrices $\{\boldsymbol{X}_t | t = t_1, \ldots, t_T\}$; (ii) on-board passenger flow, which is a time-ordered sequence of two dimensional matrices $\left\{\boldsymbol{Z}_t^L | t = t_1, \ldots, t_T\right\}$; (iii) B/A passenger flow, which is a time-ordered sequence of three dimensional tensors $\left\{\boldsymbol{Y}_t^L | t = t_1, \ldots, t_T\right\}$; (iv) transfer passenger flow, which is a time-ordered sequence of three dimensional tensors $\{\boldsymbol{M}_t | t = t_1, \ldots, t_T\}$. These different types of video-like data are then fed into four networks, i.e., SERNET, PASNET, B/ANET and TRANET, respectively. Each network extracts three types of time fragments, including *recent*, *daily-periodic* and *weekly-periodic* fragments, for modeling the temporal correlations of the historical data. Then three types of stream data are respectively fed into an ARM network component

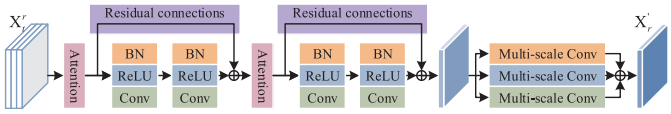


Fig. 13. The structure of ARM-Net.

with the same structure, including three modules, Attention mechanism, Residual networks and Multi-scale convolution, for capturing complex spatio-temporal dependencies, as shown in Fig. 13. Meanwhile, we integrate TRANET into B/ANET to improve the prediction performance of B/ANET, since transfer passengers belong to a part of B/A passengers. The latent representations of middle layers of SERNET, PASNET, B/ANET and internal influence factors are coupled by the *multi-task fusion* component, and trained together. In addition, we use the *context fusion* component to integrate the internal/external influence factors, and then obtain three types of prediction results, $\hat{\boldsymbol{X}}_t$, $\hat{\boldsymbol{Y}}_t^L$ and $\hat{\boldsymbol{Z}}_t^L$. Finally, we utilize the *service-level computation* component to integrate the outputs of SERNET and PASNET ($\hat{\boldsymbol{X}}_t$ and $\hat{\boldsymbol{Z}}_t^L$) to predict the service-level passenger flow $\hat{\boldsymbol{Z}}_t^S$.

A. SERNET *and* PASNET

The bus service flow and passenger flow, like the general crowd flow in a city [14], always have strong spatio-temporal dependencies. To capture different levels of temporal dependencies, Zhang *et al.* [4] proposed a deep spatio-temporal residual network to select different time fragments along the time. Inspired by this, we select similar types of temporal channels to express temporal dependencies. Specifically, we utilize three independent components, which share the same network structure, to model the *recent*, *daily-periodic* and *weekly-periodic* dependencies of the historical data, respectively.

Let d and w denote the number of time intervals contained in one day and one week, respectively. Let l_r, l_d, and l_w

denote the lengths of three types of time series fragments, respectively. Taking SERNET as an example, the three time series fragments are expressed as $\boldsymbol{X}_t = \{\boldsymbol{X}_t^r, \boldsymbol{X}_t^d, \boldsymbol{X}_t^w\}$, detailed as follows:

(1) The *recent* fragments: $\boldsymbol{X}_t^r = \{\boldsymbol{X}_{t-l_r}, \boldsymbol{X}_{t-l_r+1}, \ldots, \boldsymbol{X}_{t-1}\}$, which is a segment of time series directly adjacent to the predicting interval.

(2) The daily-periodic fragments: $\boldsymbol{X}_t^d = \{\boldsymbol{X}_{t-l_d \cdot d}, \boldsymbol{X}_{t-(l_d-1)\cdot d}, \ldots, \boldsymbol{X}_{t-d}\}$, which consists of the time fragments on the past l_d days during the same time interval (t).

(3) The weekly-periodic fragments: $\boldsymbol{X}_t^w = \{\boldsymbol{X}_{t-l_w \cdot w}, \boldsymbol{X}_{t-(l_w-1)\cdot w}, \ldots, \boldsymbol{X}_{t-w}\}$, which consists of the time fragments in the past l_w weeks with the same week attributes (i.e., weekday or weekend) during the same time interval (t).

We are now in a position to elaborate the three modules of the ARM network.

1) Attention Mechanism: After modeling the three levels of temporal dependencies, the correlations exist between the bus service or on-board passenger flows in different time slices for each level, and the mutual influences are highly dynamic under different situations. To automatically capture different weights of such influences, we introduce an attention mechanism [15] into ARM, so that some certain time intervals that are more critical to the current task could be emphasized. Meanwhile, both the efficiency and effectiveness of task processing could be improved.

Suppose a sequence of time intervals and $\boldsymbol{X}_t \in \mathbb{R}^{I \times J}$, we use $\boldsymbol{X}_t(:, j)$ to denote the value of all the features of station s_j at time t, and the pipeline of attention mechanism is defined as follows:

$$u_t = F_{pool}(\boldsymbol{X}_t) = \frac{1}{J}\sum_{j=1}^{J} \boldsymbol{X}_t(:, j), \tag{10}$$

$$f_t = F_{ratio}(u_t) = W_1 u_t + b_1, \tag{11}$$

$$g_t = F_{att}(f_t) = W_2 f_t + b_2. \tag{12}$$

The global average pooling F_{pool} is leveraged among all stations to produce a scalar summary u_t of each time interval. Then, a squeeze function F_{ratio} with attention operation F_{att} is conducted to generate adaptive weights g_t by applying non-linear transformations on u_t.

With obtained measurement g_t from the attention mechanism, we weigh each temporal data as:

$$\boldsymbol{X}'_t = g_t \odot \boldsymbol{X}_t, \tag{13}$$

where $\odot$ represents the Hadamard product, i.e., the multiplication of the corresponding elements of two vectors. $\boldsymbol{X'}_t$ is used for reweighing measurements during each time interval in the current channel.

2) Residual Block: According to the characteristic of convolution, spatial near dependencies (e.g., neighbouring bus stations of the same line) could be captured by one convolution layer. However, obvious spatial dependencies also exist between bus stations that are far away from each other. For example, humans always have regular commuting patterns, e.g., from home to work place. For another, travellers always like to leave one tourist attraction to another one. In order to capture such farther spatial dependencies, we need to increase the number of convolutional layers. However, when the network hierarchy increases to a certain large number, such deep convolution network becomes very hard to train. To accelerate the learning speed and address the gradient vanishing problem, we employ residual blocks [4], which consist of *Batch Normalization* (BN, [16]), *Rectified Linear Unit* (ReLU, [17]), and *Convolution* (Conv), to assist in the training.

