5-2021

TRIPDECODER: Study travel time attributes and route preferences of metro systems from smart card data

Xiancai TIAN
Singapore Management University, shawntian@smu.edu.sg

Baihua ZHENG
Singapore Management University, bhzheng@smu.edu.sg

Yazhe WANG
Singapore Management University, yzwang@smu.edu.sg

Hsao-Ting HUANG
National Cheng Kung University

Chih-Cheng HUNG
National Cheng Kung University

TRIPDECODER: Study Travel Time Attributes and Route Preferences of Metro Systems from Smart Card Data

XIANCAI TIAN, BAIHUA ZHENG, and YAZHE WANG, Living Analytics Research Centre, Singapore Management University, Singapore
HSIAO-TING HUANG, Department of Electrical Engineering, National Cheng Kung University, Taiwan
CHIH-CHIEH HUNG*, Department of Management Information System, National Chung Hsing University, Taiwan

In this paper, we target at recovering the exact routes taken by commuters inside a metro system that are not captured by an Automated Fare Collection (AFC) system and hence remain unknown. We strategically propose two inference tasks to handle the recovering, one to infer the travel time of each travel link that contributes to the total duration of any trip inside a metro network and the other to infer the route preferences based on historical trip records and the travel time of each travel link inferred in the previous inference task. As these two inference tasks have interrelationship, most of existing works perform these two tasks simultaneously. However, our solution TripDecoder adopts a totally different approach. TripDecoder fully utilizes the fact that there are some trips inside a metro system with only one practical route available. It strategically decouples these two inference tasks by only taking those trip records with only one practical route as the input for the first inference task of travel time and feeding the inferred travel time to the second inference task as an additional input which not only improves the accuracy but also effectively reduces the complexity of both inference tasks. Two case studies have been performed based on the city-scale real trip records captured by the AFC systems in Singapore and Taipei to compare the accuracy and efficiency of TripDecoder and its competitors. As expected, TripDecoder has achieved the best accuracy in both datasets, and it also demonstrates its superior efficiency and scalability.

CCS Concepts: • **Information systems** → **Spatial-temporal systems**; **Data mining**; • **Mathematics of computing** → *Maximum likelihood estimation.*

Additional Key Words and Phrases: metro systems, smart card data, travel time inference, route choice preference estimation, maximum likelihood estimation

ACM Reference Format:
Xiancai TIAN, Baihua ZHENG, Yazhe WANG, Hsiao-Ting HUANG, and Chih-Chieh HUNG. 2021. TRIPDECODER: Study Travel Time Attributes and Route Preferences of Metro Systems from Smart Card Data. 1, 1 (March 2021), 22 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

*The corresponding author.

Authors' addresses: Xiancai TIAN, shawntian@smu.edu.sg; Baihua ZHENG, bhzheng@smu.edu.sg; Yazhe WANG, yzwang@smu.edu.sg, Living Analytics Research Centre, Singapore Management University, Singapore; Hsiao-Ting HUANG, q36064248@gs.ncku.edu.tw, Department of Electrical Engineering, National Cheng Kung University, Taiwan; Chih-Chieh HUNG, smalloshin@gmail.com, Department of Management Information System, National Chung Hsing University, Taiwan.

XXXX-XXXX/2021/3-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

For land-scarce and metro-rely countries like Singapore, it is extremely important to improve the public transport systems in order to meet the increasing travel demands of a growing economy and population. *Mass Rapid Transit* (MRT) system is a critical part of the public transport system because of its advantages in both capacity and efficiency[1]. In order to increase the MRT ridership and encourage more commuters to take MRT, it is critical to improve the MRT service which has attracted attention from the academy.

For instance, predicting vehicle crowdedness and platform commuter intensity can help operators evaluate service quality and design structural improvements for the metro network [8, 9, 17]; understanding commuters' route choice preferences and route travel time allows operators to provide more accurate route recommendation [3, 5]; studying commuters' movement during MRT disruption enables operators to identify potentially overcrowded stations and to take more targeted remedial actions like arranging alternative transportation modes [11, 16, 20].

Studies on commuters' behaviour inside public transport systems have long been relying on external data source like field surveys [1, 5, 10, 11] and crowdsourcing [17]. However, these data sources have their own limitations. Take survey data as an example. It is easily subject to bias and errors, and conducting surveys and processing the data can be both time-consuming and labor-intensive. In addition, since most surveys are conducted with focus on particular location and time, the results are often limited in scale and diversity. Data collected via crowdsourcing suffers from similar issues. As a result, alternative data sources are required to be able to more accurately and more comprehensively understand the spatial-temporal characteristics of travel patterns, such as train control sensors [8, 9], and GPS data [3].

In this paper, we aim at inferring the travel time required by any route inside the metro network, and route preferences at both aggregation level and individual level, based on data collected from automated fare collection (AFC) systems that have emerged and widely deployed over the last decade. When the context is clear, we may use the term *AFC data* interchangeably with the term *smart card data*, *trip records* or *trip observations*, and they all refer to some key information related to trips (e.g., the time stamp and the MRT station/bus stop when a trip is started, and the time stamp and the MRT station/bus stop when a trip is ended).

As more and more public transportation systems are now using smart cards to collect trip fares, it has generated massive precious data resource for public transport scientific study. However, the smart card data has limitations. For example, because of the reliability requirement of a metro network, redundant design is adopted to tolerate faults. Consequently, there could be multiple routes available to bring a commuter from the boarding station to the alighting station. However, as most metro networks are designed as closed systems and commuters only leave traces at boarding/alighting stations for the purpose of fare collection, the exact route taken by each individual commuter remains unknown. On the other hand, the information of each commuter's movement inside the metro network is critical to the study of commuter behaviors at a microscopic level.

As mentioned above, the main objective of this paper is to infer the travel time of any route, and to infer the route preferences of commuters if there are multiple routes available to bring the commuter from the boarding station to the alighting station. These two inference tasks have interrelationship, and hence existing works on similar topics perform these two inference tasks simultaneously, which significantly increases the complexity of the problem. We adopt a very different approach. Our solution, TripDecoder, takes in a static metro network and its smart card data as inputs. By carefully studying the data, TripDecoder points out a fact that some trips inside

[1]In this paper, the term MRT system is used interchangeably with metro system.

a metro system have *only one* practical route. It makes full use of this finding, and decouples the two inference tasks into two separate steps.

During the data pre-processing stage, a route candidate set is generated for each Origin-Destination (OD) pair of stations, where unrealistic routes, such as routes that are extremely long with loops and routes with many unnecessary transfers, are removed. We then category OD pairs into two disjoint sets based on the number of available routes linking them, i.e., OD pairs with a single route and OD pairs with multiple alternative routes. The clever separation of OD pairs with single route from those with multiple routes actually motivates the design of our first inference task. Accordingly, TripDecoder strategically decomposes the travel time required by a trip into different travel links, and fully utilizes single-route OD pairs and their corresponding trips (captured by the AFC system) to derive travel time of different travel links that contribute to the travel time of any trip. Because TripDecoder only considers the trips of OD pairs with single routes, there is no ambiguity in terms of the routes taken to complete the trips. Therefore, we effectively remove the dependency of the route preference from the inferring of travel time, and are able to produce more accurate estimation of the travel time of travel links. The inferred travel time of different travel links is then used to construct travel time of routes corresponding to multi-route OD pairs, i.e., the output from the first inference task becomes an additional and useful input for the inference of route preferences. With route travel time known, the complexity of the inference of the route preferences w.r.t. multiple routes has been effectively reduced.

To illustrate and verify the proposed solution, we carry out case studies using real datasets, i.e., the city scale real trip data captured by AFC systems in Singapore and Taipei. Our result demonstrates the superior performance of TripDecoder, in terms of both accuracy and efficiency.

The remainder of this paper is organized as follows. In Section 2, we review previous studies on several related topics, including metro network travel time estimation, commuter route choice behaviour, and the use of smart card data in understanding metro operation and flow assignment. In Section 3, we present the preliminaries of TripDecoder, including the formulation of the problem studied in this paper, the extraction of the route choice sets, and the insights from data exploration. In Section 4, we present the framework of TripDecoder and detail the two-step solution algorithm to recover the route travel time and to learn the route preferences. In Section 5, we apply TripDecoder on real trip data collected from Singapore and Taipei as two case studies and report the performance of TripDecoder. We close the paper with conclusion and discussion of future research directions in Section 6. Note that without the loss of generality, in the rest of paper we use Singapore metro network and its smart card data collected during morning peak hours in 2015 December as an example to explain how the proposed framework works.

2 LITERATURE REVIEW

Understanding the commuter flow in a transportation system is an important research topic. In the existing literature, many of the works focus on studying commuter flow models based on experience [6, 7, 12]. The models depend heavily on behavior assumptions and hence lack reliable empirical data verification. Other studies are based on field surveys [1, 5, 10, 11], crowdsourcing [17], train control sensors [8, 9], and GPS data [3]. These datasets are usually expensive to obtain, small in scale, and poor in accuracy, therefore would greatly affect the analytic power of the applications built based on them.

