A STATISTICS-BASED FRAMEWORK FOR BUS TRAVEL TIME
PREDICTION

A Thesis in
Computer Science and Engineering
by
Weiping Si

© 2012 Weiping Si

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2012
The thesis of Weiping Si was reviewed and approved* by the following:

Wang-Chien Lee  
Associate Professor of Computer Science and Engineering  
Thesis Advisor

Sencun Zhu  
Associate Professor of Computer Science and Engineering

Lee Coraor  
Associate Professor of Computer Science and Engineering  
Chair of Graduate Program of the Department of Computer Science and Engineering

*Signatures are on file in the Graduate School.
Abstract

In this paper, we develop a statistics-based bus travel time prediction framework, called *Historical Trajectory based Travel/Arrival Time Prediction (HTTP)* for real-time prediction of travel time at future segments (and thus the arrival time at stops) of an on-going bus journey. The basic idea behind HTTP is to use a collection of historical trajectories “similar” to the current bus trajectory to predict the future segments. Specifically, the HTTP framework (1) samples a set of similar trajectories as the basis for travel time estimation instead of relying on only one historical trajectory best matching the on-going bus journey; and (2) explore different prediction schemes, namely, passed segments, temporal features, and hybrid methods, to identify the sample set of similar trajectories. We conduct a comprehensive empirical experimentation using real bus trajectory data collected from Taipei City to validate our ideas and to evaluate the proposed schemes. Experimental result shows that the proposed prediction schemes significantly outperforms the state-of-the-art and baseline techniques.
# Table of Contents

List of Figures vi

List of Tables vii

Chapter 1
Introduction 1
1.1 Motivation ............................................................................ 1
1.2 Background ........................................................................... 1
1.3 Research Overview .............................................................. 2
1.4 Chapter Outline .................................................................... 4

Chapter 2
Preliminaries 5
2.1 Terminologies ....................................................................... 5
2.2 Problem Formulation ............................................................ 7

Chapter 3
Related Work 8
3.1 A brief history of traffic prediction ....................................... 8
3.2 Trajectory Similarity ............................................................ 9
3.3 Trajectory Patterns ............................................................. 10

Chapter 4
Data Analysis 11
4.1 Data Scenario ..................................................................... 11
4.2 Correlation between Segments .......................................... 12
4.3 Patterns Exploration .......................................................... 13

Chapter 5
Framework Overview ............................................................... 17

Chapter 6
Algorithm 19
6.1 Passed Segments Scheme .................................................. 19
6.1.1 Clustering Algorithms ...................................................... 20
6.1.1.1 K-means Clustering .................................................... 21
6.1.1.2 V-Clustering ............................................................. 21
List of Figures

2.1 Route representation. ................................................. 5
4.1 Pearson’ Correlation. ................................................. 13
4.2 Average Pearson’ Correlation. ..................................... 14
4.3 Analysis by hours. .................................................... 15
4.4 Analysis by days. ..................................................... 16
5.1 System architecture of HTTP ........................................ 17
6.1 The window moving along. ......................................... 24
7.1 Partitioning Algorithms for route 10283. ......................... 32
7.2 Partitioning Algorithms for route 112430. ....................... 33
7.3 Partitioning Algorithms for route 11411. ......................... 34
7.4 Segment Filtering for route 10283. .............................. 35
7.5 Segment Filtering for route 112430. .............................. 36
7.6 Segment Filtering for route 11411. .............................. 37
7.7 Travel Time Estimation for route 10283. ......................... 38
7.8 Travel Time Estimation for route 112430. ....................... 38
7.9 Travel Time Estimation for route 11411. ......................... 39
7.10 Partitioning Methods. ............................................... 41
7.11 Comparison of Partitioning Methods. .......................... 42
7.12 Partitioning Methods. ............................................... 43
7.13 Comparison of Partitioning Methods for route 112430. ........ 44
7.14 Partitioning Methods. ............................................... 45
7.15 Comparison of Partitioning Methods for route 11411. ........ 46
7.16 Comparison of Prediction Schemes for route 10283. .......... 46
7.17 Comparison of Prediction Schemes for route 112430. .......... 47
7.18 Comparison of Prediction Schemes for route 11411. .......... 47
List of Tables

6.1 All historical trajectories. ....................................................... 20
6.2 Partitioned Table 6.1. .............................................................. 20
6.3 Trajectory table with ranges. .................................................. 24
7.1 Parameters and default values for route 10283 .......................... 30
7.2 Parameters and default values for route 112430 .......................... 30
7.3 Parameters and default values for route 11411 .......................... 30
Chapter 1

Introduction

1.1 Motivation

In modern society, with the ever-increasing number of vehicles on roads, traffic congestion has become one of the most serious problems nowadays, especially in some big cities, i.e., New York City, Paris and Taipei. The congested traffic are driving more and more people to take the public transportation systems, such as buses, subways and trains, instead of using their own cars. It is hard for drivers to know when and where there will be a traffic congestion and as a result a waste of time is inevitable. On the other hand, the collective transport is able to provide the public with more choices of routes. More importantly, with a schedule of predicted arrival times at each station, people can make timely plans for their upcoming activities and business. It can be expected that in the near future the public transportation will become increasingly important to many people and subsequently customer satisfactory is a high priority to public transportation services. In today’s busy society, information regarding travel time or arrival time of transport from a place to another is becoming more and more valuable. Thus, there is a high demand for an accurate estimate of arrival and travel times.

1.2 Background

Accurate estimation of travel times of public transportation is a challenging research problem that remains open for the past thirty years in the transportation research community [1, 2]. To predict the travel time on a given path, a simple approach is to adopt the average travel time derived from historical data. This approach, making constant prediction of the travel time for a path, apparently does not capture the dynamic traffic situation very well. Thus, advanced prediction techniques for travel time estimation have been proposed [1, 2, 3, 4, 5, 6, 7]. Generally speaking, these techniques share a common idea, i.e., discover certain regular patterns from the
historical data collected over time, even though the specific approaches adopted are different. For example, some propose to fit the historical data to statistical models such as Gaussian models, Bayesian network and Markov Chains in order to facilitate statistical analysis [1, 2, 3]. On the other hand, techniques based on regression models learn from historical data regression functions of estimated travel time in terms of various external factors [4, 5]. Thus, a prediction is made by using the values of those factors under current situation as input to the function. Moreover, techniques based on time series models focus on discovering the internal relationship among historical time-series data in order to identify similar patterns from historical data to make prediction under the current situation [6, 7]. Notice that there is no clear winner among these techniques as their performances are highly constrained by the quality/quantity as well as the types of data available. For example, conventional collection of traffic data is typically conducted by surveys [1, 2, 3, 4] or using expensive sensors deployed along the roads at some locations to record arrival times, traffic flow volumes, and other statistics of vehicles [5, 6, 7].

In recent years, due to the rapid advent of positioning and wireless communication technologies, wireless devices equipped with Global Positioning System (GPS) have been widely deployed on various private and public vehicles, generating massive amount of vehicle data, including instant speeds, locations, and so on, for fleet management and other transportation applications. The vehicle data, usually represented in form of trajectories, also bring a great potential for real-time estimate of the vehicle travel times. Thus, in this paper, we propose a new prediction framework to estimate the travel time of buses by exploiting collected bus trajectory data. Among the public transportation systems, the travel times of buses, which drives along with other vehicles on roads, are more difficult to predict than trains and subways, which ride on fixed rails. First, the travel condition of a bus may easily get affected by various internal and external factors, including accidents, weather, road construction, government policies and even temperature. Second, for vehicles in metropolitan areas, errors often exist in (positional) data acquisition and transmissions due to the interference and blocking of surroundings and other resources of errors. Fortunately, we do not need to capture all real-time data to make accurate travel time prediction. Historical trajectory data of buses can help!

1.3 Research Overview

Recently, research works on discovering traffic patterns from historical data collected from vehicles have receive significant attention [8, 9, 10, 11, 12]. Particularly, these works show that traffic patterns exists in road segments and thus could be used to predict the future traffic condition on the same segment. This finding provides a concrete basis for using similar trajectories to predict the travel time of an on-going bus journey. In this paper, we develop a new bus travel time prediction framework, called Historical Trajectory based Travel/Arrival Time Prediction (HTTP) for real-time prediction of travel time at coming segments (and thus the arrival time at stops) of an on-going bus journey. The basic idea behind HTTP is to use a collection of historical trajectories “similar” to the current bus journey to predict the travel times in future
segments of the bus journey. Specifically, the HTTP framework (1) samples a SET of similar trajectories as the basis for travel time estimation instead of relying on only ONE historical trajectory best matching the on-going bus journey; and (2) explore different features (e.g., travel times of passed segments as well as time/day of the bus trajectories) to identify the sample set of similar trajectories.