3) Multi-Scale Convolution: A network with only single-scale convolution kernel has limited ability to extract different levels of features. Therefore, there is still room to improve the feature learning ability. In order to further capture spatial correlations of different regions (i.e. functional similar stations with strong correlations may not be close in space), we utilize a multi-scale CNN module to extract more comprehensive spatial correlation, including global and local details. For each time channel, the matrix is recognized as one dimension stations I with one dimension feature J. The multi-scale CNN utilizes three different convolution kernel sizes on the input tensor, and then we use bitwise addition to fuse different tensors. For each single CNN, the formula of each convolutional layer k is defined as follows:

$$\boldsymbol{X}_t^{'(k)} = W^{(k)} * \boldsymbol{X}_t^{'(k-1)} + b^{(k)}, \tag{14}$$

where $*$ denotes the convolution, $W^{(k)}$ and $b^{(k)}$ are the learnable parameters in the layer.

4) PM Fusion: So far, we obtain the outputs of *recent-*, *daily-periodic*, and *week-periodic* data streams from the bus service flow (i.e., $\boldsymbol{X}'_r$, $\boldsymbol{X}'_d$, $\boldsymbol{X}'_w$) and on-board passenger flow (i.e., $\boldsymbol{Z}_r^{L'}$, $\boldsymbol{Z}_d^{L'}$, $\boldsymbol{Z}_w^{L'}$), respectively. Then the three channels can be combined using a parametric-matrix (PM) based fusion function:

$$\boldsymbol{X}_{arm} = W_r \odot \boldsymbol{X}'_r + W_d \odot \boldsymbol{X}'_d + W_w \odot \boldsymbol{X}'_w, \tag{15}$$

where W_r, W_d, W_w are the learnable parameters, representing the weights of different channels in $\boldsymbol{X}_{arm}$.

B. B/ANET *and* TRANET

As described before, both B/ANET and TRANET utilize the ARM network. The difference is that the B/A passenger flow and transfer passenger flow tensors have one additional dimension. We use two dimensional convolution to explore the spatio-temporal correlation of the tensors. The output tensors from residual blocks in different temporal channels are $\boldsymbol{Y}_t^{L'} \in \mathbb{R}^{I \times J \times 2}$ and $\boldsymbol{M}'_t \in \mathbb{R}^{I \times J \times I}$. In order to leverage the transfer information to improve the B/A flow prediction performance, $\boldsymbol{M}'_t \in \mathbb{R}^{I \times J \times I}$ is converted into $\boldsymbol{M}'_t \in \mathbb{R}^{I \times J \times 2}$ through multi-scale convolution, and then combined with B/ANET using PM based fusion as follows:

$$\begin{aligned}\boldsymbol{Y}_{arm} = {} & W_Y^r \odot \boldsymbol{Y}_r^{L'} + W_Y^d \odot \boldsymbol{Y}_d^{L'} + W_Y^w \odot \boldsymbol{Y}_w^{L'} \\ & + W_M^r \odot \boldsymbol{M}'_r + W_M^d \odot \boldsymbol{M}'_d + W_M^w \odot \boldsymbol{M}'_w,\end{aligned} \tag{16}$$

where W_Y, W_M are the learnable parameters, representing the weights of B/A passenger flow and transfer passenger flow with different channels, respectively.

C. Multi-Task Fusion

Considering the correlations between the bus service, on-board passenger and B/A passenger flow, the representations learned from SERNET, PASNET and B/ANET can be combined to improve the prediction performance. However, the three measurements $\boldsymbol{X}_{arm}$, $\boldsymbol{Y}_{arm}$, $\boldsymbol{Z}_{arm}$ from different prediction tasks have different tensor sizes. Therefore, we concatenate the three latent representation maps of $\boldsymbol{X}_{arm}$, $\boldsymbol{Y}_{arm}$ and $\boldsymbol{Z}_{arm}$. Compared with the sum-based fusion, the concatenation-based fusion can better integrate the correlations among three tasks by the mutual reinforcement. The concatenation of three latent representation maps $\boldsymbol{X}_{arm}$, $\boldsymbol{Y}_{arm}$ and $\boldsymbol{Z}_{arm}$ at the same spatial node r_{ij} across channel c is defined as follows:

$$\boldsymbol{H}(c,:,:) = W_x \boldsymbol{X}_{arm}(c,:,:), \quad c=0,\ldots,C_x-1, \tag{17}$$

$$\boldsymbol{H}(C_x+c,:,:) = W_y \boldsymbol{Y}_{arm}(c,:,:), \quad c=0,\ldots,C_y-1, \tag{18}$$

$$\boldsymbol{H}(C_x+C_y+c,:,:) = W_z \boldsymbol{Z}_{arm}(c,:,:), \quad c=0,\ldots,C_z-1, \tag{19}$$

where C_x, C_y and C_z are the numbers of channels of $\boldsymbol{X}_{arm}$, $\boldsymbol{Y}_{arm}$ and $\boldsymbol{Z}_{arm}$, and $\boldsymbol{H} \in \mathbb{R}^{(c_x+c_y+c_z)\times I\times J}$.

In order to make different tensors have geographical information, we propose some internal features (will be elaborated later in Sect. V-D) of bus lines and stations to add geographic information to tensors. Because of internal features are time invariant, we regard them as a constant tensor, and input to $\boldsymbol{H}$ via Residual Block. The final concatenation of three latent representation maps and internal features $\varepsilon_{internal}$ at the same spatial node r_{ij} should be $\boldsymbol{H} \in \mathbb{R}^{(c_x+c_y+c_z+c_{internal})\times I\times J}$.

After the concatenation, we append a convolutional layer into the three individual networks (i.e.SERNET, PASNET and B/ANET). The convolution is used to map combined latent feature maps $\boldsymbol{H}$ into different-size-channel outputs, i.e., $\boldsymbol{X}_{arm} \in \mathbb{R}^{1\times I\times J}$, $\boldsymbol{Y}_{arm} \in \mathbb{R}^{2\times I\times J}$, $\boldsymbol{Z}_{arm} \in \mathbb{R}^{1\times I\times J}$.

D. Context Fusion

Internal and external features can affect various flows. Besides common *external* features, such as weather conditions and holiday events, we add some *internal* features of bus lines and stations, i.e., using geographic information to explore passengers movement patterns in different lines and stations, including the accessibility, connectivity, and surrounding functional zones, detailed as follows:

1) Accessibility: When passengers choose a transit station, accessibility is an important factor, representing whether more places can be conveniently reached from a station through the transportation network. Specifically, we extract the accessibility feature, denoted by $R_a(s_j)$, by counting how many transportation hubs (according to the PoI categories that belong to transportation infrastructure services such as airports and railway stations) that a station s_j can reach.