In recent years, smart card data have provided us with new opportunities to perform data-centric transit behavior study. [4] develops a heuristic method to assign commuter flows inside a metro network based on AFC data. The main idea is to use train timetable to estimate the pure travel time of every trip record, and then to cluster the trips based on the pure travel time between an OD pair, with the assumption that each trip cluster corresponds to a candidate route connecting

the OD pair. The method is very efficient but it requires additional information of real-time train timetable, which is not always available. It also has accuracy issue due to the many assumptions made such as train services strictly follow the timetable, and commuters never fail to board on the immediate train after entering the stations. [13] studies the latent relationships among OD pairs, candidate routes and commuter travel time, and obtains the distribution of commuter flow on different candidate routes by a *Latent Dirichlet Allocation* (LDA) model. However, their model is not able to capture the travel time distribution on different routes, and thus could not infer local commuter flow of individual station/link segment of the metro network.

To fully exploit the AFC data and predict the local commuter flow of individual link/station and commuters' route preferences at the same time, many recent researches rely on statistical modelling based inferences [2, 14, 15, 19]. These models take commuter travel time as observations and characterize them as a mixture distribution from all potential routes. [14, 15] propose to construct posterior probability by combining the likelihood of observed commuter travel times provided by AFC data and prior knowledge about the studied transportation network. They assume the link travel time of the transit network follows the normal distribution, and the commuters' route choice probability could be represented by a logit model of various influential factors (i.e., in-vehicle travel time, transfer time). Thereafter, they perform Bayesian inference to calibrate the parameters (i.e., mean/variance of link travel time and coefficient of influence factors of the route choice probability) of the model. [19] builds a similar posterior probability model as [14, 15], where it considers not only in-vehicle travel time and transfer time factors, but also crowdedness factor for route choice probability.

To the best of our knowledge, work presented in [2] represents the state-of-the-art solution to the inference of travel time and route preferences of commuters inside a metro network based on AFC data. It proposes a different likelihood model of the observed commuter travel time by modeling the path travel time as complicated convolutions of Poisson distributions, and models the path choice probability as a logit model of the station number factor and the transfer number factor. Due to the intractability of the model, [2] also proposes approximate inference schemes to estimate the model parameters. The models discussed above assume the commuter route choice is determined by a few predetermined influential factors (e.g., route travel time, transfer number). However, factors that affect commuters' route-choice decisions could be complicated and difficult to model, and missing key influential factors may affect the accuracy of the model.

Different from existing solutions, we adopt a data-driven approach. TripDecoder models the route preferences purely based on real travel time observations reflected by the smart card data but not any explicit influential factor. In addition, there is interrelationship between these two inference tasks and all the existing works perform these two inference tasks simultaneously, which significantly increases the complexity of proposed models. The models search for the optimal parameter combination in an extremely large search space, which results in low accuracy and poor efficiency and scalability. TripDecoder is designed to address both the accuracy issue and the performance issue, as we strongly believe that an ideal solution shall be able to achieve a high accuracy and to complete the inference tasks efficiently and meanwhile is scalable. To our best knowledge, this is the first work on learning the travel time and route preferences from AFC data that considers the efficiency and scalability of the inference model, in addition to the accuracy.

A preliminary work was published in [18]. As compared with previous preliminary work, we have made following new contributions in this extended version.

- First, we have improved the inference models such that the enhanced model framework is able to achieve a higher accuracy and a better efficiency. For example, we notice the entry walking time and the exit walking time may follow different distributions, as the exiting action

happens right after a train reaches the station while the entering action could happen any time. Accordingly, in this extended version, we assume that they follow different distributions which does improve the accuracy. Instead of modeling the travel time from station s_i to its adjacent station s_j and the time from s_j to s_i differently, we simplify the model by assuming the travel time is independent of the direction. It does help reduce the complexity of the model and hence improve the efficiency, without downgrading the accuracy. When we perform the inference, we explore the impact of initial values on the performance and the new initial values used in this extended version actually are more appropriate as the training time has been reduced.

- Second, we have significantly improved the experimental study. To be more specific, we have included the work published in [2] as a new competitor; we have included the AFC data collected from Taipei as an additional dataset and reported the performance of TripDecoder and its competitors based on Taipei dataset; we have designed and implemented an evaluation framework for the inference of route preference and reported the performance of TripDecoder and its competitors; and we have included a new set of experiments to demonstrate the advantage of TripDecoder in terms of efficiency and scalability, as compared with its competitors.
- Third, we have detailed the insights we have obtained from our initial study on Singapore dataset, which suggest a simple but very novel and effective approach to perform the inference of travel time.
- Fourth, we have significantly improved the presentation and the organization of the paper.

In brief, we believe this extended version has included sufficient fresh contributions.

3 PRELIMINARY

Before we present TripDecoder, we first propose a trip reconstruction process in Section 3.1, which defines a trip as a sequence of steps to ease the inference of the travel time required. We formulate the metro system as a general graph network and introduce the notations used throughout the paper. We next introduce the concept of *candidate route set* R_{od} that is defined for a given OD pair $\langle o, d \rangle$ in Section 3.2. We use this concept to cluster all the OD pairs into two disjoint categories, the one with only one candidate route and the other with multiple candidate routes. We then perform data exploration in Section 3.3, using AFC data collected from Singapore and Taipei, and report our findings, which lay the foundation for TripDecoder. Table 1 lists the symbols that will be frequently used in the rest of the paper.

3.1 Problem Formulation

In this paper, we model a metro network as a general transportation graph $G(S, E, L)$, consisting of a set of metro stations S, a set of edges E, and a set of metro lines L. A station $s \in S$ could be either a normal station that is crossed by only one metro line or an interchange that is crossed by multiple metro lines. An edge (or a link, interchangeably) $e(s_i, s_j, l) \in E$ is a segment on a train line $l_x \in L$ that connects two stations s_i and s_j without passing any other station. Stations s_i and s_j are called adjacent if there is an edge $e(s_i, s_j, l) \in E$ between them. Note that there could be multiple edges corresponding to two adjacent stations (s_i, s_j), corresponding to different lines. In addition, we model a metro network as an undirected graph for simplicity. However, the techniques developed in this paper could be easily extended to support the case where a metro network is modelled as a directed graph.

An example metro network is depicted in Figure 1 for illustration purpose. Accordingly, we have $S = \{s_1, s_2, s_3, s_4, s_5, s_6, \cdots\}$, $E = \{e_1(s_1, s_2, l_1), e_2(s_2, s_3, l_1), e_3(s_3, s_4, l_1), e_4(s_2, s_3, l_2), e_5(s_3, s_5, l_2),$

Table 1. Frequent Symbols

Symbol	Definition
$G(S, E, L)$	a general transportation graph with S, E, L representing stations, edges, and service lines respectively
r_{ij}	a route from station s_i to station s_j
$r_{ij}.k$	the number of links travelled by a route r_{ij}
$r_{ij}.q$	the number of transfers required by a route r_{ij}
$\|r_{ij}\|$	the length of route r_{ij} which is defined as $r_{ij}.k + \alpha \times r_{ij}.q$
$T_{r_{ij}}$	the travel time required by a route r_{ij}
T_s^g	entry walking time from the turnstile to the platform at station s
T_l^w	waiting time for the service line l at the platform
T_e^c	train travel time corresponding to an link e
T_s^q	transfer time required at interchange station s
T_s^a	exit walking time from the platform to the turnstile at station s
tr	a trip record captured by AFC data, in the form of (id, s_o, s_d, t)
TR_{od}	the set of observed trips corresponding to a given OD pair $\langle o, d\rangle$, $TR_{od} = \{tr \mid tr.s_o = o \wedge tr.s_d = d\}$
R_{od}	the set of routes corresponding to a given OD pair $\langle o, d\rangle$, i.e., $R_{od} = \cup r_{od}$
r_{od}^{min}	the route corresponding to OD pair $\langle o, d\rangle$ with the shortest length
OD_s/OD_m	the set of OD pairs that have one route/multiple routes
TR_s/TR_m	the set of trip observations that corresponding to the OD pairs preserved by OD_s/OD_m

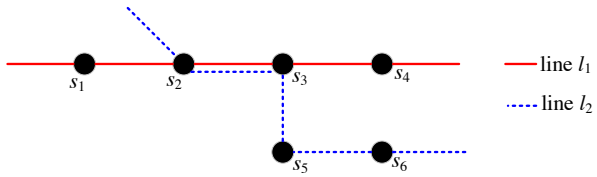


Fig. 1. Example network

$e_5(s_5, s_6, l_2) \cdots\}$, and $L = \{l_1, l_2, \cdots\}$. Station s_2 and station s_3 are adjacent, and they are connected by two edges, i.e., e_2 and e_4 corresponding to lines l_1 and l_2 respectively. Stations s_2 and s_3 are also interchanges as commuters can switch from one service line to another at both s_2 and s_3, while stations s_1, s_4, s_5 and s_6 are normal stations.