Several issues faced in design of the HTTP framework. For example, many features are associated with trajectories. Some of these features are categorical while the others are numerical. We need to use discriminative features and properly define similarity functions for those features in order to identify a sample set of similar trajectories effective for travel time prediction. To determine a set of similar trajectories based on travel time on passed trajectory segments, we consider K-Means and V-Cluster algorithms to partition the whole spectrum of travel times into a number of intervals. To determine a set of similar trajectories based on hours/days, we also use the K-Modes algorithm to partition the hours/days feature space. Accordingly, the HTTP framework is able to retrieve the sample set of similar trajectories efficiently and in turn use the returned sample set to estimate the travel times. To validate the proposed ideas and evaluate various prediction schemes proposed for HTTP, we conduct a comprehensive empirical experimentation using real bus trajectory data collected from Taipei City, Taiwan.

This research work has made a number of significant contributions as summarized below.

1. We propose a new system framework, namely, HTTP, for predicting the travel times over future segments of an on-going bus journey based on historical trajectory data. The HTTP framework consists of two major components: (i) similar trajectory retrieval; and (ii) travel time estimation.

2. We perform a detailed data analysis to investigate the correlation between bus travel times in route segments and a number of trajectory features, e.g., passed segment travel time, hours, days, etc. Based on our analysis, we select a number of trajectory features to identify similar trajectories.

3. We adopt clustering algorithms for different types of trajectory features in order to group similar trajectories together. These similar trajectory clusters allows us to efficiently and effectively retrieve a sample set of trajectories similar to the on-going bus journey.

4. We study a number of travel time estimation schemes to derive the travel time prediction for future segments of an on-going bus journey.

Through a comprehensive experimental study, using a real data set collected from buses in Taipei City, Taiwan, we validate our proposed ideas and evaluate the HTTP framework in terms of prediction accuracy. The experimental results show that all the prediction schemes proposed under HTTP significantly outperforms the baseline and state-of-the-art schemes. Among our proposals, the hybrid temporal features/passed segments (HTP) achieves the best performance.
1.4 Chapter Outline

The remainder of this paper is organized as follows. In Chapter 2 and Chapter 3, introduce terminology, formulate the research problem, and review some related works. In Chapter 4, we analyze the collected historical trajectory data. Next, in Chapter 5, we give an overview of the HTTP framework, detailing its system design. In Section 6, we further discuss the proposed prediction schemes in details. In Chapter 7, we conduct a comprehensive experimental study using the collected real data set of bus trajectories. Finally, we conclude this work in Chapter 8.
Chapter 2

Preliminaries

2.1 Terminologies

Since the buses are traveling on fixed routes, the geometrical routes in a two-dimensional space can be represented in a one-dimensional space, where the position of each point on the route is the distance from the start of the route (Figure 2.1). A route can be considered as consisting of points on the route and as a classical way, we choose a number of important points to represent a route. Those points are termed as Points of Interests. In the area of collective transportation, a natural choice of such points is the bus station, since it is of high interest to predict the arrival time at a bus station. People tend to know the arrival time at a bus station other than a random point along the route.

Def.1. A route $R$ is represented as a sequence of points, $R^{raw} = (p_0, ..., p_n)$, where each point, $p_i$, stands for a bus station and the its value denotes the total distance along the route from the start of the route to the $(i + 1)$th bus station; thus $p_i < p_{i+1}$.

![Figure 2.1. Route representation.](image)
Bus stations naturally divide a route into segments, each of which is the distance from one bus station to the next adjacent one. The goal of this project is to predict the travel time on each segment, so we tend to represent a route as a series of segments.

Def. 2. A segment $S$ is a part of a route between two adjacent bus stations. A route $R$ is represented by a sequence of segments, $\langle S_0, ..., S_{n-1} \rangle$, where the value of $S_i$ denotes $p_{i+1} - p_i$.

Due to the availability of positioning technology, buses equipped with GPS are able to update their positions (along with other bus status information) regularly and thus report the journey on a bus route as a trajectory, which consists of a time-stamped series of location points on the bus route. Notice that the location points in a trajectory is obtained in accordance with GPS-dependent sampling scheme, i.e., these sample points may not be aligned with bus stops. To address this issue, for a given trajectory, the arrival time of a bus at a bus stop is obtained by interpolation. Therefore, the travel time for each route segment can be easily computed.

Def. 3. A raw trajectory $T_{raw}$ is represented as a sequence of $\langle \langle p_0, t_0 \rangle, ..., \langle p_n, t_n \rangle \rangle$, where $p_i \in R$ and the value $t_i$ denotes the travel time of a bus traveling a distance of $p_i$. With arrival time at each bus station, it is straightforward to get the travel time between two bus stations. Therefore, on a fixed route, a trajectory $T$ is represented as a sequence $\langle \Delta t_0, ..., \Delta t_{N-1} \rangle$, where $\Delta t_i$ denotes the travel time on $S_i$ and $N$ is the number of segments on this route.

During a trajectory, a bus travels through the whole route and generates a travel time to each segment. Therefore given $M$ historical trajectories, there are $M$ travel times for each segment. To predict the arrival time of an arbitrary bus station, we just need to add together the travel times of all segments before.

Def. 4. For each segment, there is a corresponding sequence of travel time $\langle \Delta t_0, ..., \Delta t_{M-1} \rangle$, where $\Delta t_j$ is the travel time of a bus on this segment in the $(j+1)$th trajectory and $M$ is the number of historical trajectories. The sequence is defined as $ST_i^{raw}$ for $S_i$. For example, in the instance of Table 6.1, travel times in the first column belong to the corresponding segment, and they are represented as $ST_0^{raw}$.

For a specific route, given all historical trajectories along this route, we can create a table with attributes corresponding to the travel times of each segment and a record corresponding to a historical trajectory (Table 6.1).

As discussed in Chapter 4, we partition each column of Table 6.1 into ranges to get quantitative associate rules.

Def. 5. Each item in $ST_i^{raw}$ is a quantitative value that can be in an range. Using the clustering algorithms covered in Chapter 5, we are able to separate the sequence of $ST_i^{raw}$ to a sequence of ranges. So $ST_i$ for $S_i$ is represented as $\langle I_0, ..., I_{K-1} \rangle$, where $K$ is the number of ranges for $ST_i$. The number of ranges for each segment is not necessarily equal. The process from $ST_i^{raw}$ to $ST_i$ is illustrated by Table 6.1 and Table 6.2.
2.2 Problem Formulation

Def.6. Consider a bus route \( R = (S_0, ..., S_{N-1}) \) with \( N \) segments. For a bus traveling on segment \( S_i \) of its bus route, its current (and incomplete) trajectory/journey \( T_{curr} \) can be represented as a sequence of travel times of the passed segments, i.e., \( T_{curr} = (\Delta t^0_{curr}, ..., \Delta t^i_{curr}) (0 \leq i \leq N) \). Without loss of generality, the travel time prediction problem is to predict the travel times this on-going bus to spend in the remaining segments, i.e., \( \Delta t_{i+1}, ..., \Delta t_N \), on the bus route.

Given a bus route, a repository of \( M \) historical trajectories on this route, and an on-going bus traveling on the route, we aim to develop an effective travel time prediction framework and prediction schemes by exploring the patterns hidden in the massive collection of historical trajectories on the route.
Chapter 3

Related Work

To frame the research methods and concepts used in this thesis, this chapter reviews past research that has fueled my interest in prediction of movement of vehicles on known or unknown routes. I begin by giving a brief history of traffic prediction, and review the small amount of research that has focused specifically on similarity-based prediction of arrival/travel times and trajectory patterns.

3.1 A brief history of traffic prediction

Transportation study is a research area with a very long history that can be traced back to the 30s of the last century. With few vehicles on roads and under-developed technologies, it was then impossible to collect significant data about traffic conditions. Thus studies during this time were mainly about identifying certain rules that could be used to guide traffic management and the construction of transportation infrastructure. For example, the relations between traffic volumes and the weather were discovered by [13]. It justified the improvement of road surfaces during bad weather. Another typical example is that authors of [14] verified a definite relationship between highway lighting and highway accidents that in general where adequate lighting is provided there is a substantial reduction in night accidents.

Along the development of technologies and increasing of number of vehicles on roads, more data about traffic conditions could be collected, subsequently causing the emergence of research on traffic prediction in the 50s of the 20th century. However, during this period, traffic data adopted in most cases were vehicle volumes because they were easily collected by hand. For example, in [15, 16, 17], to obtain the vehicle volume on a road, observers were placed at certain locations to record the number of vehicles passed by. Such an approach was inefficient and made it difficult to collect a large amount of data. Therefore, the arrival/travel times prediction did not arise until 1970s [18, 19], when traffic sensors were widely adopted enabling researchers to have sufficient data for analysis.