2) Connectivity: The more bus lines that a station connects, the more passengers may travel through this station. Thus, we extract the connectivity feature, denoted by $R_c(s_j)$, by counting the number of bus lines across each station s_j, which reflects the attractiveness and convenience of station s_j.

3) Functional Zones: Surrounding functional zones reflect geographic features of different stations, and affect the types of passengers. We extract the functional zones feature, denoted by $R_f(s_j)$, by utilizing the entropy of surrounding PoI categories for each station s_j, which is computed as follows:

$$R_f(s_j) = -\sum_p \frac{FZ_p(s_j)}{FZ(s_j)} \times log(\frac{FZ_p(s_j)}{FZ(s_j)}), \tag{20}$$

where $FZ_p(s_j)$ denote the number of PoIs belonging to category p, and $FZ(s_j)$ denote the total number of PoIs within a certain region (e.g., a kilometer in diameter.) around station s_j.

The above three types of features constitute the internal features $\varepsilon_{internal}$ and are integrated to the multi-task fusion component as described in Section V-C.

Meanwhile, the external features include holidays (using a binary number to represent non-holiday or holiday), weather (using an integer $[0,3]$ to represent four types of weather conditions) and temperature (using an integer $[-10,21]$ to represent temperature values), as shown in Table II. To extract the relationship between each node and external features, inspired by the embedding method for natural language processing [18], we first use an embedding layer for each external feature, in which the integer values (corresponding to each type of feature) are converted to fixed-length dense vectors using the Word2vec embedding method provided by Keras [19]. Then, we utilize a full-connected layer to extract three types of relationships between each node and corresponding external feature for each task, and obtain external feature matrix for each node during each time interval $F_t^{external} \in \mathbb{R}^{I\times J}$. Then we utilize a bitwise addition between $F_t^{external}$ and the output from the multitask fusion component to obtain the final predictions for different tasks during interval t:

$$\hat{\boldsymbol{X}}_t(1,:,:) = ReLU(F_t^{external} + \boldsymbol{X}_{arm}(1,:,:)), \tag{21}$$

$$\hat{\boldsymbol{Z}}_t^L(1,:,:) = ReLU(F_t^{external} + \boldsymbol{Z}_{arm}(1,:,:)), \tag{22}$$

$$\hat{\boldsymbol{Y}}_t^L(2,:,:) = ReLU(F_t^{external} + \boldsymbol{Y}_{arm}(2,:,:)). \tag{23}$$

E. Loss Function

Let θ_1 be the learnable parameters in SERNET. The loss function for SERNET is defined as follows:

$$\underset{\theta_1}{argmin}\boldsymbol{J}_{\text{SERNET}} = \frac{1}{I}\frac{1}{J}\sum_{t\in\mathcal{T}}\sum_{i=1}^{I}\sum_{j=1}^{J}(Q_t^c \odot |\hat{\boldsymbol{X}}_t(i,j) - \boldsymbol{X}_t(i,j)| + \beta_1(\frac{\hat{\boldsymbol{X}}_t(i,j)-\boldsymbol{X}_t(i,j)}{\boldsymbol{X}_t(i,j)})^2), \tag{24}$$

where Q_t^c is an indication matrix for all the non-zero entries in $\boldsymbol{X}_t(i,j)$, β_1 is a proper weight decay as regularization, which can smooth the large prediction differences. τ is a set of available time intervals.

Let θ_2 and θ_3 be the learnable parameters in PASNET and B/ANET. The loss functions for PASNET and B/ANET are

defined using the similar formula as follows:

$$\underset{\theta_2}{argmin} \boldsymbol{J}_{\text{PASNET}} = \frac{1}{I}\frac{1}{J}\sum_{t\in\mathcal{T}}\sum_{i=1}^{I}\sum_{j=1}^{J}(Q_t^c \odot |\hat{\boldsymbol{Z}}_t^L(i,j) - \boldsymbol{Z}_t^L(i,j)| + \beta_2(\frac{\hat{\boldsymbol{Z}}_t^L(i,j) - \boldsymbol{Z}_t^L(i,j)}{\boldsymbol{Z}_t^L(i,j)})^2), \tag{25}$$

$$\underset{\theta_3}{argmin} \boldsymbol{J}_{\text{B/ANET}} = \frac{1}{I}\frac{1}{J}\sum_{t\in\mathcal{T}}\sum_{i=1}^{I}\sum_{j=1}^{J}(Q_t^d \odot |\hat{\boldsymbol{Y}}_t^L(i,j) - \boldsymbol{Y}_t^L(i,j)| + \beta_3(\frac{\hat{\boldsymbol{Y}}_t^L(i,j) - \boldsymbol{Y}_t^L(i,j)}{\boldsymbol{Y}_t^L(i,j)})^2), \tag{26}$$

where d in Q_t^d equals to $2c$.

Finally, we use a weighted sum of the three loss functions as the final loss for the multi-task fusion:

$$\underset{\theta_1,\theta_2,\theta_3}{argmin} \boldsymbol{J}_{\text{MULTITASK}} = \lambda_1 \boldsymbol{J}_{\text{SERNET}} + \lambda_2 \boldsymbol{J}_{\text{PASNET}} + \lambda_3 \boldsymbol{J}_{\text{B/ANET}}, \tag{27}$$

where λ_1, λ_2 and λ_3 are adjustable hyper parameters.

F. Optimation Algorithm

Algorithm 1 Training of MDL-SPFP Algorithm

Input: Historical observations: $\{\boldsymbol{X}_t, \boldsymbol{Y}_t^L, \boldsymbol{Z}_t^L, \boldsymbol{M}_t | t = t_1, \ldots, t_T\}$;
External and internal features: $\{\varepsilon_{t_1}, \ldots, \varepsilon_{t_T}\}$
Output: MDL-SPFP Model
// construct training instances
Initialization $\boldsymbol{D}_{train} \leftarrow \emptyset$;
for $t \in \mathcal{T}$ **do**
  Put a training instance $\boldsymbol{X}_t, \boldsymbol{Y}_t^L, \boldsymbol{Z}_t^L, \boldsymbol{M}_t, \varepsilon_t$ into $\boldsymbol{D}_{train}$;
end
// train the model
Initialize the parameters $\theta_1, \theta_2, \theta_3, \beta_1, \beta_2, \beta_3$;
repeat
  Randomly select a batch of instances $\boldsymbol{D}_{batch}$ from $\boldsymbol{D}_{train}$;
  Optimize $\theta_1, \theta_2, \theta_3, \beta_1, \beta_2, \beta_3$ by minimizing the loss (27) with $\boldsymbol{D}_{batch}$;
until *stopping criteria is met*;
Output the learned MDL-SPFP model;

Algorithm 1 outlines the training process of MDL-SPFP. First, training instances are constructed from the historical data. Second, the objective (27) is optimized on the selected batch of training instances during each iteration.