A route r_{ij} from an origin station s_i to a destination station s_j is a sequence of adjacent edges $\langle e_1, \cdots, e_k\rangle$ that could bring commuters from station s_i to station s_j. Edge $e_1(s_{i_1}, s_{j_1}, l_1)$ and edge $e_2(s_{i_2}, s_{j_2}, l_2)$ are adjacent if $e_1.s_{j_1} = e_2.s_{i_2}$, while they do not necessarily correspond to the same line. If two edges are in different lines, that is $e_1.l_1 \neq e_2.l_2$, it indicates the travel from e_1 to e_2 requires a transfer from line $e_1.l_1$ to another line $e_2.l_2$ at the station $e_1.s_{j_1}$. For example, edges e_1 and e_2 are adjacent but not edges e_1 and e_3. Route $r_{15} = \langle e_1, e_2, e_5\rangle$ provides an example route from station s_1 to station s_5 which requires a transfer at station s_3; and $r'_{15} = \langle e_1, e_4, e_5\rangle$ is another route from station s_1 to s_5 which requires a transfer at station s_2.

In this paper, we only consider simple routes without loop, so that each route only visits a station at most once. If we assume that there is no significant difference among the travel time required by each edge, the length of a route r_{ij} is determined by two parameters, the number of edges travelled

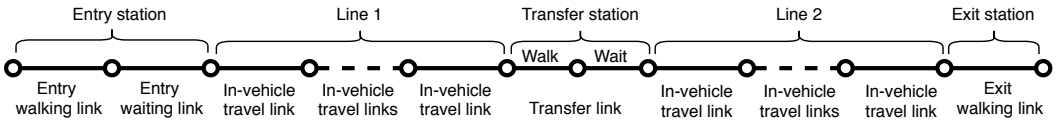


Fig. 2. Travel links of a trip in the metro system

and the number of transfers required, denoted by $r_{ij}.k$ and $r_{ij}.q$, respectively. Take the example route r_{15} as an example. We have $r_{15}.k = 3$ as it passes three edges and $r_{15}.q = 1$ as it requires one transfer at the transfer station s_3. Since a route may include transfer stations, we generalize the length of a route by taking the number of transfers into account. Given a route r_{ij}, the length of r_{ij} is defined as $|r_{ij}| = r_{ij}.k + \alpha \times r_{ij}.q$, e.g., $|r_{15}| = 3 + \alpha$. Here, α is the penalty coefficient of transfer[2].

In addition to the length of a route, we also denote $T_{r_{ij}}$ as the corresponding travel time required when a commuter takes route r_{ij} to travel from the boarding station s_i to the alighting station s_j. When the context is clear, we can use T_r to represent $T_{r_{ij}}$ for brevity. A trip starts when a commuter enters the turnstile, which consists of following four components, walking to the platform, waiting for the train, travelling via the train, and walking to the turnstile to exit the station and complete the trip. If the route taken requires transfers, an additional component (i.e., transfer) is involved. Accordingly, we can model T_r based on following five kinds of *travel links*, representing the five different travel components described above. In the rest of the paper, the term *travel link* is used to refer to one component of a trip via a metro system, which contributes to the total time required by a trip from entering the boarding station to exiting the alighting station.

- $T^g_{s_i}$: entry walking time from turnstiles to the platform at the boarding station s_i
- $T^w_{l_x}$: waiting time for the train service l_x at station s_i
- T^c_e: train travel time of every edge e
- T^q_s: transfer time required at an interchange station s
- $T^a_{s_j}$: exit walking time from the platform to turnstiles at the alighting station s_j

Here, the transfer time at station s consists of walking time from one platform to another, and waiting time for the next train. For the case of Singapore, most interchanges are crossed by two different metro lines. The only exception is the Dhoby Ghaut station in city center that is crossed by three MRT lines, thus it has three unique transfer walking time distributions. In addition, $T^w_{l_x}$ is independent of the station, as the service frequency of a service line l_x does not change from station to station. Notice in reality, the entry/exit walking time at station s also depends on the platform and turnstiles the commuters travel between, while we abuse the notation here for simplicity.

Given an edge $e(s_i, s_j, l_x)$ connecting station s_i and station s_j, we assume the travel time required from s_i to s_j via service line l_x is exactly the same as that required from s_j to s_i via the same edge. In other words, we assume that bi-directional travel costs between two adjacent stations are characterized by an identical distribution. However, TripDecoder could be easily extended to perform the inferences when we model a metro system as a directed graph, and the travel time required from station s_i to its adjacent station s_j might be different from that from s_j to s_i. We further visualize the travel links in Figure 2 to facilitate the understanding.

After decomposing a trip into five different types of travel links, we can sum the time spent on each travel link of r_{ij} in order to calculate the total travel time of $r_{ij} = \langle e_1, e_2, \cdots, e_k \rangle$, as shown in Equation (1). Note, S_m refers to the set of interchange stations on route r_{ij} where commuters have

[2]A common practice in transportation research is to set this value to 2.

to make transfers.

$$T_r = T^g_{s_i} + T^w_{l_x} + \sum_{b=1}^{k} T^c_{e_b} + \sum_{s \in S_m} T^q_s + T^a_{s_j} \tag{1}$$

The AFC system records the trips of individual commuters. Each trip observation, represented as (id, s_o, s_d, t_o, t_d), captures the details of a real trip tr made by a commuter via the metro network. Here, id is an encrypted unique string identifying a smart card, s_o is the origin station, s_d is the destination station, t_o records the time stamp when the commuter enters the station s_o, and t_d records the time stamp when the commuter exits the alighting station s_d from a turnstile. In other word, $t = t_d - t_o$ captures the real travel time required. In the rest of this paper, we represent each trip record as $tr = (id, s_o, s_d, t)$ for convenience. Given an OD pair $\langle o, d \rangle$, we collect all the trip records tr that are corresponding to $\langle o, d \rangle$ into a set TR_{od}, i.e., $\forall tr \in TR_{od}, tr.s_o = o \wedge tr.s_d = d$. In the cases that the frequencies of metro systems are different in different time of a day and different day of a week (e.g., the metro system in Singapore), we actually further decompose set TR_{od} into multiple smaller subsets, e.g., one set corresponding to the morning peak hour of weekdays, one set corresponding to non-peak hour of weekdays, one set corresponding to evening peak hour of weekdays, one set corresponding to the non-peak hour of weekend and so so. Our inference tasks are performed based on the trip records corresponding to a particular time period (e.g., morning peak of weekdays).

This paper aims at inferring the time corresponding to each travel link in order to infer the time required by all possible routes, as well as the probabilities that commuters choose each candidate route to travel for any given OD pair $\langle o, d \rangle$, given a static MRT network structure $G(S, E, L)$ and the set of trip observations captured by the AFC system. We formally define the first inference task in Definition 3.1; we will present the formal definition of the second inference task in Section 3.2, after we introduce the concept of candidate route set.

Definition 3.1. ***Inference of Route Travel Time.*** *Given a metro network $G(S, E, L)$, and a large set of trips corresponding to different OD pairs captured by AFC systems $X = \bigcup_{\langle o,d \rangle \in S \times S \wedge o \neq d} TR_{od}$, inference of route travel time is to infer the travel time of all the travel links that might contribute to the total travel time required by any route r_{ij}, which can best fit the traveling time observed in X.* □

3.2 Candidate Routes Extraction

As mentioned previously, redundant design is adopted to tolerate faults, because of the reliability requirement of a metro network. Consequently, there could be multiple routes available from an origin station s_i to a destination station s_j. We therefore introduce the concept of *candidate route set*. Let R_{od} denote the complete set of possible routes of an OD pair $\langle o, d \rangle$, and let r^{min}_{od} refer to the one with the shortest length, i.e., $\forall r \in R_{od}, |r^{min}_{od}| \leq |r| \wedge \exists r' \in R_{od}, r' = r^{min}_{od}$. Formally, we name R_{od} as the *candidate route set* corresponding to the OD pair $\langle o, d \rangle$.

To generate a candidate route set R_{od} for each OD pair $\langle o, d \rangle$, there are different strategies, such as edge elimination and k-shortest-paths. Nevertheless, the number of stations in a metro system usually is in the scale of either tens or hundreds, e.g., New York City Subway has in total 400+ stations, the most stations owned by a metro system. Consequently, we can simply adopt brute-force-search algorithm to form R_{od} for different OD pairs $\langle o, d \rangle$s.

