Efficient Constrained Planning for Deploying Additional Route on Existing Mass Transportation System with Neural-Network-Based Passenger Flow Inference

Abstract

In this work, a novel decision assistant system, called Route Scheme Assistant (RSA), is proposed to address two crucial research issues that few former researches have focused on, route-based passenger flow (PF) inference and multi-variant high-PF route planning. First, RSA can estimate PF of arbitrary user-designated routes effectively utilizing Deep Neural Network (DNN) for regression based on spatial-temporal urban informatics. Second, the proposed Bidirectional Prioritized Spanning Tree (BDPST) intelligently combines the parallel computing concept and Gaussian mixture model (GMM) for route recommendation under users' constraints running in a deliverly time. We did experiments on bus-ticket data of Tainan; the results show that the PF inference model outperforms baseline and comparative methods from 41% to 57%. Moreover, the proposed BDPST algorithm's performance is not too far away from the optimal PF and outperforms other comparative state-of-art route planning methods from 8% to 23% in large-scale route recommendations.

Introduction

Traffic deployment is highly correlated with the quality of life (Steg and Gifford 2005). For residents, they care about whether transportation construction can bring convenience. For traffic management authorities[1], they pay attention to public perception, fund allocation and public benefits. However, constructing unwanted and redundant routes or stations can lead to environmental damage and resource waste. Besides, according to our interview with civil servants in the bureau of transportation, they pointed out that the current procedure in planning new routes turns out to be lengthy due to many stakeholders involved in them. Also, the overwhelming number of requests from the public makes it difficult to decide where to construct new routes and stations.

Therefore, in this work we propose Route Scheme Assistant (RSA), a decision assistant system with the proposed Bidirectional Prioritized Spanning Tree (BDPST) algorithm, to combine and address two crucial research issues that few former researches have focused on, one is the route-based passenger flow (PF) inference and the other one is multi-variant high-PF route recommendation. These two research questions are important and beneficial for urban transportation and planning. An accurate model for inferring route-based passenger flow can help managers pre-assess the effectiveness of arbitrary new route service in advance before deployment. Therefore, for any arbitrary route proposed by a user, RSA provides an effective PF inference framework for governments or transportation companies to evaluate and verify whether the route meets the needs of the public, i.e., having a certain number of passengers to take such route. We solve the inference problem where the urban-relevant features are extracted based on our newly defined route-affecting region that take nearby areas of the routes into consideration. Given designated route with its stations labelled from users, route-based urban-relevant features are extracted and put into our inference model, the passenger flow in a certain time interval is then return to users.

On the other hand, the research question of high-PF route recommendation is also important for reality. In some cases, decision makers prefer that they can directly obtain recommended routes that will have high potential PF given users' constraints but without undergoing simulation processes of traffic planning software. Therefore, we proposed BDPST algorithm for route recommendation. By jointly utilizing our proposed inference model and BDPST algorithm, RSA can recommend routes and stations with high potential PF in a certain extent on the map along with some must-visit stations assigned by users. We combine the proposed two research solutions together with some visualization and interactive auxiliary functions to form RSA and the system structure is shown in Figure 1. More specifically, the second research issue focuses on a route planning problem where the heuristics for route planning are none-monotonic which makes most of the state-of-art route-planning algorithms (e.g. A*, Dijkstra's) not applicable. Given a set of must-visit stations and a range on the map from users, our newly proposed algorithm recommends a route with near-optimal passenger flow (based on the aforementioned inference model) per unit length.

[1] We use the terms traffic management authority and user interchangeably.

Table 1: Summary of formulations for research on designing new transportation routes

	Feature extraction											
	Station deployment	Interval deployment	Surrounding POI	Route transference	Route competition	Population structure	Human mobility	Network structure	Waiting time	Journey interval	Station distance	Trajectory length
Mauttone and Urquhart (2009)	■			■		■		■				■
Quadrifoglio and Li (2009)	■			■							■	■
Szeto and Wu (2011)	■	■		■	■			■	■			
Cancela, Mauttone and Urquhart (2015)	■			■		■		■	■			■
Pternea, Kepaptsoglou and Karlaftis (2015)	■			■				■	■			■
Ours	■		■	■	■	■	■	■		■	■	■

The number of passenger flows is undoubtedly an important indicator to measure the effectiveness of the construction. A plethora of methodologies and frameworks designed new public transportation routes relying on survey data in constructing origin-destination matrices (OD-based). However, none of them investigated and solved the research problem of inferring route-based PF by adopting route-based machine learning techniques. Moreover, in addition to origin and destination, we propose that geographical environment and urban functions of the trajectories and passing areas among stations could also affect the demand for deploying public transportation. The novelty for PF inference could sum up in the proposed feature extraction and engineering framework for temporal and spatial features of transportation routes. We are the first work to not only define route-affecting region (RAR) to model the influential region of routes, but also investigate the correlations between heterogeneous features in RAR and passenger flow.

To best of our knowledge, most of the state-of-art route-planning algorithms are not applicable in addressing the issue of route recommendations with none-monotonic heuristics / features. Although several traffic planning softwares (e.g. VISUM[2], EMME[3]) are already developed and discussed, governments or traffic management authority are still often required to provide effectiveness evaluations or near-optimal plans in a timely manner when facing overwhelming number of requests for deploying new stations or routes from the public. Therefore, we extend and combine numerous route planning techniques to propose an effective solution which utilize GMM (combined with the results of inference model from the first part) along with negative feedback based on multi-source bidirectional techniques in preventing crowding-out effect. Experimental results show the effectiveness and the efficiency of not only proposed RAR-based extraction strategy compared to traditional OD-based strategy, but also the newly proposed algorithm compared to both comparative and state-of-art route-planning algorithms.