G. Service-Level Computation

With the outputs of three tasks, we now could integrate the prediction results from SERNET and PASNET to predict the service-level passenger flows. Since passengers always move with buses, the position changes of passenger flows are always corresponding to that of bus service flows. Given a specific time interval t_{T+1} and a specific station s_j of the bus line l_i, the number of service-level on-board passengers could be computed as follows: if there is only one service, i.e., $\boldsymbol{X}_{t_{T+1}}(i,j) = 1$, then we can directly obtain the result, $\boldsymbol{Z}_{t_{T+1}}^S(i,j) = \boldsymbol{Z}_{t_{T+1}}^L(i,j)$; if there are more than one service, then we can generally compute the result by averaging the line-level on-board passenger number by the arriving bus service number:

$$\boldsymbol{Z}_{t_{T+1},k}^S(i,j) = \frac{\boldsymbol{Z}_{t_{T+1}}^L(i,j)}{\boldsymbol{X}_{t_{T+1}}(i,j)}, \quad \forall \varphi_k \in \Phi_{t,i,j},\ \forall (l_i, s_j) \in r_{ij}. \tag{28}$$

VI. EXPERIMENTS

In this section, we evaluate our proposed MDL-SPFP approach and compare it with 10 baseline approaches based on the Jinan bus operation dataset. The data from 11/01/2018 to 01/16/2019 is used for training (77 days), and the data from 01/17/2019 to 01/31/2019 (15 days) is used for testing. The Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are used as evaluation metrics.

A. Baselines

- **HA:** Historical Average model. Here, we use the average value of the last 5 time intervals to predict the next value.
- **ARIMA:** Autoregressive Integrated Moving Average model. It is a well-known time series analysis model for predicting the future values.
- **EKF:** 2-step real-time prediction model based on Extended Kalman Filter [11]. It is one state-of-the-art work used for achieving the SPFP during a short-term time interval. Here, we set two main parameters, history sequence length as 10, and input sequence length as 5.
- **RNN:** Recurrent Neural Network [20].
- **LSTM:** Long-Short-Term-Memory network [21], a special RNN model.
- **GRU:** Gate-Recurrent-Unit network [22]. It is also a special RNN model.
- **ConvLSTM:** Convolutional LSTM [23]. It is a model originally used for precipitation nowcasting using radar echo dataset. Here, different types of flow data can be regarded as sequences of images. The previous 5 time intervals are used to predict the next time interval. The model consists of two ConvLSTM layers with two BN layers.
- **ST-ResNet:** Spatio-Temporal Residual Convolutional Network [4]. It is a state-of-the-art model for flow prediction. The lengths of the three types of time series fragments are set as 3, 1 and 1, respectively. Here we use 1-D convolution in SERNET and PASNET, and 2-D convolution in B/ANET and TRANET. ST-ResNet utilizes Residual Network to model temporal dependency and spatial dependency, and utilizes a full-connected neural network to extract external features.
- **DST-ICRL:** Deep Spatio-Temporal traffic passenger flows feature learning model, which combines the Irregular Convolutional Residual Network and the LSTM Recurrent Neural Network for accurately predicting

TABLE V
CHARACTERISTICS OF 10 BASELINES AND OUR MDL-SPFP APPROACH IN TERMS OF VARIOUS CAPABILITIES

Model	Temporal	Spatial	External	Internal	Multi-task
HA	✓				
ARIMA	✓				
EKF	✓				
RNN	✓				
LSTM	✓				
GRU	✓				
ConvLSTM	✓	✓			
ST-ResNet	✓	✓	✓		
DST-ICRL	✓	✓	✓		
DMVST-Net	✓	✓	✓	✓	
MDL-SPFP [ours]	✓	✓	✓	✓	✓

line-level urban traffic passenger flows [12]. Here, we use 2-D convolution in each network, and each bus line is treated as a channel. The lengths of the three types of time series fragments are set as 3, 1 and 1, respectively. The DST-ICRL utilizes a parameter-shared convolutional module to learn temporal and spatial features among multiple channels of traffic passenger flows, and utilizes residual neural network and LSTM units to learn the high-level temporal and spatial features.

- **DMVST-Net:** Deep Multi-View Spatial-Temporal Network [24]. It is a model for taxi demand forecast using multi-view spatial-temporal network. A local CNN is used to capture spatial dependency, a LSTM model is used to capture temporal dependency, and a weighted graph of regions is used to capture semantic dependency.

The characteristics of the above baselines and our MDL-SPFP approach are summarized in Table V, in terms of the capabilities of capturing spatial-temporal dependencies and external/internal influences, and the multi-task learning.

B. Experimental Settings

1) Preprocessing: In the output of the MDL-SPFP, we use ReLU as our final activation, whose range is between 0 and $+\infty$. The Min-Max normalization method is used to scale the data into the range $[0, 1]$. For the feature extraction, we use an embedding layer to encode the internal and external data. Then, we apply the inverse of the Min–Max transformation obtained by the training set to recover the flow value and use it for the evaluations.

2) Hyperparameters: To evaluate the MDL-SPFP approach, all the experiments are compiled and tested on a Linux cluster (CPU: Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz, GPU: NVIDIA Tesla P100). The python library, Keras 2.2 is used to build our models. The detailed hyperparameter configurations for the baselines are listed in Table VI. In the multi-task part of MDL-SPFP, we set $\lambda_1 = 1$, $\lambda_2 = 1$ and $\lambda_3 = 1$ by default. The lengths of the three types of time series fragments are set as $l_r = 3$, $l_d = 1$ and $l_w = 1$, respectively. The number of convolutions for ARM are set as 12. During model training, we select 90% of the training data, and the remaining 10% as the validation set to early-stop our training algorithm. Afterwards, we train the model on the full training data for a fixed number of epochs (10 epochs). All the network parameters are trained from a random start. The batch size is set as 32. The learning rate is set as one of $\{0.01, 0.005, 0.0025, 0.00125\}$.

C. Experimental Results

In the following, we first compare the performance for each of three types of flow predictions, and then compare the final SPFP results among our MDL-SPFP and all of 10 baselines. For a fair comparison, all approaches do not utilize any internal or external feature. Table VII summarizes the experimental results in terms of MAE and RMSE. For convenience, we use SERNET, PASNET, B/ANET to represent the corresponding types of flow predictions for the baselines.