Given a candidate route set R_{od} w.r.t. an OD pair $\langle o, d \rangle$, we also notice that some routes may never be used by commuters, e.g., those that are much longer than other routes, and those with too many transfers that bring inconvenience. We, therefore, define a *restricted candidate route set* $R^{'}_{od}$ w.r.t. an OD pair $\langle o, d \rangle$, which excludes those rarely-used or never-used routes based on following criteria:

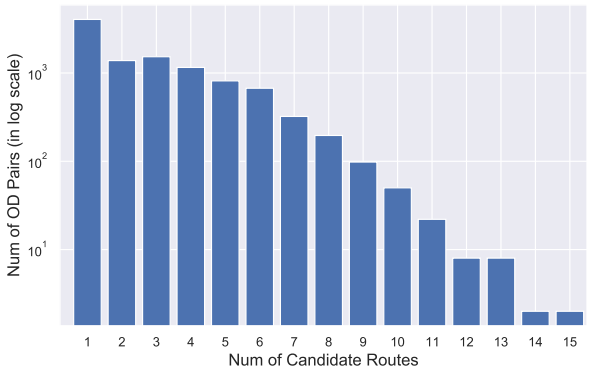


Fig. 3. Candidate routes distribution of Singapore MRT ($\beta = 2 \wedge \sigma = 2$)

- routes with any loops
- routes that are more than β (> 1) times longer than the shortest route r_{od}^{min}
- routes that are not the shortest paths but require more than σ transfers

The controlling parameters β and σ could be set according to different assumptions made on commuters' behavior. For example, in our study, we set both β and σ to two. The underlying assumptions are i) commuters might not always take the shortest route, but they are not willing to take routes that are much longer than necessary; and ii) some commuters may be willing to make transfers for comfortability or other reasons, but a route that requires more than two transfers is not preferred. However, the solution proposed in this paper is independent on the values of β or σ.

In brief, $R_{od}^{'} = \{r \in R_{od} | (r = r_{od}^{min} \vee r.q \leq \sigma) \wedge |r| \leq \beta \times r_{od}^{min}\}$. In the rest of this paper, we refer candidate routes set of an OD pair $\langle o, d \rangle$ to its restricted candidate route set $R_{od}^{'}$. The notation $|R_{od}|$ stands for the number of routes inside the candidate route set R_{od}. Based on the candidate route set, we present the second inference task that this paper wants to perform in Definition 3.2. Note that we can infer the route preference at either the aggregate level or the individual level. At the aggregate level, we can learn the route preferences of the entire commuter population based on the city-scale trip observations; at the individual level, we can learn the preference of a particular individual based on the trip records corresponding to that individual only.

Definition 3.2. ***Inference of Route Preferences.*** *Given a metro network $G(S, E, L)$ and a large set of trips corresponding to different OD pairs captured by AFC systems $X = \bigcup_{\langle o,d \rangle \in S \times S \wedge o \neq d} TR_{od}$, inference of route preferences is to infer, for an OD pair $\langle o, d \rangle$ with multiple routes (i.e., $|R_{od}| > 1$), the likelihood that each route $r \in R_{od}$ will be taken by a commuter to travel from o to d.* □

For illustration purpose, we report the size of candidate route sets of different OD pairs corresponding to Singapore metro system in Figure 3. As it can be observed, the number of candidate routes of OD pairs varies from 1 to 15. We then categorize all the OD pairs according to the sizes of their respective candidate route sets. To be more specific, *single route set* OD_s keeps all the OD pairs with single route, and *multiple route set* OD_m keeps all the OD pairs with multiple routes, i.e., $OD_s = \{\langle o, d \rangle \in S \times S | o \neq d \wedge |R_{od}| = 1\}$, and $OD_m = \{\langle o, d \rangle \in S \times S | o \neq d \wedge |R_{od}| > 1\}$. The inference of route preference only focuses on OD pairs preserved by OD_m.

Fig. 4. Singapore MRT network map (as of 2016 May)

3.3 Data Exploration Insights

As we highlight in Section 1, TripDecoder adopts an approach that is very different from all the existing solutions, i.e., decoupling the inference of the route travel time from the inferring of the route preferences. To the best of our knowledge, this is the first work that decouples these two inference tasks, which in turn benefits both the accuracy and the efficiency of the inferences. It is worth noting that although decoupling sounds simple, it is non-trivial to propose a two-step framework to not only simplify the inference tasks but also improve the accuracy, as these two inference tasks have interrelationship. Our design is partially motivated by the insights we have collected from Singapore AFC data in our data exploration, to be detailed next.

The Singapore MRT network, as shown in Figure 4, consists of 102 stations, 7 MRT lines (including two line extensions), and 114 edges between adjacent stations (as of May 2016). Trip data collected by the AFC system in 2015 December is utilized as one data source in our study. As train operation timetable differs from peak hours to non-peak hours and from weekday to weekend, we, in this paper, study trips happened during weekday morning peak and evening peak respectively, i.e., in Singapore, morning peak is from 7:30am to 9:30am, and evening peak is from 5:30pm to 7:30pm. However, the techniques developed in this paper could be applied to datasets corresponding to other time periods (e.g., non-peak hour of weekday and peak-hour of weekend).

There are in total $10,302$ (i.e., $= 101 \times 102$) OD pairs inside the Singapore MRT network. After generating candidate route sets for all OD pairs as described in Section 3.2, we find that 39.24% of OD pairs have only one candidate route, i.e., single route set OD_s consists of $10,302 \times 39.24\% = 4,042$ OD pairs, as reported in Table 2. When we further check those $4,042$ OD pairs in OD_s, we find out that their routes actually cover each single travel link that might be a component of the travel time T_r of any route r (i.e., a component of Equation (1)).

Table 2. Statistics of Candidate Routes Sets for Singapore and Taipei

	Singapore		Taipei	
	OD_s	OD_m	OD_s	OD_m
Number of OD pairs	4,042	6,260	7,258	4,298
% of OD pairs	39.24%	60.76%	62.81%	37.19%

Table 3. Number of Trips (collected during morning peak within one month) Covering Individual Travel Links

	Number of unique travel links covered	
Number of trips	Singapore	Taipei
(0,100]	10 (2.2%)	6 (1.2%)
(100,1K]	21 (4.7%)	34 (6.9%)
(1K, 10K]	93 (20.8%)	166 (33.7%)
(10K, 100K]	197 (44.0%)	205 (41.6%)
>100K	127 (28.3%)	82 (16.6%)

To be more specific, given a metro system, we could enumerate all the travel links. Take Singapore MRT network as an example. There are 7 service lines, so there are in total 7 travel links corresponding to waiting time for the train service line $T^w_{l_x}$. There are 114 edges, so there are in total 114 travel links corresponding to train travel time T^c_e. There are 102 stations with 19 being interchanges and 83 being normal stations. Each normal station contributes one travel link to entry walking time $T^g_{s_i}$ and one travel link to exit walking time $T^a_{s_i}$, while each interchange could produce multiple travel links to entry walking time $T^g_{s_i}$ and exit walking time $T^a_{s_i}$, dependent on the number of the platforms and the number of exits it has. Take Dhoby Ghaut station as an example. It is passed by 3 lines and has 2 exits located at very different locations, where each unique platform-exit combination produces a unique travel link, so in total it contributes to $3 \times 2 = 6$ travel links to both $T^g_{s_i}$ and $T^a_{s_i}$. In summary, there are 153 travel links corresponding to T^g_s and T^a_s respectively. The number of travel links corresponding to required transfer time T^q_s depends on the number of lines passing by each interchange station and the number of interchange stations. In total, there are 21 travel links. In other words, we have $(7 + 114 + 153 \times 2 + 21) = 448$ travel links corresponding to the Singapore metro system (as of May 2016). If we could derive all those 448 travel links, the travel time of any route could be recovered based on Equation (1). The real trip records corresponding to single route OD pairs captured by the AFC system actually cover each single travel link. Here, we say a trip record covers a travel link if and only if the travel link contributes to the time duration required by the trip.

This observation suggests a possibility that we actually have sufficient trip observations to perform inference of route travel time based *only* on the trips corresponding to the OD pairs in the single route set OD_s. Recall that all the OD pairs $\langle o, d \rangle$s in the single route set OD_s share a common unique feature, that is there is only one route sending a commuter from the origin station o to the destination station d. Accordingly, given an AFC trip record tr from o to d, we know the exact route taken by the commuter for the trip tr. This is to say, we can locate all the travel links travelled by tr without any ambiguity, i.e., the entry/exit station, the links $e_1, e_2, \cdots, e_k$ travelled and the transfer S_m required in Equation (1) are known. In other words, we can take all the trip records trs that are corresponding to OD pairs inside the single route set OD_s to infer the travel time of different travel links. This significantly simplifies the inferring of the travel time, which will be further demonstrated by our experimental study to be presented in Section 5.