Estimation of arrival/travel times prediction of collective transportation, especially buses,
started to attracted increasing attentions since the 80s of the last century [1, 2, 3]. Along with the development of the society, congestions happened increasingly frequently in cities, driving people to use public transportation rather than their own vehicles in daily life and care about the quality of public transportation service. As the most important aspect of public transportation service, arrival/travel times prediction became the most critical topic in traffic prediction area. At early stage of the research on this topic, constrained by technologies, researchers had to work on data collected from traffic sensors and surveys. Thus the prediction was off-line. Since the development of GPS devices and wireless network, it is available to collect large volume of traffic-related data in real-time manner. Therefore, Real-Time arrival/travel times prediction has become a hot topic since the GPS devices and wireless network are widely applied in public transportation system. Over decades, researchers applied different models and methods on real-time arrival/travel times prediction. In [20], the authors develop mathematical models taking into account the travel times on links, dwell times at stops, and delays at intersections. The algorithm proposed in [21] is to provide real-time bus arrival information based on the bus location data, the schedule information, the difference between scheduled and actual arrival times, and the waiting time at time-check stops. Predicting methods based on historical data are also developed in [22]. To a greater extent, with the development of Artificial Intelligence, researchers has widely adopted Artificial Intelligence methods in real-time arrival/travel times predictions. The most widely adopted methods are Kalman filtering [23, 24, 25] and artificial neural networks (ANNs) [26, 27, 28]. Kalman filtering takes into account the stochastic properties of the process disturbance and the measurement noise. It works well for short-term prediction, but not for long-term prediction. ANNs has a huge advantage that it can process complex nonliner relationships. However, it is limited by the extremely long training time. Besides, other machine learning methods are also popular in recent years. Real-time prediction using Support Vector Regression (SVR) and Support Vector Machine (SVM) has become a hot topic recently.[29, 30] Similar to ANNs, SVR is too expensive in training to do a real-time update.

3.2 Trajectory Similarity

That vehicles run back and force on a fixed route causes the naturally existence of similar trajectories in the historical data. Therefore the similarity-based approach is the straightforward approach to predict future travel times. Great amount of work has done on identifying similar trajectories or similar time series, in both one-dimension and multi-dimensions. [31] proposed $L_p$-norm as to compute the Manhattan Distance or Euclidean Distance. $L_p$-norm is widely applied in various application but is only available for time series with same length. Therefore, other similarity measures are developed and adopted. Berndt and Clifford [32] introduced Dynamic Time Warping (DTW) that is adopted in [33, 34]. The concept of edit distance was introduced in [35] and the most widely used distance based on edit distance is LCSS (Longest Common SubSequence) distance. [36, 37, 38] apply LCSS as the distance measure to fetch similar trajectories or time series. However, these algorithms tend to emphasize on the overall similarity of
the whole trajectory, without considering the similarity of trajectories in individual or subsets of segments. Additionally, while LCSS and DTW are applicable to our data, they are highly sensitive to noises and errors. In this project, we propose our own similarity measures in HTTP.

Recently, predicting methods based on historical trajectory data have also been developed in [39, 12, 22]. The authors show that the similarity between historical trajectories and current position data of a bus can be exploited to predict bus arrival time at bus stations, which share the same intuition with our research work in this paper. In TransDB [22], the system searches the historical trajectory database for the most similar trajectory to the passed segments of the current bus trajectory in order to make a good prediction. The basic idea is that, based on the proposed trajectory similarity function, the nearest neighborhood trajectory (NNT) and the trajectory of current bus ride are anticipated to exhibit similar traveling behavior (in terms of travel time). Based on this assumption, the NNT serves as a good basis for predicting the future travel time of current bus ride without explicitly taking into account various external and internal factors. The HTTP system proposed in this paper also aim to exploit the patterns in similar historical trajectories for making predictions. However, we argue that the historical trajectory most similar to the passed segments of the current bus trajectory alone may not provide the best prediction of the on-going bus ride. Thus, we collect a set of similar trajectories and adopt a statistical approach to make predictions. Additionally, we exploit different features associated with trajectories and develop different similarity functions to find similar trajectories that, as our experimental results show in Chapter 7, make significantly more accurate travel time predictions than TransDB.

3.3 Trajectory Patterns

Patterns of historical trajectories are described in two classes: trend and periodicity. The trend represents a general systematic linear or nonlinear component that changes over time and does not repeat or at least does not repeat within the time range captured by data. The periodicity represents the component repeats itself in certain intervals over time. In [8], the author display the daily periodicity from historical data of travel times around the same location. [9] also conducted an analysis to verify the existence of periodicity of speeds over time on a route segment. [10, 11] verify the pattern by measuring the correlation between the traffic on a specific route of different time periods. Such pattern verify the possibility of using historical data of a certain segment to predict the future traffic condition on the same segment.
Chapter 4

Data Analysis

Given a number of trajectories that consists of travel times, we perform an analysis to explore the correlation and patterns inside the trajectory data. Out goals of the data analysis are two-fold: i) we would like to verify the availability of using historical trajectory to the current bus ride to do prediction of the future travel times; and ii) we would like to explore any pattern inside the data that can be used for the prediction.

4.1 Data Scenario

We have all our data collected of a year, from March 2010 to March 2011, from buses of the city of Taipei. Each bus in Taipei is equipped with the GPS device that records the status of the bus along with its movement. The data consists of the instant speed, GPS coordinates and a time stamp. Each bus and each route have their own identification respectively. We also get information about bus stations on routes. Each bus station has its own name, GPS coordinates as well as the ID of the routes it belongs to. Therefore we can find all bus stations for each route. For a specific route, a bus station has a sequence number among all bus stations belonging to this route. In most cases, there are more than one bus traveling on a route. And for each bus, it travels on a fixed route for several times a day and the number of times is dynamic. Even in a day, the bus does not necessarily keep running all the time and sometimes are out of services for unknown reasons. Taking the data of one day for example, on March 16th, 2010, there are totally 3,893 buses running on 394 routes.

The route with an ID 10283 is taken to do experiments. This route has 64 stops, namely 63 segments, with the first stop Xinzhuang and the last stop Dr.Sun Yat-sen Memorial Hall. The distance of it is 47.4 kilometers. From March 2010 to April 2011, there are totally 24,985 trajectories.
4.2 Correlation between Segments

First, we would like to see whether there are correlations between segments that can be used to verify that we are able to use historical trajectory to the current one in terms of passed segments to predict the travel times of the future segments. Before processing the analysis, we explain why we utilize correlations between segments. Suppose a correlation in terms of travel times commonly exists between segments, previous segments are related to later segments along the route. Given a current trajectory and its similar historical trajectory in terms of passed segments, the correlation works for both of them. Therefore, both of their travel times of future segments are likely performing in a similar way, and subsequently their future travel times are also similar with a high opportunity.

We use Pearson’s correlation as the tool to measure the correlation between segments. Pearson Product Moment Correlation (Pearson’s correlation for short) is widely used to measure the linear association between two variables. The value of Pearson’s correlation is between -1 and 1. Positive values mean positive correlations and negative values mean negative correlations. The stronger the correlation is, the farther the value should be from 0. Given two variables \(X\) and \(Y\) with means \(\bar{X}\) and \(\bar{Y}\) respectively and standard deviations \(S_X\) and \(S_Y\) respectively. The correlation \(\gamma\) is computed as

\[
\gamma = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n - 1)S_XS_Y}
\]  

where \(n\) is the number of elements in \(X\) and \(Y\).

Usually the correlation between segments should not keep constant all the time. The farther two segments are from each other, the weaker the influence of one on another one should be. To detect the Pearson’s correlation for any two segments as well as its trend along with the increasing of distance between two segments, we use a specific figure, Figure 4.1, to represent the Pearson’s correlation between segments. The travel time of a segment is defined as a variable, and the historical travel times on it are the values of the variable. We use Y-axis values to represent the Pearson’s correlation between two variables and X-axis values to represent the number of segments between them, which is termed as Segment-Distance. For example, given a Pearson’s correlation between segment 20 and segment 25, a corresponding point is drew on the figure with the X-value being 5, which is 25 minus 20. However, such a figure is not able to offer a clear illustration of the change of Pearson’s correlation because there are too many points for each X-axis value. To solve this problem, we provide another figure, Figure 4.2, to represent the average value of all points for each X value.

As shown in Figure 4.1, the Pearson’s correlation exists commonly between any arbitrary segments. However, the correlation does not appears to be high for most pair of segments. Specifically, when two segments are near to each other, for example, adjacent, the Pearson’s correlation is remarkable and obviously higher than others. Therefore a segment is more related to near segments than farther ones. Moreover, Figure 4.2 indicates an apparent decline curve
from 1 along the X-axis. We can conclude that historical trajectories similar to the current trajectory in terms of close passed segments are more reliable for prediction than that in terms of all passed segments.

### 4.3 Patterns Exploration

The second goal of the data analysis is to explore any pattern inside the data that can be used for the prediction. Intuitively, travel times of a segment are not only related to that of near segments, but also to some features that can be explored by analyzing possible patterns inside the historical data. For example, in a city area, the traffic conditions are usually the worst during a day in rush hours in the morning and in the afternoon. Therefore, we can associate the travel times to a timely feature. Similarly, the travel times of a same segment may appear differently in weekdays and weekends. In weekends, the travel time would be higher than that in weekdays because people drive out making congestions than usually. On the other hand, the travel time could also be lower than those in weekdays because of less vehicles in rush hours.