1) Bus Service Flow Prediction: From Table VII, we can observe that our MDL-SPFP performs apparently better than other baselines for SERNET. Specifically, the results of RMSE demonstrate that MDL-SPFP is 48.6% better than HA, 53.4% better than ARIMA, 46.1% better than EKF, 41.0% better than RNN, 39.3% better than LSTM, 40.6% better than GRU, 7.4% better than ConvLSTM, 27.9% better than ST-ResNet, 33.9% better than DMVST-Net, and 26.3% better than DST-ICRL. Generally, the three traditional shallow prediction models (i.e., HA, ARIMA and EKF) perform worse than other deep learning based prediction models, owing to lack of the ability of capturing complex non-linear spatial-temporal dependencies.

2) On-Board Passenger Flow Prediction: Similar to SERNET, it can be observed that our MDL-SPFP performs apparently better than other baselines for PASNET. Specifically, the results of RMSE demonstrate that MDL-SPFP is 25.8% better than ST-ResNet, 35.8% better than DMVST-Net, and 28.7% better than DST-ICRL.

3) Boarding/Alighting Passenger Flow Prediction: It can be observed that our MDL-SPFP achieve better performance than the baseline approaches except DMVST-Net. Specifically, the MAE and RMSE of MDL-SPFP are very close to the best results achieved by DMVST-Net, i.e., with just 1.58% and 5.78% differences respectively.

4) Service-level On-Board Passenger Flow Prediction: It can be observed that our MDL-SPFP outperforms all of other baselines. Specifically, the results of MAE demonstrate that MDL-SPFP is 65.96% better than HA, 58.88% better than ARIMA, 51.92% better than EKF, 51.34% better than RNN, 50.69% better than LSTM, 49.61% better than GRU, 22.39% better than ConvLSTM, 53.42% better than ST-ResNet, 39.16% better than DMVST-Net, and 35.70% better than DST-ICRL. Meanwhile, the results of RMSE demonstrate that MDL-SPFP is 55.04% better than HA, 62.19% better than ARIMA, 47.52% better than EKF, 44.40% better than RNN, 42.29% better than LSTM, 42.78% better than GRU, 15.66% better than ConvLSTM, 30.39% better than ST-ResNet, 32.26% better than DMVST-Net, and 25.40% better than DST-ICRL.

D. Evaluation on Model Hyper-Parameters

1) Effect of Filter Number: Figure 14 presents the effect of filter number on MDL-SPFP. As the filter number become

TABLE VI
NETWORK CONFIGURATION OF DIFFERENT BASELINES

Model	RNN	LSTM	GRU	ConvLSTM	ST-ResNet	DST-ICRL	DMVST-Net
# of hidden units in RNN	Layer1: 128 Layer2: 128	Layer1: 128 Layer2: 128	Layer1: 128 Layer2: 128	-	-	Layer1: 128	Layer1: 64 Layer2: 64
# of hidden units in full-connected network	Layer1: 981	Layer1: 981	Layer1: 981	-	Layer1: 981 Layer2: 981	Layer1: 981	Layer1: 981
Convolution kernel number	-	-	-	Layer1: 64 Layer2: 1	Layer1- Layer11: 64 Layer12: 1	Layer1- Layer5: 64	Layer1: 64 Layer2: 64 Layer3: 64
Convolution kernel size	-	-	-	Layer1: 3 Layer2: 3	Layer1- Layer12: 3	Layer1- Layer5: 3	Layer1: (3,3,1,64) Layer2: (3,3,64,64) Layer3: (3,3,64,64)
Convolution layer stride	-	-	-	-	Layer1- Layer12: 1	Layer1- Layer5: 1	Layer1: 1 Layer2: 1
Activation function	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU

TABLE VII
COMPARISONS WITH BASELINES FOR THREE TYPES OF FLOW PREDICTIONS AND THE FINAL SPFP RESULTS

Model	SerNet		PasNet		B/ANet		SPFP	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
HA	0.5369	0.82878	5.95654	11.4095	1.44002	2.54288	5.30379	8.68689
ARIMA	0.55351	0.91361	5.31458	11.58237	1.16553	2.45014	4.39018	10.32956
EKF	0.44898	0.78931	4.65628	9.63339	1.11711	2.33737	3.75499	7.44153
RNN	0.45677	0.72097	4.76537	10.00399	0.94099	2.3494	3.70983	7.024
LSTM	0.45018	0.70167	4.68038	9.63384	0.94933	2.3453	3.66145	6.76656
GRU	0.43042	0.71687	4.60918	9.7326	0.94292	2.34826	3.5828	6.82513
ConvLSTM	0.26537	0.45971	2.87513	5.92626	0.94382	2.34946	2.32625	4.63053
ST-ResNet	0.33457	0.59066	3.16732	6.49932	0.92089	2.23749	2.7957	5.61002
DMVST-Net	0.36871	0.64368	3.68405	7.51492	**0.85038**	**1.93113**	2.96759	5.76478
DST-ICRL	0.32597	0.57739	3.44136	6.76746	**0.87472**	**1.8666**	2.80773	5.23479
MDL-SPFP[ours]	**0.21811**	**0.42562**	**2.18178**	**4.82412**	0.86399	2.04956	**1.80537**	**3.90521**

larger, the MAE of the model decreases, and the computation consumption becomes higher, which verifies that more filters can be utilized to capture more useful information.

2) Effect of Network Depth: Figure 15 presents the effect of network depth on MDL-SPFP. As the network becomes deeper (i.e. the number of residual units increases), the MAE of the model first decreases, verifying that deeper network can capture more complicated spatial dependencies. However, when the network becomes much deeper, the MAE starts to increase, implying that the training process becomes more difficult.