Backgrounds

PF inference. Some works in designing new transportation routes focused on reducing transportation time through route adjustment or shift (Cancela, Mauttone and Urquhart 2015)(Mauttone and Urquhart 2009)(Pternea, Kepaptsoglou and Karlaftis 2015)(Quadrifoglio and Li 2009)(Szeto and Wu 2011). There are also some works that optimized route planning with distance, time, transference considered (Guihaire and Hao 2010)(Yan et al. 2012). Some works studied the problem of predicting arrival time (Chien, Ding and Wei 2002)(Lin et al. 2013)(Cheng, Liu and Zhai 2010) or future PF (Arabghalizi and Labrinidis 2019)(Yap, Cats and Arem 2018)(Wei and Chen 2012) based on regression analysis. However, the above works focused on dealing with existing routes, which are not our target problem.

Therefore, by focusing on designing new transportation routes, Table 1 first presents a summary of formulations for previous works, listing the aspects each research approached. By analyzing the considered and extracted features, we generalize six kinds of relevant urban features in inferring the passenger flow for new routes deployed in transportation networks. Most importantly, all the previous works utilized these features on an OD-based analysis.

Route planning algorithms. Research on route planning algorithms in transportation networks has developed over years, where the network is usually modeled as a directed graph in order to utilize Dijkstra's algorithm to compute a best route between two nodes (Delling et al. 2009). Several improvements have been made to run Dijkstra's algorithm in almost linear time or using little memory thereafter (Abraham et al. 2011) (Goldberg 2008). However, it is too slow for practical applications in real-world transportation networks, which consist of millions of nodes (grids), while instant results are requested (Julian, Thomas and Dorothea 2015). Therefore, to speed up searching an unimodal transportation network, bidirectional search, goal direction

[2] http://vision-traffic.ptvgroup.com/en-us/products/ptv-visum/

[3] https://www.inrosoftware.com/en/products/emme/

(Goldberg and Harrelson 2005)(Wagner, Willhalm and Zaroliagis 2005) (Yoshizumi, Miura and Ishida 2000), transportation hierarchy, distance table, and separator-based methods (Delling et al. 2011) are generally used (Bast 2009)(Bast et al. 2010)(Delling et al. 2009)(Ehrgott and Klamroth 1997)(Zhang, Kabadi and Punnen 2011).

However, some of the relevant urban features (e.g. the entropy of urban functions, or the relationship between new and existing routes) generalized in inferring the PF is not superimposable, making the standard solution for one-to-all path problem (Dijkstra's algorithm, which updates values by superimposing connections iteratively) (Dijkstra 1959) inappropriate for our case. Therefore, using these speedup techniques mentioned above, along with the concept of multivariate Gaussian mixture model, which is mostly utilized in solving background modeling for real-time tracking (Stauffer and Grimson 1999), we propose a target-prioritized searching algorithm in this work.

Preliminaries

In this section, we introduce the backgrounds and their key concepts, formally definitions of our two problems would thereafter be defined in respective section.

Definition 1: *Grid*. We divide the city into disjointed grids and store all features which are correlated with passenger flow (e.g. number of each type of POI located in this grid, human mobility data that pick-up / drop-off at this grid, whether existing routes passed this grid, etc.) into corresponding grid.

Definition 2: *Grid-like graph*. Grid-like graph is composed of disjointed grids that records connections as original road network based on OpenStreetMap (OSM). Each grid stores the connections between adjacent grids in its eight directions if there exists a road in OSM that connects each other.

Definition 3: *Station*. Station is a facility or area for passenger to regularly get-into or get-off the mass transit transportation. (Note that the mass transit transportation here refers to city bus, light rails, trolley bus, etc.) Passenger need to pay by smart card when getting-into or/and getting-off (depends on authority) the mass transit at a station. Station in original mass transit data or as input given by users as system is a point with latitude and longitude; but turns into a grid that the point located at in the grid-like graph.

Definition 4: *Trajectory*. Trajectory is the path that certain mass transit regularly takes between two stations. Trajectory in original mass transit data or as input given by users as system is a series of road junctions; but turns into a series of connecting grids in the graph-like grids for further PF inference and route recommendation.

Definition 5: *Route*. Route is a set of combination of trajectory and stations. Note that same series of trajectories with different set of stations does refer to different route. Route is a series of connecting grids and several grids labelled as stations in the grid-like graph; however, since we divide the city into disjointed grids with a meticulous size, the actual route in real world (OSM) can be easily reproduced given a sequence of grids. Therefore, though some re-projections from grid to actual road network are needed, no other superfluous process needed to handle in post-processing.

Definition 6: *Passenger Flow (PF)*. Since the price is fixed fare for Chicago and Tainan bus transit system, and most of mass transit transportation system in other cities, the passenger flow along the route here refers to the total passengers who passed any point along the route. To be more specific, passenger flow is counted once someone pay by smart card when getting-into or getting-off the mass transit at a station of a route. Note that passenger flow would not be counted twice if one pay when getting-into and -off the same transit.

Definition 7: *Transference*. Direct transference between transits are not allowed in bus transit system of both cities, which means one must get-off from first transit before getting-into the other at a station. Since one must pay at least once when getting-off the first transit or getting-into the other, records of passenger flow for these two routes are secured. We assume that transference between transits are plausible if the distance between stations that belongs to different transits is below the walking tolerance for pedestrians.