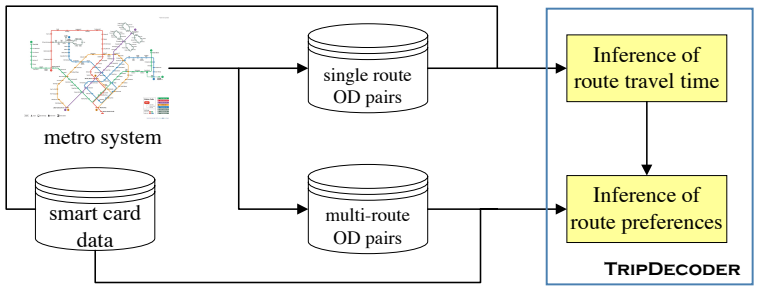


Fig. 5. TripDecoder framework

To further verify our conjecture and test if there are sufficient trip records to perform the inference, for each travel link in the metro network G, we further count the number of trips corresponding to only the single route OD pairs that cover the travel link, as reported in Table 3. Notice that the counts (and the percentages) reported are based on one month of morning peak trip records collected by AFC system in Singapore. As can be observed, 97.8% of the travel links in Singapore MRT network are covered by more than 100 trip records, and 93.1% of the travel links are covered by more than 1,000 trip records, which suggests that the trips corresponding to single-route OD pairs are indeed sufficient to perform robust inferences of the travel time of each travel link. Note that the number of trips will be further increased when the duration corresponding to the data collection is extended. Consequently, we would like to conclude that the finding of the trip records of single-route OD pairs being sufficient to infer the travel time is NOT a coincidence that is only observed from the Singapore dataset. For example, we have performed a similar study on Taipei dataset, again based on one month city-scale data collection. As reported in Table 2 and Table 3, the above statement is also valid on Taipei dataset.

4 SOLUTION ALGORITHMS

As highlighted before, we propose to decouple the inference of route travel time from the inference of route preferences. Accordingly, there are two major components in TripDecoder, the framework proposed in this paper to perform the inference tasks. Figure 5 depicts the architecture of TripDecoder. In the following, we detail how TripDecoder performs these two inference tasks.

4.1 Travel Time Inference

As stated in Equation (1), the travel time of a trip consists of five types of travel links, represented by T_s^g, T_l^w, T_e^c, T_s^q and T_s^a respectively. In this project, we assume that T_s^g, T_l^w, T_e^c, T_s^q and T_s^a all follow normal distribution. We do understand that normal distribution might not be the best distribution to model certain type of travel links (e.g. the waiting time T_l^w for a particular service line). However, the simplicity and additive properties of normal distribution make it (and some of its variants) a very popular choice for modelling some random variables.

$$T_s^g \sim N(\mu_s^g, \sigma_s^g) \tag{2}$$

$$T_l^w \sim N(\mu_l^w, \sigma_l^w) \tag{3}$$

$$T_e^c \sim N(\mu_e^c, \sigma_e^c) \tag{4}$$

$$T_s^q \sim N(\mu_s^q, \sigma_s^q) \tag{5}$$

$$T_s^a \sim N(\mu_s^a, \sigma_s^a) \tag{6}$$

where the probability distribution function of normal distribution $N(\mu, \sigma)$ is defined as

$$N(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-u)^2}{2\sigma^2}} \tag{7}$$

As mentioned before, we assume the waiting time of a service line l is independent on the stations, while different MRT lines can have different $\{\mu_l^w, \sigma_l^w\}$.

For the inference of route travel time, it has to learn the mean and the variance in the normal distribution for each travel link. The travel time from station i to station j which follows a normal distribution is denoted by $T_{ij} \sim N(\mu_{ij}, \sigma_{ij}^2)$. Since T_{ij} assembles the travel time of every travel link covering the route, the distribution of T_{ij} can be approximated by a normal distribution and the mean μ_{ij} and variance σ_{ij}^2 could be derived as follows:

$$T_{ij} \sim N(\mu_{ij}, \sigma_{ij}) \tag{8}$$

$$\mu_{ij} = \mu_{s_i}^g + \mu_{l_x}^w + \sum_{b=1}^{K} \mu_{e_b}^c + \sum_{s \in S_m} \mu_s^q + \mu_{s_j}^a \tag{9}$$

$$\sigma_{ij}^2 = {\sigma_{s_i}^g}^2 + {\sigma_{l_x}^w}^2 + \sum_{b=1}^{K} {\sigma_{e_b}^c}^2 + \sum_{s \in S_m} {\sigma_s^q}^2 + {\sigma_{s_j}^a}^2 \tag{10}$$

For OD pair $\langle i, j \rangle$ with a single route (i.e., $\langle i, j \rangle \in OD_s$), the likelihood of observing travel time t is

$$L(\mu_{ij}, \sigma_{ij}^2; t) = N(t; \mu_{ij}, \sigma_{ij}^2) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{\frac{-(t-u_{ij})^2}{2\sigma_{ij}^2}} \tag{11}$$

Let parameter set Θ [3] include all the travel time parameters to be inferred, i.e., $\{\mu_s^g, \sigma_s^g\}$, $\{\mu_s^q, \sigma_s^q\}$, and $\{\mu_s^a, \sigma_s^a\}$ for all the stations $s \in S$, $\{\mu_e^c, \sigma_e^c\}$ for all the edges $e \in E$, and $\{\mu_l^w, \sigma_l^w\}$ for all the lines $l \in L$. We propose to use maximum likelihood methods to estimate them. Given a set of history trip records TR_s corresponding to the OD pairs in the single route set OD_s, i.e., $TR_s = \{(id_i, o_i, d_i, t_i) | \langle o_i, d_i \rangle \in OD_s\}$, the full likelihood of TR_s is

$$L(\Theta; TR_s) = \prod_{i=1}^{|TR_s|} \left[\frac{1}{\sqrt{2\pi\sigma_{o_i d_i}^2}} e^{\frac{-(t_i - u_{o_i d_i})^2}{2\sigma_{o_i d_i}^2}} \right] \tag{12}$$

Taking the logarithm of Equation (12), we can get the log-likelihood as

$$l(\Theta; TR_s) = \sum_{i=1}^{|TR_s|} \left[-\frac{1}{2} \ln(2\pi\sigma_{o_i d_i}^2) - \frac{(t_i - \mu_{o_i d_i})^2}{2\sigma_{o_i d_i}^2} \right] \tag{13}$$

To begin with, we initialize Θ based on prior knowledge of the metro network or empirical observations. Take Singapore MRT network as an example. μ_e^c is set based on statistics provided by Singapore Land Transport Authority (LTA), and the corresponding σ_e^c is set to $\mu_e^c/10$. $\{\mu_s^g, \sigma_s^g\}$, $\{\mu_s^q, \sigma_s^q\}$, $\{\mu_s^a, \sigma_s^a\}$ and $\{\mu_l^w, \sigma_l^w\}$ are set based on empirical observations, as reported in Table 4. Thereafter, we perform stochastic gradient descent (SGD) to tune the parameters by maximizing $l(\Theta; TR_s)$.

[3] Θ is time dependent, as the train frequency of metro systems in most cities varies from peak hour to non-peak hour, from weekday to weekend.

Table 4. Initial Values for Travel Steps

Travel link	μ (seconds)	σ (seconds)
Normal station entry/exit walking	60	12
Interchange entry/exit walking	120	24
Transfer walking	60	12
Train service waiting	tf * 0.5	tf * 0.05