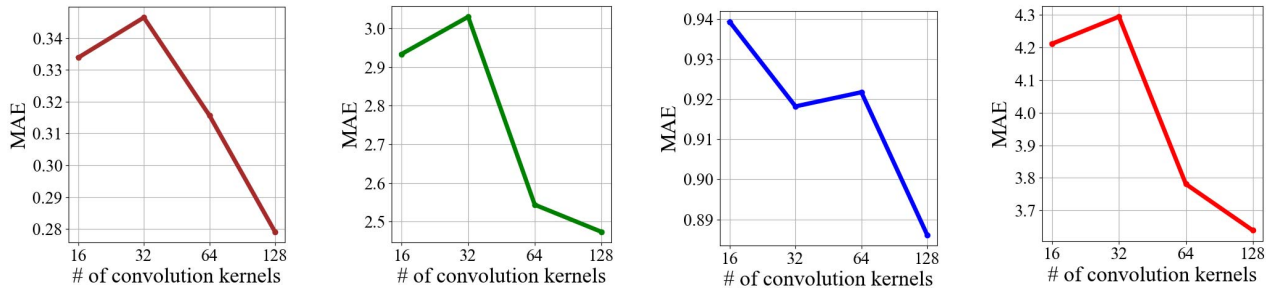
E. Evaluation on Different Components/Modules

To evaluate the effects of different components/modules, we remove each component/module (including attention mechanism, multi-scale convolution, SERNET, PASNET, B/ANET, and TRANET) one by one from MDL-SPFP. Recall that we directly integrate TRANET into B/ANET to improve the performance of B/ANET, so we now also evaluate its effect by comparing an alternative policy, i.e. regarding TRANET as an independent task. For convenience, all the approaches for evaluations do not utilize the context fusion component by default. Finally, we specially add the context fusion component to MDL-SPFP for comparison. The detailed evaluation results for each individual task, the overall loss and the final SPFP are listed in Table VIII.

From Table VIII, it can be observed that both two modules of attention mechanism and multi-scale convolution have great influences to all the results. Specifically, these two modules reduce the final SPFP result of MAE by 31.87% and 6.95% respectively, and that of RMSE by 23.31% and 3.32% respectively. Second, it is interesting to observe that, if either one of SERNET and PASNET is removed, a severe performance degradation will be brought about for the other. It implies strong dependencies between the bus service flow and the on-board passenger flow. By contrast, B/ANET has less impact, but it can still reduce the SPFP result of MAE by 6.39%, and that of RMSE by 2.86%, respectively. Third, the performance becomes worse when we either remove TRANET or regard it as an independent task, although the latter is slightly better than the former. In short, our entire MDL-SPFP approach achieves the best performance,

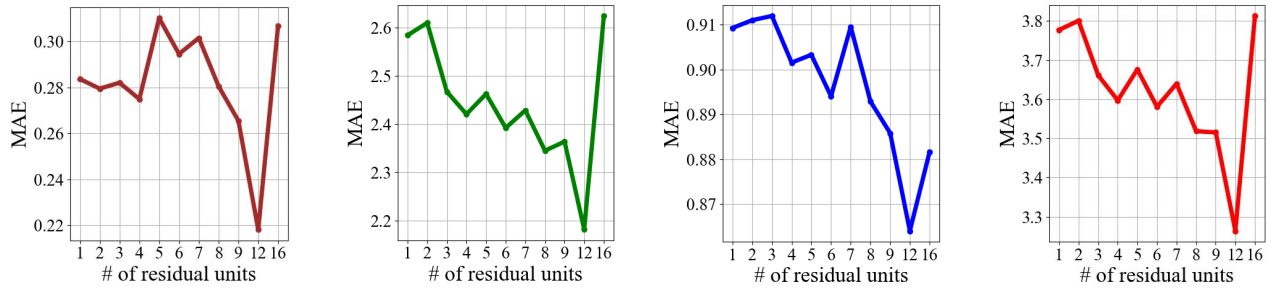
TABLE VIII
EVALUATION ON DIFFERENT COMPONENTS/MODULES (C.F. IS SHORT FOR "CONTEXT FUSION")

Components/ modules	SerNet		PasNet		B/ANet		Loss	SPFP	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	/	MAE	RMSE
w/o attention	0.338	0.581	3.038	6.189	0.939	2.338	4.314	2.65	5.092
w/o multi-scale conv.	0.246	0.439	2.265	4.87	0.879	2.127	3.39	1.94	4.04
w/o SERNET	/	/	2.321	5.011	0.913	2.22	/	/	/
w/o PASNET	0.309	0.574	/	/	0.892	2.144	/	/	/
w/o B/ANET	0.244	0.474	2.231	4.836	/	/	/	1.929	4.02
w/o TRANET	0.257	0.469	2.31	5.023	0.865	2.043	3.433	2.05	4.277
TRANET as a task	0.289	0.51	2.278	4.94	0.864	2.056	3.431	2.03	4.194
MDL-SPFP	0.218	0.426	2.182	4.824	0.864	2.05	3.264	1.805	3.905
MDL-SPFP w/ c.f.	0.198	0.376	2.084	4.73	0.901	1.924	3.183	1.71	3.845



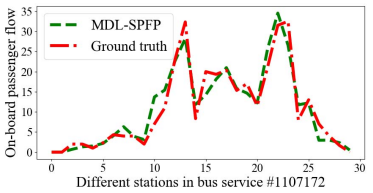
(a) Bus service flow (b) On-board passenger flow (c) B/A passenger flow (d) Overall loss

Fig. 14. Effect of filter number.

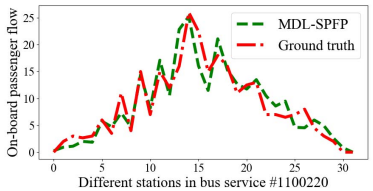


(a) Bus service flow (b) On-board passenger flow (c) B/A passenger flow (d) Overall loss

Fig. 15. Effect of network depth.



(a) Trip #1



(b) Trip #2

Fig. 16. Prediction results of MDL-SPFP against the ground truths.

confirming the effectiveness of each component/module and the entire MDL framework. In addition, the results also verify the effectiveness of the context fusion component, which reduces the SPFP result of MAE by 5.27% and that of RMSE by 1.55%, respectively.

F. Service-Level Passenger Flow Prediction

Figure 16 presents the SPFP results achieved by our MDL-SPFP approach for two randomly selected bus services from one bus line, against the ground truth in Jinan City. It can be visually observed that our MDL-SPFP approach is very accurate in tracing the ground truth curves (including sudden changes), which demonstrates the superior performance of our proposed approach.

VII. Related Work

Traffic flow prediction has been extensively investigated due to its importance to people's daily life. We summarize existing work in Table IX with a three-dimension taxonomy: (i) prediction models, i.e., traditional shallow prediction models or deep learning based prediction models; (ii) traffic flows types, i.e., general traffic flows, which mainly focuses on the total number of individual objects such as taxis and bicycles passing through a region during a period [25], or passenger flows, which mainly focus on the total number of passengers passing through a bus or subway station; (iii) spatial granularity, i.e., region-level, which is mainly applicable to general traffic flow prediction, line/station-level, which is mainly applicable to bus/subway passenger flow prediction, or service-level, which is mainly applicable to bus passenger flow prediction. Note that, if we can predict passenger flows for any subway line at any station, then the service-level passenger flow prediction can be achieved directly, but which cannot apply to bus passenger flow prediction.