Definition 8: *Origin-destination (OD) matrix*. An origin-destination (OD) matrix is essential for efficient traffic control and management (Yang and Zhou 1998), which has been utilized in modeling congestion and estimating travel time by specifying the travel demands between two nodes in network. The input data is usually based on survey data from the region of origin and destination (Peterson 2007).

Definition 9: *Admissibility*. A search algorithm is admissible if it is guaranteed to find a minimal path to a solution whenever such a solution exists. That is, the cost to reach the destination is never overestimated. For example, breadth-first-search is admissible since it enumerates every states before considering states at next level.

Definition 10: *Monotonicity*. A function is monotonic if and only if it is either entirely non-increasing or entirely non-decreasing. To be more specific, f(x) is monotonic if for all x and y such that $x \leq y$ one has either $f(x) \leq f(y)$ or $f(x) \geq f(y)$. That is, the costs between any two states are always underestimated throughout the searching space. Therefore, for any admissible cost function f, one can always construct a monotone admissible function f' which is at least as informed as f. (Korf 1985)

Definition 11: *Heuristic algorithms*. A well-known and state-of-art heuristic algorithm is A* algorithm (Hart, Nilsson and Raphael 1968). A* could always find the path with cheapest cost if the heuristic function is admissible. Therefore, based on Definition 9, if the heuristic function (cost function) is monotonic in the problem space, such algorithm could be proved to find the optimal solution.

Definition 12: *Non-monotonicity*. Visiting certain grid in the route planning process could decrease current inferred PF per unit length since all input features are renewed and put into the model to re-calculate the inferred PF instead of adding new input features on former values directly. In other words, the heuristic function is not admissible. Note that neither input features nor inferred PF is monotonic in our case.

PF inference

Problem definition

Given a set of trajectories for the designated route with its stations labeled from users, our goal is to infer the passenger flow PF($l_i t_j$) for each route l_i in certain time intervals t_j. In other words, we devise RSA for users to plan their own routes and stations. Then, the system derives the passenger flow of the user-designated trajectory and stations in a certain time interval.

The framework of PF inference in RSA is shown in Figure 1, which mainly consists of three components. In data preprocessing, we divide the city into disjointed grids (e.g., 0.1km × 0.1km) (Silman, Barzily and Passy 1974), and all features are fetched and stored in grids for further extraction. The second component is training models. The feature set for each existing route is extracted and integrated as the training data along with its corresponding ticket data, which is associated with the timestamp and PF for each route. We treat various features as inputs and PF values as the predictive label. We adopt DNN for regression as training models. In the third component, the pre-trained model is utilized for the query route given by the user to infer PF value.

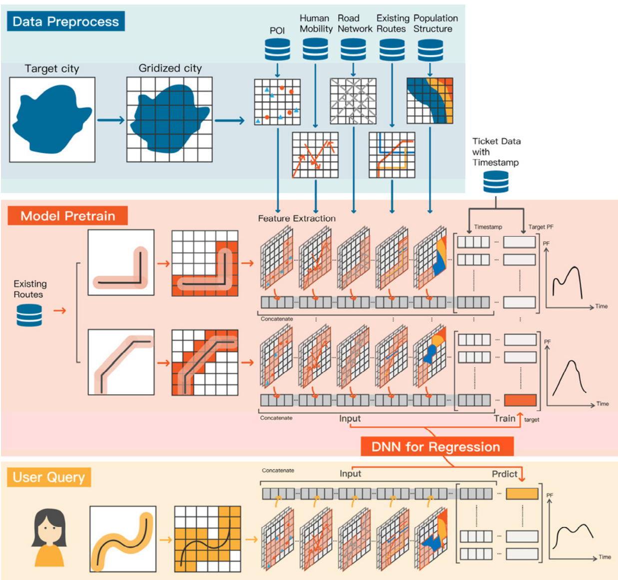

Figure 1: PF inference procedure in RSA.

Route affecting region

The demand for public transportation is not only based on the origin and destination, but also the nearby geographical environment and urban functions of nearby areas. Thus, we propose RAR for considering PF-related features. A route can comprise multiple segments that contain successive points close to each other. Then we can draw a circle for each point, where we consider each point as the center of a circle, and then RAR formed by a set of circles. Based on Design Manual for Urban Sidewalks (Su and Kuan 2003), the walking tolerance for pedestrians is 400 to 800 meters; we thereafter extract corresponding features within RAR.

Feature extraction based on RAR

To infer the PF value of a trajectory correctly, we consider six kinds of relevant urban features in RAR:

POI-related features. Various POIs (specific point location such as transportation hubs or entertainment venues) and their density in RAR indicate the function of a region, which might have high correlation to the PF of a route. For example, a high PF might be associated with route to many shopping centers. We consider two aspects of POI features:

POI Density. The density of POI indicates the popularity of a certain activity type in RAR. As the example mentioned above, a high density of certain types of POI such as shopping centers and schools can result in high PF value.

POI Entropy. The POI entropy in RAR shows the diversity of purpose when people visit the nearby area of a route. The entropy for trajectory li is based on Information Theory (Cover and Thomas 1991):

$$\text{Entropy}(l_i) = -\sum_{\gamma\in\Gamma}\left(\frac{N_\gamma(l_i,r)}{N(l_i,r)}\times\log\frac{N_\gamma(l_i,r)}{N(l_i,r)}\right) \quad (1)$$

Where Γ indicates set of POI, and γ refers to certain type of POI. Besides, N (l_i, r) displays the total number of POI in RAR of trajectory l_i based on radius r, Nγ (l_i, r) displays the number of type-γ POI in RAR of trajectory l_i relatively.