Limited by data sources, we are not able to explore every feature. In this project, we process analysis on time and day to see whether they can be used as features for travel times. As mentioned, during peak hours in the morning and afternoon, congestions happen with a high probability. Therefore the travel time of a segment maybe high during peak hours and low in non-peak hours. We partition a day into 24 time periods, each of which is of 1 hour. Figure 4.3 indicate the trend of travel times along a day. First, for each of the segments, travel times of it are allocated into 24 sets in terms of the hour the travel time happened at. we can see the
Figure 4.2. Average Pearson’ Correlation.

X-axis representing each hour. The Y-axis represents the travel time in seconds. This figure is generated from Segment29, Segment33 and Segment45 of one year’s data. For each boxplot, the red line in the middle of the box is the median. The upper edge and lower edge of the box is the %75 and %25 of the data. Some data regarded as outliers are marked as red crosses.

According to Figure 4.3, travel times in the morning from 7am to 9am and in the afternoon from 4pm to 7pm are relatively higher than others, just as we suppose. The rush hours, which is termed as peak hours in our project, consists of two hours from 7am to 9am and from 4pm to 7pm. The other hours are termed as non-peak hours. It can be expected that travel times of a segment happening in peak hours are more similar to that also in peak hours, and travel times in non-peak hours are also likely to be similar to each other.

Travel times are not only related to the hour it happens at, but also to the day it is in. To verify the correlations between travel times and the day this travel time happened at, we classify the day into 7 classes, each of which represent a day in a week. And we also use boxplot to represent the highest, lowest and average travel times of a segment in each day.

As Figure 4.4 indicates, travel times in weekdays does not differ from each other too much. However, we can see that the travel times drop dramatically during weekends. The difference between weekday and weekend are apparent. Such a trend supports the hypothesis that the travel times change in a similar way in a week. From Monday to Friday, the travel times do not vary too much. The travel times in Friday are slightly higher probably because people who do not return home before Friday, for example, students, also take vehicles to home, causing more serious traffic congestions. And in the weekend, travel times are obviously lower than that in weekdays.
Figure 4.3. Analysis by hours.
Figure 4.4. Analysis by days.
Chapter 5

Framework Overview

Through the data analysis presented earlier, we observe the correlations between the segment travel time and the various trajectory features. Accordingly, we aim to exploit the travel time patterns exhibited in similar trajectories to develop a novel travel time prediction framework, called Historical Trajectory based arrival/travel Time Prediction (HTTP), based on a large collection of historical bus trajectories. In this section, we first provide an overview of the proposed HTTP system framework and then, in Chapter 6 discuss a number of similar trajectory based prediction schemes proposed under this framework. Figure 5.1 shows our system design of the HTTP framework.

As illustrated, the proposed HTTP system (i.e., an location based service server) continu-
ously collects bus trajectory data from GPS-equipped buses which report the latest bus status including time-stamped geographical coordinates of the bus and instant speed. The HTTP server is responsible for receiving and storing the trajectory data, monitoring the not-yet-completed trajectory journeys of on-going buses serving on their corresponding bus routes, and making prediction of bus travel time on the routes in responding to (i) passenger enquiries and (ii) real time update of bus arrival time at bus stops. As shown in Figure 5.1, the HTTP server consists of three modules: a) Bus Status Monitoring (BSM) module; b) Travel Time Prediction (TTP) module; and c) Similar Trajectory Search (STS) module.

The BSM module is responsible for communicating with the buses to receive bus status information and GPS data updates of the on-going trajectories. Once an update from a bus reaches the server, BSM catches the bus status (such as instant speed, current bus coordinate and new time stamp) of , extracting features associating with the developing trajectory , and store the information as part of in the historical trajectory repository.

The TTP module is responsible for predicting the arrival times of buses to bus stops, which can be reduced to a problem of predicting the travel times of buses on their remaining route segments. As mentioned, the TTP module can be invoked to make predictions by (i) a passenger enquiry; or (ii) the realtime updates of bus arrival information at stops. The former arrives on demand and the latter usually happens periodically. Without loss of generality, we consider the latter scenario as (i) can be considered as a simplified case of (ii). In this paper, for simplicity, we focus on predicting the travel time of a bus, given its current location, on remaining segments of its journey on the bus route. Moreover, instead of constantly making predictions, we assume that TTP is invoked every time when BSM receives the updated bus status (including the GPS data of bus location) and passes the required input parameters for prediction to TTP. Our idea behind the TTP module is very simplefind a sample set of historical trajectories similar to the ongoing bus journey as a statistical base to estimate the travel time for prediction.

Obviously, TTP relies on the STS module to search for similar trajectories effectively and efficiently. As there could be different ways to identify the sample set of similar trajectories, different notions of similarity could be explored to ensure the effectiveness of TTP. On the other hand, with a massive amount of historical data, it is infeasible to make exhaustive comparison between the trajectory of current bus journey against all the historical trajectories in the database. To ensure the search efficiency, we create indexes of trajectories and related patterns in the STS module to avoid retrieval of redundant trajectories that are not to be used for our travel time estimation. In other words, we only fetch a relatively small set of candidate trajectories and return them back to TTP.
Chapter 6

Algorithm

As discussed earlier, we design HTTP as a general framework to support travel time prediction. Based on this framework, the remaining issue is to devise similarity trajectory based prediction schemes which first invoke the STS module to retrieve a sample set of trajectories for making effective travel time estimation in the TTP module. Based on our data analysis, we observe the travel time correlation between two segments and the travel time patterns corresponding to some temporal features such as hours and days. Therefore, we follow these observations to introduce two different schemes based on passed segments (PS) and temporal features (TF). As their names suggest, these two schemes use the passed segments and temporal features of an on-going bus journey, respectively, to identify similar trajectories for prediction.

6.1 Passed Segments Scheme

The PS scheme is to do prediction by finding the historical trajectories “similar” to the current one in terms of the travel times on road segments between passed stops. Thus a similarity measuring algorithm has to be taken into consideration. As mentioned before, the conventional algorithms measuring similarity of time series, $L_p$-norm, DTW and LCSS, are not appropriate for our project. First, those algorithms are highly sensitive to any error or outlier in the data. As a result, a slightly variation in the collected data might results in dramatically dismatches between the current trajectory and historical data. Second, these algorithms only evaluate the overall similarity of the whole trajectory, but leaving the similarity of trajectories on each segment unknown.

To address the above problems, we propose a new similarity measure that take into account the similarity between two trajectories on each segment. Given two trajectories, $\langle \Delta t_0, \ldots, \Delta t_n \rangle$ and $\langle \Delta t'_0, \ldots, \Delta t'_n \rangle$, we compare each pair of travel times $\Delta t_i$ and $\Delta t'_i$. If the the difference between each pair is less than the threshold specifically for this segment, the two trajectories are considered “similar”. This method improves the conventional distance measure algorithms in that for two “similar” trajectories, not only the whole one, but also on each segment they are
Table 6.1. All historical trajectories.

<table>
<thead>
<tr>
<th>Trajectory0</th>
<th>Segment0</th>
<th>Segment1</th>
<th>Segment2</th>
<th>Segment3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trajectory1</td>
<td>60s</td>
<td>500s</td>
<td>90s</td>
<td>120s</td>
</tr>
<tr>
<td>Trajectory2</td>
<td>90s</td>
<td>200s</td>
<td>100s</td>
<td>60s</td>
</tr>
<tr>
<td>Trajectory3</td>
<td>120s</td>
<td>120s</td>
<td>100s</td>
<td>60s</td>
</tr>
<tr>
<td>Trajectory4</td>
<td>180s</td>
<td>150s</td>
<td>100s</td>
<td>50s</td>
</tr>
</tbody>
</table>

Table 6.2. Partitioned Table 6.1.

<table>
<thead>
<tr>
<th>Segment0</th>
<th>Segment1</th>
<th>Segment2</th>
<th>Segment3</th>
</tr>
</thead>
<tbody>
<tr>
<td>50-99s</td>
<td>120-149s</td>
<td>60-89s</td>
<td>30-59s</td>
</tr>
<tr>
<td>100-149s</td>
<td>150-199s</td>
<td>90-119s</td>
<td>60-89s</td>
</tr>
<tr>
<td>150-199s</td>
<td>200-250s</td>
<td>120-140s</td>
<td>90-139s</td>
</tr>
<tr>
<td>200-230s</td>
<td>–</td>
<td>100s</td>
<td>140-149s</td>
</tr>
<tr>
<td>–</td>
<td>–</td>
<td>–</td>
<td>150-200s</td>
</tr>
</tbody>
</table>

“similar” enough.

However, this method is limited by the low efficiency that is caused by searching for similar travel times for each segment, especially when the number of historical trajectories is large. Fortunately, we do not need to search through historical data, but to allocate travel times into clusters and match a current travel time into a cluster.

To better illustrate this problem, we provide the following example. For a specific route, given a number of historical trajectories on this route, we can create a table with attributes corresponding to the travel times of each segment of a route and a record corresponding to each historical trajectory (Table 6.1).