A. General Traffic Flow Prediction

1) Traditional Shallow Prediction Models: Traffic flow prediction can be treated as a time series prediction problem. Many approaches, such as ARIMA and Kalman filtering, could be used to capture temporal dependencies very well [26]–[28], but failed to capture spatial dependencies. Recent work was conducted to utilize spatial information

TABLE IX
TRAFFIC FLOW PREDICTION SURVEY

Traffic flows types	Spatial granularity	Prediction models	
		Shallow models	Deep learning models
General traffic flows	Region-level	[26]–[34]	[4], [24], [35]–[38]
Passenger flows	Line/Station-level	[5]–[9]	[12], [13]
	Service-level	[10], [11]	*MDL-SPFP*

(e.g., adding regularizations on model similarity for nearby positions [29]–[31]) and external data (e.g., adding features of weather condition and local events [32]–[34]) to improve the prediction performance. However, these approaches heavily relied on feature engineering and were difficult to capture complex non-linear spatial-temporal dependencies.

2) Deep Learning Based Prediction Models: Recently, deep learning has been utilized for traffic flow prediction by virtue of its remarkable capability to capture non-linear spatiotemporal dependencies [25]. Zhang *et al.* [4], [35] treated the traffic flow within the whole city as an image and utilized CNN to capture spatial dependencies. Meanwhile, they proposed to select different time fragments to model temporal dependencies. Several researchers proposed to utilize RNN-based models to better capture temporal dependencies, especially under extreme conditions [36], [37]. Some work further integrated both temporal and spatial dependencies by combining CNN and LSTM [24], [38].

B. Passenger Flow Prediction

1) Traditional Shallow Prediction Models: In recent years, many models have been utilized for passenger flow prediction, including ARIMA model [5], Seasonal ARIMA (SARIMA) model [7], [8], Kalman filtering model [6], [11], support vector machine (SVM) model [9], [10], etc. As described before, these models failed to capture complex non-linear spatial-temporal dependencies. In addition, most of them [5]–[9] focused on station-level subway passenger flow prediction.

2) Deep Learning Based Prediction Models: More recently, a few studies started to leverage deep learning for passenger flow prediction. Liu *et al.* [13] proposed to combine pre-trained unsupervised stacked autoencoders with the supervised DNN to predict the passenger flow for any specified hour. However, it was only limited to the station-level prediction for bus rapid transit (BRT) and subway systems. Du *et al.* [12] proposed a deep irregular convolutional residual LSTM network model called DST-ICRL for urban traffic passenger flow prediction. However, it was only limited to predicting line-level passenger flows in each region (i.e., grid cell) instead of distinguishing different bus stations or services.

VIII. CONCLUSION

We investigate the fine-grained service-level passenger flow prediction problem for bus transit systems. We propose a novel multitask deep learning (MDL) approach, *MDL-SPFP*, for simultaneously predicting arriving bus service flow, line-level on-board passenger flow and line-level boarding/alighting passenger flow, which can mutually reinforce the prediction of each type of flow, and finally integrate the outputs to achieve the fine-grained service-level prediction. For each flow prediction task, a novel deep neural network, *ARM*, is designed to combine attention mechanism, residual block and multi-scale convolution for well capturing various complex non-linear spatio-temporal dependencies and influence factors. Extensive experiments based on a large-scale realistic bus operation dataset verify the superior performance of MDL-SPFP over 10 baselines. We hope our approach could be used for other passenger flow predictions such as subway systems and the entire public transit systems. It is also significant to utilize passenger flow prediction results for implementing various applications, such as transit resource scheduling and crowdedness-aware route recommendations.

REFERENCES

[1] (2019). *Buses and Coaches-Getting Everybody Onboard*. [Online]. Available: https://www.iru.org/who-we-are/about-mobility/bus-and-coach-transport

[2] (2017). *Ministry of Transport of China: 250 Million People Travel by Bus Every Day*. [Online]. Available: https://www.yicai.com/news/5373677.html

[3] (2019). *Transit Crowdedness Trends From Around the World, According to Google Maps* [Online]. Available: https://www.blog.google/products/maps/transit-crowdedness-trends-around/

[4] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1–7.

[5] C. Ding, J. Duan, Y. Zhang, X. Wu, and G. Yu, "Using an ARIMA-GARCH modeling approach to improve subway short-term ridership forecasting accounting for dynamic volatility," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1054–1064, Apr. 2018.

[6] P. Jiao, R. Li, T. Sun, Z. Hou, and A. Ibrahim, "Three revised Kalman filtering models for short-term rail transit passenger flow prediction," *Math. Problems Eng.*, vol. 2016, pp. 1–10, Mar. 2016.

[7] M. Milenković, L. Švadlenka, V. Melichar, N. Bojović, and Z. Avramović, "SARIMA modelling approach for railway passenger flow forecasting," *Transport*, vol. 33, no. 5, pp. 1113–1120, 2018.

[8] M. Ni, Q. He, and J. Gao, "Forecasting the subway passenger flow under event occurrences with social media," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1623–1632, Jun. 2017.

[9] Y. Sun, B. Leng, and W. Guan, "A novel wavelet-SVM short-time passenger flow prediction in beijing subway system," *Neurocomputing*, vol. 166, pp. 109–121, Oct. 2015.

[10] Q. Chen, W. Li, and J. Zhao, "The use of LS-SVM for short-term passenger flow prediction," *Transport*, vol. 26, no. 1, pp. 5–10, 2011.

[11] J. Zhang *et al.*, "A real-time passenger flow estimation and prediction method for urban bus transit systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 3168–3178, Nov. 2017.

[12] B. Du *et al.*, "Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 972–985, Mar. 2020.

[13] L. Liu and R.-C. Chen, "A novel passenger flow prediction model using deep learning methods," *Transp. Res. Part C, Emerg. Technol.*, vol. 84, pp. 74–91, Nov. 2017.

[14] J. Zhang, Y. Zheng, J. Sun, and D. Qi, "Flow prediction in spatio-temporal networks based on multitask deep learning," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 3, pp. 468–478, Mar. 2020.