Human mobility. Human mobility is extracted from taxi data in three ways: transition density, incoming flow, and leaving flow. The transition density indicates the ratio of transitions occurred in the same RAR. The incoming flow shows the total records entering the RAR; on the contrary, leaving flow displays the total records exiting RAR.

Road network structure. Road network structure, including degree and closeness centrality, are considered since it might be correlated with real traffic conditions. We extract network structures from OpenStreetMap (OSM), where degree centrality identifies total number of reachable vertices for all intersections in RAR, and closeness centrality shows average distance between intersections in RAR.

Competition and transference with existing routes. Two routes might form a competitive relationship if their RAR is similar. However, intersected routes with considerable extended segments would encourage passengers to transfer between them. Therefore, we seek intersections between designated routes and existing ones, calculate the extended, nearby, and overlap grids each transferable existing route has; then sum up each type of grids as features of model.

To be more specific, grids that holds designated routes and each existing route can be labelled respectively. Then a simple algorithm is run to calculate the number of grids that are labelled as (i) the designated route together with existing routes, or (ii) the designated route but nearby grids in its RAR contain existing routes. Through such process, we derive the number of overlap grids and nearby grids between designated route and each existing ones; meanwhile, if intersection of designate route and certain existing route exists, the rest grids of that existing route are viewed as extended grids. Dividing these values by total grids of each existing route, several floating-point numbers are derived representing overlap/nearby/extended region respectively.

Population structure. People in RAR of different ages and genders can have different intentions for taking public transportation. Consequently, we extract the population data of the target city, and then the population for each age group and gender are normalized as features.

Time information and granularity. Seasons and holidays can influence the passenger flow of public transportation. We adopt one-hot encoding to record the time information for each ticket record.

Inference model construction

We adopt DNN for Regression to derive the PF for the designated route. The input data includes all the features extracted based on the RAR of the user-designated route, including POI-related properties, human mobility, road network structure, correlations of existing routes, population structure, and time information. As it turns out, the output is the inferred PF value of the user-designated route. The architecture of DNN for Regression is a feed-forward neural network with many levels of non-linearities allowing them to represent a highly non-linear regression function. Meanwhile, all our input features are normalized / rescaled to 0~1, the type of the hidden units for 4 dense layers is ReLU (Rectifie Linear Units), and the output unit is linear.

High PF route recommendation

Problem definition

Given a set of must-visit stations $S_M = \{S_{M0},\ldots,S_{Mi}\}$ and the range along with constraints including the number of recommended stations r, our goal is to recommend a trajectory in the given area along with a set of stations $S=S_M+S_R$ to maximizes the inferred PF value per unit length for the route (combination of trajectory and stations), where S_M refers to the set of must-visit stations, and S_R is the recommended stations $\{S_{R0},\ldots,S_{Rr}\}$.

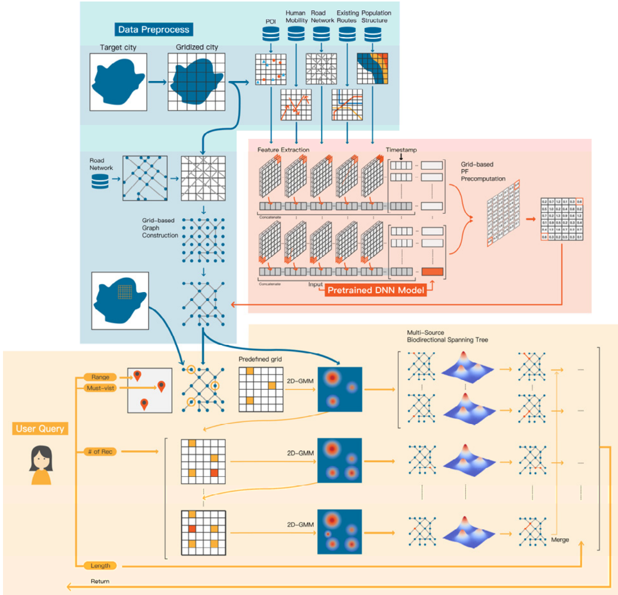

Figure 2: High PF route recommendation procedure in RSA

The optimal solution is quite difficult to obtain in a large urban space since there are too many combinations of route segments and stations for forming a route. According to our experiments, some exhaustion-based methods are not feasible due to high execution time. Meanwhile, considering the non-monotonic characteristics, most of existing route planning algorithms such as Dijkstra's algorithm, Prim's Algorithm, or heuristic algorithms including A* are inappropriate for our case (Bast et al. 2016). Since such "multiple choice branching" problem turns out to be NP-Complete and due to the non-monotonicity, we focus on approaching optimal answers in small ranges and outperforming other comparative methods in large ranges. Therefore, based on the idea of parallel computing of bidirectional search and goal-directed priority from best-first search, we propose the **Bid**irectional **P**rioritized **S**panning **T**ree (BDPST) to help retrieve a passable solution in a reasonable time frame.

The framework of high-PF route recommendation is shown in Figure 2. Next, we present our strategies, followed by the detailed algorithm and a brief complexity comparison with existing algorithms.