[a] tf represents train frequency of a MRT line

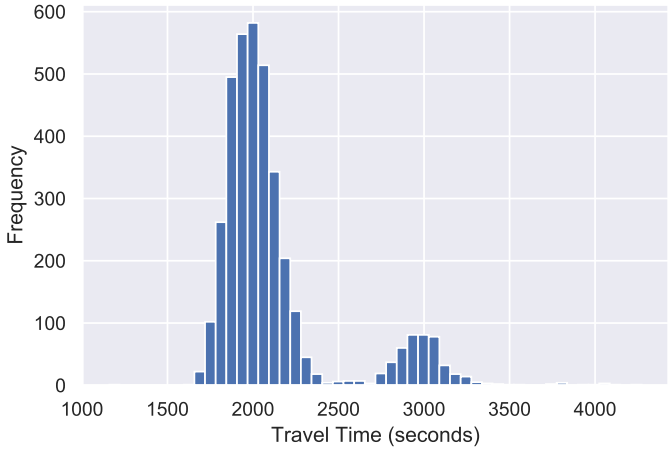


Fig. 6. Travel time observations from Bishan to Jurong East

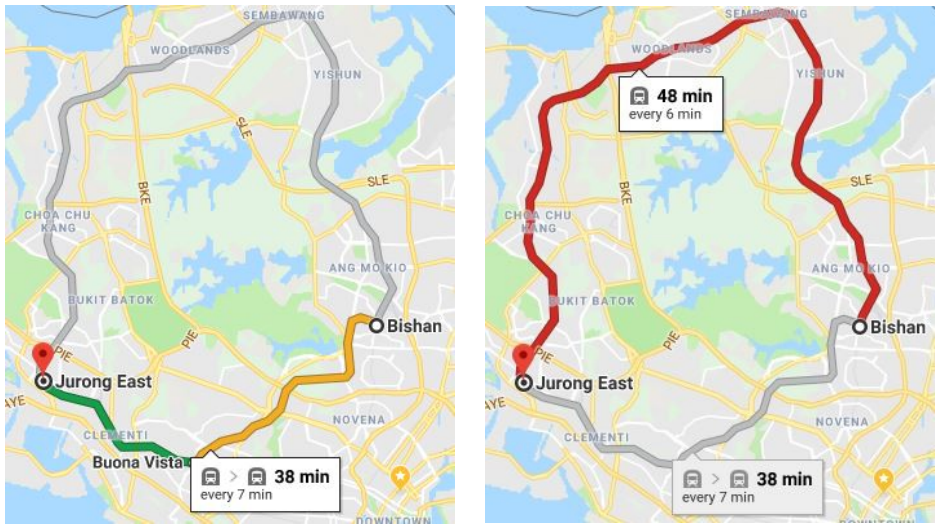
4.2 Route Preferences Inference

Based on travel time deduced in Section 4.1, we are ready to assemble travel time distribution of any route. For a given OD pair $\langle o, d\rangle \in OD_m$, each candidate route in R_{od} has a unique travel time distribution. Take the routes from Bishan station to Jurong East station as an example. The real travel time distribution based on TR_{od} is depicted in Figure 6, where we could observe two different patterns, one pattern having an average travel time of about 33 minutes, and the other having an average travel time of about 50 minutes. It is very likely that these two patterns of the travel time represent the two different candidate routes, as suggested by Google Map shown in Figure 7. Thus, TR_{od} can be modeled as a mixture of distributions from the different candidate routes,

$$TR_{od} \sim \sum_{r \in R_{od}} \pi(r) N(t; \mu_r, \sigma_r^2) \tag{14}$$

where μ_r and σ_r of each candidate route have already been derived in Section 4.1. $\pi(r)$ refers to the probability that commuters take r when traveling from o to d and thus $\sum_{r \in R_{od}} \pi(r) = 1$.

Assume we have a set of historical trip records TR_m corresponding to the OD pairs in the multiple route set OD_m, i.e., $TR_m = \{(id, o_i, d_i, t_i) | \langle o_i, d_i\rangle \in OD_m\}$. Let Π represent the set of parameters related to route preference to be derived, i.e., $\Pi = \{pi(r) | r \in R_{od} \wedge |R_{od}| > 1\}$. Then, the full likelihood of TR_m can be written as,



Route 1 Route 2

Fig. 7. Two candidate routes from Bishan to Jurong East on Google Map

$$L(\Pi; TR_m) = \prod_{i=1}^{|TR_m|} [\sum_{r \in R_{o_i d_i}} \pi(r) N(t_i; \mu_r, \sigma_r^2)] \tag{15}$$

Taking the logarithm of Equation (15), we get the log-likelihood as

$$l(\Pi; TR_m) = \sum_{i=1}^{|TR_m|} \ln(\sum_{r \in R_{o_i d_i}} \pi(r) N(t_i; \mu_r, \sigma_r^2)) \tag{16}$$

To begin with, we assign equal likelihood to every candidate route of any OD pairs in OD_m, i.e.,

$$\pi(r) = 1/|R_{od}|, \forall r \in R_{od} \ \wedge \ \forall \langle o, d \rangle \in OD_m$$

To get the maximum of Equation (16), again we perform SGD to tune route preferences Π.

5 CASE STUDY

To demonstrate the superior performance of TripDecoder, we conduct studies based on two real sets of city-scale trip records collected in Singapore and Taipei. As mentioned before, we consider both the accuracy and the efficiency as the goals when we design TripDecoder. Consequently, we will compare the accuracy and the efficiency of TripDecoder with the state-of-the-art works. In the following, we first briefly explain the real datasets used in this study, and then present the two sets of experiments that evaluate the accuracy and the efficiency of TripDecoder as well as other state-of-the-art works.

5.1 Experiments Settings

5.1.1 Datasets. Our study is based on two sets of city-scale real trip records, captured by the AFC system in Singapore and that in Taipei respectively. In particular, EZ-Link card and EasyCard are used as the smart cards for payment of public transport in Singapore and Taipei, respectively. EZ-Link card data collected in Singapore in December 2015, and EasyCard data collected in Taipei

Table 5. Smart Card Data Sample

id	type	entry date	entry time	exit date	exit time	origin id	destination id
02***5F	adult	2015-12-02	08:20:04	2015-12-02	08:27:27	35	12
02***5F	adult	2015-12-02	18:13:57	2015-12-02	18:21:25	12	35
02***5F	adult	2015-12-03	08:13:51	2015-12-03	08:21:21	35	12
02***5F	adult	2015-12-03	18:31:45	2015-12-03	18:38:11	12	35
02***5F	adult	2015-12-03	18:47:45	2015-12-03	19:01:16	35	12

Table 6. Smart Card Dataset Attributes

attribute	notation	description
id	c_{id}	unique identifier of a smart card
type	$type$	commuter type (i.e., child, adult, senior)
entry date	$date_{in}$	starting date of a ride
exit date	$date_{out}$	ending date of a ride
entry time	t_{in}	starting time of a ride
exit time	t_{out}	ending time of a ride
origin id	id_{in}	unique identifier of the origin MRT station/bus stop
destination id	id_{out}	unique identifier of the destination MRT station/bus stop

in August 2018 are used in this study. As our study is based on metro networks, we exclude the data corresponding to bus rides in this study.

As listed in Table 5, each EZ-Link record in our data collection is corresponding to one MRT ride, including the boarding and alighting MRT stations and the corresponding timestamps. Other information such as the commuter type and the fare charge are also recorded. Apart from that, each smart card is associated with an encrypted unique identifier, so that we can identify all the rides taken by one commuter with commuter's real identity being well protected. Table 6 lists the attributes captured by each EZ-Link record. Taipei EasyCard data is no different from EZLink data except that it doesn't release the encrypted unique identifier of each card so we are not able to differentiate the trips made by one commuter from those made by other commuters.

Due to defects of an AFC system, sometimes it generates duplicate records or trip records with unrealistic travel duration (e.g., trips that last more than multiple hours or less than two minutes). These records may bias our analytics and hence are removed.

5.1.2 Baselines. Our proposed method TripDecoder is compared against a naive average method, commercial Apps and academic research works. First, the naive average method, denoted as *Historical Average*, simply utilizes the historical average travel time corresponding to each OD pair $\langle o, d \rangle$ as the predicted travel time of any route from o to d. Second, *Google Map*, the most popular direction service in both Singapore and Taipei, is included as the main representative of commercial Apps. In addition, we also include *Gothere* (https://gothere.sg), a very popular direction service used in Singapore. Third, the inference model published in NIPS 2017 [2] is the latest work which is employed as the representative of the state-of-the-art academic research work, denoted as *NIPS*.

5.1.3 Metrics. We employ the *prediction error* and the *execution time* to evaluate model effectiveness and efficiency respectively. Given a set of trip observations X and a prediction method ρ, let μ_{od} be the expected travel time of route r_{od} predicted by ρ. The prediction error of travel time of ρ is

Table 7. Travel Time Prediction Error

	Singapore		Taipei	
	Morning Peak	Evening Peak	Morning Peak	Evening Peak
TripDecoder	**8.53**%	**10.38**%	10.26%	**10.88**%
Historical Average	8.59%	10.70%	**10.03%**	10.93%
NIPS	17.40%	20.19%	19.63%	21.67%
Google Map	19.94%	19.49%	23.67%	25.45%
Gothere	12.06%	13.57%	-	-

defined as:

$$error_{X,\rho} = \frac{1}{|X|} \sum_{(s_o,s_d,t)\in X} \frac{|\mu_{od} - t|}{t} \tag{17}$$

The execution time reported in this paper is obtained by running the two inference tasks on Microsoft Windows 10 Education instances, each of which is shipped with a Intel Core i8-8700 CPU @3.20GHz and a 32.0GB RAM.

5.2 Accuracy Evaluation

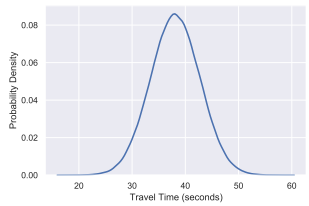
TripDecoder deduces both travel time parameters Θ and route preference Π. To report its performance in a more comprehensive way, we conduct different sets of experiments to compare the performance of TripDecoder with its competitors, for both the prediction error of derived Θ (corresponding to the inference of travel time) and that of Π (corresponding to the inference of route preferences).

Evaluation of Travel Time Parameters. In Section 4.1, we use trip observations set TR_s to infer travel link time parameters, which are then used to construct travel time of any route. Recall that trip observations in TR_s are corresponding to OD pairs in OD_s, the set of OD pairs having *only one* route. In other words, for any trip $tr \in TR_s$, it corresponds to one OD pair $\langle o, d\rangle$ in OD_s that has exactly one route only. Accordingly, we know the exact route taken by tr and hence all the exact travel links that contribute to the travel time of tr. Consequently, we can derive μ_{od} for each trip record tr in TR_s and derive the prediction error following Equation (17).