[15] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

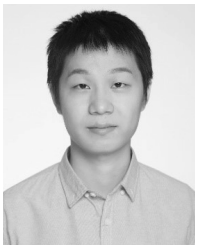
[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

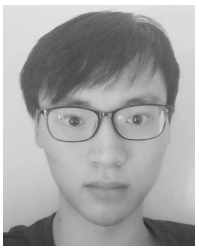
[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[19] (2019) *Docs.Layers.Embedding Layers*. [Online]. Available: https://keras.io/layers/embeddings/
[20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[22] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*. [Online]. Available: http://arxiv.org/abs/1406.1078
[23] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.
[24] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.
[25] Z. Liu, Z. Li, K. Wu, and M. Li, "Urban traffic prediction from mobility data using deep learning," *IEEE Netw.*, vol. 32, no. 4, pp. 40–46, Jul. 2018.
[26] S. Shekhar and B. M. Williams, "Adaptive seasonal time series models for forecasting short-term traffic flow," *Transp. Res. Record, J. Transp. Res. Board*, vol. 2024, no. 1, pp. 116–125, Jan. 2007.
[27] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.
[28] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 871–882, Jun. 2013.
[29] T. Idé and M. Sugiyama, "Trajectory regression on road networks," in *Proc. 25th AAAI Conf. Artif. Intell.*, 2011, pp. 1–6.
[30] J. Zheng and L. M. Ni, "Time-dependent trajectory regression on road networks via multi-task learning," in *Proc. 27th AAAI Conf. Artif. Intell.*, 2013, pp. 1–8.
[31] D. Deng, C. Shahabi, U. Demiryurek, L. Zhu, R. Yu, and Y. Liu, "Latent space model for road networks to predict time-varying traffic," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1525–1534.
[32] B. Pan, U. Demiryurek, and C. Shahabi, "Utilizing real-world transportation data for accurate traffic prediction," in *Proc. IEEE 12th Int. Conf. Data Mining*, Dec. 2012, pp. 595–604.
[33] F. Wu, H. Wang, and Z. Li, "Interpreting traffic dynamics using ubiquitous urban data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*, 2016, p. 69.
[34] Y. Tong *et al.*, "The simpler the better: A unified approach to predicting original taxi demands based on large-scale online platforms," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 1653–1662.
[35] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "DNN-based prediction model for spatio-temporal data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*, 2016, p. 92.
[36] Z. Cui *et al.*, "Deep stacked bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction," in *Proc. 6th Int. Workshop Urban Comput. (UrbComp)*, 2017, pp. 1–9.
[37] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. SIAM Int. Conf. Data Mining*, 2017, pp. 777–785.
[38] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2017, *arXiv:1707.01926*. [Online]. Available: http://arxiv.org/abs/1707.01926

Dan Luo is currently pursuing the Ph.D. degree with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China. Her research interests include urban computing and artificial intelligence.

Dong Zhao (Member, IEEE) received the B.S. degree from the Department of Computer Science and Technology, Henan University, China, in 2008, and the Ph.D. degree from the School of Computer Science, Beijing University of Posts and Telecommunications, China, in 2014. He was a Visiting Ph.D. Student with the Department of Computer Science, Illinois Institute of Technology, from 2012 to 2013. He is currently an Associate Professor with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications. His research interests include the Internet of Things, mobile crowd sensing, urban computing, and data science. He has published over 50 articles and one book on these fields. He was awarded the China Computer Federation (CCF) Outstanding Doctoral Dissertation Award in 2015, the ACM Beijing Doctoral Dissertation Award in 2015, and the Natural Science Award of the Ministry of Education, China, in 2017. He is a member of ACM.



Qixue Ke received the B.S degree from the School of Geography and Information Engineering, China University of Geosciences, in 2018. He is currently pursuing the master's degree with the School of Software Engineering, Beijing University of Posts and Telecommunications. His research interests focus on traffic data mining and intelligent transportation technology.



Xiaoyong You is currently pursuing the master's degree with the School of Computer Science, Beijing University of Post and Telecommunications, China. His research interests include data visualization and artificial intelligence.

Liang Liu (Member, IEEE) received the B.S. degree from the Department of Computer Science and Technology, South China University of Technology, Guangzhou, China, in 2004, and the Ph.D. degree from the Department of Computer, Beijing University of Posts and Telecommunications, Beijing, China, in 2009. He was a Visiting Ph.D. Student with the Networking and Information Systems Laboratory, Texas A&M University, College Station, TX, USA, from 2007 to 2008. He is currently a Professor with the Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia and the Vice Dean of the School of Computer Science, Beijing University of Posts and Telecommunications. He has published over 100 articles. His current research interests include the Internet of Things and intelligent sensing technologies.

Desheng Zhang (Member, IEEE) has been an Assistant Professor with the Department of Computer Science, Rutgers University, since 2016. His research interests include cyber physical systems, cyber human systems, and data science in extreme-scale data-intensive urban infrastructure from an interdisciplinary perspective.

Huadong Ma (Senior Member, IEEE) received the B.S. degree in mathematics from Henan Normal University in 1984, the M.S. degree in computer science from the Shenyang Institute of Computing Technology, Chinese Academy of Science, in 1990, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Science, in 1995. From 1999 to 2000, he held a visiting position with the University of Michigan, Ann Arbor, USA. He is currently a Chang Jiang Scholar Professor and the Director of the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, and the Executive Dean of the Institute of Network Technology, Beijing University of Posts and Telecommunications, China. His current research focuses on the Internet of things and sensor networks, and multimedia computing. He has published over 300 articles in journals (such as ACM/IEEE TRANSACTIONS) or conferences, such as ACM SIGCOMM/MobiCom/MM and the IEEE INFOCOM, and five books. He is an Editorial Board Member of the IEEE T-MM, the IEEE IoT JOURNAL, *ACM T-IoT*, and MTAP. He serves as the Chair of ACM SIGMOBILE China. He was awarded the Natural Science Award of the Ministry of Education, China, in 2017. He was a recipient of the 2019 Prize Paper Award of IEEE TRANSACTIONS ON MULTIMEDIA, the 2018 Best Paper Award from the IEEE MultiMedia, the Best Paper Award in the IEEE ICPADS2010, and the Best Student Paper Award in IEEE ICME2016 for his coauthored articles. He received the National Funds for Distinguished Young Scientists in 2009.

Xingquan Zuo (Senior Member, IEEE) received the Ph.D. degree in control theory and control engineering from the Harbin Institute of Technology, Harbin, China, in 2004. He is currently a Professor with the School of Computer Science, Beijing University of Posts and Telecommunications. From 2004 to 2006, he was a Postdoctoral Research Fellow with the Automation Department, Tsinghua University. From 2012 to 2013, he was a Visiting Scholar with the Industrial and System Engineering Department, Auburn University, AL, USA. He has published over 100 research articles in journals and conferences, two books, and several book chapters. His research interests are in optimization and scheduling, data mining, artificial intelligence, and intelligent transportation systems. He is a Senior Member of China Computer Federation and the Chinese Association for Artificial Intelligence. He is the Committee Member of the Transportation Model and Simulation Society and Intelligent Simulation Optimization and Scheduling Society of Chinese Association for System Simulation. He served in a program committee or as Program Chair of more than ten international conferences.