The strategies

To reduce the searching complexities, the strategies for our BDPST consist of four aspects:

Gird-based graph construction. Similar to ideas from other route planning algorithms, our proposed BDPST needs to construct a graph to search possible routes. We propose *grid-like graph* instead of the original road network for planning routes, where the length for each edge is either 1 or √2 unit length. Such grid-like graph can retain the connectivity of the original road network but merges vertices into fewer grids. The benefits of using the grid-like graph is three-fold. First, the number of nodes can be reduced from 237,866 (in the original road network) to 94,282 since only 94,282 grids (0.1km × 0.1km) contain road segments. Second, given a route, the computational loading and time of extracting features in RAR can be saved due to the grid indexing. Third, the grid is clear and easy to visualize for both user queries and route recommendation in map.

PF precomputation and lookup table construction. A recommended route can be represented by multiple connected nodes (girds). Before route construction, we first pre-infer the PF value for each grid in the map area, and PF values of grids are then stored in a lookup table. Thereafter, given a query range from a user, only the subset of PF values would be fetched and used for route recommendation.

Another lookup table is also constructed for each grid to record the connectivity between the grid itself and adjacent ones in its eight direction. Since the distance between each grid (node) is set to either 0.1km or 0.1√2km for two kinds of directions, we only need to store the connectivity information in the table and BDPST will traverse and generate routes according to the information from lookup tables.

Gaussian mixture model for modeling spatial influence. We adopt the idea of Gaussian mixture model (GMM) in two ways. First, for each must-visit station or recommended station, we model its spatial influence (i.e. the attractiveness of inferenced passenger in spatial aspect) on other nodes using two-dimensional Gaussian distribution. Therefore, for each node in the graph, we can compute its gained Gaussian distributed PF value from each must-visit station or recommended station. Such Gaussian distributed PF values can solve non-monotonic problems for some features such as entropy and relationship with existing routes and propagate their effects smoothly. That is, Gaussian distribution can smooth a non-monotonic function, which decreases the incorrectness in the number of non-monotonic features when a grid is selected. The corresponding feature values then can be easily accumulated by our search process. Modeling spatial influence using GMM has another benefit. Properties of nearby grids are somewhat propagated with a strictly decreasing function (Gaussian function) based on physical distances to complement the lack of specific information. For example, when considering whether to recommend a grid, it would be better if we can estimate its ability of transference to existing routes or POI-related attractiveness from nearby grids. Therefore, by accumulating all Gaussian distributed PF value from each must-visit station, the total number of potential passengers extracted from each grid is estimated.

Second, GMM is useful for selecting deployed stations from nodes. Assuming we only select locations which have high PF values as recommended stations, it might lead to settle several stations in a very small region since the grids in that region have high inferenced PF values. To avoid this herding effect, we propose to subtract the accumulated Gaussian distributed PF value from the pre-inferenced PF value of each grid. The calculated PF value is the expected gained passengers considering negative effects from other locations. Then, stations could be recommended based on such re-calculated PF values through an iterative process until the number of recommend stations S_R is satisfied. Thus, each recommended station selected is not only based on high inferenced PF value of the grid itself, but also are kept away from each other by adopting this strategy.

Multi-source bidirectional spanning. A simple idea to improve over the plain Dijkstra's algorithm is to search from the source and target node at the same time until two search frontiers meet (Bast 2009). To be more specific, the spanning tree from the source will proceed and search, while the tree from the target will search in the opposite direction. Eventually, the two will find an intersection with the lowest cost and complete route construction. Therefore, multi-source bidirectional spanning performs spanning search from several sources (must-visit stations) simultaneously.

More precisely, the multi-source bidirectional spanning tree maintains multiple queues and utilizes specific searching strategies such as Dijkstra's algorithm to form a route connecting two selected grids by merging two queues when the frontier goes to a grid that has been visited by the other queue. It should be noted that we set a constraint in which one source could only connect to a certain number of sources. Such restriction is to prevent from constructing a radial route. The algorithm will repeatedly merge queues until there is a queue that contains all the sources. Then the route will be constructed successfully.

The algorithm

The pseudo code for proposed BDPST Algorithm for stations and route recommendation is listed in supplementary materials, which consists of three parts, including grid-preprocessing (line 1-6), station-recommending (line 7-13) and trajectory-routing (line 14-56). In this code, *pf(g)* represents the inferenced PF value of grid *g*, *G(g',g)* represents Gaussian function in two dimensions between grid *g* and *g'*, with *gd(g',g)* represents the distributed PF from grid *g*.

In grid-preprocessing, it calculates the PF value of each must-visit grid utilizing the inference model proposed in previous section; then it evaluates its spreading impact on other grids in the area based on Gaussian function in two dimensions. To be more specifically, several independent Gaussian distributions representing potential PF of each grid are involved in our proposed BDPST.

Second, based on the negative Gaussian feedback of inference PF from must-visit and selected stations, the scores of the Gaussian mixture model for all grids are derived and we greedily select the grid with maximal PF as the recommended station.

In the third component, BDPST minimizes depth of search by performing multi-source bidirectional search and prunes the breadth of search space on the basis of the spreading impact of positive Gaussian feedback from other stations, which makes it act as a breadth-first-based, target-prioritized spanning tree growing from multiple stations (starters) simultaneously.

Complexity

The time complexity for our proposed BDPST ends up in ***O(EV)*** since the worst case is to traverse each grid in all directions, where E refers to the branching factor, and V indicates the number of grids labelled with road in given area. Detail explanation is attached in supplementary material. The schematic search space of Dijkstra's algorithm, bidirectional search, and our BDPST algorithm is shown in figure 3, where BDPST visits fewer grids than bidirectional as the target-prioritized approach restricts the breadth based on the tendency to other targets.