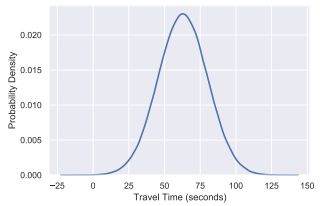
Table 7 reports the prediction error of TripDecoder and its competitors, with numbers in bold indicating the best performers. For both Google Map and Gothere, we submit 20 queries for each OD $\langle o, d\rangle$ pair in OD_s and report the average performance. Each of those 20 queries has the boarding station o as its current location, alighting station d as its destination, and the trip start time is randomly selected from the duration (e.g., for morning peak in Singapore, the trip start time is randomly selected from 7:30am to 9:30am). As could be observed from the results, TripDecoder and Historical Average produce very similar results, i.e., μ_{od} derived from TripDecoder is very close to its corresponding empirical mean $\overline{T_{od}}$. This is consistent with our expectation. Given an OD pair $\langle o, d\rangle$ in OD_s, its travel time is assumed to follow normal distribution and is estimated based on maximum likelihood method, and hence the derived μ_{od} is expected to be very close to $\overline{T_{od}}$. Actually, $\overline{T_{od}}$ is the analytical solution of μ_{od}, for $\langle o, d\rangle \in OD_s$. As compared to other competitors, TripDecoder demonstrates a superior accuracy performance. For example, TripDecoder reduces the prediction error of Google Map by 14.25% and 17.57%, for Singapore dataset and Taipei dataset respectively.

For demonstration purpose, we decompose the route from Buona Vista station to Jurong East station in Singapore, and report the travel time distribution for each of its travel links in Figure 8.

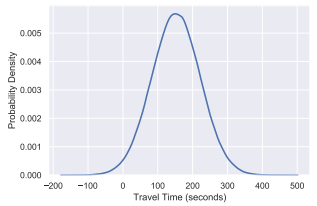
Note that both Buona Vista station and Jurong East station are located at East-West (EW) line, as shown in Figure 4. They are EW21 and EW24 respectively, about 3 stations away along the EW line. There are in total six travel links that contribute to the travel time from Buona Vista station to Jurong East station, as reported in Figure 8, including a) entry walking time at Buona Vista station, b) waiting time for EW line at Buona Vista station, c) train travel time from Buona Vista (EW21) to Dover station (EW22), d) train travel time from Dover (EW22) to Clementi station (EW23), e) train travel time from Clementi (EW23) to Jurong East station (EW24), and f) exit walking time at Jurong East station.



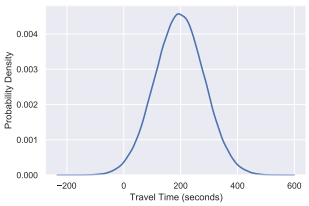
(a) Entry walking time at EW21 (μ=38)



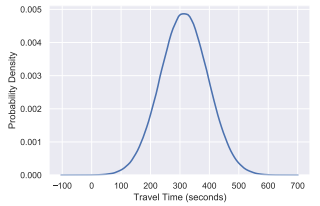
(b) Waiting time at EW21 (μ=63)



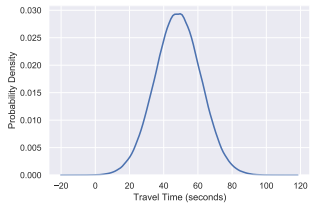
(c) Train travel time from EW21 to EW22 (μ=152)



(d) Train travel time from EW22 to EW23 (μ = 196)



(e) Train travel time from EW23 to EW24 (μ=314)



(f) Exit walking time at EW24 (μ=48)

Fig. 8. Travel time distribution of travel links for trips from Buona Vista (EW21) to Jurong East (EW24)

Evaluation of Route Preferences. As stated in Definition 3.2, the inference of route preferences is to infer the likelihood that a commuter, when travelling from o to d, takes a specific route $r \in R_{od}$ when the route candidate set R_{od} has multiple routes, i.e., $\langle o, d\rangle \in OD_m$. However, we do not have the ground truth of the exact routes taken by commuters. Consequently, how to evaluate the accuracy of the second inference task remains challenging. In this study, we propose a novel evaluation plan that is based on an assumption that a commuter who has to regularly travel from station o to station d has her own preferences, if there are multiple routes available. The main idea is to learn individual commuter's preferred route for the trip from station o to station d based on the historical trips from o to d made by the commuter, and use the learned individual route preference as the prediction of the route taken by the commuter when she is about to make the trip again from o to d. Note, $\langle o, d\rangle$ considered in this evaluation plan must be from OD_m.

In order to implement this evaluation plan, we strategically group the trip records in TR_m based on $\langle id, o, d\rangle$. That is to say that trip records sharing the same id, o and d values form one group, denoted as TR^{id}_{od}, i.e., the trip records in TR^{id}_{od} correspond to all the trips from o to d made by one single commuter. Formally, $TR_m = \cup_{\langle o,d\rangle \in OD_m \wedge id \in \cup id} TR^{id}_{od}$, such that $\forall id \neq id' \vee \langle o, d\rangle \neq \langle o', d'\rangle$,

$TR^{id}_{od} \cap TR^{id'}_{o'd'} = \emptyset$. We next order the groups according to the descending order of their cardinality $|TR^{id}_{od}|$, the number of trip records in each group. We then select the top 10% of the groups based on their set cardinality. In Singapore dataset, top 10% groups have their set cardinality ranging between 20 and 23. For each of such selected groups TR^{id}_{od}, we partition the trip records into two disjoint subsets based on the ratio of 1 : 1, denoted as $TR^{id}_{od\_train}$ and $TR^{id}_{od\_test}$, with $TR^{id}_{od\_train}$ being the training subset and $TR^{id}_{od\_test}$ being the testing subset. We then feed the training subset $TR^{id}_{od\_train}$ to TripDecoder to learn the preferred route r_{od} of the commuter whose encrypted identifier is id when travelling from o to d, and use the learned preferred route r_{od} to predict the route taken by the commuter for the trips (again from o to d) in the testing subset. Note that we have learned the travel time corresponding to each travel link in the travel time inference task, so we can derive the time required by the preferred route r_{od} and compare that with the ground truth travel time ($t_{out} - t_{in}$) captured by the trips in $TR^{id}_{od\_test}$ to calculate the prediction error rate based on Equation (17).

Table 8. Route Preferences Prediction Error (dataset: Singapore)

	Morning Peak	Evening Peak
TripDecoder	**6.20**%	**7.02**%
Historical Average	7.40%	9.27%
Shortest Route	7.40%	8.00%
NIPS	15.99%	16.52%

Table 8 shows the prediction error of TripDecoder and NIPS by using Singapore dataset. We exclude Taipei dataset from this set of experiments as *id* is not available in Taipei dataset. Note that in addition to NIPS, we also implement the *Historical Average* method presented in Section 5.1.2, which predicts the travel time of any trip in the testing set as the empirical mean $\overline{T_{od}}$, and *Shortest Route*, which refers to a very common assumption made by many existing works, i.e., commuters tend to take the shortest route when there are multiple routes available. We want to highlight that although we include *Historical Average* as one competitor, it is NOT able to differentiate between/among different candidate routes corresponding to the same $\langle o, d \rangle$ pair and hence cannot reveal the actual routes taken by commuters. Again, TripDecoder incurs the lowest prediction error. It is worth noting that TripDecoder outperforms *Shortest Route* even in the morning peak, when most of the regular commuters are expected to be more sensitive to the travel time required. It also implies that commuters do not always take the shortest route even during weekday morning peak.

5.3 Efficiency Evaluation

As we mentioned in Section 1, TripDecoder is designed to not only improve the accuracy of the two inference tasks but also address the efficiency issue. Ideally, we prefer a model that can complete the inference tasks with low prediction error within a short duration of time. Therefore, we can evaluate the efficiency of a model in two perspectives, including *execution time* and the *rate of convergence*.

Figure 9 reports the execution time of 50 iterations with the size of training data varied. It can be seen that the execution time of both models increases linearly. However, the execution time of TripDecoder grows slightly whereas that of NIPS grows significantly. Consequently, TripDecoder has a much better scalability and is more suitable to process city-scale trip records.

We also study the number of iterations required by the inference models to achieve a stable prediction error and report the result in Figure 10(a). First, we could observe that both TripDecoder and NIPS are able to achieve a higher prediction accuracy as the number of iterations increases.