By clustering algorithm that will be discussed later, we partition each column into several non-overlap ranges. Each range contains at least one value and each value only falls in one range. Table 6.1 can be transferred into a table of the following form, Table 6.2, where the number of ranges for each segment is not necessarily equal.

Given a current trajectory \( \langle \Delta t_0, \Delta t_1, \Delta t_2 \rangle \), the passed segments are segment0, segment1 and segment2. \( \Delta t_2 \) falls in a certain range for segment2 and we take this one as the match to \( \Delta t_2 \). All trajectories whose travel time of segment2 falls in the matching range are marked. The same operation is applied to segment1 and segment0 and then we can find trajectories whose travel times of the three past segment fall in all matching ranges. Since for each segment, their travel times are similar to the current trajectory, they can be considered as “similar” to the current trajectory and used for prediction.

### 6.1.1 Clustering Algorithms

In this section we consider the algorithms used to partition each sequence of quantitative values, \( ST_i^{raw} \), into a sequence of ranges \( ST_i \). Since \( ST_i^{raw} \) is a sequence of numerical values and can
be represented in a one-dimensional space. Splitting such a sequence is actually to allocate a set of one-dimensional data into clusters. We adopt two clustering algorithms. The first one is an widely adopted clustering algorithm, K-means algorithm, which is commonly applied to different applications since it works efficiently and effectively for various kinds of data. The second algorithm, V-Clustering, is specifically for one-dimensional data.

### 6.1.1.1 K-means Clustering

K-means is one of the simplest unsupervised and widely used learning algorithms that solve the clustering problem. Given a data set and a number, K, the basic idea of K-means is to allocate the data into K clusters by defining K centroids.

The procedure of K-means consists of 4 steps. Firstly, initial centroids for K clusters are defined. Those centroids should be as much as possible far away to each because their locations are highly related to the final clustering result. Next, a loop is generated to associate each data point to its nearest centroid. In this loop we re-calculate the new centroid for each cluster as the mean of the data belonging to this cluster. After K new centroids are calculated, a new binding is done between the same data set points and the nearest new centroid. As a result of this loop the K centroids change their location step by step until they do not change any more.

This algorithm aims at minimizing an objective function, which is a squared error function

$$ J = \sum_{j=1}^{k} \sum_{i=1}^{n} \| x_i^{(j)} - c_j \|^2 $$

where $\| x_i^{(j)} - c_j \|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centroid $c_j$. We use the objective function as an indicator of the distance of the n data points from their respective cluster centers.

### 6.1.1.2 V-Clustering

V-Clustering algorithm is introduced to allocate a sorted list of one-dimensional data into clusters. Authors of [40] proposes the algorithm to allocate travel times of transitions pertaining to a landmark edge. Given a list of one-dimensional data, $L$, we first sort $L$ according to the values, and then partition the sorted $L$ into several sub-lists in a binary-recursive way. In each iteration, we first compute the variance of all the data in $L$. Later, we find the “best” split point having the minimal weighted average variance (WAV) defined as:

$$ WAV(i; L) = \frac{|L_A^{(i)}|}{L} V(L_A^{(i)}) + \frac{|L_B^{(i)}|}{L} V(L_B^{(i)}) $$

where $L_A^{(i)}$ and $L_B^{(i)}$ are two sub-lists of $L$ split at the $i_{th}$ element and $V$ represents the variance. This best split point leads to a maximum decrease of

$$ \Delta V^{(i)}(L) = V(L) - WAV(i; L) $$
The algorithm terminates when $\max_i \{\Delta V(i)\}$ is less than a threshold, which is termed as $V_{\text{thresh}}$. As a result, we can find out a set of split points dividing the whole list $L$ into several clusters $C = c_1, c_2, ..., c_m$, each of which represents an range of travel times. For a cluster $c_i$, we are able to use the maximum and minimum value in it as the upper bound and lower bound of the corresponding range.

### 6.1.1.3 The comparison between K-means and V-Clustering

The differences between K-means and V-Clustering are on two aspects. Firstly, K-means can be applied to multi-dimensional data, but V-Clustering is only used when data is one-dimensional. Secondly, k-means finds clusters by measuring the distance between them. A data has shortest distance to the cluster it belongs to than others. On the other hand, V-Clustering finds a cluster by measuring the variance of a cluster. As well known, the variance is a measure of how far a set of numbers is spread out from its mean. So a set of data with relatively low variance is considered as a cluster.

Even though K-means is an efficient and widely adopted clustering algorithm, it is faced with two limitations. Firstly, the initial centroids are chosen randomly and different chosen may cause different clustering results. Another issue is how to determine the value of K. With no common direction on this problem, it is hard to offer a perfect value of K. V-Clustering, on the other hand, is highly constrained by the value of the threshold because the number of clusters is controlled by it. The number of clusters increases or decreases when the threshold is large or low, respectively. Just as same as K-means, we cannot know the perfect number of clusters at the beginning of the algorithm.

### 6.1.2 Range Matching

Now that a quantitative table with historical travel times successfully transferred into that with ranges by the clustering algorithm, we are able to do PS Prediction. Each time a bus travels to a new bus station and a new travel time of the most recently passed segment is obtained, we are able to discover the range where this new travel time falls in. With the travel time of each previous segment, actually we have known the corresponding ranges to all segments passed.

We offer an example to illustrate the process of Range Matching. Consider Table 6.3 as a table of historical trajectories with each attribute corresponding to a segment. As discussed before, we partition values of each attribute into ranges. Suppose that a bus has just passed segment2 and going to start segment3, because we have got the travel time of this bus on segment0, segment1 and segment2, we are able to find the range where each of these travel times falls in. In this example, they are $I_{01}$, $I_{12}$ and $I_{20}$ (marked by red color in Table 6.3).

In practise, travel times similar to the current one might not be only those in the matching range. Since there are errors and outliers existing in data, the value of a travel time may vary from its real value. For example, given two adjacent ranges $I_a$ and $I_b$, a travel time falling in $I_a$ highly likely belongs to $I_b$. In this case, although $I_a$ is the matching range, we should also take
travel times in $I_b$ into consideration. In practise, for a range, the travel time falling in upper
adjacent range or lower adjacent range is also considered as belonging to it.

The core of PS Prediction is to discover a RangeSet from a table like Table 6.3. Suppose the
range of segment3, which we are going to predict, is $I_{33}$ (marked by blue color), the RangeSet
we are looking for is \{$I_{01}, I_{12}, I_{20}, I_{33}$\}. Apparently, the first three ranges has been discovered
via a matching from historical trajectories to the current trajectory. But how can we find $I_{33}$?

As mentioned previously, PS Prediction aims to use similar trajectories to the current one in
terms of travel times on passed segments. A trajectory whose travel times of passed segments
falling in corresponding ranges can be considered as similar to the current trajectory. Thus,
firstly we try to explore the similar trajectories from ranges $I_{01}$, $I_{12}$ and $I_{20}$. The procedure is
termed as Segment-Filtering. Secondly, the future range, $I_{33}$, is defined as the range that future
travel times most likely fall in. Therefore, we propose to discover $I_{33}$ in a statistical way.

6.1.3 Segment-Filtering

We aim to find historical trajectories whose travel times of segment 0, segment 1 and segment 2
all fall in $I_{01}$, $I_{12}$ and $I_{20}$. Such a process is termed as Segment-Filtering.

Usually, it is expected that the most recent segment of a trajectory is more important for
predicting the future than the older segments. Such a hypothesis is verified in Data Analysis.
Based on Figure 4.2, we can find the correlations between a segment and its near segments are
obviously stronger than that between it and its farthest segments. Given two sets of trajectories,
$SET_a$ and $SET_b$, trajectories of $SET_a$ have travel times of segment 2 falling in $I_{20}$, and trajecto-
ries of $SET_b$ have travel times of segment 0 falling in $I_{01}$. We can conclude that chance of that
the trajectories in $SET_a$ have travel times of segment 3 falling in $I_{33}$ is bigger than that in $SET_b$.
Therefore, for two trajectories, $T_a$ and $T_b$, if $T_a$ is in $SET_a$ and $T_b$ is in $SET_b$, $T_a$ is more likely in
the set of trajectories with travel times in $I_{30}$. We denote $T_r$ as a set keeping record of trajectories.
$T_r$ records all historical trajectories initially. From the most recent segment, $S_2$, to oldest one,
$S_0$, we get rid of trajectory if its travel time of this segment is not in the corresponding range.
This process is termed as Segment-Filtering. Apparently, the number of $T_r$ is decreased during
each step of Segment-Filtering. The purpose of Segment-Filtering is to obtain trajectories that
have most influences on future segments.