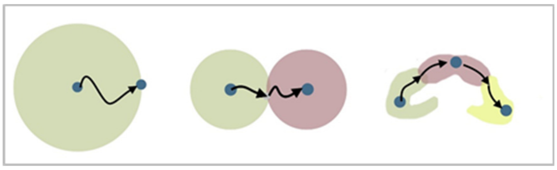

Figure 3: Schematic search space of Dijkstra's algorithm (left), bidirectional search (middle), and BDPST algorithm (right)

Evaluation

We use bus-ticket data on realistic public transit networks from Tainan City Government for evaluation, which contains 14,336,226 ticket records (01-12/2017). The city bus system holds 104 routes and 6575 stations. Tainan is divided into 505,296 disjointed grids (0.1km × 0.1km). Meanwhile, static features including POI (8,734 records in total) and road network (237,866 nodes and 414,409 edges) are extracted from GoogleMap and OpenStreetMap. Furthermore, we use bike trips (139,418 records) that list pick-up and drop-off location representing human mobility.

PF inference

To compare the importance between different features, Table 2 shows the components of feature sets used in preliminary evaluation. Apart from time information, which is included in all feature sets, each of the six urban relevant features is selected respectively from set *I* to set *V*. Set *VII* refers to the static location-based features, including POI-related ones and population structure; similarly, set *VIII* represents the static transportation-based features consists of road network and competition/transference with existing routes. Furthermore, set *IX* combines all the static features, and human mobility is later considered in set *X*.

Table 2: Feature sets to be used in preliminary evaluation

Features	Feature Set									
	I	II	III	IV	V	VI	VII	VIII	IX	X
Timestamp	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
POI-density	✓						✓		✓	✓
POI-entropy						✓	✓		✓	✓
Human mobility		✓								✓
Road network				✓				✓	✓	✓
Existing routes					✓			✓	✓	✓
Population structure			✓				✓		✓	✓

The evaluation is based on the leave-one-out method. We leave one route PF data out of the complete data, and then use the rest of data to train the model and infer the value of the left out route based on the features extracted from the RAR of the route. Then we compare the inference value with the ground-truth. Thereafter, we leave the next PF and use the remaining data to train and infer again until each PF is inferred and compared with its ground-truth. Finally, we retrieve the normalized root-mean-square error (RMSE).

Considering the walking tolerance for pedestrians (0.4 km to 0.8 km), we summarize the preliminary evaluation by illustrating the normalized RMSE of PF inference results based on DNN for regression under different feature sets and RAR/Radius settings for RAR-based/OD-based scenario in Figure 4a and 4b respectively. Where the RAR/Radius range of 0.4 km gains the better normalized RMSE.

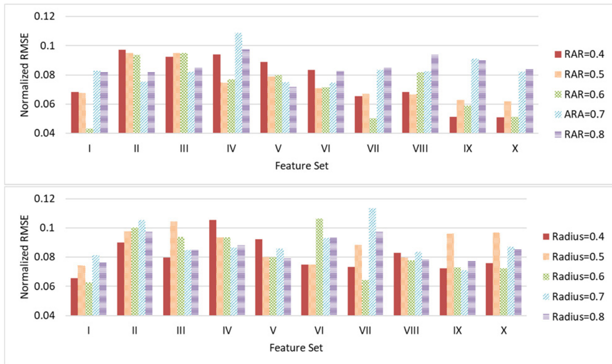

Figure 4: Normalized RMSE of PF inference results based on DNN for regression under different feature sets and ranges for (a)(up) RAR-based scenario and (b)(down) OD-based scenario.

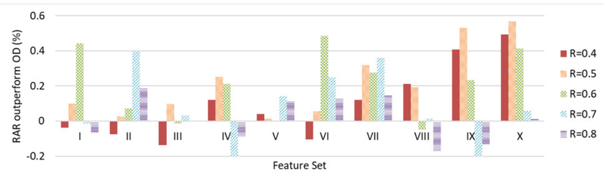

Figure 5: Percentage RAR-based outperforms OD-based scenario under different feature sets and range settings.

We identified that feature set *I* and *VI* show better results than the others, which indicates that POI-related features turn out to have much more impact among all features. Furthermore, location-based ones outperform transportation-based ones for 13% and 10%. On the other hand, although the advantage of the proposed RAR strategy is not significant for each single urban relevant feature (feature sets *I* to *VI*), the RAR-based scenario consistently gains the best results considering multiple heterogeneous features when range set at 0.4 km (sets *VII* to *X*), as shown in Figure 5.

Route recommendation

In this section, we developed six comparative methods: (a) *Dijkstra's Algorithm (Dijkstra's)* starts at one must-visit station and searches for another must-visit station as a destination on Dijkstra's Algorithm. Where after route towards the current destination is settled, treat this grid as the next origin to start the search for another must-visit station iteratively. (b) *Breadth-First Search (BFS)* starts at one must-visit station and explores all of the neighbor grids next to the current grid prior to traversing grids that cannot be reached in one step. The route is formed when all the must-visits are visited. (c) *Iterative Deepening Depth-First Search (IDDFS)* is a depth-limited version of the depth-first search to run repeatedly with increasing depth limits until all the must-visit stations are traversed. (d) *Best-First Search (Best-First)* iteratively explores neighbor grid that holds the highest pre-calculated PF of the grid itself. (e) *Distance-Based A* (Distance-A*)* performs A* algorithm from must-visit station to another must-visit station iteratively with a heuristic that demonstrates the distance between destination and candidate grid. (f) *Passenger-Flow-Based A* (PF-A*)* performs A* algorithm from must-visit station to another must-visit station iteratively with a heuristic that predicts the PF between destination and candidate grid. Baseline method: *Brute-Force (BF)* systematically enumerates all possible combinations and retrieves the optimal one.