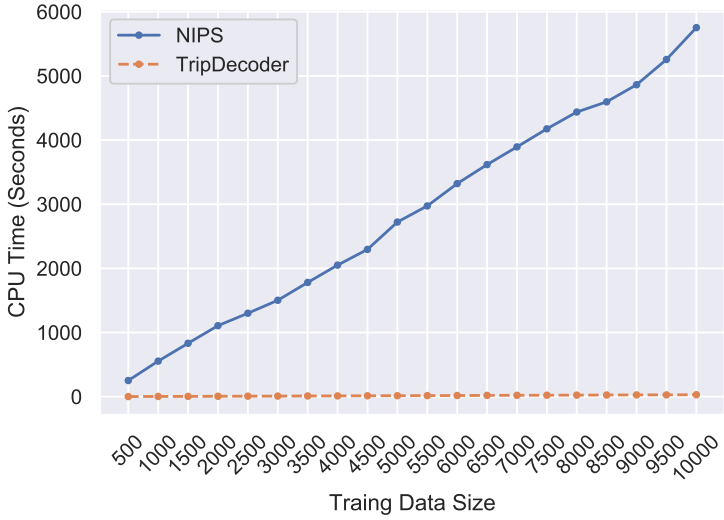
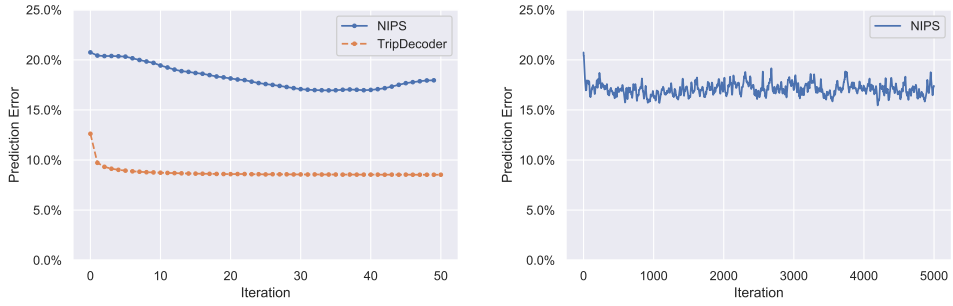


Fig. 9. CPU time of 50 iterations vs. training data size represented by the total number of trip records



(a) TripDecoder vs. NIPS (b) NIPS

Fig. 10. Travel time prediction error vs. training iterations

Second, prediction error of TripDecoder actually decreases significantly in the first few iterations of training and achieves a stable and close-to-optimal performance only after a small number of iterations; whereas NIPS cannot decrease its prediction error in the first 50 iterations steadily. To further investigate the convergence rate of NIPS, we keep tracking the prediction error of NIPS by increasing the number of iterations from 50 to 5000, as reported in Figure 10(b). It can be observed that the prediction error of NIPS keeps vibrating and tends not to converge. We can conclude that TripDecoder demonstrates a superior efficiency and the so-called super convergence capability which is very desirable in model training whereas NIPS results in limited scalabity in practice.

6 CONCLUSION

In this paper, we target at recovering the exact routes taken by commuters inside a metro system that are not captured by an AFC system and hence remain unknown. In 2016, London Tube system run a 4-weeks' trial to log more than 500 million WiFi connection requests from around 5.6 million

devices. One of the main objectives was to track the journeys around the network and to recover how commuters move inside the network. Without incurring additional cost, TripDecoder is able to achieve the same goal based on available data already captured by an AFC system.

We strategically propose two inference tasks to handle the recovering, one to infer the travel time of each travel link that contributes to the total duration of any trip inside a metro network and the other to infer the route preference based on historical trip records and the travel time of each travel link inferred in the previous inference task. As these two inference tasks have interrelationship, most of existing works perform these two tasks simultaneously. However, we adopt a totally different approach when we design TripDecoder. TripDecoder fully utilizes the fact that there are some trips inside a metro system with only one practical route available and smartly decouples these two inference tasks. To be more specific, it only takes those trip records with only one practical route as the input for the first inference task of travel time, and feeds the inferred travel time to the second inference task as an additional input, which not only improves the accuracy of both inference tasks but also effectively reduces the complexity of the inference tasks. We have conducted comprehensive experiments based on real data captured by AFC systems in Singapore and Taipei to compare the performance of TripDecoder and its competitors, including both commercial services and academic contributions. Consistent with our expectation, TripDecoder has demonstrated a much better performance in terms of both the accuracy and the efficiency. In the near future, we plan to extend TripDecoder to predict the commuting flows of each individual stations inside a metro network.

ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Singapore under its International Research Centres in Singapore Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

REFERENCES

[1] Ka Kee Alfred Chu and Robert Chapleau. 2010. Augmenting transit trip characterization and travel behavior comprehension: Multiday location-stamped smart card transactions. *Transportation Research Record* 2183, 1 (2010), 29–40.

[2] Nicolò Colombo, Ricardo Silva, and Soong Moon Kang. 2017. Tomography of the London underground: a scalable model for origin-destination data. In *Advances in Neural Information Processing Systems*. 3062–3073.

[3] Thomas Holleczek, Shanyang Yin, Yunye Jin, Spiros Antonatos, Han Leong Goh, Samantha Low, Amy Shi-Nash, et al. 2015. Traffic measurement and route recommendation system for mass rapid transit (mrt). In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1859–1868.

[4] Ling Hong, Wei Li, and Wei Zhu. 2017. Assigning passenger flows on a metro network based on automatic fare collection data and timetable. *Discrete Dynamics in Nature and Society* 2017 (2017).

[5] Fanglei Jin, Enjian Yao, Yongsheng Zhang, and Shasha Liu. 2017. Metro passengers' route choice model and its application considering perceived transfer threshold. *PloS one* 12, 9 (2017), e0185349.

[6] Yulin Liu, Jonathan Bunker, and Luis Ferreira. 2010. Transit Users' Route-Choice Modelling in Transit Assignment: A Review. *Transport Reviews* 30, 6 (2010), 753–769.

[7] Shoichiro Nakayama and Ryuichi Kitamura. 2000. Route choice model with inductive learning. *Transportation Research Record* 1725, 1 (2000), 63–70.

[8] Bo Friis Nielsen, Laura Frølich, Otto Anker Nielsen, and Dorte Filges. 2014. Estimating passenger numbers in trains using existing weighing capabilities. *Transportmetrica A: Transport Science* 10, 6 (2014), 502–517.

[9] Agostino Nuzzolo, U Crisalli, Luca Rosati, and Angel Ibeas. 2013. STOP: A Short term Transit Occupancy Prediction tool for APTIS and real time transit management systems. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. 1894–1899.

[10] Peter Rickwood and Garry Glazebrook. 2009. Urban structure and commuting in Australian cities. *Urban Policy and Research* 27, 2 (2009), 171–188.

[11] Ricardo Silva, Soong Moon Kang, and Edoardo M. Airoldi. 2015. Predicting traffic volumes and estimating the effects of shocks in massive transportation systems. *Proceedings of the National Academy of Sciences* 112, 18 (2015), 5643–5648.

[12] Tony Smith, Chao-Che Hsu, and Yueh-Ling Hsu. 2008. Stochastic User Equilibrium Model with Implicit Travel Time Budget Constraint. *Transportation Research Record: Journal of the Transportation Research Board* 2085 (2008), 95–103.

[13] Guandong Sun, Yun Xiong, and Yangyong Zhu. 2017. How the passengers flow in complex metro networks?. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*. ACM, 23.

[14] Lijun Sun and Jian Gang Jin. 2015. Modeling Temporal Flow Assignment in Metro Networks Using Smart Card Data. *2015 IEEE 18th International Conference on Intelligent Transportation Systems* (2015), 836–841.

[15] Lijun Sun, Yang Lu, Jian Gang Jin, Der-Horng Lee, and Kay W. Axhausen. 2015. An integrated Bayesian approach for passenger flow assignment in metro networks. *Transportation Research Part C: Emerging Technologies* 52 (2015), 116 – 131.

[16] Xiancan Tian and Baihua Zheng. [n.d.]. Using smart card data to model commuters' response upon unexpected train delays. In *Proceedings of the IEEE Conference on Big Data*. 831 – 840.

[17] Gilles Vandewiele, Pieter Colpaert, Olivier Janssens, Joachim Van Herwegen, Ruben Verborgh, Erik Mannens, Femke Ongenae, and Filip De Turck. 2017. Predicting train occupancies based on query logs and external data sources. In *Proceedings of the 7th International Workshop on Location and the Web*.

[18] Yazhe Wang, Chih-Chieh Hung, Baihua Zheng, and Ee-Peng Lim. 2018. TripDecoder: Inferring Routes of Passengers of Mass Rapid Transit Systems by Smart Card Transaction Data. In *Proceedings of the IEEE International Conference on Data Mining Workshops*.

[19] Xinyue Xu, Liping Xie, Haiying Li, and Lingqiao Qin. 2018. Learning the Route Choice Behavior of Subway Passengers from AFC Dat. *Expert Systems with Applications* 95 (2018), 324–332.

[20] Haodong Yin, Baoming Han, Dewei Li, Jianjun Wu, and Huijun Sun. 2016. Modeling and Simulating Passenger Behavior for a Station Closure in a Rail Transit Network. *PloS one* 11, 12 (2016).