In practice, the number of trajectories in $T_r$ is dramatically reduced through each step and
therefore $T_r$ is likely to be empty after the Segment-Filtering if the procedure is from the nearest
to the farthest segment. Even though $T_r$ is not empty, a relatively small size of it might cause
the result of prediction based on statistics insignificant. To solve this problem we introduce a
window that moves forward when a new travel time is submitted. The window sets a limit on
the length of Segment-Filtering. We only do Segment-Filtering in the window to make sure that
there are plenty of trajectories inside $T_r$ in the end. The window is visualized in Figure 6.1. The
bus moves along with time. It stops at bus station 3 at time $t_0$. At time $t_1$ and $t_2$, the bus
stops at bus station 4 and 5 respectively. Suppose the window is of a length of 3, we need to do
Segment-Filtering for the most recent 3 segments every time when the bus arrives at a new bus station.

However, the window cannot guarantee enough trajectories in $T_r$ every time. Minimum Number of Trajectory (MNT) is also introduced in the algorithm to set a minimum requirement of the number of trajectories in $T_r$. If the number of trajectories in $T_r$ is less than MNT during Segment-Filtering after an range, trajectories in this range are too different from those in other ranges, which means the current travel time this range corresponds to is an outlier. Therefore we undo the filtering on this range, skip it and move forward to the next one. After Segment-Filtering, to predict the travel time of a specific segment, for example, $S_3$ in Table 6.3, we use trajectories remaining in $T_r$ to discover $I_{33}$.

### 6.1.4 Statistical Predicting

We define the predicting method as Statistical Predicting. For the Table 6.3, the range to be predicted is $I_{33}$ and the ranges before this segment discovered through matching are $I_{01}, I_{12}$ and $I_{20}$. We are looking for a relationship from the passed ranges to $I_{33}$. We define such a relationship as $\langle I_{01}, I_{12}, I_{20} \rangle \Rightarrow \langle I_{33} \rangle$. $\langle I_{01}, I_{12}, I_{20} \rangle$ is termed as the antecedent and $\langle I_{33} \rangle$ is termed as the consequent.

After Segment-Filtering, we got $T_r$ with several trajectories in it. For each one of these trajectories, we are able to find out which range of segment 3 its travel time falls in. Given the
size of $T_r$ is $s$, for each range $I_{3i}$ in segment 3, the number of trajectories belonging to $T_r$ with a travel time of segment 3 falling in $I_{3i}$ is $s_i$. Since there are 4 ranges for segment 3, $\sum_{i=0}^{3} s_i = s$. We define the possibility that the trajectories falling in $I_{01}, I_{12}$ and $I_{20}$ fall in $I_{3i}$ as Confidence, which is computed as

$$Conf(I_{01}, I_{12}, I_{20} \Rightarrow I_{3i}) = \frac{s_i}{s}, \quad (6.4)$$

which indicates the probability of $I_{3i}$ given $I_{01}, I_{12}$ and $I_{20}$, $P(I_{3i}|I_{01}I_{12}I_{20})$.

Suppose $(I_{01}, I_{12}, I_{20} \Rightarrow I_{33})$ has the highest confidence, $I_{33}$ is the range that the travel time of $S_3$ of the current trajectory most likely falls in. We take $I_{33}$ as where the future travel time of $S_3$ is in. By a statistical way, we use the value in $I_{33}$ that appears the most times of all historical trajectories as the prediction for segment 3.

### 6.1.5 Algorithm of PS Scheme

First of all, we look for the range of the most recent passed segment in terms of the current trajectory, as well as the trajectories in this range (line1-line10). Secondly, Segment-Filtering is operated to get a set of trajectories (line11-line21). Finally, the prediction is discovered by looking for an range with highest confidence and a travel time in this range with the highest support (line22-line31).

### 6.2 Temporal Features Scheme

Besides PS Prediction, we also propose another prediction method that uses some features inside the historical data that are directly related to the travel time to do prediction. Using similar trajectories, we are able to provide satisfactory predictions. However, this method can not guarantee accuracy under all circumstances. It is limited under some cases. For example, when unusual events happening on a future segment, it is hardly to discover a reliable prediction from historical data because this event might never happen before, subsequently no similar travel times recorded. Fortunately, resorting to features related to traffic information on the current segment, we are able to make a prediction of travel times without information of previous segments. For example, the time when a bus enter this segment is important because the traffic changes along with time in a day. It is common that during peak hours in the morning and afternoon, congestions happen with a high probability. That’s why we use those important factors to discover the traffic condition that cannot be explored from similar historical trajectories and previous segments.

The TF Prediction consists of two steps. Firstly, we determine a series of features that directly influence the travel time of a segment; and do clustering on all historical travel times of each segment in terms of the features. Secondly, with the current traffic features, we decide which cluster a future travel time should be in.
6.2.1 Clustering in terms of features

The travel time on a segment is controlled by a series of features. As verified in data analysis, we extract two factors as the features of travel times: the time stamp when a bus runs on this segment and the day of the current trajectory. Time stamp is a feature highly related to travel times, which follows a hourly pattern. Thus travel times happening in the same rush period of the day, for example, 8am in the morning, are most likely to higher than that in other hours. We split a day into 24 time periods, and each of them is one hours. The similar pattern also exists for the week. Travel times in the weekdays are higher than that in weekends.

Given a segment $S_i$, the series of historical travel times on it, is $ST_{i}^{raw} = \langle \Delta t_0, ..., \Delta t_{M-1} \rangle$, where $M$ is the number of all historical trajectories. As discussed above, we extract a series of features for the travel time. So for each $\Delta t_j$ belonging to $ST_{i}^{raw}$, there is a vector, $V_j$, of features associated to it, which is termed as feature vector. Because we use two features, each $V_j$ is represented as $\langle v_0, v_1 \rangle$. We define the series of such vectors as $SV_i = \langle V_0, ..., V_{M-1} \rangle$, where each $V_j$ is corresponding to $\Delta t_j$ in $ST_{i}^{raw}$.

Given $ST_{i}^{raw} = \langle \Delta t_0, ..., \Delta t_{M-1} \rangle$ and its corresponding $SV_i = \langle V_0, ..., V_{M-1} \rangle$, we allocate $SV_i$ into $K$ clusters. Because $\Delta t_j$ and $V_j$ are associated, actually travel times are also allocated at the same time. In a cluster, all travel times share the same or similar features. Since the features are highly related to the value of the travel time, the travel times in a same cluster are likely to be close by value. The clusters we get is represented as $C_i = \{c_0, ..., c_K\}$, where $K$ is the number of clusters for $S_i$.

We tend to allocate the data into clusters in a natural way. As verified in data analysis, there are apparently peak hours and non-peak hours in a day, and the travel times in weekdays and weekends are very different. Therefore We allocate the historical data into four clusters, namely, the first cluster contains the travel times happening in non-peak hours and weekdays; the second cluster contains the data in non-peak hours and weekends; the third cluster contains the data in peak hours and weekdays and the data happening in peak hours and weekends are in the final cluster. Another possible partitioning method is to allocate the data according to the exact hour and day the data happened at. Since one day consists of 24 hours and a week has 7 days, there will be totally 168 clusters for this method. In the evaluation section, we will evaluate both partitioning method.

6.2.2 Cluster Matching

Every time the bus arrives at a new stop, features of the current travel times can be obtained. As mentioned, historical travel times of this segment has been partitioned into clusters, namely, all historical trajectories are partitioned. Through matching the current travel time to a cluster in terms of the features, we are able to discover which cluster the current travel time belongs to. To illustrate this process, we use the example of Table 6.3. To predict the travel time on segment 3, we have its historical travel times $ST_{3}^{raw}$, associated vectors $SV_3$ and a control vector, $V_{curr}$, of the current traffic features. $V_{curr} = \{v_{curr0}, v_{curr1}\}$. The current travel time belongs to the
cluster to which $V_{curr}$ has the corresponding time and date. Let $c_1$ is the matching cluster, we believe the current travel time on segment 3 is generated from $c_1$. And trajectories belonging to his cluster are most possible ones that are similar to the current trajectory. Therefore, we use the trajectories in this set to predict the travel time of future segment. For a specific segment, the travel time that happens most times in the set are chosen to be the predicted travel time.

6.3 Hybrid Schemes

PS scheme and TF scheme work from different aspect but also have something in common. The former is based on the correlations between future and passed segment. The trajectories discovered similar to the current trajectory is in terms of travel times. By contrast, TF scheme concentrates on the features that directly influence the travel time. Feature-Based Prediction also explores trajectories “similar” to the current trajectory, but the “similarity” is based on features of the current segment.

Since PS scheme and TF scheme are doing predictions via a set of trajectories, they can be applied to each other on the set of trajectories that generated from the other method. Therefore, we propose two different ways to do Hybrid scheme that combines them together as a single framework.