The evaluation is based on average PF per unit length and execution time. All methods run 1,000 randomly generated testing cases for area range from 0.25 km^2 to 16 km^2 on same condition. When comparing against comparative methods, PF per unit length for each method is divided by the value of the proposed BDPST algorithm into a unit PF ratio. We then compare the PF ratio and runtime of BDPST with comparative methods by varying two independent variables, including area range and the number of must-visit stations. Results are shown in Figure 6 and 7.

All methods are implemented in Java and evaluation for runtime is based on the single core of an Intel i7-7700 CPU @ 3.60GHz with 16 GB of RAM. Furthermore, considering practical applications in real-world transportation networks, queries executed for over 60 seconds are identified as failures (failure trials would not affect the runtime illustrated in following figures); for methods that have more than half failure cases to respond queries under the parameter setting, their unit PF ratio and runtime would not be considered or displayed in corresponding figures.

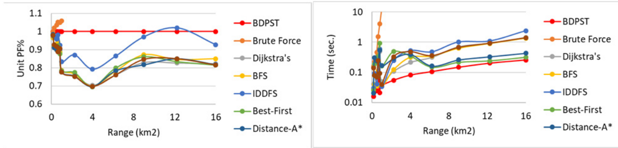

Figure 6: Unit PF% and execution time for different methods under area range. Runtime is in logarithmic scale.

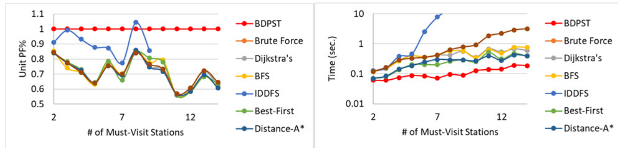

Figure 7: Unit PF% and execution time for different methods under number of must-visit stations. Runtime is in logarithmic scale.

Figure 6 shows that proposed BDPST performs close to optimal solutions in small space and in the large-scale region (>9 km^2), BDPST maintains with high PF per unit length and success rate. Although *IDDFS* gains better PF results dealing with requests in some cases, the runtime of *IDDFS* is larger than BDPST for at least one logarithmic scale. As the main difference between BDPST and *IDDFS* is that the latter is a DFS-based algorithm resulting in a higher chance of exploring regions where no station exists. Result in Figure 7 shows that when given a large number of requested must-visit stations, *IDDFS* fails to construct routes in 60 seconds for more than half of the trials.

Experimental results on the runtime also meet our analysis that the time complexity for BDPST is mainly related to the area range. Compared to other comparative algorithms, the scale of runtime for BDPST is close to *Best-First Search* and *Distance-A**. To conclude, proposed BDPST algorithm outperforms other comparative methods from 7% to 22% in large scale (>9 km^2) and reasonably close to optimal in small space. Besides, *Brute-Force* can obtain optimal solutions but is only feasible for very small ranges (<1 km^2)

Conclusions

This work propose RSA that addresses route-based PF inference and constrained route planning. Given heterogeneous features and faced with the competitive and transfer effects of existing routes, our proposed RAR and feature engineering methods are effective for handling dynamic and static data. Utilizing parallel computing concepts and GMM, the proposed BDPST algorithm recommends high-PF route under users' constraints running in a deliverly time. Experimental results on Tainan bus-ticket data show that our proposed PF inference model and BDPST algorithm outperform baseline and comparative methods. Moreover, the proposed BDPST algorithm is feasible for real-time large-scale route recommendations.

References

Abraham, I. et al. 2011. A Hub-Based Labeling Algorithm for Shortest Paths in Road Networks. In *Proceedings of the 10th International Symposium on Experimental Algorithms*. Lecture Notes in Computer Science, Vol. 6630. Springer, 230–241.

Arabghalizi, T., and Labrinidis, A. 2019. How full will my next bus be? A Framework to Predict Bus Crowding Levels. In *Proceedings of the 8th International Workshop on Urban Computing*. Anchorage, Alaska: ACM.

Bast, H. 2009. Car or public transport – two worlds. *Efficient Algorithms*. Lecture Notes in Computer Science, Vol. 5760. Springer, 355–367.

Bast, H. et al. 2010. Fast Routing in Very Large Public Transportation Networks Using Transfer Patterns. In *Proceedings of the 18th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science, Vol. 6346. Springer, 290–301.

Bast, H. et al. 2016. Route Planning in Transportation Networks. *Algorithm Engineering*. Lecture Notes in Computer Science, vol 9220. Springer.

Cancela, H.; Mauttone A.; and Urquhart, M. E. 2015. Mathematical programming formulations for transit network design. *Transportation Research Part B: Methodological* 77: 17-37.

Cheng, S.; Liu B.; and Zhai, B. 2010. Bus arrival time prediction model based on APC data. *6th Advanced Forum on Transportation of China*.

Chien, S. I.-J.; Ding, Y.; and Wei, C. 2002. Dynamic Bus Arrival Time Prediction with Artificial Neural Networks. *Journal of Transportation Engineering* 128(5): 429-438.

Cover, T. M., and Thomas, J. A. 1991. Entropy, relative entropy and mutual information. *Elements of Information Theory*, ch. 2, sec. 1, pp. 12-13.

Delling, D. et al. 2009. Engineering route planning algorithms. *Algorithmics of Large and Complex Networks*. Lecture Notes in Computer Science, Vol. 5515. Springer, 117–139.

Delling, D. et al. 2011. Customizable route planning. In *Proceedings of the 10th International Symposium on Experimental Algorithms*, Vol. 6630. Springer, 376–387.

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik*, 269–271.

Ehrgott, M., and Klamroth, K. 1997. Connectedness of efficient solutions in multiple criteria combinatorial optimization. *European Journal of Operational Research* 97(1): 159-166.

Garey, M. R.; Johnson, D. S.; and Stockmeyer, L. 1976. Some simplified NP-complete graph problems. *Theoretical Computer Science* 1(3): 237-267.

Garey, M. R.; Johnson, D. S.; and Tarjan, R. E. 1976. The Planar Hamiltonian Circuit Problem is NP-Complete. *SIAM Journal on Computing* 5(4), 704–714.

Garey, M. R., and Johnson, D. S. 1979. Computers and intractability: a guide to the theory of NP-completeness. *W. H. Freeman and Company*. Appendix B

Goldberg, A. V., and Harrelson, C. 2005. Computing the shortest path: A* search meets graph theory. In *Proceedings of the 16th Annual ACM–SIAM Symposium on Discrete Algorithms*. SIAM, 156–165.

Goldberg, A. V. 2008. A Practical Shortest Path Algorithm with Linear Expected Time. *SIAM Journal on Computing* 37(5): 1637–1655.

Guihaire, V., and Hao, J.-K. 2010. Transit network timetabling and vehicle assignment for regulating authorities. *Computers & Industrial Engineering* 59(1).

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* (4)2: 100–107.

Hassin, R., and Levin, A. 2012. An efficient polynomial time approximation scheme for the constrained minimum spanning tree problem using matroid intersection. *SIAM Journal on Computing* 33(2): 261–268.

Henn, S. T. 2007. Weight-constrained minimum spanning tree problem. Doctoral dissertation, Department of Mathematics, University of Kaiserslautern, Germany.

Julian, D.; Thomas, P.; and Dorothea, W. 2015. User-Constrained Multi-Modal Route Planning. *Journal of Experimental Algorithmics*. 19(3):1.1-1.19.

Korf, R. E. 1985. Depth-first iterative-deepening: an optimal admissible tree search. *Artificial Intelligence* (27)1: 97–109.

Lin, Y. et al. 2013. Real-Time Bus Arrival Time Prediction: Case Study for Jinan, China. *Journal of Transportation Engineering*, 1133-1140.

Mauttone, A., and Urquhart, M. E. 2009. A route set construction algorithm for the transit network design problem. *Computers & Operations Research* 36: 2440-2449.

Peterson, A. 2007. The Origin–Destination Matrix Estimation Problem — Analysis and Computations. Doctoral dissertation, Department of Science and Technology. Linköping University, Sweden.

Pternea, M.; Kepaptsoglou, K.; and Karlaftis, M. G. 2015. Sustainable urban transit network design. *Transportation Research Part A: Policy and Practice* 77: 276-291.

Quadrifoglio, L. and Li, X. 2009. A methodology to derive the critical demand density for designing and operating feeder transit services. *Transportation Research Part B: Methodological* 43(10): 922-935.

Silman, L. A.; Barzily, Z.; and Passy, U. 1974. Planning the route system for urban buses. *Computers & Operations Research* 1(2): 201-211.

Stauffer, C. and Grimson, W. E. L. 1999. Adaptive background mixture models for real-time tracking. *IEEE CVPR*. Fort Collins, CO, USA, pp. 246-252 Vol. 2. DOI: 10.1109/CVPR.1999.784637

Steg, L., and Gifford, R. 2005. Sustainable transportation and quality of life. *Journal of Transport Geography* 13(1): 59-69

Su, H.-M., and Kuan, C.-C. 2003. Planning and design guidelines. In *Design Manual for Urban Sidewalks*, ch. 4, sec. 1, pp. 1-4.

Szeto, W. Y., and Wu, Y. 2011. A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong. *European Journal of Operational Research* 209(2): 141-155.

Yan, Y. et al. 2012. Robust optimization model of schedule design for a fixed bus route. *Transportation Research Part C: Emerging Technologies* 25: 113-121.

Yang, H., and Zhou, J. 1998. Optimal traffic counting locations for origin–destination matrix estimation. *Transportation Research Part B: Methodological* 32(2): 109–126.

Yap, M.; Cats, O.; and Arem, B. 2018. Crowding valuation in urban tram and bus transportation based on smart card data. *Transportmetrica A: Transport Science*. DOI: 10.1080/23249935.2018.1537319.

Yoshizumi, T.; Miura, T.; and Ishida, T. 2000. A* with Partial Expansion for Large Branching Factor Problems. In *Proceedings of the 17th National Conference on Artificial Intelligence*. AAAI Press, 923-929.

Wagner, D.; Willhalm, T.; and Zaroliagis, C. 2005. Geometric containers for efficient shortest-path computation. *Journal of Experimental Algorithmics* 10, 1.3: 1–30.

Wei, Y. and Chen, M.-C. 2012. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. *Transportation Research Part C:* 21: 148-162.

Zhang, R.; Kabadi, S. N.; and Punnen, A. P. 2011. The minimum spanning tree problem with conflict constraints and its variations. *Discrete Optimization* 8: 191-205.