The first approach is to find a set of trajectories similar to the current trajectory in terms of travel times of passed segments and then filter the remaining trajectories using the temporal features cluster of the current bus journey. Thus we call this scheme hybrid passed segments/temporal features (HPT). On the other hand, the second approach, called hybrid temporal features/passed segments (HTP), first applies the TF scheme and then the PS scheme in clustering. Notice that the major difference between HPT and HTP lies in the process of filtering. For HPT, as it performs PS first, the segment filtering process remains the same and the matching with temporal feature clusters is executed by post-processing. On the other hand, for HTP, the clustering for both TF and PS can be pre-computed. Thus, the resulted clusters are fine-grained, capturing similar trajectories in both temporal features and passed segments.
Algorithm 1 Algorithm of PS Scheme

Input:

- $M$ historical trajectories;
- Current trajectory $T_{curr} = \langle \Delta t_{0}^{curr}, ..., \Delta t_{l}^{curr} \rangle (0 \leq l \leq N - 1)$;
- The number of segments, $N$; The most recent segment is $S_{l}$;
- $ST_{l}$, the sequence of ranges for $S_{l}$, is represented as $\langle I_{0}, ..., I_{K-1} \rangle$;
- The RangeSet based on travel time in $T_{curr}$, $SET_{interval}$;
- The set of historical trajectories, $SET_{all}$;
- window;
- $MNT$;

Output: Sequence of predicted travel times of future segments, $T_{pr}$;

1: for each $I_{k} \in ST_{l}$ do do
2: if $\Delta t_{l}^{curr}$ falls in $I_{k}$ then then
3: add $I_{k}$ into $SET_{interval}$
4: for each $T_{i} \in SET_{all}$ do do
5: if $\Delta t_{l}^{i} \in T_{i}$ falls in $I_{k}$ then then
6: add $T_{i}$ into $T_{r}$
7: end if;
8: end for;
9: end if;
10: end for;
11: for $i \in l - 1...l - window$ do do
12: $T_{r}' = T_{r}$
13: for each $T_{k} \in T_{r}$ do do
14: if $\Delta t_{i}^{i} \in T_{k}$ not falls in $I_{i} \in SET_{interval}$ then then
15: remove $T_{k}$ from $T_{r}$
16: end if;
17: end for;
18: if $|T_{i}| < MNT$ then then
19: $T_{r} = T_{r}'$
20: end if;
21: end for;
22: for $i \in l + 1...n$ do do
23: $I_{k} \leftarrow$ the range for $S_{i}$ to form a association rule with highest confidence based on $SET_{lr}$
24: for each $T_{j} \in T_{r}$ do do
25: if $\Delta t_{i}^{j}$ in $T_{j}$ falls in $I_{k}$ then then
26: add $\Delta t_{i}^{j}$ in $T_{r}$
27: end if;
28: end for;
29: add $\Delta t_{k}$ with the highest support in $T_{r}$ in $T_{pr}$
30: end for;
31: return $T_{pr}$;
Chapter 7

Performance Evaluation

In this section, we evaluate the performance of different prediction schemes and test various parameters. We use the data (Route ID 10283) presented in data analysis. The trajectory repository covers the whole year (March 2010 - Feb. 2011) of trajectories on the route. We use 508 trajectories in the first week of March 2011 as the testing data and ground truth to evaluate the accuracy of predictions.

To better illustrate the performance of our programs, we also use the data of another two routes. Route with an ID 112430 is taken to do experiments. This route has 23 stops, namely 22 segments, with the first stop Sanchong and the last stop Taipei Station. The distance of it is 19.6 kilometers. We use 928 trajectories in the first week of March 2011 as the testing data. Another route is with an ID of 11411. This route has 49 stops, namely 48 segments, with the first stop FuJen Catholic University and the last stop Yongchun High School. The distance of it is 43 kilometers. We use 504 trajectories in the first week of March 2011 as the testing data. For route 11411, this route is partitioned into two parts by a bridge connecting the Taipei city and another city. To get a consistent performance, we only consider the part in the city of Taipei, which has 24 segments.

A comprehensive set of experiments were performed. A number of parameters are tested in the experiments. Table 7.1 summarizes various parameters used in prediction schemes for route 10283 with their default values. Note that Vthresh denotes the threshold used in V-Clustering of the passed segment scheme. Window size and MNT denote the number of segments filtered and the minimum number of trajectories to maintain in the segment-filtering process of the same prediction scheme. K is used for K-Mode clustering in the temporal features scheme.

Table 7.2 and Table 7.3 summarizes the default parameters used for route 112430 and route 11411.

We use the \textit{prediction error} of travel times on future segments as the performance metric in our experiments. However, since the lengths of segments are different from each other, comparing the absolute errors in travel time is not reasonable. As we expect that the longer a segment is, the larger an error in travel time on this segment may likely be, we adopt the \textit{normalized error} in
Table 7.1. Parameters and default values for route 10283

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Size</td>
<td>12 months</td>
</tr>
<tr>
<td>Vthresh</td>
<td>100000</td>
</tr>
<tr>
<td>Window length</td>
<td>2</td>
</tr>
<tr>
<td>MNT</td>
<td>400</td>
</tr>
<tr>
<td>K</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 7.2. Parameters and default values for route 112430

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Size</td>
<td>12 months</td>
</tr>
<tr>
<td>Vthresh</td>
<td>500000</td>
</tr>
<tr>
<td>Window length</td>
<td>2</td>
</tr>
<tr>
<td>MNT</td>
<td>400</td>
</tr>
<tr>
<td>K</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 7.3. Parameters and default values for route 11411

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Size</td>
<td>12 months</td>
</tr>
<tr>
<td>Vthresh</td>
<td>500000</td>
</tr>
<tr>
<td>Window length</td>
<td>1</td>
</tr>
<tr>
<td>MNT</td>
<td>400</td>
</tr>
<tr>
<td>K</td>
<td>3</td>
</tr>
</tbody>
</table>

To illustrate the predicting results of different parameters, we propose to use the average normalized errors of all segments as the measurements to evaluate the overall performance of each value of the parameters.

In the evaluation, we aim to evaluate the accuracy of the proposed travel time prediction schemes in HTTP, including (1) passed segments (PS) scheme; (2) temporal features (TF) scheme; (3) hybrid passed segments/temporal features (HPT) scheme, and (4) hybrid temporal features/passed segments (HTP) scheme. Additionally, we use random prediction (RP), i.e., randomly select a trajectory to predict the travel time on the given segment, and average prediction (AV), i.e., use average of all travel times to predict the travel time on the given segment, as the baselines for comparison. Nevertheless, as the PS and TF schemes both employ clustering algorithms to partition trajectories into groups of similar trajectories in order to facilitate efficient retrieval, we first fine tune the two schemes. Additionally, given the sample set of similar trajectories retrieved, there are multiple methods to make a prediction based on the sample set. Thus, we next evaluate these different methods. Finally, we make comparison of all the prediction schemes in HTTP.
7.1 Tuning the Passed Segments Scheme

Selection of Partitioning Algorithms. We would like to decide which of K-Means and V-Clustering is a more suitable partitioning algorithm for the PS scheme. Thus, we first use the well-known K-Means algorithm to cluster the trajectories based on travel time of segments. We vary $K$ from 10 to 100 in step of 10 (with other parameters set in default) to observe the performance. Figure 7.1(b) shows the average normalized error of the $K$ from 10 to 100. As for V-Clustering, the number of clusters is not explicitly determined like in K-Means but controlled by the setting of $V_{thresh}$. The larger $V_{thresh}$ is, the smaller number of clusters are produced. We test V-Clustering with 10 thresholds, varying from 50000 to 500000 with step of 50000. The experiments were on all segments of the route and the number of clusters resulted for each segment are different. As shown in Figure 7.1(a), the performance is the worst when $V_{thresh}$ is 50000 and becomes better as $V_{thresh}$ increases. From Figure 7.1, we can tell that the PS scheme using V-Clustering obviously results in better performance than using K-Means. Therefore, we adopt 100000 as the default value for $V_{thresh}$ and use V-Clustering as the default partitioning algorithm for the PS scheme.

As for another two routes, we also use the K from 10 to 100 in step of 10 (with other parameters set in default) to observe the performance. Figure 7.2(b) and Figure 7.3(b) shows the average normalized error. We also test V-Clustering with 10 thresholds, varying from 50000 to 500000 with step of 50000. As shown in Figure 7.2(a) and Figure 7.3(a), the performances are both the worst when $V_{thresh}$ is 50000 and becomes better as $V_{thresh}$ increases. From Figure 7.2 and 7.3, as the same to route 10283, the PS scheme using V-Clustering is better than using K-Means. Therefore, we use V-Clustering as the default partitioning algorithm and the default value for $V_{thresh}$ of the two routes are both 500000.

Segment Filtering. In the PS scheme, Segment Filtering is used to select a set of trajectories similar to the current bus journey. In order to ensure there are reasonable number of similar trajectories returned for prediction, two mechanisms, namely, window of segment filtering (or window for short) and minimal number of trajectories (MNT), are placed in the PS scheme. In the following, we first evaluate the impact of different window size on the performance. As shown in Data Analysis, a segment correlated strongly only to its nearby segments. Therefore, instead of extending the experiment up to the maximal window size of 62 (i.e., there are 63 segments in total), we vary the window size from 1 to 8 with step of 1.

From Figure 7.4(a), we can notice that the predicting performance is obviously worse than the other situations when the number is from 3 to 8, which means the correlation between segments reduces dramatically when the number is larger than 2. Besides, there is no clear evidence that the performance with a window of 1 is better than that with a window being 2, or the opposite. However, when the window length is set to 1, there are too many trajectories left after the Segment-Filtering that increase the cost of our program. Therefore, we take 2 as the default value.

For route 112430, from Figure 7.5(a), we can notice that the predicting performance is obvi-
ously worse than the other situations when the number is from 3 to 8, which means the correlation
between segments reduces dramatically when the number is larger than 2. However, for route
11411, as illustrated in Figure 7.6(a), the predicting result achieves the best when the window is
1. Therefore, we take 2 as the default window for route 112430 and 1 as the default window for
route 11411.

We have to make sure plenty of trajectories left after Segment-Filtering. Besides the window
required, we also introduce the MNT. MNTs evaluated in these experiments are from 50 to 400,
with a step of 50. As illustrated in Figure 7.4(b), the predicting performance become better
along with the increasing of MNT. Actually, with a enhancement of MNT, there will be more
trajectories in $T_r$ after Segment-Filtering, which means the process of predicting a future travel times from $T_r$ is more significant in a statistical way because there are enough samples for a statistical process. Therefore, we adopt MNT of 400 as the default value. We can also notice similar trend of performance along with the change of MNT for route 112430 and route 11411. Therefore, we take 400 as the default value for route 112430 and route 11411.

**Prediction Methods.** In the PS scheme, we not only set values for different parameters, but also make a comparison between prediction methods. Besides PS, we provide another 2 prediction methods: RP and Survival Analysis (termed as SA). As mentioned before, random prediction (RP) is to randomly select a trajectory to predict the travel time on the given segment.
Survival Analysis is a widely adopted statistical analyzing process. With $T_r$ generated by Segment-Filtering from the historical data, any trajectory in $T_r$ is regarded as a similar trajectory to the current one. Therefore, the current trajectory should follow a similar distribution of $T_r$ for a travel time of the future segment. We create a normal distribution generator by $T_r$ to generate a travel time as the prediction.

As illustrated by Figure 7.7, Figure 7.8 and Figure 7.9, for the three routes, compared with RP and SA, PS scheme performs the best of all. Because both RP and SA only offer a single trajectory each time for the prediction, it can not be guaranteed that such a trajectory is good from $T_r$ generated from Segment-Filtering.

Figure 7.3. Partitioning Algorithms for route 11411.
enough for the prediction. That’s why PS can performs the best of all three methods.

7.2 Tuning the Temporal Feature Scheme

As mentioned in the Algorithm section, we are going to allocate the historical travel times of each segment into clusters by K-modes. Two temporal features are adopted: the time when a bus enter a certain segment, time, and the day when the bus ride happens at, day. The time can be classified into peak hours or non-peak hours and the days can be classified into weekdays or weekends. In this experiment, we firstly evaluate the different classifying methods for the two
features and pick up the best performance with the most appropriate K.

We propose the four classifying methods: First, naturally, time is classified into 24 and day is classified into 7, which is termed as 24-7. Secondly, time is classified into 24 and day is classified into 2: weekdays and weekends, which is termed as 24-2. Thirdly, we classify time into 2, peak hours and non-peak hours, and day into 7, which is termed as 2-7. Finally, we classify both time and day into 2 classes, which is termed as 2-2. For each of these, we did experiments with different Ks. We did experiments with different Ks. If the K is set to 1, there will be no clustering actually. Thus we set the K from 2 to 20, with a step of 1. The predicting results are displayed by 7.10.

Figure 7.5. Segment Filtering for route 112430.
Figure 7.6. Segment Filtering for route 11411.

As Figure 7.10 indicate, we can clearly notice that from 2 to 20, the predicting errors for the four classifying methods are all increasing slightly. However, the error of K being 2 is slightly worse than that with K being 3. If the K is set too small, for example, 2, historical data can not be allocated precisely that causes the prediction from a cluster to be not accurate enough. Therefore, with the performance achieving the best when the K is 3, we choose 3 as the default number of K.

We also did experiment to compare the four possible classifying methods with a K being 3. As illustrated in Figure 7.11, there is almost no differences between them.

Such experiments are also applied to another routes. For route 112430, along with the change
of $K$, the predicting performances of the four classifying methods are different. However, when we compare the four methods with their best $K$, their performances are almost the same. Since classifying time into 24 and day into 7 can make a refiner classification, we adopt 24-7 as the default classification methods the default $K$ is 5. For route 11411, the four methods are almost the same the different of performance is not obvious. Therefore, we adopt 24-7 as the default classifying method and the default $K$ is 3.
7.3 Hybrid Prediction

After tuning the PS and TF schemes, respectively, we now evaluate the performance of proposed prediction schemes along with the average prediction (AP) and TransDB, the state-of-the-art technique for travel time prediction using historical trajectories. In this experiment, we compare the schemes under evaluation by considering segment distance, i.e., the number of segments between the current segment of bus and the predicted segment. For example, for segment distance equals 1, we predict the travel time of next segment from the bus position in every bus status report and compute their average normalized error; for segment distance equals 2, we predict the travel time of segment located two segment away from the bus position in every bus status report and compute their average normalized error; and so on. Figure 7.16 plots the experimental result. As shown, for all prediction schemes evaluated, the prediction error increases as the segment distance increases because it is more difficult to predict segments far away from the current bus location. AP and TransDB perform obviously worse than the four methods we proposed in HTTP. Since only ONE trajectory (i.e., the NNT) is fetched to make prediction, TransDB can not guarantee the accuracy of its prediction all the time. On the other hand, while AP uses the average value of all historical data to make prediction, the result is not satisfactory as it accommodates too many different situations and consequently compromises its prediction accuracy.

Unfortunately the differences between the four proposed schemes cannot be visualized clearly in the general plot of Figure 7.16. Thus, we zoom in to observe the performance in segment distance 1-5 (see the box within Figure 7.16; the result in other segment distances are consistent with the observation here). As shown, the two hybrid schemes, HPT and HTP, are better than PS and TP. Between PS and TP, TP is generally better than PS as it results in clusters in finer
granularity (as explained earlier in Chapter 6). Between the two hybrid schemes, the HTP scheme is slightly better than HPT because each time a new travel time is received, TF is performed before PS. Therefore, the final set of trajectories that are used to estimate the travel time are similar to the current bus journey in terms of passed segments not only by travel times but also by features. On the other hand, for HPT, TF is applied to the set of trajectories obtained from PS. Therefore the returned trajectories are similar to the current bus journey in terms of features corresponding to the current segment. Somehow HTP filtering is more strict than HPT. That’s why the HTP outperforms HPT.

For another two routes, we can notice similar phenomenon to route 10283 in Figure 7.17 and Figure 7.18. However, for route 11411, the performance of HPT and HTP in segment distance 1-5 are not obviously different from each other. Therefore, we represent the performance in segment distance 13-17, in which we can notice the HTP is obviously better than HPT.
Figure 7.10. Partitioning Methods.
Figure 7.11. Comparison of Partitioning Methods.
Figure 7.12. Partitioning Methods.
Figure 7.13. Comparison of Partitioning Methods for route 112430.
Figure 7.14. Partitioning Methods.
Figure 7.15. Comparison of Partitioning Methods for route 11411.

Figure 7.16. Comparison of Prediction Schemes for route 10283.
Figure 7.17. Comparison of Prediction Schemes for route 112430.

Figure 7.18. Comparison of Prediction Schemes for route 11411.
Chapter 8

Conclusion

In this paper, we study the problem of predicting bus arrival/travel time using historical trajectories. We propose a novel prediction framework, namely, HTTP, and develop two basic travel time prediction schemes, passed segment (PS) and temporal features (TF). The former is based on the correlation between travel times of nearby segments in the same trajectory. The PS scheme identifies similar trajectories by using travel times on passed segments as the statistical basis of prediction. The latter is based on the temporal patterns of bus travel times in route segments. The TF scheme uses two temporal features “hours of a day” and “days of a week” to identify similar trajectories in terms of these features to make travel time prediction. As HTTP is a general framework, these two basic prediction schemes are combined to provide two hybrid prediction schemes, namely, hybrid passed segment/temporal features (HPT) and hybrid temporal features/passed segments (HTF).

We conduct a set of comprehensive experiments using a real data set collected from buses in Taipei City, Taiwan, to evaluate the effectiveness of the proposed prediction schemes with the average prediction (AP) scheme (a baseline) and the TransDB scheme (a state-of-the-art technique for bus travel time prediction). The result shows that all the prediction schemes proposed under HTTP significantly outperforms AP and TransDB. Meanwhile, the HTP scheme performs the best, which demonstrates that a better matching of similar trajectories through a strict filtering produce excellent prediction results.

As for the future work, we plan to explore more features and patterns from the historical trajectory data. Currently, we only explore temporal features. It will be interesting to discover the correlation of segment travel time with other possible features (e.g., weather). Finally, we aim to further enhance the prediction accuracy by considering historical trajectories not only in one bus route but also those in nearby routes.
Bibliography


